

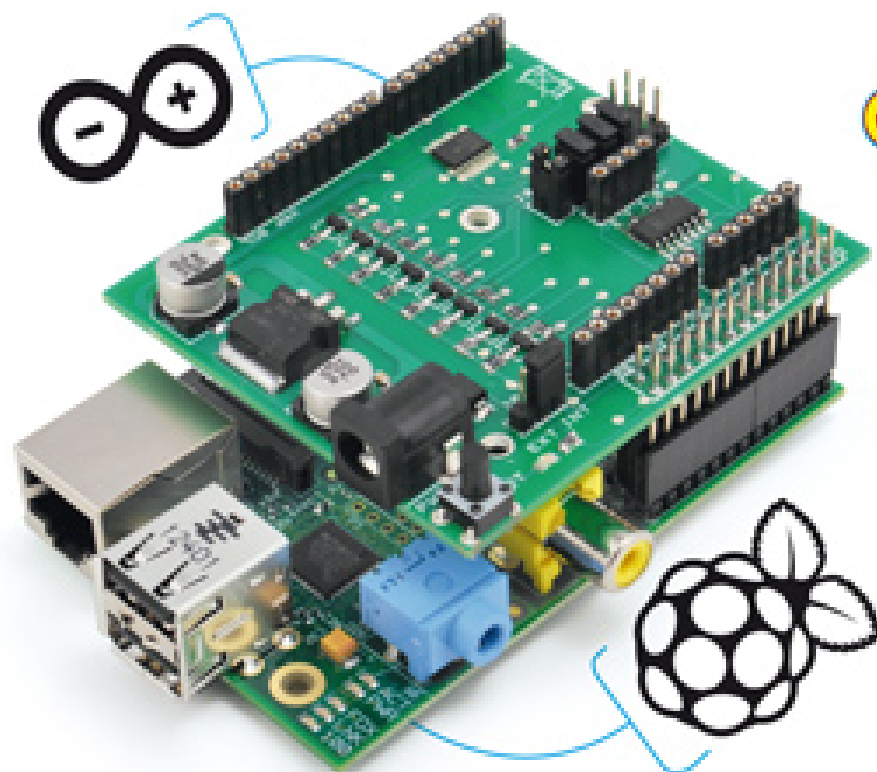
# Elettronica In

oltre l'elettronica

www.elettronica.in.it

Anno XIX - n. 173  
Febbraio 2013

€ 6,00



**OpenWheels**  
l'elettronica  
di controllo

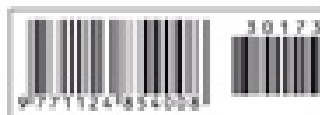


## Shield RaspberryPi compatibile Arduino

- Demoboard Bluetooth con RN-42
- Centrale Antifurto 24 zone, la costruzione
- Figgjo Clock: design Ikea, tecnologia Arduino
- myKIT, misure di peso con LabVIEW e myDAQ
- Applicazioni più smart con il nuovo MK802III
- RFTide: la wireless pan di Aurel
- Convertitore USB per programmazione Arduino
- Streaming audio con casse Bluetooth
- TV Tennis Game
- Robotica: gare e news



**Acquisizione con Kinect  
e stampa con 3Drag**

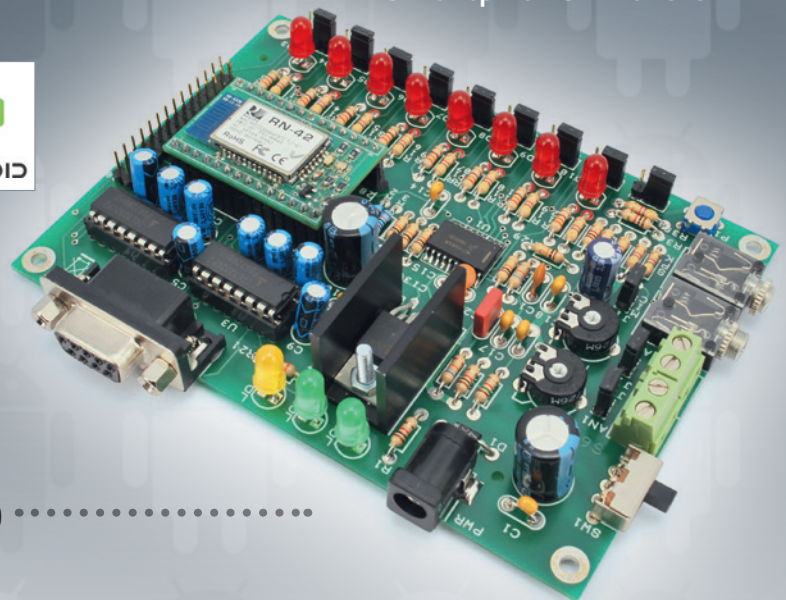






# DEMOBOARD BLUETOOTH RN-42

Risolveremo un progetto bluetooth pubblicato alcuni anni fa aggiornandolo con la nuova tecnologia attualmente disponibile sul mercato ed espandendolo con il collegamento ai moderni Smartphone Android.



degli Ingg. TOMMASO e ALESSANDRO GIUSTO .....

**A**lcuni numeri fa abbiamo presentato un primo progetto basato sul nuovo modulo bluetooth Roving Networks RN-42. A seguito della pubblicazione abbiamo ricevuto talmente tante richieste di maggiori informazioni sul nuovo componente che abbiamo deciso di approfondirne lo studio

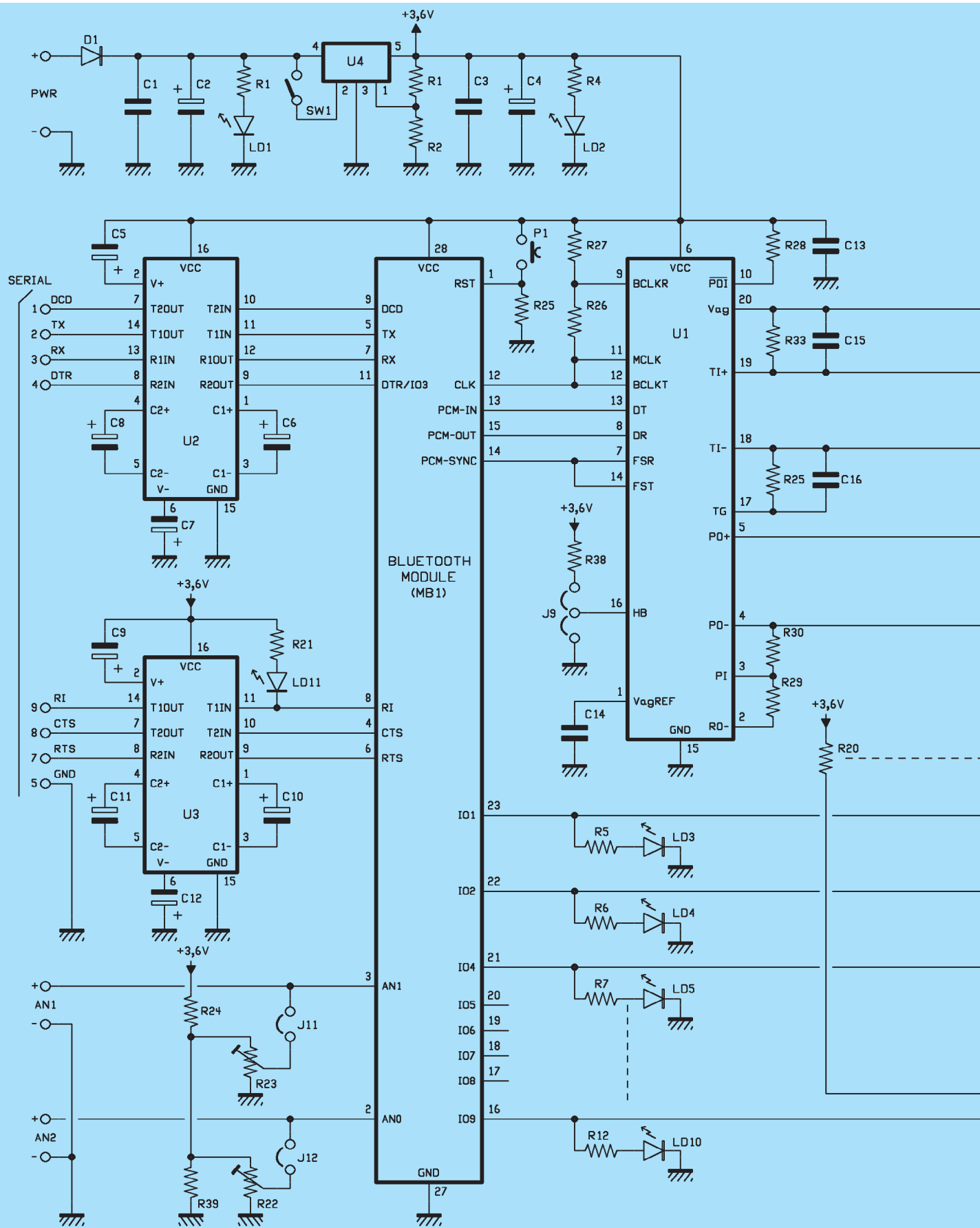
e fornirvi maggiori strumenti e conoscenze per la programmazione ed integrazione nei vostri sistemi embedded. Come strumento base per i nostri studi e relativi esperimenti abbiamo utilizzato un precedente nostro progetto, cioè la Demoboard per Bluetooth, presentata nel numero di marzo 2006

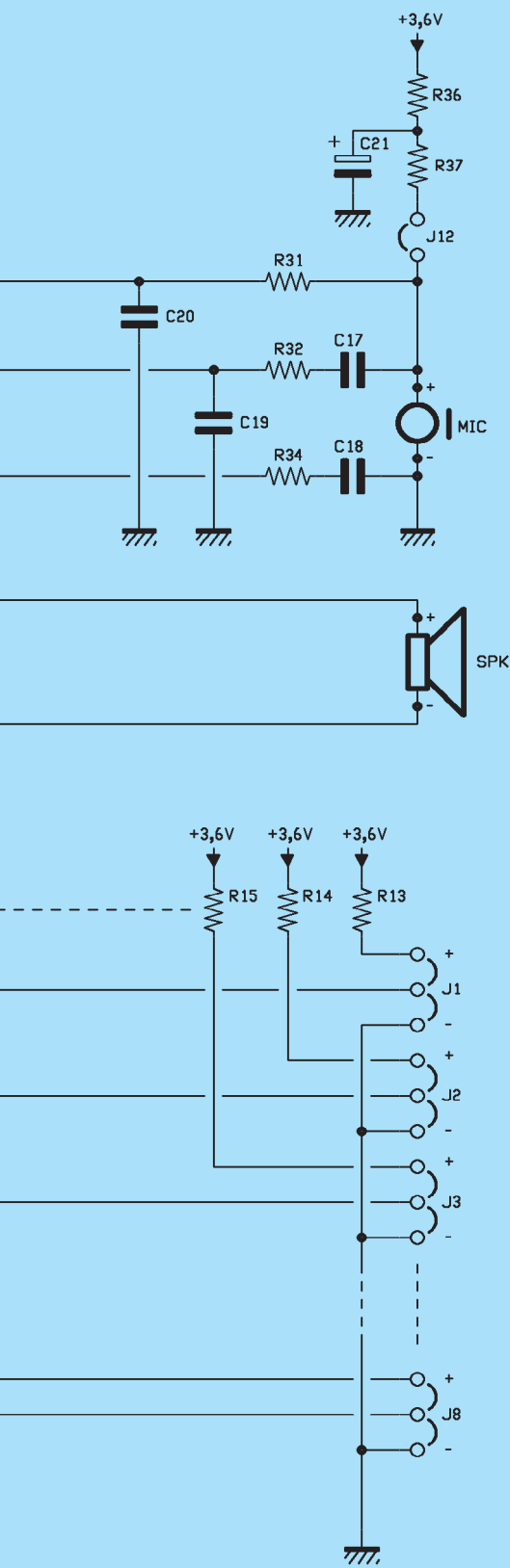
della rivista, della quale, dato il tempo trascorso, riportiamo lo schema elettrico ed il piano di montaggio.

## TECNOLOGIA BLUETOOTH

Il bluetooth è una tecnologia hardware e software di comunicazione RF tra più dispositivi in modalità full duplex. La

frequenza della portante utilizzata è a 2,4 GHz e la banda disponibile viene suddivisa in diversi canali. Quando due dispositivi vogliono comunicare stabiliscono l'utilizzo di un canale comune utilizzato per lo scambio dei dati; nel caso in cui la qualità del canale scenda al di sotto di una qualità minima, i





due sistemi scelgono un nuovo canale e contemporaneamente “deviano” la comunicazione su questo nuovo. Il protocollo software prevede due diverse classi di dispositivi: Slave e Master. Della prima classe fanno parte quegli apparati che sono sempre in attesa della richiesta di nuove connessioni (ma che non possono mai avviarne una nuova di propria iniziativa) mentre della seconda fanno parte i sistemi che possono eseguire scansioni per rilevare i dispositivi bluetooth presenti nel proprio raggio d’azione e che, in qualsiasi istante possono avviare e stabilire una nuova connessione. Dal punto di vista della sicurezza, già nativamente il Bluetooth è un sistema sicuro in quanto comprende un sistema per prevenire l’ascolto e l’accesso indesiderato alle comunicazioni (spoofing) basato sull’autenticazione degli utenti; in più il flusso dei dati è criptato e chiavi di accesso possono essere generate e cambiate durante la connessione. Sono presenti tre entità per garantire la sicurezza: una chiave pubblica (a 48 bit) per ogni dispositivo, una chiave privata (a 128 bit) generata e variata ad ogni inizializzazione ed infine un numero casuale (a 128 bit) generato per ogni nuovo scambio di dati.

#### HARDWARE MODULO RN-42

L’RN-42 è un modulo Bluetooth 2.1 di classe 2 con antenna di trasmissione integrata nel PCB del componente con capacità di comunicazione di circa 3 Mbps. La versione con antenna integrata ha un range di trasmissione e di utilizzo teorico di circa 15/20 metri che può salire anche fino a 1000 metri nella versione con antenna esterna. Il modulo supporta le modalità di funzionamento denominate SPP (Serial Port Profile) e HID (Human Interface

Device); implementa un efficace meccanismo di riduzione del consumo che lo porta, nelle migliori condizioni, a consumare circa solo 26uA anche se, nelle normali condizioni di utilizzo il consumo aumenta. Nella modalità SPP il baud rate di comunicazione è di 240 kbps per quanto riguarda i dispositivi slave e di 300 kbps per i master; velocità a nostro parere più che sufficienti per le più svariate applicazioni bluetooth (anche per quelle più esigenti). L’RN-42 dispone di una semplice interfaccia hardware e software UART che può essere utilizzata per il collegamento a qualsiasi microcontrollore per realizzare sistemi embedded; inoltre avendo integrato al proprio interno lo stack software bluetooth, non richiede il collegamento a MCU particolarmente potenti. Come prevede il protocollo bluetooth è presente la criptazione della comunicazione (vengono utilizzati algoritmi a 128 bit) e la correzione automatica degli errori di trasmissione; queste ultime due caratteristiche rendono l’RN-42 ideale anche per applicazioni particolarmente critiche in cui viene richiesta la massima sicurezza e trasmissioni dati senza perdite. Al momento unica nota negativa che ci sentiamo di segnalare è che il modulo non supporta la trasmissione dell’audio, ma solo di dati numerici. Infatti nonostante sul pin out siano presenti i 4 pin del formato PCM, in realtà questi non sono utilizzati e anche il datasheet li indica

#### Range Characteristics (Approximate range in office environment)

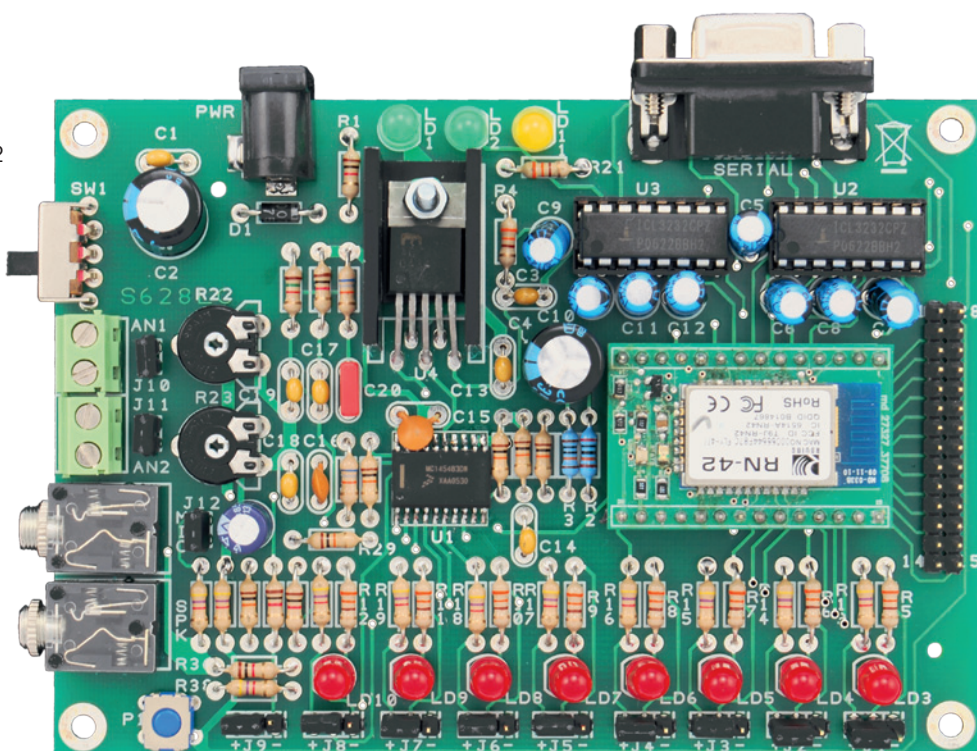
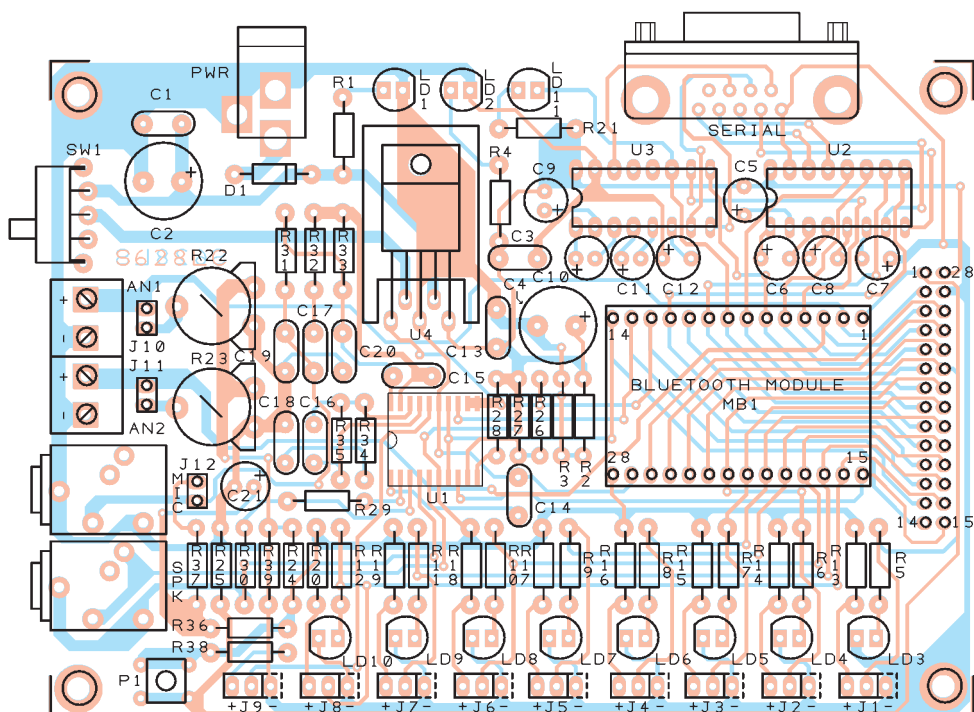
| Range             | RN-42   |
|-------------------|---------|
| After One Wall    | 55 feet |
| After Two Walls   | 60 feet |
| After Three Walls | 36 feet |

Fig. 1



## Elenco Componenti:

- R1: 1 kohm  
 R2: 200 Kohm 1%  
 R3: 100 Kohm 1%  
 R4÷R12: 330 ohm  
 R13÷R20: 4,7 kohm  
 R21: 330 ohm  
 R22: trimmer 47 kohm  
 R23: trimmer 47 kohm  
 R24: 1 kohm  
 R25: 4,7 kohm  
 R26÷R30: 10 kohm  
 R31: 1,5 kohm  
 R32: 1 kohm  
 R33: 68 kohm  
 R34: 10 kohm  
 R35: 68 kohm  
 R36: 1 kohm  
 R37, R38: 4,7 kohm  
 R39: 1 kohm  
 C1: 100 nF multistrato  
 C2: 470 µF 25 VL elettrolitico  
 C3: 100 nF multistrato  
 C4: 470 µF 25 VL elettrolitico  
 C5÷C12: 1 µF 100 VL elettrolitico  
 C13, C14: 100 nF multistrato  
 C15, C16: 470 pF ceramico  
 C17÷C19: 100 nF multistrato  
 C20: 10 nF 100 VL poliestere  
 C21: 47 µF 35 VL elettrolitico  
 U1: MC145483DW  
 U2, U3: MAX3232  
 U4: MIC2941  
 MB1: Socket Bluetooth con RN-42 (cod. FT1018M)  
 D1: 1N4007  
 P1: Microswitch  
 LD1, LD2: led 5 mm verde  
 LD3÷LD10: led 5 mm rosso  
 LD11: led 5 mm giallo  
 SW1: Deviatore slitta 90°  
 Varie:  
 - Morsettiera 2 poli (2 pz.)  
 - Plug alimentazione  
 - Connettore DB9 90° femmina  
 - Presa jack 3,5 mm (2 pz.)  
 - Vite 3 MA 10 mm  
 - Dado 3 MA  
 - Dissipatore (ML26)  
 - Zoccolo 8+8 (2 pz.)  
 - strip femmina 14 pin (2 pz.)  
 - strip maschio 14 pin (2 pz.)  
 - strip maschio 3 pin (9 pz.)  
 - strip maschio 2 pin (3 pz.)  
 - Jumper (12 pz.)  
 - Circuito stampato codice S628



come NON-CONNECT. Il pin out e relativa descrizione del modulo l'abbiamo già descritto nel precedente articolo; in questo riprendiamo brevemente il datasheet per mostrarvi in Fig. 1 le distanze di comunicazione rilevate all'interno di un classico ambiente pratico. Come si vede nella tabella vengono mostrate le distanze di comunicazione nel caso in cui siano presenti degli ostacoli (muri) tra le due unità: nel caso di un solo muro sono supportate distanze di 55 piedi; nel caso di 2 muri di 60 piedi mentre di 3 muri di 36 piedi. Nell'articolo scorso abbiamo presentato anche il socket adattatore del modulo RN-42; come già anticipato il pin out del nuovo socket è stato pensato per essere compatibile con le precedenti schede bluetooth prodotte dall'azienda Futura Elettronica; in questo articolo utilizzeremo la demoboard per mostrarvi come configurare e lavorare con il nuovo dispositivo.

### SOFTWARE MODULO RN-42

Riprendiamo ed approfondiamo le informazioni fornite nel precedente articolo. Dal punto di vista software, il modulo viene gestito tramite opportuni comandi inviati attraverso il collegamento UART.

All'accensione il modulo si avvia in configurazione Data (trasmissione e ricezione trasparente dei dati); per entrare nella configurazione Command in cui il modulo accetta i comandi è necessario inviare i caratteri \$\$\$\$. Per uscire dalla stessa inviare i caratteri --- seguiti da <cr>.

Il modulo può funzionare secondo 6 distinte modalità bluetooth, selezionate tramite il comando SM:

- 1) Slave Mode (comando SM,0): modalità di default in cui il

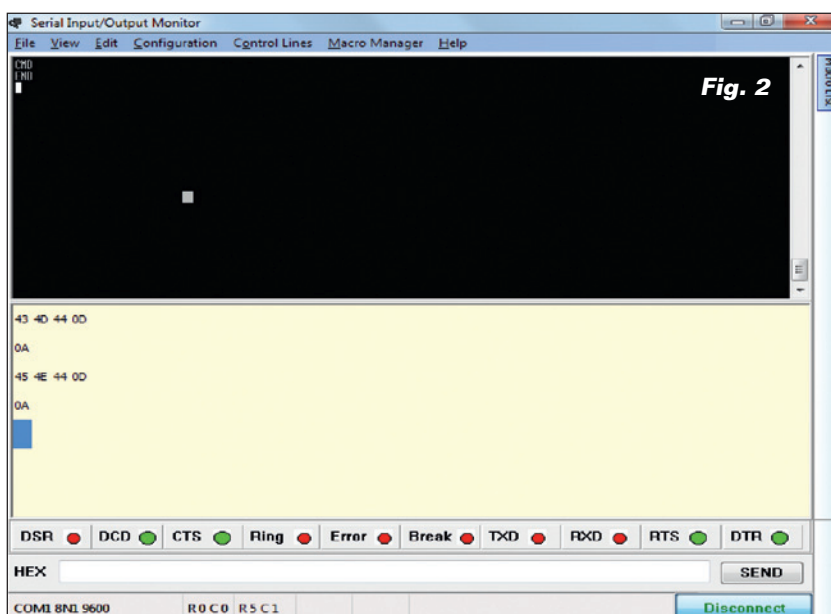


Fig. 2

modulo è in continua attesa di richieste di connessione che può accettare o rifiutare (non può invece avviare connessioni di propria iniziativa);

- 2) Master Mode (comando SM,1): modalità opposta alla precedente in cui il modulo può avviare connessioni di propria iniziativa ma non può essere localizzabile da altre periferiche;

- 3) Trigger Master Mode (comando SM,2): in questa modalità il modulo, ogni qual

volta viene ricevuto un carattere sulla UART locale, si connette automaticamente a uno slave a cui era stato precedentemente abbinato. La connessione rimane attiva per un intervallo programmabile tra 1 e 255 secondi;

- 4) Auto-connect (Master Mode) (comando SM,3): in questa modalità, all'accensione, il modulo si connette automaticamente a uno slave a cui è stato precedentemente abbinato. La connessione rimane attiva per

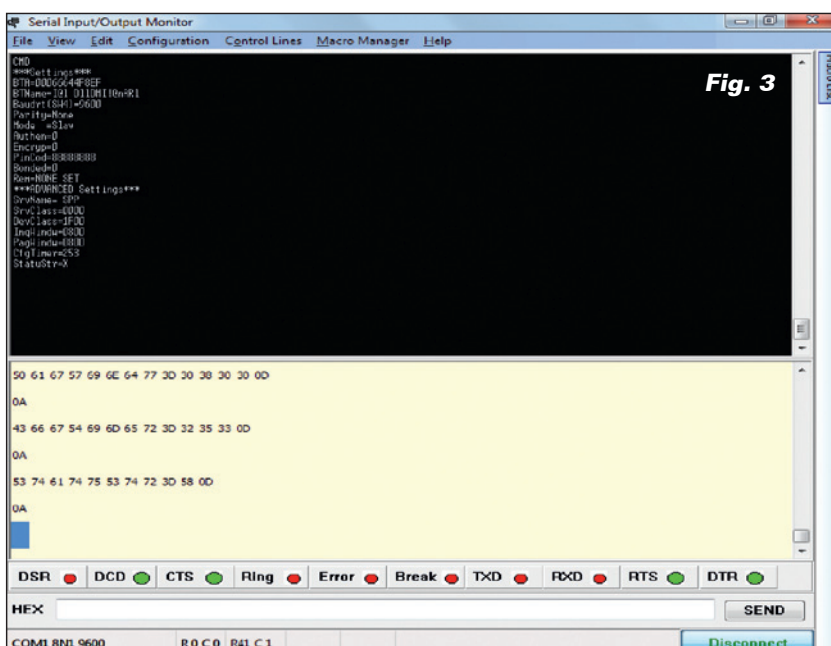


Fig. 3

tutto il tempo in cui il modulo risulta alimentato; nel caso in cui la comunicazione cada il modulo tenta automaticamente di ricollegarsi allo stesso slave;

5) Auto-connect (DTR

Mode) (comando SM,4): modalità simile alla precedente, tranne che è possibile avviare e chiudere la connessione tramite lo stato del pin PIO6.

6) Auto-connect ANY Mode

(comando SM,5): modalità simile alla precedente, tranne che ogni volta che varia lo stato del PIO6, viene avviata una nuova procedura di abbinamento a uno slave.

I comandi supportati dal modulo sono composti da 1 o 2 caratteri più alcuni parametri e vengono suddivisi in 5 classi principali:

- SET COMMANDS: utilizzati per la configurazione del modulo;
- GET COMMANDS: utilizzati per leggere la configurazione corrente del modulo;

- CHANGE COMMANDS: utilizzati per modificare temporaneamente la configurazione del modulo;
- ACTION COMMANDS: utilizzati per richiedere al modulo l'esecuzione di alcune azioni (avvio connessione, interrogazione dispositivi; ecc...);
- GPIO COMMANDS: utilizzati per configurare e manipolare i pin di I/O generici (GPIO) del modulo.

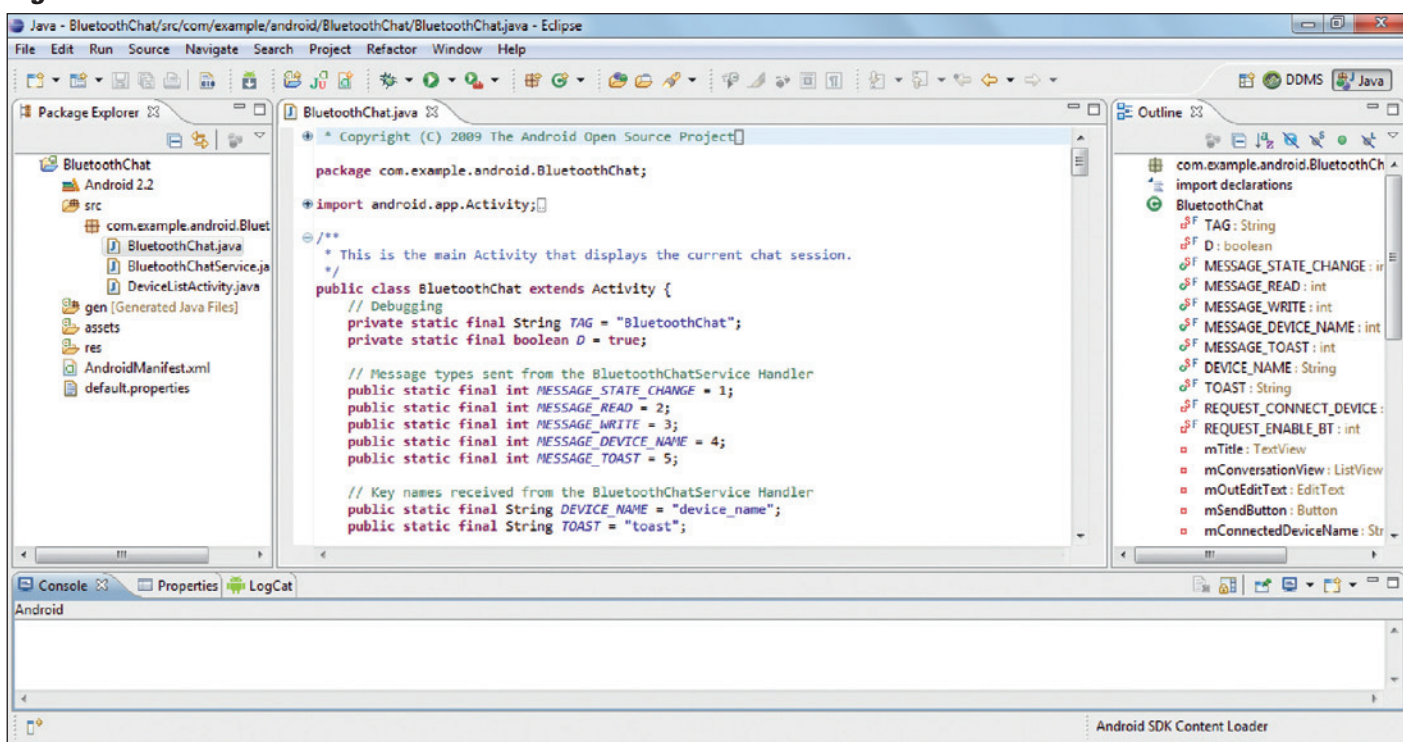
In questo articolo riprendiamo in considerazione molti dei comandi supportati:

- comando S7,<1,0>: attivazione/disattivazione della modalità di comunicazione seriale a 7 bit;
- comando SA,<1,0>: attivazione/disattivazione autenticazione. Se attivata, per ogni nuova connessione, verrà richiesta l'autenticazione;
- comando SB,<timer>: invia un segnale di BREAK al trasmettitore;
- comando SE,<1,0>: attivazione/disattivazione criptazione

comunicazione;

- comando SF,1: forzatura configurazione di fabbrica (default);
- comando SI,<hex word>: imposta il valore della finestra temporale denominata "Inquiry Scan Window", cioè la percentuale di tempo che il modulo passa nell'essere "rilevabile" dagli altri dispositivi bluetooth.
- comando SJ,<hex word>: imposta il valore della finestra temporale denominata "Page Scan Window", cioè la percentuale di tempo che il modulo passa nell'essere "collegabile" dagli altri dispositivi bluetooth.
- comando SL,<E,O,N>: settaggio parità UART. Può essere pari (E, Even), dispari (O, Odd) oppure nessuna (N, None).
- comando SM,<5,4,3,2,1,0>: impostazione modalità di funzionamento (comando appena analizzato sopra);
- comando SN,<name>: impostazione nome bluetooth

Fig. 4

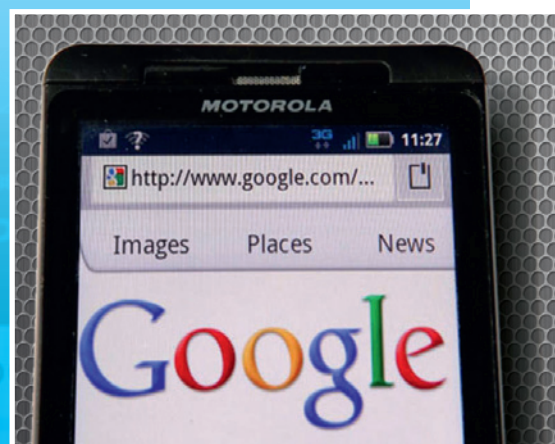




- associato al modulo RN-42;
- comando SO,<text>: imposta la stringa denominata "Extended Status String", cioè una stringa (massimo di 8 caratteri) che verrà inviata sulla porta seriale dal modulo RN-42 in caso di nuova connessione o disconnessione bluetooth. Per esempio, in caso di nuova connessione verrà inviata la stringa "<text>CONNECT" mentre in caso di disconnessione verrà inviata la stringa <text>DISCONNECT;
- comando SP,<text>: impostazione pin code sicurezza Bluetooth;
- comando SU,<rate>: impostazione baudrate porta seriale UART;
- comando SW,<hex word>: abilita la modalità "low power SNIFF" (di default disabilitata). La modalità SNIFF permette una notevole riduzione dei consumi in quanto il modulo RN-42 entra in "deep sleep" e si sveglia ogni 625 \* <hex word> microsecondi per inviare o ricevere dati;
- comando S|,<value>: impostazioni modalità "low power connect". Disabilita la sezione radio bluetooth e timer dei led nel periodo di tempo in cui il modulo RN-42 non è connesso. Il modulo continuamente passa da attivo (rilevabile e collegabile) a deep sleep. In questo modo viene realizzato un efficace sistema di riduzione dei consumi;
- comando D: lettura impostazioni base bluetooth del modulo;
- comando E: lettura impostazioni estese bluetooth del modulo;
- comando O: lettura ulteriori impostazioni bluetooth del modulo;
- comando GK: lettura stato corrente connessione bluetooth (1: modulo connesso; 0: modulo

## Arriva l'X Phone, Google lancia la sfida a Apple e Samsung

Google torna a sfidare Apple e Samsung con uno smartphone "rivoluzionario", pronto a competere con iPhone 5 e Galaxy. Allo 'X Phone', che dovrebbe essere presentato nei primi mesi del 2013, stanno lavorando gli ingegneri di Motorola Mobility, acquistata da Google nell'agosto del 2011. Top secret le caratteristiche del nuovo smartphone che dovrebbe essere dotato di un display "flessibile" e di novità



legate all'acquisizione da parte di Motorola della startup Viewdle, specializzata nell'elaborazione e nell'interpretazione di immagini e gesti. Secondo il Wall Street Journal, X Phone sarà solo il primo step nella 'riscossa' non solo di Google, ma anche del glorioso marchio Motorola, finito nell'ombra per il successo dei cellulari intelligenti.

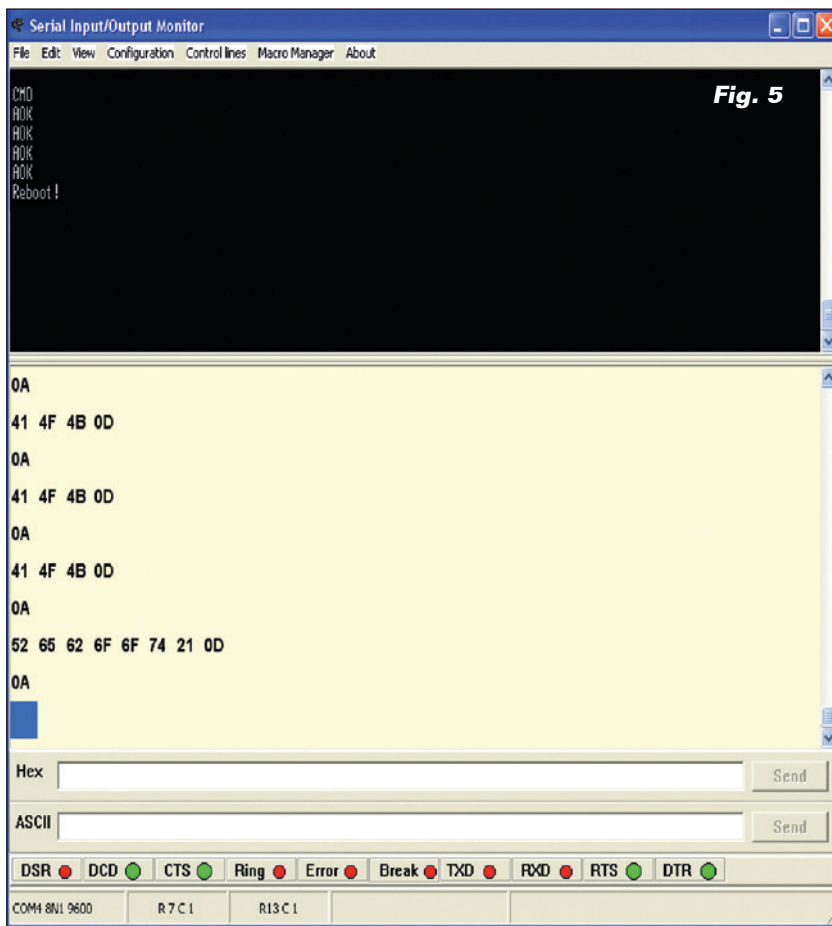
- disconnesso);
- comando G&: lettura stato pin I/O PIO;
- comando V: lettura versione firmware del modulo;
- comando \$\$\$: passaggio dalla modalità dati alla modalità comandi dell'RN-42 (dopo i caratteri \$\$\$, tutti i caratteri ricevuti attraverso la porta UART vengono considerati dal modulo come dei comandi);
- comando ---<cr>: passaggio dalla modalità dati dell'RN-42 (tutti i caratteri ricevuti dopo --- vengono considerati come dati da trasmettere attraverso il canale bluetooth);
- comando C,<address>: richiesta connessione ad un dispositivo specificato attraverso indirizzo MAC;
- comando I,<time>,<COD>: richiesta scansione "inquiry scan" (rilevamento dispositivi bluetooth presenti nel raggio di copertura). <Time> indi-

ca i secondi di durata della scansione (default: 10 secondi) mentre <COD> il codice numerico della classe bluetooth entro cui restringere la scansione. I dispositivi bluetooth rilevati vengono indicati secondo il formato <BT address>,<BT name>,<COD> dove il primo campo indica l'indirizzo MAC bluetooth, il secondo campo il nome associato al dispositivo ed infine l'ultimo la classe del dispositivo;

- comando K,: chiusura forzata connessione attiva;
- comando L: lettura indice qualità connessione attiva;
- comando R,1: forzatura reboot software modulo RN-42.

### DEMOBOARD BLUETOOTH

Come detto in precedenza nell'articolo, il nuovo socket per RN-42 è compatibile con le precedenti schede progettate alcuni anni fa dall'azienda Futura Elettronica. Per mostrare le potenzialità, confi-



**Fig. 5**

gurazione e programmazione del nuovo componente utilizziamo come base hardware una scheda Demoboard per Bluetooth (presentata nel numero di Marzo 2006 di Elettronica In) in cui sostituiamo il socket Ezurio con uno di nuovo tipo.

A questo punto è sufficiente realizzare una connessione RS232 tra un

PC e la scheda, aprire un software terminale seriale (HyperTerminal fornito con Windows può andare più che bene) e configurare la porta seriale con quella standard (9600 bps, nessuna parità, 8 bit di dati, 1 di stop e nessun controllo di flusso). Digitando i caratteri dei comandi nel terminale questi verranno trasmessi al modulo e

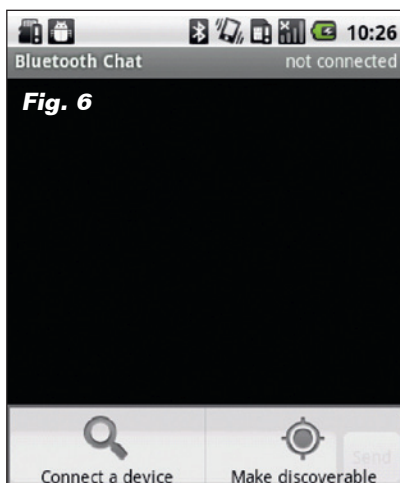
quindi è possibile verificare l'esecuzione degli stessi.

In **Fig. 2** mostriamo la risposta dell'RN-42 a due dei comandi da noi ritenuti basilari, \$\$\$ e ---<cr> (in figura mostriamo la schermata del terminale seriale da noi scelto, cioè software SLOW fornito con il compilatore per PIC CCS PCWH da noi solitamente utilizzato per la programmazione). Come detto i comandi permettono di passare dalla modalità dati a quella command e viceversa (il modulo risponde rispettivamente con CMD ed END).

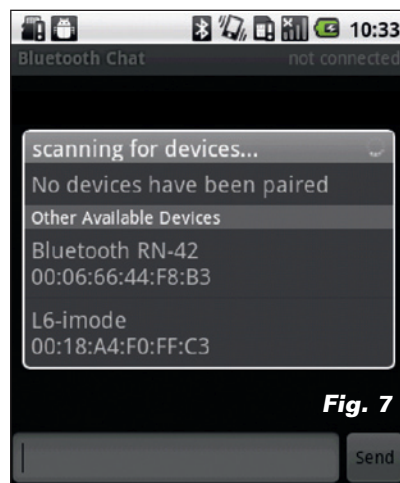
In **Fig. 3** mostriamo invece la risposta dell'RN-42 al comando di lettura configurazione D e E. Partendo da questa base è possibile provare ad inviare tutti i comandi supportati e verificare le varie risposte ricevute e come cambia il comportamento del modulo.

## RN-42 SLAVE CON COLLEGAMENTO ANDROID

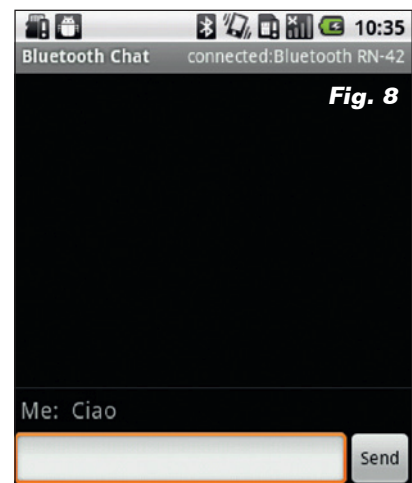
Come primo esempio applicativo del nuovo modulo, mostriamo come configurare l'RN-42 in modalità slave e come realizzare un collegamento bluetooth con software Android; dal lato hardware utilizziamo la demoboard bluetooth più socket RN-42 e dal lato smartphone un software demo bluetooth (software Blue-



**Fig. 6**



**Fig. 7**

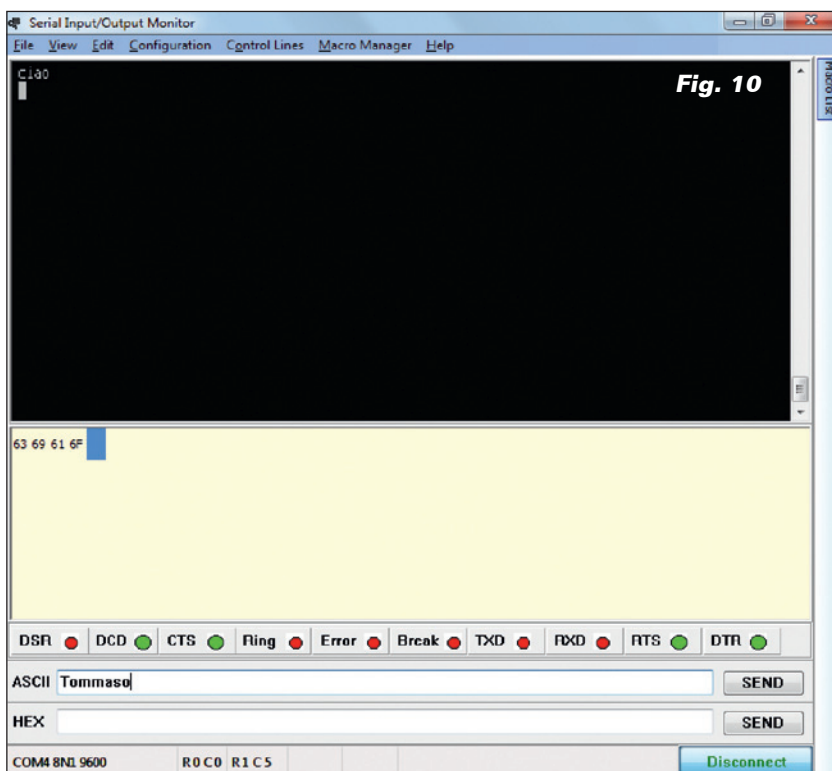
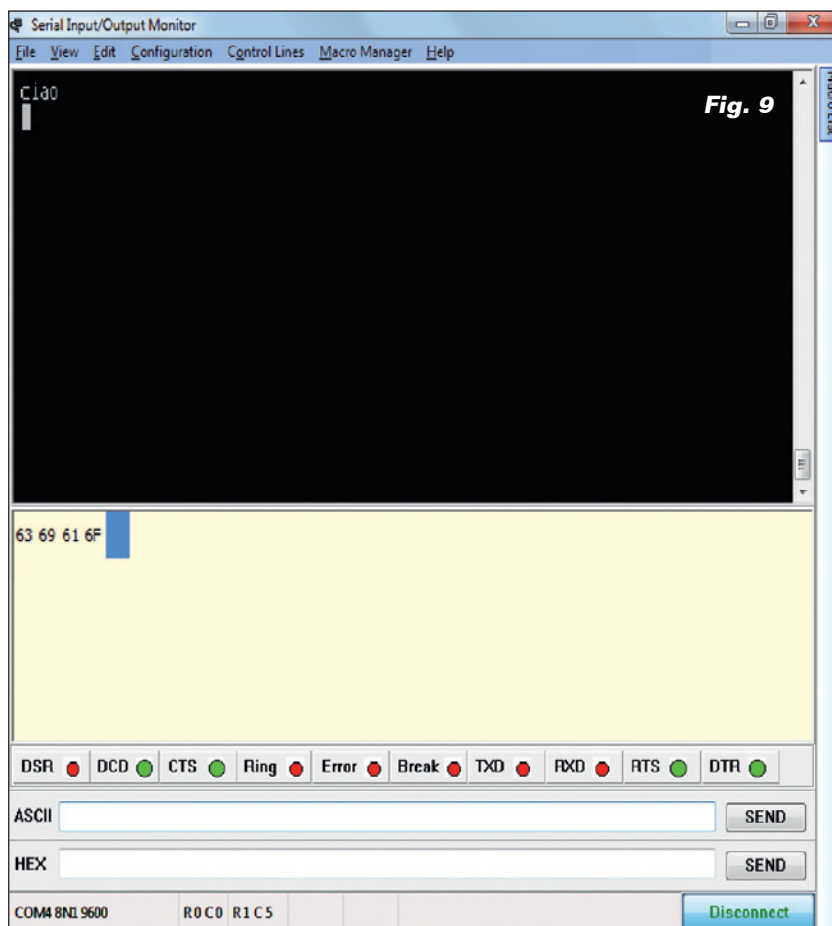


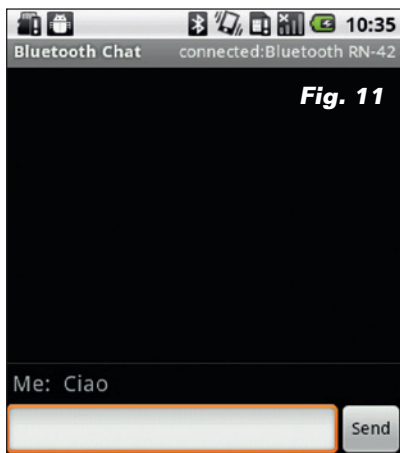
**Fig. 8**



toothChat) fornito con l'ambiente di sviluppo standard per Android (Android SDK integrato con IDE Eclipse entrambi scaricabili gratuitamente dai rispettivi siti internet). In Fig. 4 mostriamo il sorgente del software di esempio aperto nell'ambiente di sviluppo; unica modifica da effettuare nel sorgente Android è nel file BluetoothChatService.java alla variabile MY\_UUID che andrà impostata a 00001101-0000-1000-8000-00805F9B34FB (questa variabile indica il tipo di dispositivo bluetooth a cui collegarsi e il sorgente originale è impostato per telefoni generici mentre il nostro software andrà configurato per dispositivi embedded). A questo punto programmiamo il modulo RN-42 come dispositivo slave bluetooth; dal terminale seriale inviamo in sequenza gli opportuni comandi "\$\$\$" (selezione Command Mode), "SF,1<cr>" (reset software), "SM,0<cr>" (impostazione mode a slave), "SN,Bluetooth RN-42<cr>" (settaggio nome bluetooth), "SP,1234<cr>" (settaggio PIN bluetooth) ed infine "R,1<cr>" (reboot) (Fig. 5 mostriamo la finestra di esempio).

A questo punto lanciamo il software sullo smartphone e, tramite la voce "Connect a device" del menù (visualizzata in Fig. 6) selezioniamo il dispositivo a cui collegarsi tra quelli rilevati e mostrati in lista (in Fig. 7 mostriamo i dispositivi rilevati dal software in esecuzione su un nostro smartphone) e in seguito, dopo aver completato con successo il collegamento, tramite l'opportuno campo di testo presente nel fondo della pagina grafica, inviamo i caratteri che verranno ricevuti dal modulo bluetooth (e quindi visualizzati a video tramite software terminale seriale). In Fig. 8 mostriamo i caratteri inseriti nell'opportuno campo di testo





**Fig. 11**

ed inviati tramite bluetooth; in **Fig. 9** mostriamo invece gli stessi caratteri ricevuti e visualizzati dal terminale seriale. Viceversa, posizionandosi sul terminale seriale e digitando dei caratteri, questi verranno trasmessi dal modulo RN-42 e ricevuti dallo smartphone che li visualizzerà sul proprio display grafico. In **Fig. 10** mostriamo i caratteri inviati tramite software SLOW mentre in **Fig. 11** gli stessi caratteri visualizzati sul display grafico dello smartphone.

## RN-42 MASTER E RILEVAZIONE SLAVE

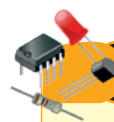
Dopo aver mostrato come è possibile utilizzare l'RN-42 come slave bluetooth e gestire le richieste di comunicazione provenienti dall'esterno, analizziamo la situazione opposta; cioè configuriamo il nuovo modulo come master bluetooth ed utilizziamolo per rilevare la presenza nelle vicinanze di slave bluetooth. L'operazione che permette a un master di eseguire una scansione del canale radio si chiama "Inquiry Scan" e dall'RN-42 è gestito attraverso il comando "I,<time>,<COD>". Iniziamo l'esempio programmando il modulo RN-42 come dispositivo master; dal terminale seriale inviamo in sequenza i comandi "\$\$\$\$" (selezione Command Mode), "SF,1<cr>" (reset software), "SM,1<cr>" (impostazione mode a master), "SN,Bluetooth RN-42<cr>" (settaggio nome blue-

tooth), "SP,1234<cr>" (settaggio PIN bluetooth) ed infine "R,1<cr>" (reboot). A questo, sempre da terminale seriale, selezioniamo nuovamente la modalità Command Mode (comando "\$\$\$") ed inviamo la stringa "I<cr>" per richiedere la scansione (in **Fig. 12** mostriamo la risposta al comando). Come si vede nel nostro campo di copertura sono stati rilevati 3 slave bluetooth; il primo con nome LG-P500 (il nostro smartphone) appartenente alla classe con codice 5A020C; il secondo con nome L6-imode (il nostro cellulare) della classe 500204 ed infine l'ultimo con nome Giovanni (il cellulare del nostro vicino di casa) della classe 5A020C. Come è facile intuire, la modalità Master del modulo RN-42 può essere utilizzata per realizzare un sistema di rilevazione RFID con attivazione e disattivazione di alcune funzioni a seconda dei dispositivi rilevati.

## CONCLUSIONI

In conclusione di articolo ci sentiamo di dare giudizi estremamente positivi sul nuovo componente; pensiamo che dopo un breve periodo di tempo di studio l'utilizzo e l'integrazione risulterà abbastanza semplice, veloce ed intuitiva. Ottimo strumento per prendere mano e "confidenza" con l'RN-42 potrebbe essere sicuramente la demoboard analizzata in questo articolo. Altra nota estremamente positiva e che vi assicuriamo è sempre ben vista da tutti i progettisti elettronici è il prezzo del componente che rispetto a soluzioni simili o equivalenti riteniamo decisamente interessante e competitivo. A nostro parere il modulo bluetooth RN-42 può essere utilizzato con successo in tutti quei sistemi embedded che richiedono possibilità di comunicazione verso l'esterno e potrebbe sostituire alcune tecnologie finora utilizzate; per esempio

potrebbe benissimo sostituire un collegamento seriale di tipo RS-232 risultando più comodo grazie al fatto che non sono presenti scomodi cavi di collegamento oppure potrebbe essere utilizzato al posto di una rete Wi-Fi con il vantaggio di non richiedere nessun componente (ma soprattutto costo) aggiuntivo come il router o gateway Wi-Fi. Come già sottolineato all'interno dell'articolo, la continua espansione e diffusione degli smartphone o palmari dotati come standard del bluetooth, permette il collegamento del modulo RN-42 a questi dispositivi e relativa gestione a distanza attraverso interfacce utente grafiche semplici, chiare ed intuitive. A nostro parere è proprio quest'ultimo punto che rende il componente presentato in questo articolo una valida scelta per i propri futuri progetti. ■



## per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono di facile reperibilità. Il firmware ed il software descritti possono essere scaricati gratuitamente dal sito della rivista ([www.elettronicaonline.it](http://www.elettronicaonline.it)) così come i master dei CS utilizzati nei progetti. La demoboard (descritta dettagliatamente nei numeri 106 e 107) è disponibile in scatola di montaggio al prezzo di 47,00 Euro (cod. FT628K). Il kit non comprende il socket Bluetooth comprensivo del modulo RN-42 che è disponibile già montato al prezzo di 29,00 Euro (cod. FT1018M). Ricordiamo che questo circuito è stato descritto dettagliatamente sul fascicolo 168 della rivista. Tutti i prezzi si intendono IVA compresa.

Il materiale va richiesto a:  
Futura Elettronica, Via Adige 11,  
21013 Gallarate (VA)  
Tel: 0331-799775 • Fax: 0331-792287  
<http://www.futurashop.it>



# STESSO NOME, MA NUOVE PRESTAZIONI, SEMPRE PIÙ SMART

Il Mini PC MK802III  
condivide con il  
modello precedente  
il nome e poco  
altro: CPU dual core  
a 1.6GHz e 8 GB di  
memoria Flash con  
Jelly Bean 4.1.1



di SIMONE MAJOCCHI

**I**l successo dei mini PC Android è un dato di fatto e nella migliore tradizione dei produttori del Far East, non è stato lasciato passare molto tempo prima che ci fosse una terza generazione pron-

ta a spostare l'asticella ancora più in alto. Con la versione III di MK802 cambiano ancora alcune cose, iniziando dalla forma che resta simile a una chiavetta USB, ma questa volta è più

sottile e larga, rinunciando alla serie di porte USB sparpagliate sul perimetro e portando la connettività alle due estremità. Da

un lato c'è l'HDMI e dall'altro c'è la porta USB HOST di tipo A, mentre l'alimentazione resta su microUSB, accanto a quella HOST. Su un lato si trova lo

## CARATTERISTICHE TECNICHE del Rikomagic MK802 III



- Sistema Operativo: Android 4.1.1 (Jelly Bean)
- Modello: MK802 III
- CPU: RK3066 Dual Core
- GPU: Quad- Core 2D/ 3D/ OpenGL ES2.0(AMD Z430)/ OpenVG1.1
- OS: Android 4.1
- Frequenza CPU: Cortex-A9, Up to 1.6GHZ
- RAM: 1G DDR3
- Memoria Flash: 8GB
- Slot Micro SD: T-Flash(Max 32GB)
- Browser: Jelly Bean
- Formati Video: WMV/ASF/MP4/3GP/3G2M4V/AVI/MJPEG/RV10/DivX/VC-1/MPEG-2/MPEG-4/H.263/H.264/1280\*720P HD 30 fps, 1080P/720\*480 D1 30fps
- Formati Audio: MP3/WMA/APE/FLAC/AAC/OGG/AC3/WAV
- Musica: Supporta lo streaming da Internet
- Immagini: Max.8000x8000 JPEG BMP GIF PNG
- WiFi: 802.11 b/g/n
- Flash player: Supports flash 11.1
- Porte: HDMI(maschio), Slot micro-SD, USB host, USB alimentazione; LED(Blu)
- Accessori inclusi: HDMI cable, cavetto USB alimentazione, manuale
- Dimensioni: 90 x 40 x 13 mm

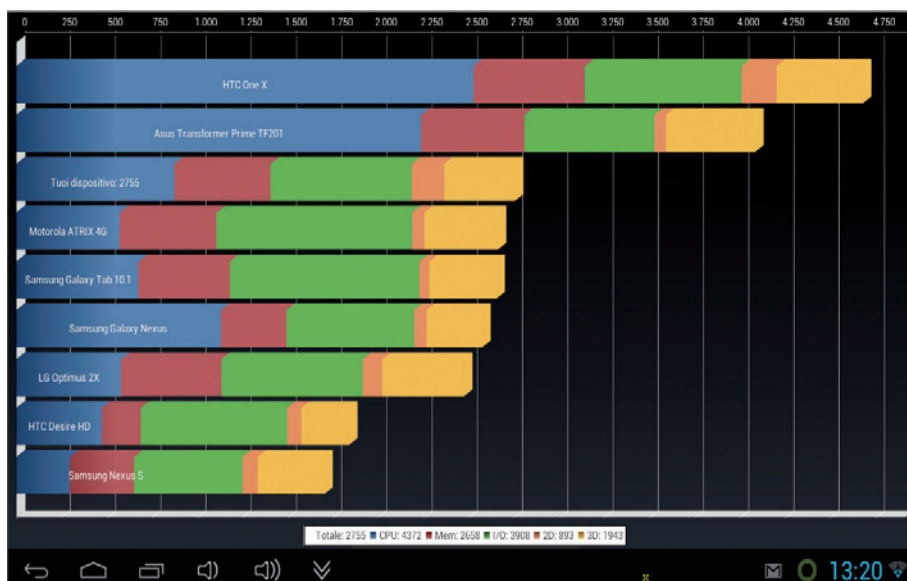
Oltre al collegamento WiFi, il Mini PC è compatibile con gli adattatori Ethernet USB, come quello riportato nell'immagine, per rendere il collegamento più semplice o più affidabile.



Questo benchmark ci permette di posizionare le prestazioni di MK802 III rispetto ad altri prodotti basati su Android. L'indice superiore riguarda la "produttività", mentre quello inferiore è invece legato alle prestazioni grafiche, per i giochi.

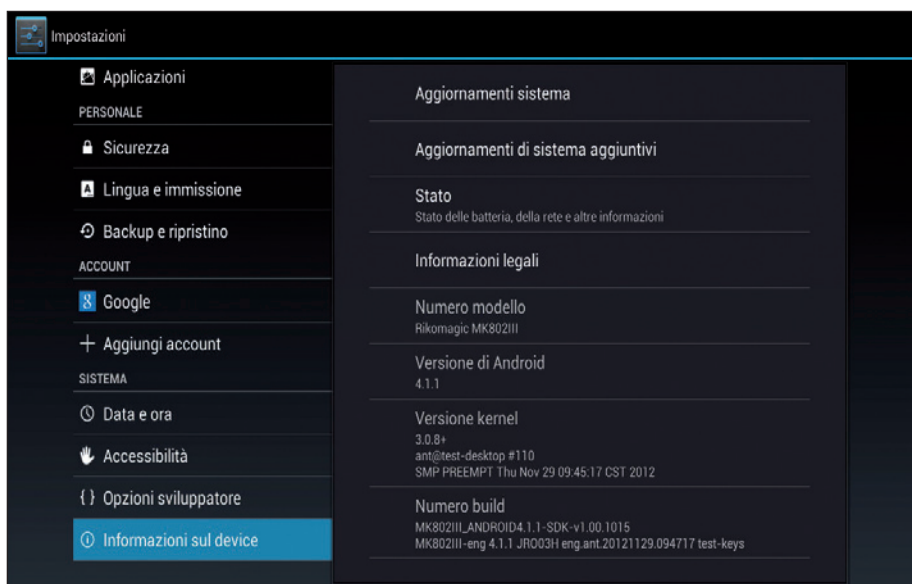
slot per la microSD anche di tipo SDHC e con capacità fino a 32 GB. Con solo una porta Host risulta indispensabile aggiungere un hub, autoalimentato o alimentato esternamente, per collegare le periferiche di memorizzazione e quelle di input. Chi vuole può anche

utilizzare l'hub per collegare un adattatore USB Ethernet grazie al quale si può spegnere l'interfaccia Wifi ed attivare quella Ethernet, riducendo un po' il carico di dissipazione termica interno al dispositivo. Questa soluzione può anche migliorare la comunicazione



L'aggregazione delle varie prestazioni è la chiave di lettura di questo benchmark. Il nostro dispositivo è posizionato nella terza riga dall'alto, con prestazioni migliori di alcuni prodotti di riferimento, come il Galaxy Tab.

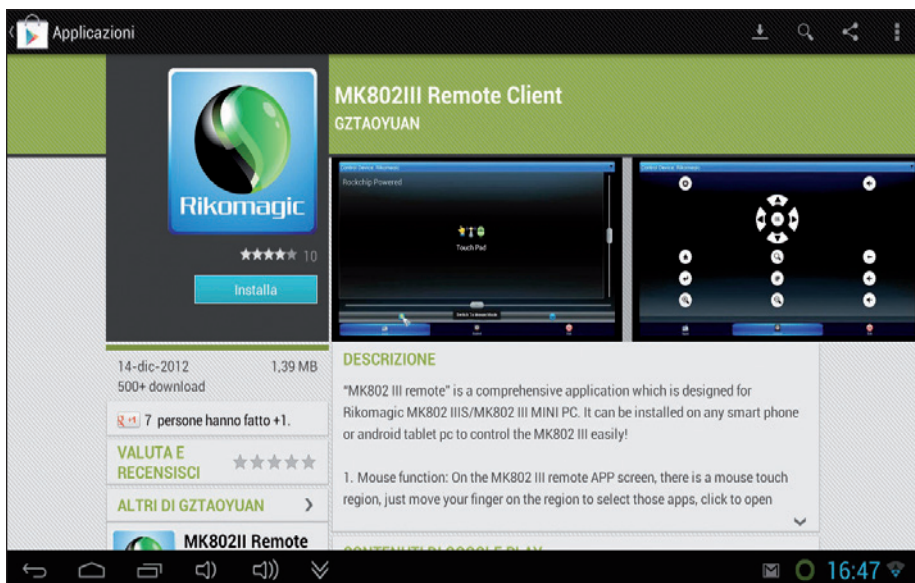




*Ecco la schermata con le informazioni di sistema. Da notare come le prime due voci in alto siano per l'aggiornamento del sistema operativo e delle parti specifiche del processore Rockchip 3066. La versione di Jelly Bean installata è già la 4.1.1.*

in rete nel caso in cui il vostro WiFi sia condiviso fra numerosi dispositivi. Tornando alle caratteristiche di MK802 III, l'elemento di maggiore rilievo è il passaggio da un chip Allwinner a un Rockchip 3066. Chi non segue il mercato dei dispositivi pro-

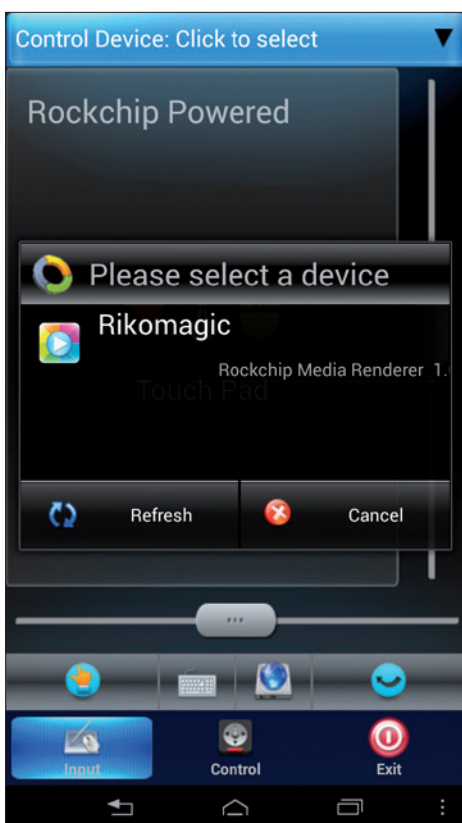
dotti nel Far East non avrà probabilmente mai sentito parlare di Rockchip, ma quest'azienda è da anni attiva nella produzione di soluzioni di tipo SoC (System on Chip) per mercati che vanno dai player multimediali ai sistemi embedded, passando dai TV box e dai tablet.



*Rikomagic offre la possibilità di montare su un altro dispositivo Android un programma che permette di controllare in remoto MK802III, senza ricorrere a tastiere e mouse esterni. Grazie a questo client, è anche possibile accedere a funzioni multitouch.*

In passato abbiamo potuto utilizzare dei prodotti multimediali basati sui chip di questa azienda che, pur non disponendo della parte di produzione vera e propria, ha sviluppato dal 2001 ad oggi numerose soluzioni ad elevate prestazioni e con un ottimo contenuto tecnologico. I prodotti sviluppati da Rockchip hanno vinto per ben cinque anni il premio "China Chip" ed è probabile che diversi sistemi che state utilizzando abbiano un prodotto di quest'azienda all'interno. Il chip RK3066 scelto da Rikomagic per questa nuova versione di mini PC si posiziona molto vicino ai processori Exynos della serie 4 di Samsung, offrendo due core Cortex A9 con frequenza di clock dinamica fino a 1,6 GHz. Abbiamo eseguito dei benchmark e in effetti il posizionamento di questo mini PC è molto vicino a Samsung Nexus e al Galaxy TAB, soprattutto in termini di prestazioni per il Gaming. Al proposito, questo SoC dispone dei quattro core del Mali 400, clockati a 250 MHz, per il supporto di Open GL ES 2.0 e Open VG 1.1.

Come per l'Allwinner, anche il RK3066 dispone di un processore video dedicato, capace di gestire da hardware la riproduzione di filmati in full HD senza perdita di frame. A differenza del modello precedente, però, qui non abbiamo il player con le funzionalità 3D incorporate e probabilmente dovremo attendere qualche aggiornamento del firmware per ritrovare questa funzionalità. La memoria RAM è 1GB di tipo DDR3, per elevate prestazioni, mentre la memoria FLASH su cui è presente il sistema operativo, la partizione inter-



*Ecco la schermata che appare sul dispositivo Android usato come remote client per interagire con un MK802III collegato alla stessa rete WiFi.*

na e quella della SD virtuale ammonta a 8 GB, di cui 504MB come memoria principale, 5,82 GB come SD virtuale e il resto come ROM.

Il sistema supporta l'aggiornamento Over The Air (OTA) tramite il collegamento a Internet e prevede sia il sistema operativo, sia la parte specifica di Rockchip (come visibile nella schermata delle informazioni). Quando vi abbiamo presentato l'MK802II vi abbiamo anche suggerito di acquistare del Google Play l'app DroidMote, utile per gestire da uno smartphone direttamente il dispositivo. Ora non è più necessario in quanto è lo stesso produttore a offrire un'app gratuita e dedicata, capace di collegarsi



*Google Earth a tutto schermo, su un 40 pollici e con i modelli dei palazzi in 3D, fa sempre il suo effetto. Con questa versione di MK802 III è anche migliorata la resa durante il movimento e il sorvolo dei Landmark.*

via WiFi (o rete cablata purché smartphone ed MK802 III siano nella stessa sottorete) per mandare i comandi multitouch, i caratteri digitati su una tastiera remota e una serie di combinazioni di tasti dedicata a funzioni specifiche. Ovviamente l'App non va caricata su MK802 III, ma sullo smartphone o sul tablet che intendete usare per questo controllo remoto.

Il supporto, inoltre, è nativo e diretto, quindi si può usare immediatamente questo client remoto per l'interazione con MK802 III senza ricorrere a nessuna tastiera e mouse fisici. Che le prestazioni di questa terza versione siano superiori rispetto alla precedente è chiaramente visibile anche semplicemente operando sull'interfaccia di base, mentre con le applicazioni, anche impegnative come un gioco 3D o Google Earth, l'elevata risoluzione non crea problemi significativi. In caso di necessità si può comunque abbassare il numero di pixel dello schermo, lasciando al televisore il compito di scalare

l'immagine a schermo intero. Questo riduce il carico di calcoli e rende più fluidi i giochi. Per i video full HD, invece, la riproduzione è impeccabile anche a 1920 x 1080 pixel. Nel complesso, quindi, questa terza versione permette di portare la vostra "smart TV" a nuovi livelli di prestazioni video e 3D, rendendola quasi una console per videogiochi oltre che un sistema di visualizzazione di pagine web anche in Flash.

Aggiungendo un lettore DVD o un disco esterno con interfaccia USB il sistema diventa una vera postazione multimediale "integrata" con il vostro televisore. Il prezzo è quello del modello precedente, con prestazioni superiori, questo ci ricorda che Gordon Moore non è ancora stato smentito, neanche stavolta.

**Il Mini PC dual-core MK802III è disponibile al prezzo di 82,00 Euro (IVA compresa) presso Futura Elettronica srl ([www.futurashop.it](http://www.futurashop.it)); la tastiera wireless Mini8 costa invece 38,00 Euro, sempre IVA compresa.**



Qualcuno l'aveva immaginato, qualcun altro lo aveva sperato e gli sviluppatori lo hanno realizzato: bastano una Kinect e un computer per creare il modello 3D di cose e persone con una qualità sufficiente a duplicarle con stampa 3D.

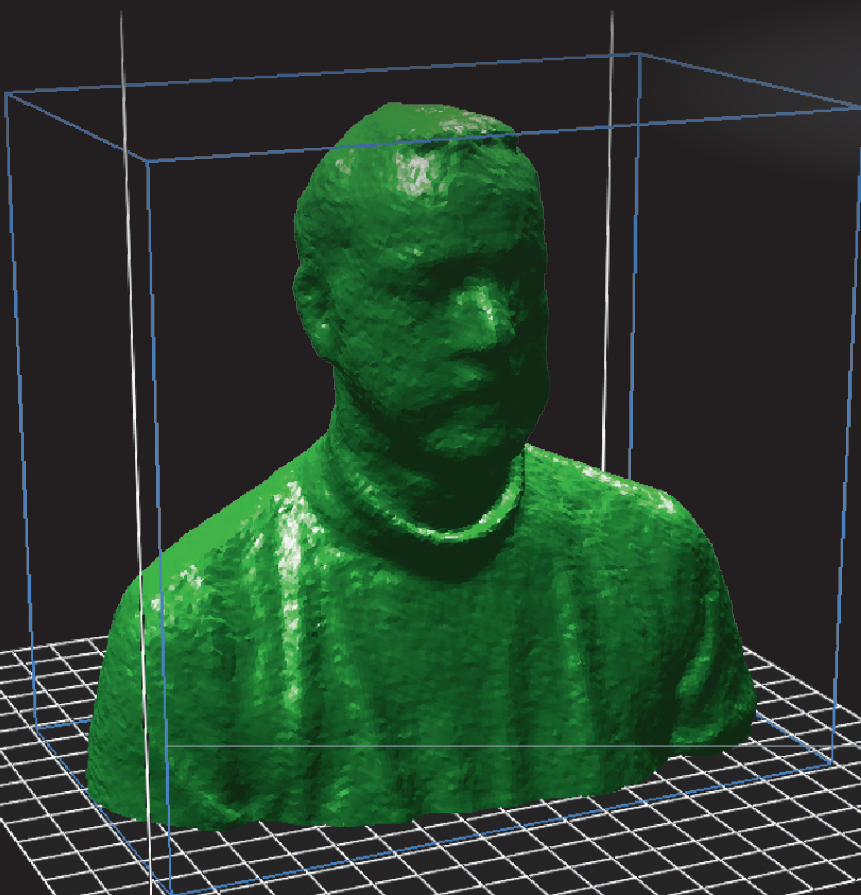
# KINECT PER SCANSIONI 3D

di SIMONE MAJOCCHI



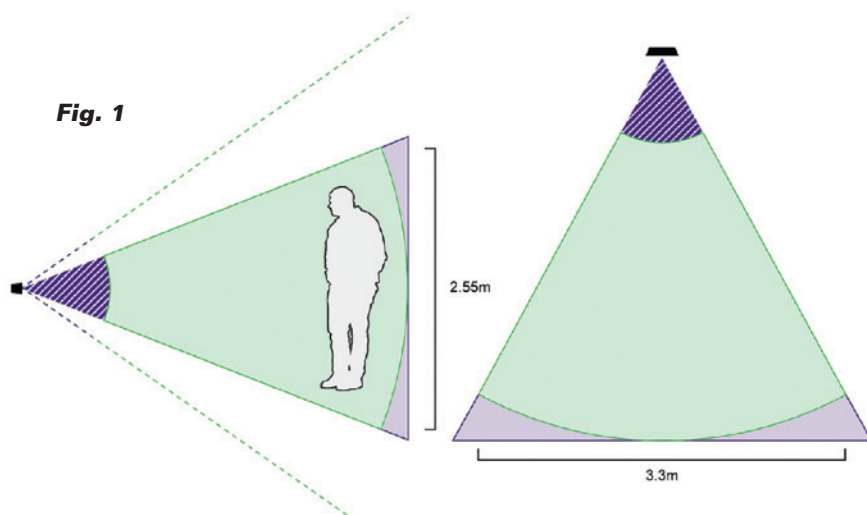
**L**e tecnologie a disposizione per creare modelli 3D di oggetti reali sono numerose, ma non sempre alla portata degli hobbisti. Così come le stampanti 3D professionali esistono da oltre 30 anni, ma solo da un paio sono nati i progetti e le soluzioni desktop sotto i 1000 euro,

anche nel settore della scansione 3D stanno nascendo numerose soluzioni che cercano di portare in un ambito hobbistico le tecnologie attualmente di fascia professionale. Oltre ai sistemi basati su laser, proiettori video, macchine fotografiche e webcam che, con software





**Fig. 1**



più o meno open, permettono di creare una "point cloud" da cui poi generare una superficie 3D, si può ricorrere a un hardware dedicato che è già in grado di fornire una rappresentazione 3D di quanto gli sta di fronte. Ci stiamo riferendo a Kinect, di cui abbiamo iniziato a parlare ormai due numeri fa, proprio in preparazione di questo nuovo articolo. Kinect, da quando è stata "aperta" ad opera di alcuni tenaci sviluppatori, si sta affermando come la soluzione più semplice ed efficace per acquisire modelli tridimensionali di dimensioni né troppo piccole, né troppo grandi. Dal mezzo busto ad un mobile o a una persona intera, Kinect con l'opportuno software può fare dei piccoli miracoli e, soprattutto, non è richiesta la costruzione di alcun hardware di supporto. Non si tratta della soluzione giusta per fare un "duplicato" di qualche piccolo oggetto, a causa della "risoluzione" del dispositivo e del volume all'interno del quale Kinect opera in modo ottimale, ma se il vostro obiettivo è creare e stampare modelli di persone e mobili, avete trovato quello che fa al caso vostro.

#### **I FERRI DEL MESTIERE**

Gli ingredienti di questo metodo di acquisizione sono sostanzialmente tre: il sensore Kinect nelle

versioni per Xbox, per PC o il dispositivo "compatibile" Asus Xtion Pro, un personal computer con scheda grafica accelerata ATI o NVIDIA e infine il software ReconstructMe con gli opportuni driver per il sensore utilizzato. Non si tratta di una soluzione open source, ma di un SDK reso disponibile in forma gratuita e con limitazioni per l'uso non commerciale, mentre per scopi industriali e professionali è prevista una licenza a pagamento. I pacchetti di contorno per l'uso dell'SDK da console o tramite interfaccia sono resi disponibili anch'essi in forma gratuita per uso non commerciale. Un programma alternativo è Kscan3D, ma in questo caso la versione Trial non permette di salvare il modello e quindi non ci permette di arrivare in fondo al processo. Per i nostri esperimenti basati su Kinect e in attesa di qualche altra soluzione più "open", abbiamo comunque quello che ci serve; vediamo quindi come partire da un soggetto reale per trasformarlo in un busto che ci metteremo sulla scrivania come ricordo di questa nuova avventura tecnologica.

#### **LE DIMENSIONI CONTANO**

Quando si sceglie Kinect quale dispositivo di scansione, si fa anche una scelta circa le dimensioni

di quanto potrà essere acquisito dal punto di vista pratico. Lo scopo di Kinect era infatti quello di acquisire l'intera figura per individuarne i movimenti, quindi le ottiche e il firmware che produce i dati di profondità sono stati tarati su un volume opportuno. Abbiamo riportato l'area gestita nella figura 1 per darvi un riferimento.

Come si può quindi intuire, con Kinect potremo fare scansioni di oggetti grandi quanto una sedia, oppure un portaombrelli, ma non riusciremo ad avere la definizione sufficiente a scansionare un accendino recuperandone anche i dettagli. L'obiettivo di Microsoft di permettere un riconoscimento delle espressioni facciali piuttosto che di alcune caratteristiche del volto è stato raggiunto ed è utilizzato nelle applicazioni e nei giochi, ma la definizione di 320 x 240 punti della mappa di profondità va tenuta inevitabilmente in considerazione quando pianificate la scansione.

Per migliorare il risultato, si possono utilizzare varie tecniche di smoothing (ammorbidimento) e campionamento multiplo, sfruttando anche il fatto che il soggetto viene "esplorato" e non rimane immobile. Si tratta di un approccio comunque molto impegnativo sotto il profilo dei calcoli e infatti gli sviluppatori di ReconstructMe hanno pensato di sfruttare i processori grafici a bordo delle schede video per avere un risultato accettabile in termini di frame gestite al secondo. Il software, infatti, deve fare il tracking degli oggetti, aggiungendo alla nuvola di punti le nuove misurazioni correlandole a quelle precedenti. In pratica quando un soggetto ruota, il software riconosce la struttura della superficie già acquisita anche nella nuova mappa di profondità

## Downloads

### Latest ReconstructMe Console



**ReconstructMe\_Installer\_NonCommercial\_405.exe**

25318 downloads

### Kinect for Xbox, ASUS Xtion family sensor drivers



**OpenNI Sensor Driver Package x86**

**1.5.2.zip**

56645 downloads

### Kinect for Windows sensor driver



**KinectRuntime\_v15\_Setup.exe**

9885 downloads

e quindi riesce ad “agganciare” i punti comuni, mentre aggiunge quelli non ancora gestiti.

L’uso delle GPU delle schede grafiche comporta dei limiti di compatibilità e infatti assieme all’SDK viene mantenuta una lista di compatibilità delle varie schede grafiche. L’SDK ha due modalità di funzionamento: normale con un volume di 1x1x1m suddiviso in 256 elementi per ciascuna dimensione, oppure alta risoluzione con il medesimo volume, ma 512 elementi. Il funzionamento delle varie schede non è garantito per entrambe le modalità e infatti nelle nostre prove abbiamo riscontrato alcune difficoltà per la modalità ad alta risoluzione su un pc con una gloriosa Nvidia GT8500, mentre con una Ati Radeon siamo stati più fortunati, anche se il frame-rate di questa seconda scheda era inferiore.

### PREPARIAMO IL SOFTWARE

Come abbiamo parzialmente già visto nella scorsa puntata dedicata a Kinect, per creare il nostro modello 3D dobbiamo aver installato il software ReconstructMe che si compone di un SDK, di un’interfaccia

in modalità console e di una serie di driver che permettono di utilizzare le varie versioni di sensore Kinect (per Xbox, per PC e quello introdotto da Asus con il nome Asus Xtion Pro Live). Si parte dal sito di riferimento all’indirizzo <http://reconstructme.net/projects/reconstructme-console/> dove possiamo scaricare il kit software di ReconstructMe che già include la parte di SDK e una serie di profili predefiniti. Dei file di batch permetteranno di avviare

**Fig. 2**

le varie modalità di acquisizione corrispondenti.

Per non obbligarvi a recuperare gli articoli precedenti, ecco rapidamente l’installazione di quanto serve per gli scopi di questa puntata. Scaricate il file “ReconstructMe\_Installer\_NonCommercial\_405.exe” e il kit di driver per il tipo di sensore che avete a disposizione (nel nostro caso “OpenNI Sensor Driver Package x86 1.5.2.zip”) e procedete con l’installazione prima dei driver e poi del software. Nel pacchetto compresso dei driver trovate le indicazioni per la procedura di installazione del vostro driver specifico, mentre l’installer di ReconstructMe sostanzialmente vi chiede una directory in cui installare il tutto (Fig. 2). Per la parte relativa alla scheda video, dovete obbligatoriamente avere dei driver che supportano OpenCL (CL e non GL...!) e può essere una buona idea fare un salto sul sito del produttore della scheda o del computer per scaricare i driver più recenti. Come indicato nella tabella di compatibilità (Tabella 1) che riportiamo in



**Fig. 3**

| Model                    | Standard | Highres |
|--------------------------|----------|---------|
| ATI FirePro V3800        | 3        | 1       |
| ATI FirePro V4800        | 3        | 1       |
| ATI FirePro V5900        | 3        | 3       |
| ATI Radeon HD 2600 Pro   | 2        | 1       |
| ATI Radeon HD 4570M      | 1        | 1       |
| ATI Radeon HD 4870       | 1        | 1       |
| ATI Radeon HD 5145M      | 1        | 1       |
| ATI Radeon HD 5650M      | 3        | 2       |
| ATI Radeon HD 5700       | 3        | 3       |
| ATI Radeon HD 6490M      | 2        | 1       |
| ATI Radeon HD 6570       | 3        | 2       |
| ATI Radeon HD 6850       | 3        | 3       |
| ATI Radeon HD 7640G      | 2        | 1       |
| ATI Radeon HD 7850       | 3        | 3       |
| INTEL Dual Quadcore      | 2        | 2       |
| INTEL HD Graphics 3000   | 2        | 1       |
| NVIDIA GeForce 9500GT    | 2        | 1       |
| NVIDIA GeForce 9800GT 3D | 3        | 3       |
| NVIDIA GeForce 9800GTX   | 3        | 3       |
| NVIDIA GeForce G 102M    | 2        | 1       |
| NVIDIA GeForce GT 240    | 3        | 2       |
| NVIDIA GeForce GT 330M   | 2        | 1       |
| NVIDIA GeForce GT 430    | 3        | 3       |
| NVIDIA GeForce GT 440    | 3        | 3       |
| NVIDIA GeForce GT 555M   | 3        | 3       |
| NVIDIA GeForce GT 630M   | 3        | 3       |
| NVIDIA GeForce GTX 220   | 2        | 2       |
| NVIDIA GeForce GTX 280   | 3        | 2       |
| NVIDIA GeForce GTX 295   | 3        | 1       |
| NVIDIA GeForce GTX 460   | 3        | 3       |
| NVIDIA GeForce GTX 550Ti | 3        | 3       |
| NVIDIA GeForce GTX 560   | 3        | 3       |
| NVIDIA GeForce GTX 560M  | 3        | 3       |
| NVIDIA GeForce GTX 570   | 3        | 3       |
| NVIDIA GeForce GTX 590   | 3        | 3       |
| NVIDIA GeForce GTX 660M  | 3        | 3       |
| NVIDIA GeForce GTX 670M  | 3        | 3       |
| NVIDIA GeForce GTX 680   | 3        | 3       |
| NVIDIA GT 540M           | 3        | 3       |
| NVIDIA Quadro 2000       | 2        | 2       |
| NVIDIA Quadro 3000M      | 2        | 2       |
| NVIDIA Quadro 4000       | 3        | 2       |
| NVIDIA Quadro FX 2800M   | 3        | 1       |
| NVIDIA Quadro FX 3500M   | 3        | 1       |
| NVIDIA Quadro FX 4800    | 3        | 3       |
| NVIDIA Quadro FX 580     | 3        | 1       |
| NVIDIA Quadro NVS 140M   | 1        | 1       |
| NVIDIA Quadro NVS 295    | 2        | 1       |
| NVIDIA Quadro NVS 3100M  | 2        | 1       |
| NVIDIA Quadro NVS 4200M  | 2        | 1       |

**Chart Legend**  
 Smooth realtime experience  
 Jerky realtime experience  
 Does not work at all

**Tabella 1**

queste pagine, solo alcuni modelli di schede video dispongono di questi driver e la sperimentazione da parte degli utenti è l'unico feedback certo disponibile. Se la vostra scheda NON è in elenco e dispone di OpenCL, provate a usare ReconstructMe nelle due modalità disponibili per scoprire cosa funziona e qual è il frame rate a cui potrete lavorare. Eventualmente provate a cimentarvi nella modifica dei parametri dei file di configurazione seguendo le indicazioni della documentazione disponibile sul sito, magari definendo un diverso volume di scansione e dei parametri diversi. Già curiosando nei file di testo forniti come configurazione scoprirete che ReconstructMe ha molte funzionalità interessanti.

### PREPARIAMO IL SET

Quando si utilizza questo software senza chiave di attivazione commerciale, si hanno alcune limitazioni di tipo applicativo (non ci si può lavorare producendo reddito in modo diretto o indiretto), ma soprattutto c'è un limite di tipo tecnico dovuto alla perdita del tracking a intervalli regolari. Praticamente si ha un tempo limitato per cercare di arrivare alla fine della scansione prima che il software inizi a fare dei capricci e ci renda la vita decisamente complicata. Per non perdere tempo, va quindi preparato un "set" dove realizzeremo le nostre acquisizioni. Elemento indispensabile se volete realizzare il vostro "busto" è la presenza di una seconda persona che possa manovrare computer e Kinect. Anche un cavalletto di quelli da almeno 1,5 metri è molto utile per evitare che la Kinect vada tenuta in mano (Fig. 3). L'illuminazione è un elemento tutto sommato trascurabile in quanto non dobbiamo acquisire

anche l'immagine a colori per ricostruire un nostro avatar, ma ci basta la parte legata al 3D che viene creata con un'illuminazione a infrarossi generata direttamente dalla Kinect. Un ultimo elemento che a nostro avviso rende tutto più facile è uno sgabello rotante o una sedia, sempre rotante, a cui è stato temporaneamente rimosso lo schienale. Il soggetto potrà così ruotare su se stesso all'interno del volume gestito in acquisizione.

### VIA CON LE PROVE

Con questa soluzione, è indispensabile prendere confidenza con il sistema di scansione, con il volume gestito, con la velocità di rotazione del soggetto e con i comandi da tastiera che ReconstructMe in modalità consolle permette di usare.

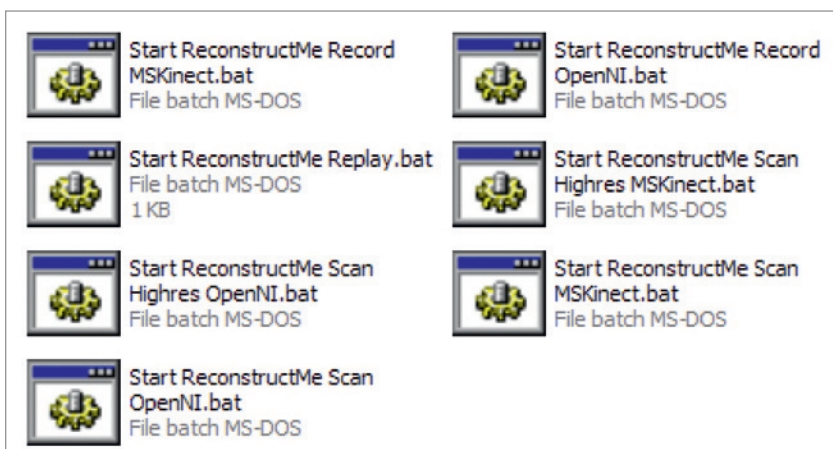
Come indicato prima, è abbastanza improbabile riuscire a fare da soli e quindi dovete avere l'aiuto di qualcuno che sia in grado di apprendere assieme a voi gli accorgimenti e le tecniche indispensabili per raggiungere un buon risultato. Noi stessi abbiamo provato per un po' di tempo prima di trovare la soluzione più funzionale e potrebbe darsi che per la struttura della stanza dove operate ci possano essere dei problemi che vanno risolti in modo diverso rispetto al setup che vi abbiamo proposto.

Innanzitutto, fate in modo che Kinect sia saldamente fissata al cavalletto e che questo possa essere facilmente alzato e abbassato per consentire la scansione del soggetto da due angolazioni verticali diverse.

Lo sgabello o la sedia dovrebbero avere alle spalle una situazione il più neutra e lineare possibile (un bel muro anche un po' distante è ideale).

Infine il soggetto non dovrebbe





**Fig. 4** - I file disponibili nella cartella.

indossare capi con una superficie poco definita come un maglione di lana molto morbido o un colletto di pelo. Abbiamo anche notato che i dettagli sottili come una montatura di occhiali non vengono rilevati, mentre degli occhiali da sole con una montatura decisa e lenti scure potrebbe essere visibile. In generale, ricordate che la profondità viene gestita proiettando una matrice di punti molto complessa e fitta, con luce infrarossa. Quello che non riflette in modo nitido la matrice crea conseguentemente dei problemi di acquisizione.

Avviando uno dei file di batch presenti nella cartella di ReconstructMe potete rapidamente verificare come si comporta il sensore con quanto avete addosso e allo stesso tempo potete prendere confidenza con l'area attiva di acquisizione (**Fig. 4**). I tipi di batch disponibili sono tre – Record, Scan e Scan Hires – e sono ripetuti per ciascuna delle due piattaforme hardware gestite, ovvero OpenNI per Xbox Kinect e Asus, oppure MSKinect per Kinect PC.

Come indicato prima, potreste avere problemi ad acquisire con l'alta risoluzione, quindi provate sia il .bat normale, sia quello Hires per capire dove vi conviene concentrare gli sforzi.

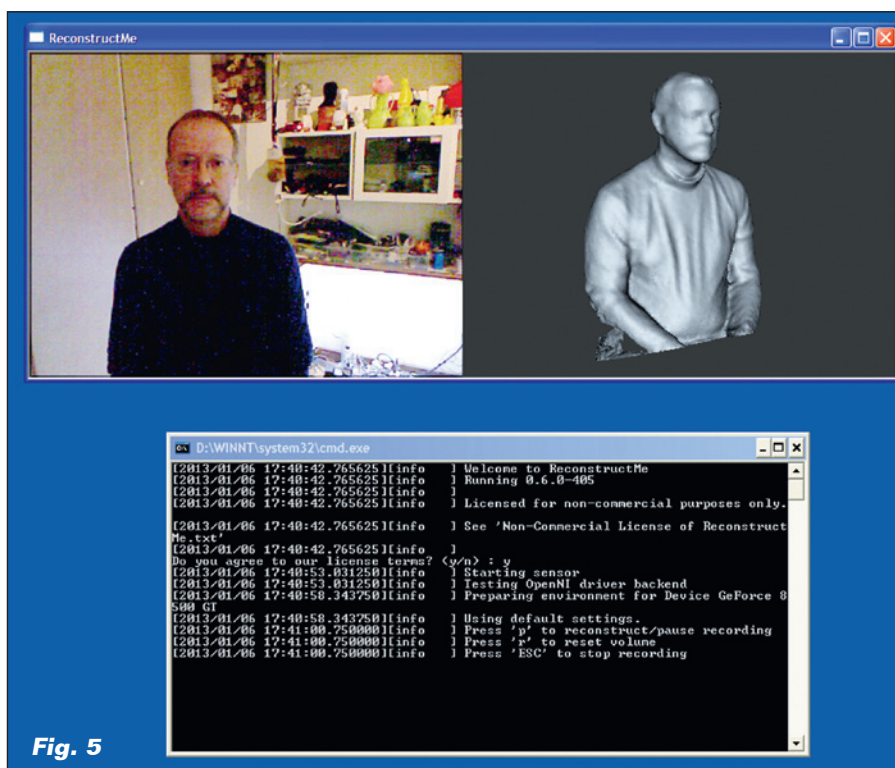
Se usate uno dei batch per lo scan, ricordate che avete a disposizione tre comandi da tastiera

– riportati anche nella finestra di CMD che appare quando lanciate il batch - : “p” fa partire o ferma l'acquisizione, “r” cancella la nuvola di punti e di fatto riavvia l'elaborazione, “esc” finisce la fase di acquisizione e permette di salvare il file in formato PLY (un formato comune per lo scambio di dati di mesh e modelli 3D). Per le prove, è quindi indispensabile premere almeno una volta “p” dopo aver accettato le condizioni d'uso dando “y” e invio.

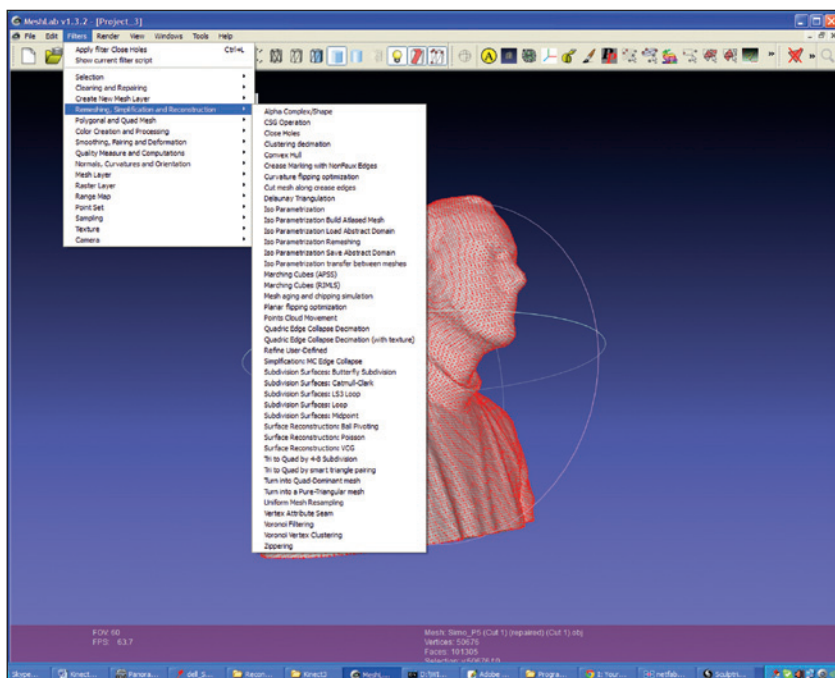
L'aspetto a nostro avviso più importante di questa fase è tro-

vare la posizione in cui il sensore lavora al meglio e riesce a produrre un ragionevole dettaglio di quanto inquadrato. Fate pratica provando a ruotare il soggetto ripreso, interrompendo con il tasto “p” e riprendendo anche da un'altra angolazione verticale. Solo con un po' di pratica riuscirete a fare l'acquisizione del modello in più riprese, spostando l'altezza e il puntamento del sensore all'interno di una medesima scansione.

Ricordate, durante la fase di sperimentazione, che quando sentite un “bip” dall'altoparlante interno al PC, avete perso il tracking del soggetto. Questo non è sempre causato dal movimento troppo veloce, ma fa parte delle “limitazioni” della versione non commerciale: a intervalli regolari il software sgancia il tracking intenzionalmente per renderne l'uso “commerciale” decisamente scomodo. Imparare il ritmo con cui il tracking viene sganciato



**Fig. 5**



**Fig. 6**

salvate. Ripetete questa operazione per produrre due o tre versioni del soggetto prima di passare alle fasi successive del processo di creazione del modello stampabile.

Prima lavorazione: Meshlab

Chi ci ha seguito nei vari articoli in cui abbiamo trattato il tema dei modelli 3D e della loro stampa, ricorderà che c'è un set minimo di ferri del mestiere che riteniamo indispensabili. Fra questi c'è Meshlab.

È disponibile gratuitamente all'indirizzo <http://meshlab.sourceforge.net/> per varie piattaforme ed è una specie di coltellino svizzero in open source per la lavorazione di numerosi formati 3D.

Per questa nostra specifica applicazione, di Meshlab ci servono due filtri: quello che "chiude" i buchi e quello che ammorbidisce e smussa il modello come se fosse passato con della carta vetrata. Nella schermata vedete la posizione della funzione di chiusura dei buchi: è la terza voce dall'alto nell'elenco (Fig. 6).

In alternativa potete usare la fun-

permette di capire quando effettivamente c'è un problema nel movimento attorno al soggetto (o nella rotazione del soggetto di fronte al sensore fisso). Quando usate la pausa ("p") potete intervenire con il mouse nel lato destro della finestra, ruotando e ispezionando il modello acquisito fino a quel momento (Fig. 5). Ovviamente, se il soggetto cambia forma o posizione, non sarà più possibile per il software trovare il tracking e il modello risulterà deformato o inutilizzabile. Questo è uno dei motivi per cui c'è bisogno dell'aiuto di una persona oltre al soggetto da riprendere e anche il soggetto dovrebbe essere in grado di restare immobile per tutta la durata dell'acquisizione.

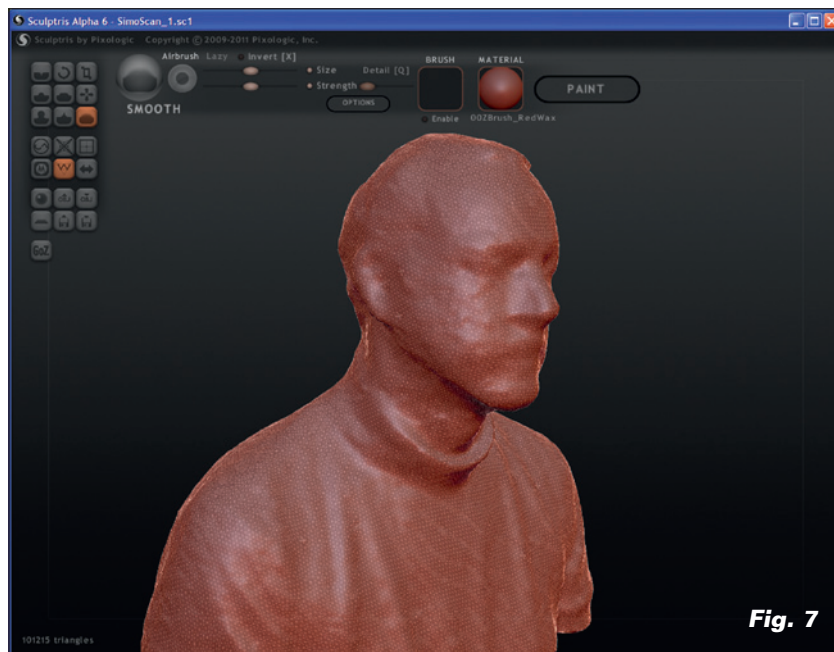
## CIACK SI GIRA

Diamo per scontato che siate riusciti a prendere la dovuta confidenza con i vari aspetti dell'acquisizione e che il vostro soggetto sia pronto sullo sgabello rotante, con il sensore posizionato o in alto o in basso e pronto all'acquisizione.

Fate prendere un bel respiro al soggetto, date il via all'acqui-

sizione con "p" e dopo un primo momento di stabilizzazione date un "r" per iniziare con l'acquisizione vera e propria. Il soggetto deve ruotare con lo sgabello alla velocità che precedentemente si è rivelata quella ottimale; alla fine del primo giro spostate il sensore nella seconda posizione verticale e ripetete il giro, così da non avere buchi né nella parte superiore della testa, né sotto il mento.

Premete "esc" per interrompere e



**Fig. 7**

**Fig. 8**

zione di chiusura dei buchi che si attiva cliccando sulla terza icona a destra della A in un cerchio giallo (è una H in una nuvola di punti). Se cercate fra le opzioni, potete usare ancora altri metodi per gestire il problema dei buchi, mentre per il taglio e la “messa in forma” del busto utilizzeremo un altro strumento.

Quando siete soddisfatti sia del taglio, sia dello smoothing, potete passare alla voce File -> Export Mesh As e creare un file di tipo OBJ del vostro modello.

### UN RITOCCHINO CON SCULPTRIS

Il modello catturato da Kinect è sicuramente molto somigliante, ma anche poco dettagliato e quindi – solo se avete un minimo di pratica con il programma – potete caricare il file 3D in Sculptris e utilizzare le sue notevoli caratteristiche per aggiungere un po’ di dettagli mancanti al modello. Sculptris non potrà “chiudere i buchi” se ne sono rimasti, ma vi permetterà di fare una serie di interventi interessanti anche aggiungendo pezzi al modello, se necessario.

Non essendo particolarmente abili con Sculptris, ci siamo solo assicurati della correttezza del processo con l’importazione del file OBJ e la sua successiva esportazione una volta terminate le modifiche, mentre ci siamo astenuti dall’intervenire (Fig. 7).

### E ADESSO NETFABB

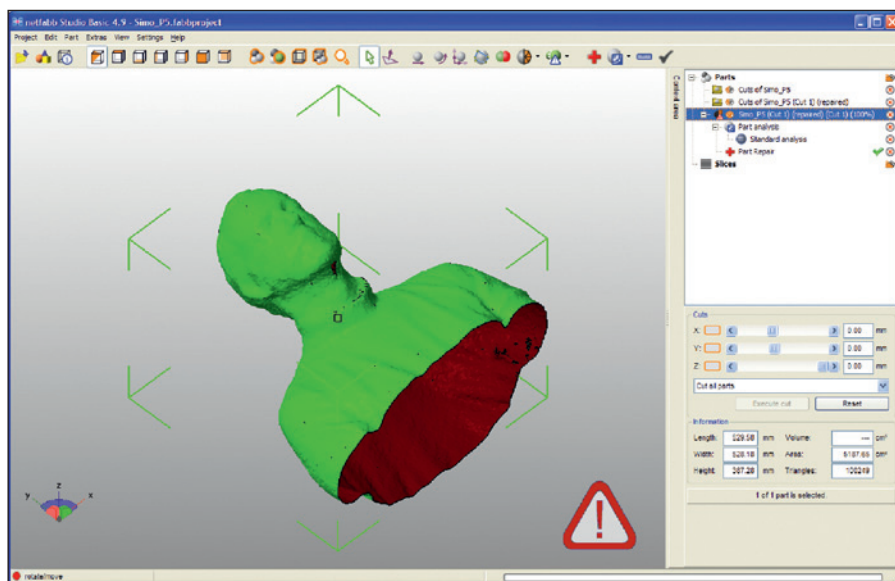
Ci stiamo avvicinando alla fine del processo di preparazione del nostro file, ancora in formato OBJ, ma già riparato e reso più liscio grazie ai passaggi in Meshlab e Sculptris.

Ora con Netfabb possiamo sistemare per bene la base di appoggio ed eventualmente correggere l’orientamento.

Abbiamo notato che durante la scansione, è molto facile che il soggetto non sia propriamente con la schiena dritta e lo sguardo “fiero”, mentre è più probabile che sia un po’ incassato e con la schiena curva, proprio perché per restare immobile ha scelto una posizione comoda.

Con Netfabb possiamo correggere la posa ruotando di -20 gradi il modello sull’asse X (questi valori nascono dai nostri esperimenti ed è molto probabile che siano adatti anche alle vostre acquisizioni); agite sul valore di rotazione per avere una posizione più o meno corretta, ricordando che se il mento non risulta orizzontale, ma è leggermente piegato verso l’alto, ci sarà possibile stampare probabilmente senza supporti, mentre se è dritto, dovremo attivare i supporti. Il secondo intervento da fare con Netfabb è il taglio della base con un piano allineato sull’asse Y. Questo vale solo se non avete ancora fatto rotazioni al modello che, per come viene creato da Kinect e ReconstructMe, si trova di fatto sdraiato sul piano di stampa.

La base di appoggio deve preferibilmente essere piatta ed estesa, così da garantire una buona presa sul piatto di stampa (Fig. 8).



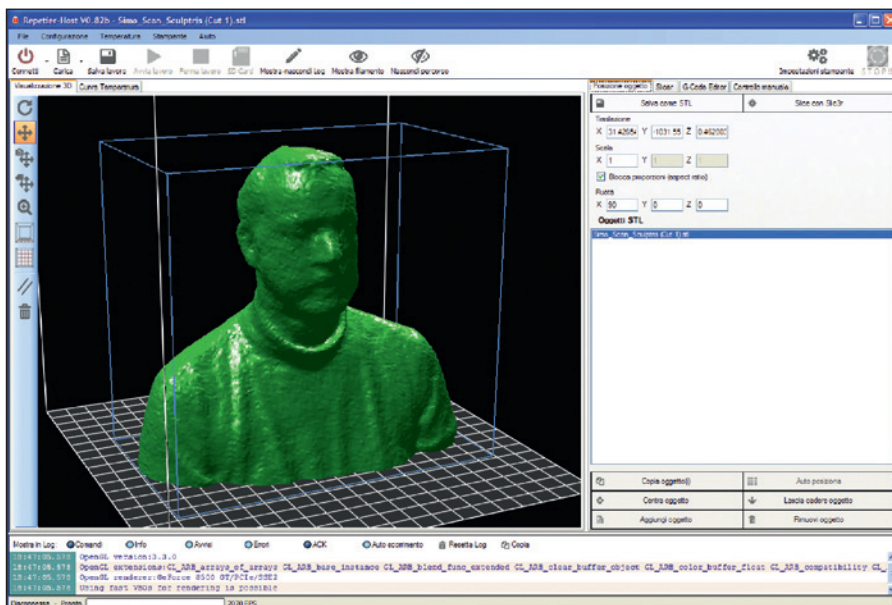
Sempre con Netfabb si può dare un’ulteriore “sistemata” al modello grazie alla funzione di riparazione. Nell’immagine precedente si nota il punto esclamativo che segnala problemi potenziali per la stampa: in questo caso il software propone automaticamente la riparazione in fase di esportazione. Solo dopo aver effettuato lo slicing si può però sapere se è tutto a posto o meno. L’esportazione, inoltre, va fatta con il formato di file STL, abbandonando l’OBJ usato finora.

### LA STAMPA SI AVVICINA

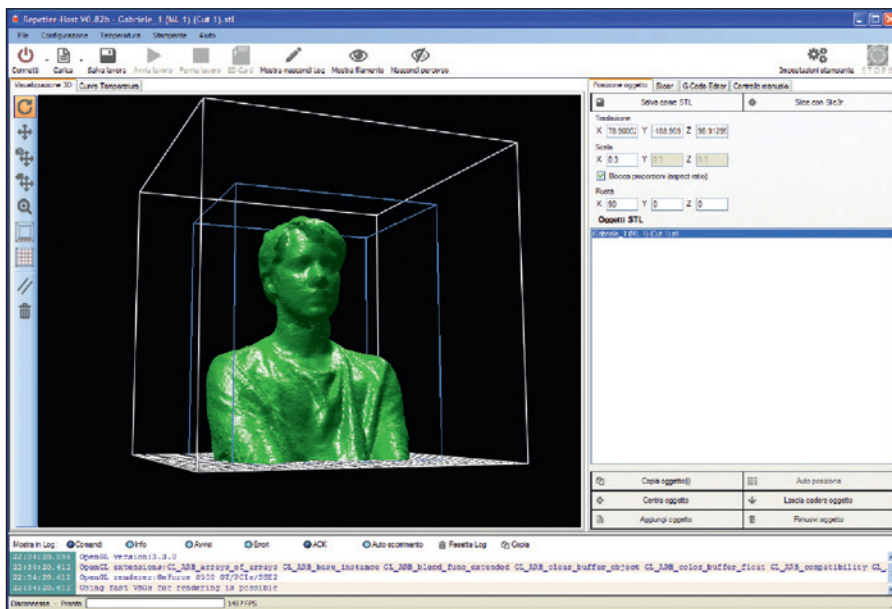
Con il file finalmente ripulito, tagliato e in formato STL, possiamo passare all’ultima fase che utilizza Repetier Host come applicazione di slicing e stampa. Il caricamento vi genererà sicuramente un errore in quanto la nostra stampante gestisce un volume di stampa da 20 x 20 x 20 cm, mentre se avete acquisito un busto di persona, questo sarà da almeno 60 x 60 x 40. Repetier Host non impedisce il caricamento, ma vi mette il modello in un colore di “allarme” e lo fa pulsare. Usate la funzione di ridimensionamento e impostate la scala a “0.2” per cominciare. Usate i pulsanti “Centra oggetto” e “Lascia cadere oggetto” per metterlo



**Fig. 9**



**Fig. 10**



in centro al piatto di stampa. La rotazione per mettere “in piedi” il busto è di 90 gradi sull’asse X. Ripetete la centratura e l’allineamento. Con il busto posizionato correttamente potete lavorare sulla percentuale di riduzione per ottenere il risultato migliore. A seconda della fisionomia del soggetto, potete infatti avere dimensioni e proporzioni diverse (Fig. 9 e Fig. 10). Ispezionate il modello da stampare utilizzando la rotazione

della telecamera attorno al piano (la prima icona in alto a sinistra con la freccia circolare) facendo soprattutto caso alle inclinazioni dietro alla nuca e sotto al mento per capire se ci sono potenziali problemi sulla stampa. Il passaggio successivo, quello dello slicing, potrebbe richiedere anche una quindicina di minuti dato che il modello avrà sicuramente un elevato numero di triangoli. Per le impostazioni, vi suggeriamo di limitare i perime-

tri a 2 o 3 e di tenere il fill a un valore contenuto, tipo 15 o 20% per non perdere troppo tempo nel riempimento del busto. Al termine dello slicing, vi suggeriamo ancora di ispezionare il risultato finale e verificare che nelle zone critiche di nuca e mento non ci siano dei perimetri che si appoggiano al nulla. Se ne trovate potete diminuire lo spessore degli strati, aumentando però il tempo di stampa, oppure tornare a Netfabb per modificare l’inclinazione del busto e la conseguente angolatura del mento o della nuca.

## E ADESSO CI VUOLE PAZIENZA

Con lo slicing pronto – di cui vi suggeriamo il salvataggio per poter eventualmente lanciare e rilanciare la stampa in un secondo momento – date una rinfrescata al piano di stampa o con un po’ di vinavil diluito o con un po’ di carta vetrata fine seguendo la prassi che avete trovato più adatta alla tenuta delle stampe sul piano. Il busto che state per stampare è grande come superficie di appoggio e tenderà a distaccarsi ai bordi se non c’è una buona adesione, anche se la parte che è a contatto dovrebbe avere pochi strati pieni e poi subito gli strati con il fill a bassa densità, quindi con una conseguente bassa tensione dovuta alla contrazione da raffreddamento.

Se nelle vostre esperienze precedenti, il distacco è per voi un problema reale, utilizzate il BRIM abbondante con il primo strato al massimo da 0,3 mm di spessore. Se la vostra acquisizione è simile alla nostra, preparatevi a circa 5 ore di tempo di stampa impostando a 90 mm/sec la velocità di estrusione. Potete usare una velocità più elevata se siete in grado di intervenire negli ultimi strati di stampa e disponete di

una ventola di raffreddamento: sicuramente la parte superiore della testa risulterà critica da stampare in velocità senza deformazioni perché è fatta di perimetri e strati molto piccoli. Solo rallentando e raffreddando si evita che ogni strato collassi sul successivo non ancora sufficientemente freddo e stabile. Anche a 90 mm/sec, comunque, un occhio di riguardo per gli ultimi 50 strati può evitare di avere un risultato con la testa schiacciata (Fig. 11).

#### COSA ACQUISIRE

Ora che abbiamo portato a termine con successo l'intera procedura, dall'acquisizione alla pulizia fino alla stampa, possiamo ripartire puntando alla specializzazione: qualcuno usa questo sistema per i volti, qualcuno per le figure intere. Quello che va ricordato è la risoluzione disponibile. Per nostra sfortuna, non avevamo una scheda video adatta al funzionamento in alta risoluzione – che fornisce otto volte il numero di punti per unità spaziale rispetto a quella da noi usata – per valutare la qualità di acquisizione massima. Nella modalità standard, il risultato ricorda i rapinatori con le calze di nylon in faccia, ma le persone sono comunque riconoscibili. Se puntate alla realizzazione di volti, sarà necessario rendere l'alta risoluzione funzionante ricorrendo a una delle schede pienamente supportate. Se riuscite a realizzare qualche acquisizione particolarmente ben riuscita, fatecelo sapere; saremo felici di mostrare sul blog i vostri risultati e, se ce lo permettete, potremmo aggiungere il vostro file fra quelli disponibili per la stampa. ■



Fig. 11



# Fiera

dell'informatica elettronica e radiantismo

# Rovigo

**Cen.Ser. - Viale Porta Adige 45**

## 23 e 24 febbraio 2013

apertura dalle ore 9,00 alle ore 19,00

**RIDUZIONE DI 2 EURO**  
 presenta questo coupon alle casse  
 per ottenere il biglietto a prezzo ridotto

per tutta la durata della manifestazione saranno ospitate le iniziative

**TRASHWARE**

riutilizzo di materiale informatico  
 per scopi di utilità sociale, a cura di  
 Faber Libertatis

**MERCANTE IN FIERA**

mostra mercato dell'hobbista,  
 il luogo migliore per scambi tra  
 appassionati di radiantismo e  
 cine-foto-ottica

info: 0425 27401 - [www.arearebus.com](http://www.arearebus.com) - [fiera@arearebus.com](mailto:fiera@arearebus.com)





# SHIELD DI ESPANSIONE ADC PER RASPBERRYPI COMPATIBILE CON ARDUINO

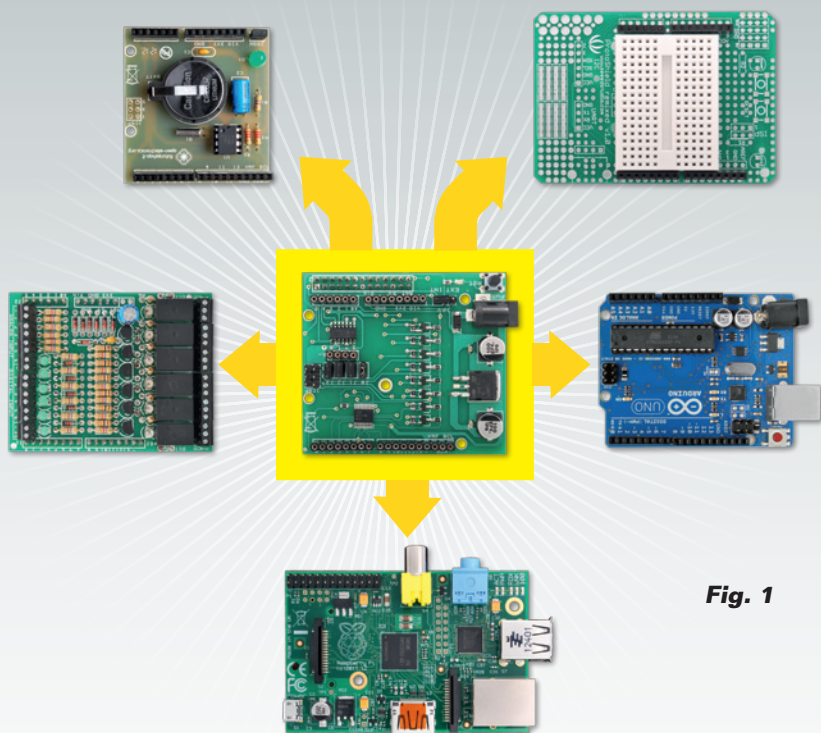


di MARCO MAGAGNIN

Esandiamo le funzionalità di RaspberryPi con uno shield compatibile con la piedinatura di Arduino e con a bordo un convertitore analogico digitale a 16 bit di risoluzione. Ulteriori espansioni che presenteremo in futuro permetteranno di aggiungere un convertitore DAC, una serie di I/O digitali aggiuntivi e la possibilità di collegarsi anche a schede e PC sprovvisti di GPIO.

**C**ontinuiamo la nostra serie di appuntamenti legati al mondo RaspberryPi ed alle sue applicazioni tornando ad occuparci più approfonditamente delle funzionalità della periferica GPIO. In questo e nei prossimi articoli vi proponiamo una serie di dispositivi che permettono diverse possibilità di integrare periferiche e dispositivi hardware sia collegandoli al GPIO della scheda RaspberryPi, sia ad altri microPC non dotati di GPIO ed anche ai comuni PC portatili e desktop.

Nella Fig. 1 potete vedere un'immagine d'insieme delle integrazioni possibili con la scheda che vi presentiamo in questo articolo, uno shield di espansione ADC che permette di aggiungere a RaspberryPi la possibilità di eseguire letture da sensori con uscita analogica. Come vedremo nel prosieguo dell'articolo, questo shield può essere alimentato sia a 12V che a 5V ed esegue



**Fig. 1**

la conversione dei livelli di tensione da 3,3V a 5V per i pin digitali del GPIO e per le linee di comunicazione dei bus i2C, SPI e seriale. Inoltre questo shield possiede sia un connettore adatto ad inserirsi nel connettore GPIO di RaspberryPi sia le strip che corrispondono ai connettori tipici dei microcontrollori della famiglia Arduino e dei relativi shield di espansione.

Con questo shield potremo utilizzare con RaspberryPi molti degli shield già disponibili per Arduino, ed anche per mettere in comunicazione la scheda RaspberryPi con lo stesso microcontrollore Arduino. Nel prossimo futuro vi proporremo un ulteriore shield con a bordo un integrato DAC ed un integrato di espansione I/O con sedici pin, entrambi pilotati via i2C.

Per permettere anche ai dispositivi Linux non dotati di GPIO come i microPC Mk802, Mk802 III ed anche normali PC, di utilizzare periferiche di ingresso/uscita analogiche e digitali

vi proporremo dei convertitori USB/i2C e USB/SPI che potranno essere utilizzati con questo shield.

Chiaramente completeremo questa panoramica con esempi software per utilizzare questa architettura nei vari linguaggi e con i diversi dispositivi Linux.

Prima di iniziare la descrizione dello schema elettrico dello shield è però necessario

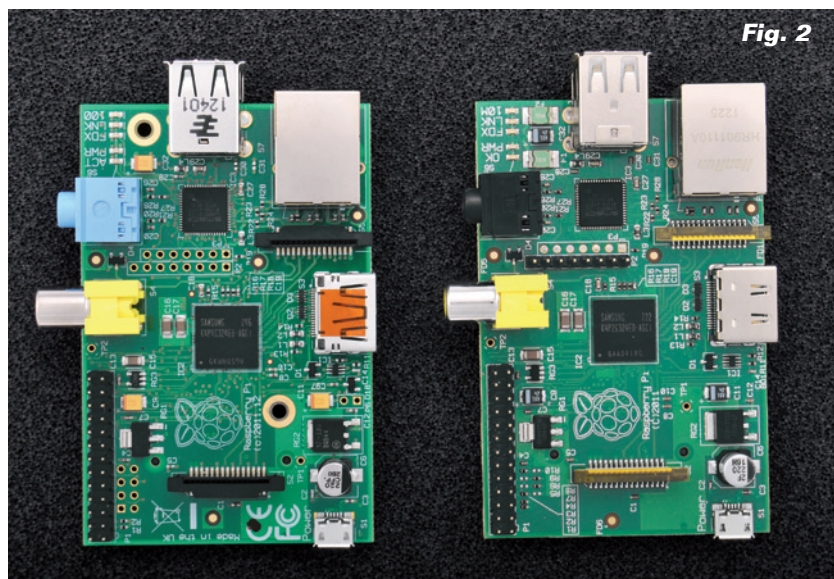
tenere conto ancora una volta delle modifiche e degli aggiornamenti che si susseguono in un mondo in evoluzione accelerata come quello legato al sistema operativo Linux e che non risparmia la nostra RaspberryPi.

Quella attualmente in vendita ed in distribuzione è la rev. 2 della versione B di RaspberryPi, caratterizzata principalmente, rispetto alla rev. 1, dall'aumento della memoria RAM da 256 a 512 Mb e da alcune variazioni ai riferimenti di I/O del connettore GPIO.

Per verificare il modello di RaspberryPi in vostro possesso potete fare riferimento alla **Fig. 2**; a destra la scheda rev. 1, a sinistra la scheda rev. 2, riconoscibile, tra l'altro, dai due evidenti fori di fissaggio.

Nella **Fig. 3** sono visibili le piedinature del connettore GPIO nelle due versioni della scheda; le principali differenze sono i riferimenti agli I/O digitali per i pin 3, 5, 7, 13 e per i pin del bus i2C che nella rev. 1 riportano le linee del bus(0) e nella rev. 2 quelle del bus(1).

Vediamo ora la descrizione del



**Fig. 2**



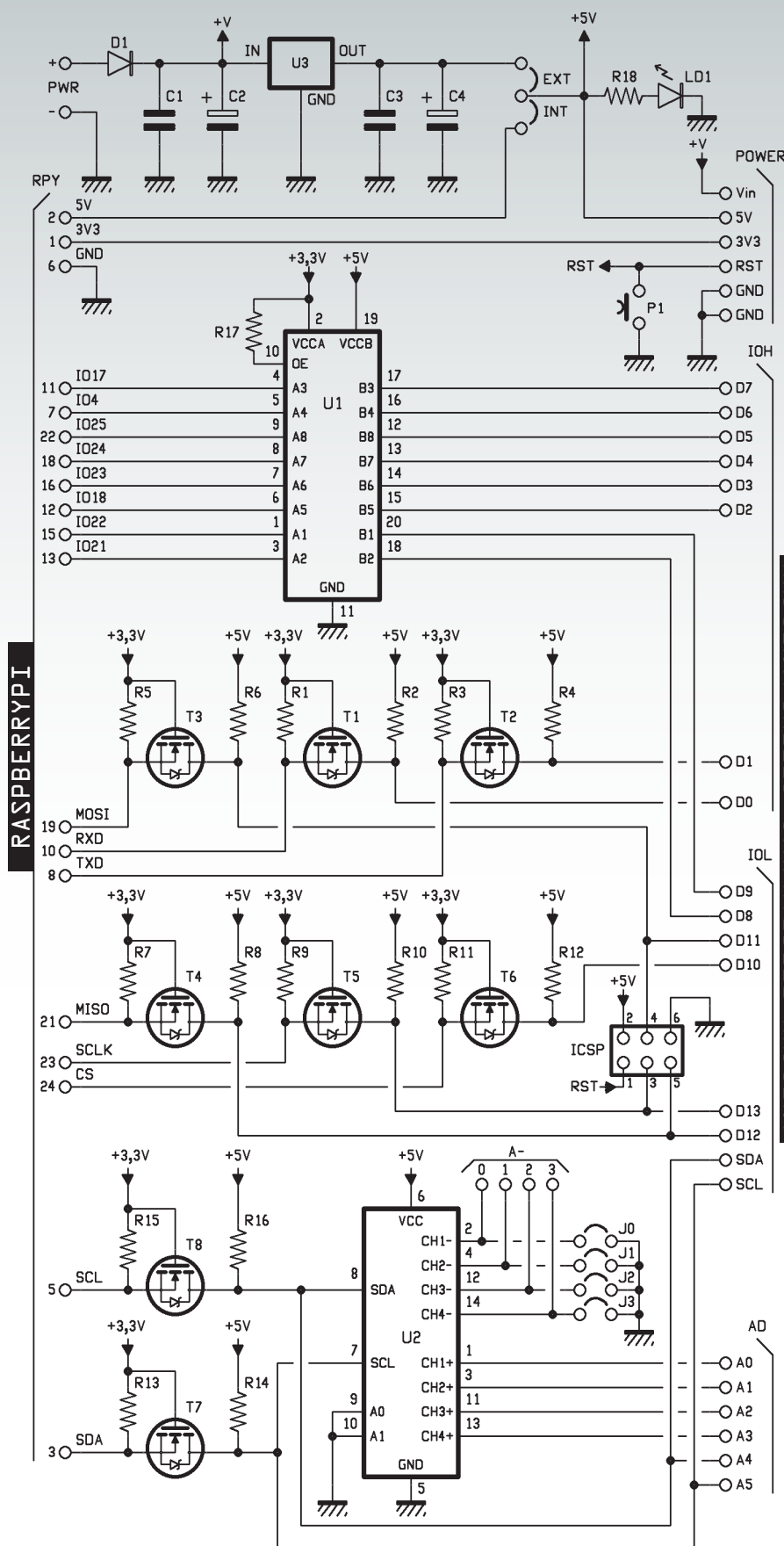
circuito elettrico dello shield e poi gli aspetti software per l'utilizzo dello stesso.

### SCHEMA ELETTRICO

Lo shield di integrazione del GPIO è stato progettato per offrire le seguenti possibilità:

- Estendere le funzionalità del GPIO con un convertitore ADC a quattro canali singoli o differenziali, direttamente sulla basetta e, con un ulteriore shield, un'espansione a ulteriori 16 I/O digitali ed un convertitore DAC a 4096 passi, entrambi su bus i2C;
- Alimentazione diretta dello shield prelevando i 5V da RaspberryPi oppure con alimentatore esterno sia a 5V sia a 12V per compatibilità con gli shield di Arduino;
- Conversione dei livelli da 3,3V a 5V, e viceversa, degli I/O digitali, dei bus i2C, SPI e della porta seriale, per rispettare sia i requisiti elettrici di RaspberryPi che dei sensori e dispositivi esterni;
- Possibilità di utilizzare il convertitore ADC sia con gli ingressi in modalità differenziale che lineare;
- Connettore a 26 pin per il collegamento con il connettore GPIO di RaspberryPi;
- Connettori per il collegamento diretto ad una scheda Arduino e agli shield previsti per quest'ultima;
- Possibilità di collegarsi a convertitori aggiuntivi esterni USB/i2C e USB/SPI.

Analizziamo lo schema elettrico per verificare come i requisiti posti sono stati realizzati in pratica. Iniziamo dalla alimentazione costituita dalla classica configurazione centrata sul regolatore 7805. I condensatori filtrano e stabilizzano la tensio-







**Fig. 3**

ne in uscita. I ponticelli EXT e INT permettono di scegliere se ricevere l'alimentazione a 5V dal pin 2 del connettore GPIO oppure dall'alimentazione esterna tramite il regolatore 7805. In questo modo è possibile evitare un'alimentazione esterna, quando non sia necessario. Il pin 2 di RaspberryPi è in grado di erogare 500 mA nella rev. 1 e 300 mA nella rev. 2. Per shield e sensori con assorbimenti maggiori o per shield che richiedono tensioni maggiori come lo shield relè, è necessario utilizzare un alimentatore esterno. Veniamo ora ai circuiti di conversione dei livelli, che sono realizzati con due approcci diversi. La conversione dei livelli dei pin di I/O digitale è realizzata utilizzando l'integrato TBX0108 un traslatore di livelli ad 8 bit con due linee di alimentazione, del quale trovate la descrizione nel riquadro dedicato. Sul lato A dell'integrato sono collegate le linee provenienti dal connettore GPIO di RaspberryPi indicato nello schema con RPY, sul lato B le linee collegate ai connettori IOL, IOH, POWER e AD, che mappano i rispettivi connettori I/O di Arduino. Il terminale GND è collegato a massa, i terminali  $V_{CCA}$  e  $V_{CCB}$  sono i riferimenti di tensione per la conversione dei livelli e sono collegati alle linee di tensione, rispettivamente a 3,3V e 5V, il terminale OE (Enable), quando è posto a livello alto, abilita l'operatività dell'integrato ed è collegato al livello  $V_{CCA}$  (tensione 3,3V). Nella tabella di Fig. 4 sono riportate le corrispondenze tra i pin del GPIO lato RaspberryPi ed i corrispondenti pin sui connettori Arduino; le due sezioni della tabella riportano le corrispondenze sia per la

configurazione della rev. 1 sia della rev. 2. Nell'integrato TBX0108 la conversione dei livelli funziona in entrambe le direzioni in modo automatico, in quanto l'integrato stesso è in grado di determinare la direzione del flusso dati per ciascun pin di I/O (autosensing). Questa caratteristica, utile per la conversione dei livelli dei pin di I/O digitale rende poco affidabile l'impiego dello stesso integrato per la conversione dei livelli delle linee del bus i2C, in quanto dalla parte di RaspberryPi le linee del bus i2C sono dotate di resistenze di pull up, che interferiscono con la funzione di autosensing dell'integrato TBX0108 (vedi riquadro dedicato). Per le linee del bus i2C, del bus SPI e della porta seriale è stata adottata una soluzione basata sui MOS-FET BSS138, transistor ad effetto di campo a canale N enhancement mode con una soglia VGS di 1,3V.

Il circuito di conversione di livello è identico per ciascuna linea di segnale. Come esempio ci riferiamo alla linea SDA del bus i2C. Il gate (g) del MOS-FET BSS138 (T7) è collegato alla linea di alimentazione a 3,3V, il source (s) alla linea di segnale a livello inferiore (3,3V) ed il drain (d) alla linea di segnale a livello più alto (5V). Tra il source ed i 3,3V è collegata la resistenza di pull up R13 mentre tra il drain ed i 5V è collegata la resistenza di pull up R14.

Vediamone il funzionamento nelle diverse condizioni:

1. Nessuna comunicazione in corso. - Nessun dispositivo porta le linee di segnale a livello basso e di conseguenza la linea di segnale a 3,3V è mantenuta a livello alto dalla resistenza R13 di pull-up. Il gate ed il source del MOS-FET sono entrambi a 3,3V, di conseguenza la VGS (gate-to-source voltage) è al di sotto della soglia di innesco ed il MOS-FET non conduce. In questa condizione la linea di segnale a tensione maggiore (5V) è mantenuta a livello

**Fig. 4**

| Arduino | RPI rev. 1 | RPI rev. 2 |
|---------|------------|------------|
| D2      | GPIO18     | GPIO18     |
| D3      | GPIO23     | GPIO23     |
| D4      | GPIO24     | GPIO24     |
| D5      | GPIO25     | GPIO25     |
| D6      | GPIO4      | GPIO4      |
| D7      | GPIO17     | GPIO17     |
| D8      | GPIO21     | GPIO27     |
| D9      | GPIO22     | GPIO22     |

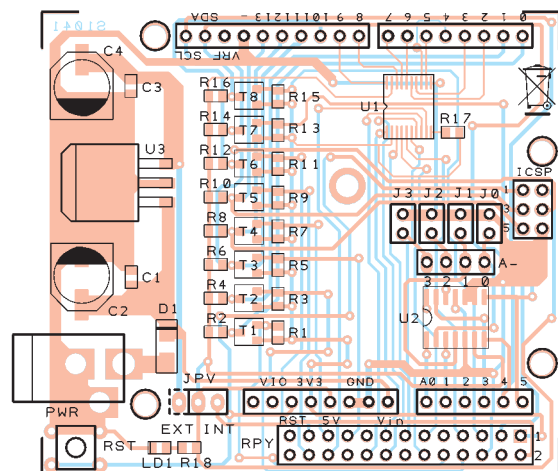
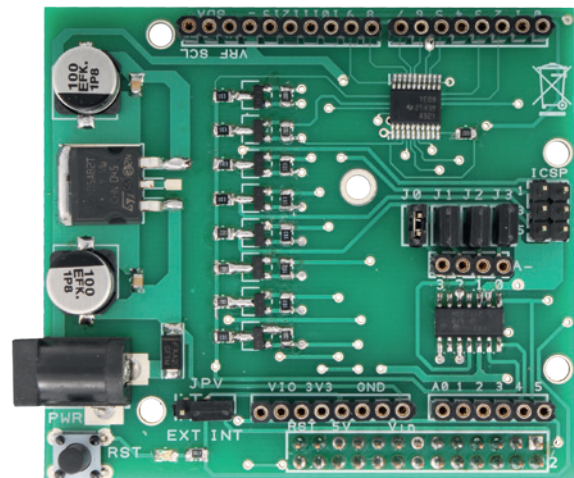
alto dalla resistenza di pull-up R14. Entrambe le sezioni della linea di segnale sono a livello alto, ma ai rispettivi livelli di tensione.

2. Comunicazione proveniente dal dispositivo a 3,3V. - Un dispositivo collegato alla sezione a 3,3V della linea di segnale porta la linea stessa a livello basso. Il source del MOS-FET T7 viene portato anch'esso a livello basso, mentre il gate si mantiene al livello di 3,3V. La VGS viene a trovarsi al di sopra della soglia di innesco ed il MOS-FET entra in conduzione. In questa condizione la sezione della linea di segnale a livello maggiore viene portata anch'essa a livello basso, pilotata dal dispositivo a 3,3V e dal MOS-FET in conduzione. In conclusione entrambe le sezioni della linea di segnale vengono a trovarsi a livello basso.
3. Comunicazione proveniente dal dispositivo a 5V. - Un dispositivo collegato alla sezione a 5V della linea di segnale porta la linea stessa a livello basso. In questa condizione la sezione a 3,3V viene portata a livello basso dal diodo drain-substrate finchè la VGS supera la soglia di innesco ed il MOS-FET T7 entra in conduzione. Ora la sezione a 3,3V della linea di segnale viene tenuta a livello basso pilotata dal dispositivo a 5V e dal MOS-FET in conduzione. In conclusione anche in questo caso entrambe le sezioni della linea di segnale vengono a trovarsi a livello basso.

In questo modo i livelli logici sono convertiti e trasferiti in entrambe le direzioni, indipendentemente dal device che guida la comunicazione. Le condizioni 2 e 3 realizzano la funzione "wired AND" tra le due sezioni della linea di segnale come richiesto dalle specifiche i2C, e non sono affette dalle resistenze di pull up presenti sul bus stesso.

Veniamo ora alla conversione ADC per la quale è stato adottato l'integrato MCP3428 della Microchip (U2).

Nella conversione ADC si è dovuto accettare un piccolo compromesso nella mappatura dei pin sui connettori che accolgono gli shield Arduino. Arduino possiede sei ingressi ADC ma due di questi, attestati sui pin A4 e A5 sono condivisi con gli ingressi i2C. In ambiente Arduino, le diverse destinazioni dei pin sono configurabili a programma, in base alle esigenze della applicazione. Questa possibilità non è riproducibile con il nostro shield che impiega in permanenza i pin del bus i2C per colloquiare con il convertitore

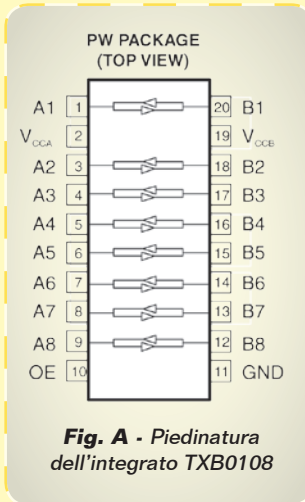


### Elenco Componenti:

|                                       |                                  |
|---------------------------------------|----------------------------------|
| R1 ÷ R17: 10 kohm (0805)              | T6: BSS138W-7-F                  |
| R18: 470 ohm (0805)                   | T7: BSS138W-7-F                  |
| C1: 100 nF ceramico (0805)            | T8: BSS138W-7-F                  |
| C2: 100 µF 25 VL<br>elettrolitico (E) | D1: GF1M-E3/67A                  |
| C3: 100 nF ceramico (0805)            | RST: Microswitch                 |
| C4: 100 µF 25 VL<br>elettrolitico (E) |                                  |
| U1: TXB0108PWR                        | Varie:                           |
| U2: MCP3428-E/SL                      | - Plu alimentazione              |
| U3: MC7805ABD2T<br>(D2PAK)            | - Strip Maschio 2 vie (4 pz.)    |
| LD1: LED verde (0805)                 | - Strip maschio 3 vie (3 pz.)    |
| T1: BSS138W-7-F                       | - Strip maschio 4 vie            |
| T2: BSS138W-7-F                       | - Strip femmina 2x13 vie (1 pz.) |
| T3: BSS138W-7-F                       | - Strip femmina 6 poli           |
| T5: BSS138W-7-F                       | - Strip femmina 8 poli (2 pz.)   |
|                                       | - Strip femmina 10 poli          |
|                                       | - Jumper (5 pz.)                 |
|                                       | - Circuito stampato              |

# L'integrato TXB0108

L'integrato TXB0108 (**Fig. A**) è un traslatore di livelli ad otto bit con due linee di alimentazione configurabili, e consente di far comunicare dispositivi e sistemi che presentano interfacce digitali caratterizzate da livelli di segnale



**Fig. A - Piedinatura dell'integrato TXB0108**

a tensioni differenti, per esempio 3,3V e 5V. L'architettura del TXB0108, della quale in **Fig. B** è visibile una sezione relativa ad una linea di segnale, è in grado di determinare automaticamente la direzione del flusso di dati, e quindi non richiede un segnale di controllo per impostare a priori la direzione del flusso di dati da un device all'altro. Quando una sezione della linea è attivata per traslare dati in una direzione, i driver di output dell'IC mantengono tutta la linea al livello impostato, alto o basso. Data la bassa corrente di mantenimento del livello, prevista allo scopo in progettazione, un device esterno che impegna la linea in direzione opposta con un livello di corrente che supera quello di mantenimento dell'IC, è in grado di provocare l'inversione di direzione del flusso di dati. Per questa ragione i device che pilotano le linee di I/O dell'integrato devono poter mantenere una

corrente di almeno  $\pm 2$  mA. Quando un dispositivo impegna la linea in una direzione, il circuito one-shot rileva il fronte di salita o di discesa, sulle porte A o B a seconda della provenienza del flusso dati. Durante il rilevamento del fronte di salita il circuito one-shot mette in conduzione i transistor PMOS (T1, T3) per un breve periodo, in modo da velocizzare la transizione dal livello basso a quello alto. In modo simile, durante il rilevamento di un fronte discendente il circuito one-shot mette in conduzione i transistor NMOS (T2, T4) per un breve periodo, velocizzando la transizione dal livello alto a quello basso. Il TXB0108 è progettato per pilotare carichi capacitivi fino a 70pF. I driver di output dell'IC sono caratterizzati da una corrente di pilotaggio molto bassa. Se si collegano resistenze di pull up o pull down esterne sulle linee di I/O il loro valore deve essere superiore ai 50 kOhm per essere certi che non contrastino i driver di output

dell'IC. Per la stessa ragione è consigliabile non utilizzare il TXB0108 in applicazioni che richiedono la conversione di livelli dei bus i2C o 1-wire dove di solito i driver collegati alle linee di I/O sono in configurazione open drain con resistenze di pull-up che interferiscono con il meccanismo di auto sensing dell'IC.

La porta A è pilotata dalla tensione presente sul pin  $V_{CCA}$  ed accetta livelli di tensione compresi tra 1,2 V e 3,6 V. La porta B è pilotata dalla tensione presente sul pin  $V_{CCB}$  ed accetta livelli di tensione compresi tra 1,65 V e 5,5 V. La tensione presente sul pin  $V_{CCA}$  non deve mai essere maggiore di quella presente sul pin  $V_{CCB}$ . Quando il terminale output-enable (OE) è al livello basso tutte le uscite sono poste nello stato di alta impedenza e la conversione di livello viene inibita. In funzionamento normale il terminale OE deve essere mantenuto a livello alto, collegato alla linea di alimentazione  $V_{CCA}$ .

ADC, impegnando i relativi pin e lasciandone solo quattro pin liberi per gli ingressi ADC. In compenso l'integrato MCP3428 permette conversioni sino a 16 bit di precisione sia di segnali lineari che differenziali.

I terminali da CH1+ a CH4+ sono collegati rispettivamente ai pin da A0 ad A3 sul connettore AD di Arduino. I terminali da CH1- a CH4- sono attestati sullo strip A-, e possono essere posti a massa singolarmente mediante i jumper J0, J1, J2 e J3. In questo modo è possibile configurare ciascun pin per acquisire segnali analogici sia lineari sia differenziali. I pin SDA e SCL dell'integrato sono collegati da un lato ai pin corrispondenti sul connettore Arduino e dall'altra al drain dei transistor T7 e T8. I pin ADR0 e ADR1 permettono di assegnare diversi indirizzi all'integrato a seconda della combinazione dei livelli alto e basso assegnati ai pin stessi. Nel nostro caso teniamo entrambi i pin a livello basso collegandoli a massa in modo da impostare l'indirizzo 0x68. Ovviamente il pin  $V_{CC}$  è collegato ai 5V ed il pin  $V_{SS}$  a massa. Le linee del bus SPI, MISO, MOSI, SCLK e CS vanno dai piedini 19, 21, 23 e 24 del connettore GPIO ai pin SPI del connettore Arduino ed anche al connettore ICSP. Le linee del bus seriale provenienti dai pin 8 e 10 del connettore GPIO vanno anch'essi ai corrispondenti circuiti a transistor di conversione dei livelli e poi ai pin TXD e RXD del connettore Arduino.

## UTILIZZIAMO LO SHIELD SU RASPBERRYPI

Per procedere nelle operazioni successive, utilizziamo sempre RaspberryPi, modalità terminale, le due finestre SSH ed SCP aperte e login come utente root, come descritto negli articoli precedenti dedicati a RaspberryPi.

Per prima cosa, se RaspberryPi è in funzione, spegniamola con il comando

```
shutdown -h now
```

oppure

```
halt
```

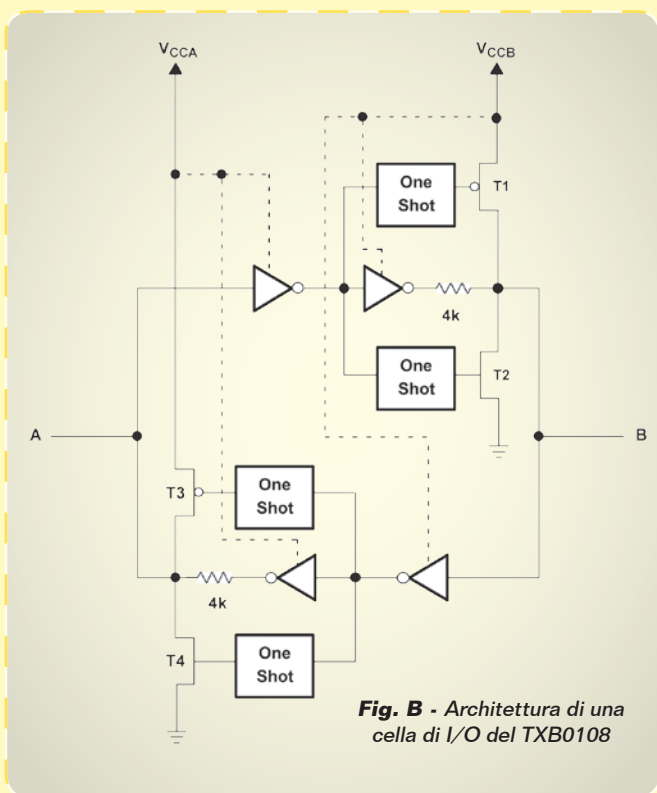
e poi togliamo l'alimentazione. Colleghiamo il connettore RPY dello shield sul connettore GPIO di RaspberryPi e ridiamo tensione.

Ora dobbiamo attivare i moduli di gestione (nel mondo Windows chiamati driver) i2C ed SPI oppure uno solo dei due, in base alle vostre esigenze.

A tal proposito vi è da sottolineare la diversità di approccio ai moduli di gestione delle periferiche esterne nei mondi Windows e Linux che illustriamo nel riquadro di approfondimento dedicato.

Come potete verificare nel riquadro di approfondimento i moduli di gestione dei bus i2C ed SPI sono compilati nel kernel di Raspbian come





**Fig. B** - Architettura di una cella di I/O del TXB0108

moduli esterni e non facenti parte integrale del kernel. Oltre a questo, nella distribuzione di Raspbian che utilizziamo per i nostri articoli sono anche blacklisted ovvero oscurati all'utilizzo normale. Questo perché le linee dei bus condividono alcuni pin digitali del GPIO che vengono privilegiati nella configurazione predefinita. Per ... resuscitare e rendere utilizzabili uno o entrambi i moduli di gestione dei bus i2C e SPI è necessario toglierli dalla blacklist e poi "aggiungerli" all'insieme di moduli conosciuti dal kernel.

Visto che i moduli di gestione dei bus i2C ed SPI sono stati introdotti nelle versioni più recenti della distribuzione Raspbian, iniziamo sempre con l'aggiornamento della distribuzione presente sulla nostra scheda con i comandi:

```
apt-clean
apt-get update
apt-get upgrade
```

Ora apriamo il file di configurazione che contiene l'elenco dei moduli blacklisted (oscurati), con il comando :

```
nano /etc/modprobe.d/raspi-blacklist.conf
```

Nano è un editor di testo minimale che funziona in ambiente terminale.

Eliminiamo i moduli i2C ed SPI dalla blacklist cancellando le righe o, come abbiamo preferito noi, commentandole con un "#" (Fig. 5). Premiamo Ctrl-X e poi Y alla richiesta di salvare il file dopo le modifiche.

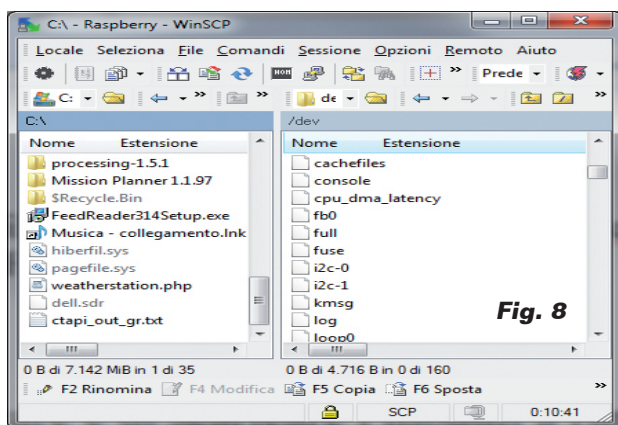
Ora dobbiamo fare in modo che i due moduli "liberati" vengano caricati e diventino parte integrante del kernel.

Per questa operazione abbiamo due possibilità. La prima ci permette di caricare i moduli a comando, ed ha validità per tutto il tempo nel quale RaspberryPi rimane acceso. Al boot successivo i moduli dovranno essere ricaricati a comando.

La seconda ci permette di caricare i moduli direttamente al boot del sistema operativo, e renderli disponibili alle applicazioni immediatamente dopo il boot, condizione indispensabile in un sistema server unattended.

La prima possibilità richiede l'utilizzo del comando modprobe. Scriviamo (Fig. 6).

| Module         | Size   | Used by                           |
|----------------|--------|-----------------------------------|
| i2c_dev        | 5587   | 0                                 |
| bneip          | 10514  | 2                                 |
| rftcomm        | 33663  | 0                                 |
| bluetooth      | 157711 | 10 rftcomm, bneip                 |
| snd_bcm2835    | 12808  | 0                                 |
| snd_pcm        | 74834  | 1 snd_bcm2835                     |
| snd_timer      | 19698  | 1 snd_pcm                         |
| snd            | 52489  | 3 snd_timer, snd_pcm, snd_bcm2835 |
| snd_page_alloc | 4951   | 1 snd_pcm                         |
| spidev         | 5136   | 0                                 |
| i2c_bcm2708    | 3681   | 0                                 |
| spi_bcm2708    | 4401   | 0                                 |



**Fig. 8**

*modprobe i2c-dev*

Possiamo vedere il buon esito della attivazione dei driver con il comando che mostra la lista di tutti i moduli installati (Fig. 7):

*lsmod*

Dato che in Linux tutto (o quasi) è un file, se andiamo nella cartella /dev vediamo apparire i file di collegamento ai device i2c-0 ed i2c-1 (Fig. 8). Il comando *modprobe* permette di caricare e di scaricare i moduli a run time e mantiene i suoi effetti fintanto che RaspberryPi è acceso. In caso di spegnimento, o anche solo di reboot, i nostri moduli dovranno essere ricaricati manualmente. Questa condizione non è adatta a funzionare con una applicazione stand alone, che deve funzionare in modo automatico. Il comando *modprobe*, con l'opzione *remove*, può essere utilizzato anche per disattivare un modulo caricato in precedenza, per esempio:

*modprobe -r i2c-dev*

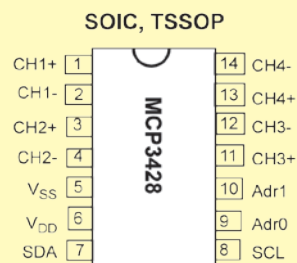
Se si desidera che i moduli vengano caricati alla accensione di RaspberryPi è necessario abilitare il caricamento permanente del driver i2C, che si realizza modificando opportunamente il file di configurazione /etc/modules, che contiene la lista dei driver da caricare al momento del boot. Per modificare il file possiamo usare il comando:

*nano /etc/modules*

ed aggiungere una nuova linea al file di configurazione (Fig. 9)

*i2c-dev*

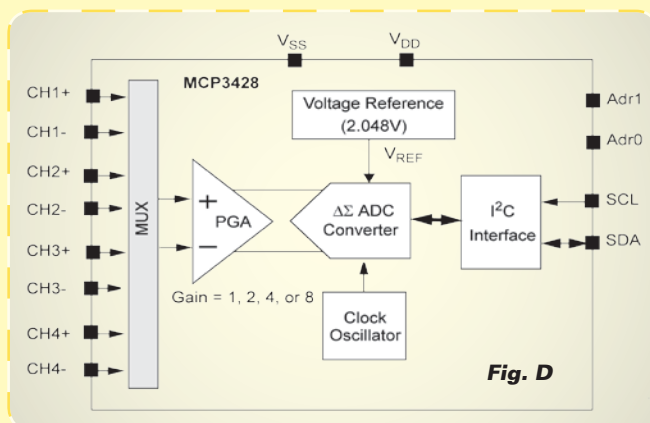
L'integrato MCP3428 è un convertitore analogico-digitale Delta-Sigma a basso rumore che può raggiungere una risoluzione fino a 16 bit. Sono disponibili quattro canali di ingresso differenziali; il layout del dispositivo è visibile in Fig. C.



**Fig. C**

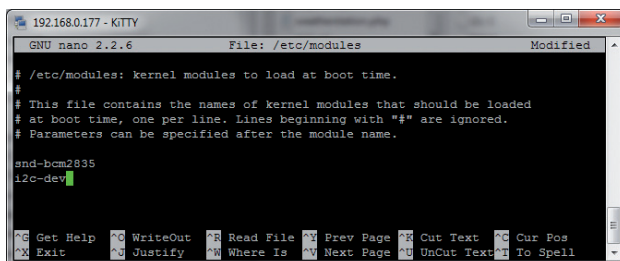
La frequenza di campionamento dipende dalla risoluzione che si vuole ottenere; con una risoluzione di 16 bit si possono ottenere 15 misurazioni al secondo, 60 misurazioni

al secondo alla risoluzione di 14 bit e 240 misurazioni alla risoluzione di 12 bit. Il dispositivo si calibra automaticamente eliminando gli errori di offset e di guadagno dovuti alle variazioni di temperatura e di tensione di alimentazione. Come visibile nella Fig. D, la tensione di riferimento di 2,048V è gestita internamente al dispositivo, il che permette un intervallo di tensioni di ingresso di  $\pm 2.048V$  differenziali. Il segnale in ingresso può essere amplificato per mezzo dell'amplificatore di guadagno interno (PGA nella Fig. D), programmabile dall'utente per ottenere fattori di guadagno di 1, 2, 4, o 8 volte. È anche possibile programmare la modalità di conversione, che può essere a campionamento continuo o singolo. In modalità continua il dispositivo esegue letture in continuazione ed



**Fig. D**

Premete CTRL X e poi y per salvare le modifiche al file ed uscire. Ora installiamo il pacchetto i2c-tools che ci fornisce una serie di funzioni utilizzabili a linea di comando per verificare il funzionamento del bus i2c:



**Fig. 9**

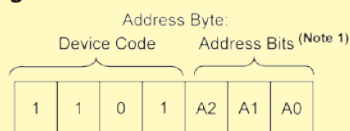
| I <sup>2</sup> C Device Address Bits |    |    | Logic Status of Address Selection Pins |               |
|--------------------------------------|----|----|----------------------------------------|---------------|
| A2                                   | A1 | A0 | Adr0 Pin                               | Adr1 Pin      |
| 0                                    | 0  | 0  | 0 (Addr_Low)                           | 0 (Addr_Low)  |
| 0                                    | 0  | 1  | 0 (Addr_Low)                           | Float         |
| 0                                    | 1  | 0  | 0 (Addr_Low)                           | 1 (Addr_High) |
| 1                                    | 0  | 0  | 1 (Addr_High)                          | 0 (Addr_Low)  |
| 1                                    | 0  | 1  | 1 (Addr_High)                          | Float         |
| 1                                    | 1  | 0  | 1 (Addr_High)                          | 1 (Addr_High) |
| 0                                    | 1  | 1  | Float                                  | 0 (Addr_Low)  |
| 1                                    | 1  | 1  | Float                                  | 1 (Addr_High) |
| 0                                    | 0  | 0  | Float                                  | Float         |

**Fig. E**

aggiorna il buffer con l'ultima lettura eseguita. In modalità singola la lettura viene eseguita a seguito di un comando ricevuto e poi il device entra in standby, in attesa della richiesta successiva. Quest'ultima modalità permette di ridurre considerevolmente i consumi di corrente. La comunicazione con l'integrato avviene sul bus I<sup>2</sup>C con l'invio dei registri di indirizzo e di comando, di due byte ciascuno.

## Registro di indirizzo

Il registro di indirizzo serve ad identificare quale dispositivo si vuole raggiungere sul bus I<sup>2</sup>C. È composto da 7 bit i primi 4 dei quali sono fissi, rappresentano il codice device univoco assegnato al produttore ed assumono la configurazione 1101. Gli ultimi tre sono determinati da



come configuriamo i due pin di indirizzo del dispositivo, la tabella in **Fig. E** contiene le diverse configurazioni dei tre bit a seconda dello stato dei pin di address. Nel nostro caso abbiamo collegato entrambi i pin a massa e di conseguenza i tre bit assumono valore 000. La configurazione completa del byte di indirizzo diventa 1101000 che equivale a 0x68 in esadecimale.

## Registro di configurazione

Un po' più complessa è la gestione del registro di configurazione, che è utilizzato per impostare le modalità di funzionamento del convertitore ADC e per richiedere la lettura

del buffer di uno specifico canale. Il byte di configurazione è il risultato del concatenamento di diverse impostazioni espresse in formato binario come visibile nella **Fig. F**.

## bit 7 RDY: Bit di Ready

Il bit costituisce l'indicatore che i dati nel buffer sono pronti da leggere. Nella modalità one-shot configurare a "1" il bit istruisce il device ad iniziare una nuova lettura.

In lettura il bit RDY assume i seguenti significati:

- 1 = Il buffer di output non è stato aggiornato.
- 0 = Il buffer di output è stato aggiornato con l'ultima lettura.

Significato del bit RDY in scrittura:

In modalità di conversione continua viene ignorato. In modalità di conversione a campionamento singolo:

- 1 = Inizia una nuova conversione.
- 0 = Nessun significato.

## bit 6-5 C1-C0: Bit di selezione del canale di lettura

- 00 = Canale 1
- 01 = Canale 2
- 10 = Canale 3
- 11 = Canale 4

## bit 4 O/C: Bit che indica la modalità di conversione

- 1 = Modalità di conversione a campionamento continuo.
- 0 = Modalità di conversione a campionamento singolo.

## bit 3-2 S1-S0: Bit di configurazione della risoluzione della conversione

- 00 = 12 bit con 240 conversioni al secondo
- 01 = 14 bit con 60 conversioni al secondo
- 10 = 16 bit con 15 conversioni al secondo

## bit 1-0 G1-G0: PGA Bit di configurazione del fattore di guadagno

- 00 = x1
- 01 = x2
- 10 = x4
- 11 = x8

Se vogliamo leggere il canale 1 in modalità one-shot, con risoluzione di 16 bit e guadagno x1 dovremo comporre il registro con la seguente sequenza:

1-00-0-10-00

che in esadecimale corrisponde al valore 0x88 che corrisponde al registro usato nel nostro programma.

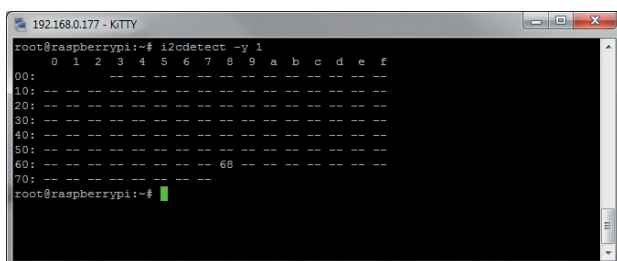
**Fig. F**

| R/W-1 | R/W-0 | R/W-0 | R/W-1 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| RDY   | C1    | C0    | O/C   | S1    | S0    | G1    | G0    |
| 1 *   | 0 *   | 0 *   | 1 *   | 0 *   | 0 *   | 0 *   | 0 *   |
| bit 7 |       |       |       | bit 0 |       |       |       |

`apt-get install i2c-tools`

Aggiungiamo il nostro utente pi al gruppo i2c:

`adduser pi i2c`



**Fig. 10**

Eseguiamo il reboot di RaspberryPi per attivare le nuove configurazioni con il comando

`reboot`

Dopo che RaspberryPi si è riavviata e vi siete riconnessi con Putty o Kitty verificate se sul bus I<sup>2</sup>C è visibile il convertitore ADC con il comando

`i2cdetect -y 0` per RaspberryPi rev. 1 oppure `i2cdetect -y 1` per RaspberryPi rev. 2

Dovrete ottenere un risultato simile a quello visibile in **Fig. 10** dove l'indirizzo 0x68 identifica il



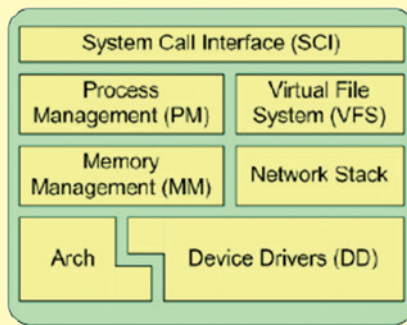
# Gestione moduli in Linux

Quello di GNU-Linux è un mondo immenso, ed una trattazione approfondita delle sue logiche di funzionamento, oltre a non essere nel campo di interesse di questa rivista, richiederebbe centinaia e centinaia di pagine e di schemi. Tuttavia pensiamo sia opportuno approfondire alcuni aspetti che possono facilitare la comprensione del perché si eseguono certe operazioni o che possono dare una impostazione alla risoluzione di problemi che possono presentarsi. Prendiamo spunto dalla necessità di attivare i moduli di gestione dei bus i2C ed SPI per approfondire maggiormente gli aspetti reconditi della gestione dei Device Driver in Linux.

Ci ricollegiamo a quanto descritto nel box di approfondimento sulla architettura del sistema operativo GNU-Linux presentata nel numero 168 di ElettronicaIn. Abbiamo descritto le caratteristiche del kernel, che ne hanno permesso la portabilità sui processori più disparati, e quindi il principale fattore della sua diffusione.

Se approfondiamo l'architettura del kernel possiamo considerarlo composto da un certo numero di sottosistemi come visibile nella **Fig. G**.

Sofferamoci per brevità su due sottosistemi particolari, quello denominato Arch nella **Fig. G** che può essere considerato il vero cuore del kernel, che si occupa dell'interfaccia verso il processore hardware specifico al quale è destinato e che deve essere riscritto, almeno in parte, per ogni nuovo processore: il cosiddetto processo di "porting". L'altro sottosistema, fortemente integrato con il primo è l'insieme dei Device Drivers, l'insieme di moduli che permettono di gestire le periferiche più disparate da parte del kernel stesso e delle applicazioni utente.



**Fig. G**

La maggior parte dei dieci milioni di righe di codice che costituiscono i sorgenti del kernel di GNU-Linux sono concentrati proprio nel sottosistema Device Drivers e sono compilati congiuntamente al kernel. Sono disponibili due opzioni di compilazione dei Device Driver, la prima consiste nel compilare i Device Driver all'interno del kernel in modo da ottenere un unico oggetto monolitico. Questa opzione è utilizzata per integrare nel kernel i Device Driver di utilizzo obbligatorio come console, terminali e dischi oppure per rendere accessibili automaticamente (plug and play) i dispositivi come dongle e periferiche USB, in modo da rendere queste operazioni totalmente trasparenti all'utilizzatore.

La seconda modalità permette di compilare i Device Driver come moduli caricabili nel kernel separatamente, e solo se necessario, come è il caso dei Device Driver per i bus i2C e SPI.

Come abbiamo già visto negli articoli scorsi le periferiche gestite dai Device Driver vengono poi presentate alle applicazioni come file nella cartella /dev (gpio4 o ttyUSB0 per fare degli esempi che abbiamo già incontrato).

Vediamo per sommi capi la sequenza di azioni e collegamenti che permettono di creare la catena di trasmissione di dati e comandi tra la periferica hardware reale, gestita dal kernel, e la sua rappresentazione applicativa, nella cartella /dev.

Quando si desidera compilare una nuova distribuzione di Linux si inizia col disporre i sorgenti che si desidera includere nella compilazione in una struttura di cartelle opportunamente organizzate.

Uno dei passi successivi più impegnativi è la definizione delle caratteristiche che dovrà assumere la distribuzione finale, obiettivo che si ottiene configurando le centinaia di opzioni e parametri previste all'interno dei sorgenti. Per fare questa operazione si utilizza un configuratore a menù del quale, nelle **Fig. H** e **Fig. I** si vedono due pagine della sezione dedicata ai Device Driver.

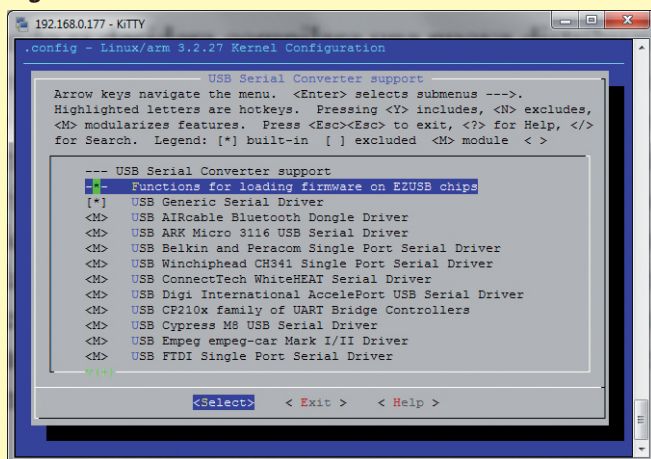
Nella **Fig. H** si può notare, per esempio, il driver FTDI (quello utilizzato per la Centralina meteo, mentre nella **Fig. I** si può vedere il Device Driver per il bus i2C. I moduli da includere nel corpo del kernel sono selezionati con "\*", quelli da compilare separatamente sono contrassegnati con una "M".

Dopo aver terminato, e salvato il file di configurazione, si esegue la compilazione effettiva del kernel utilizzando il compilatore "C" (pacchetto g++). Al termine si installano i componenti compilati del kernel nelle rispettive partizioni di boot e di root. Il kernel riconosce ciascun Device Driver per mezzo dei cosiddetti numeri di device major e minor.

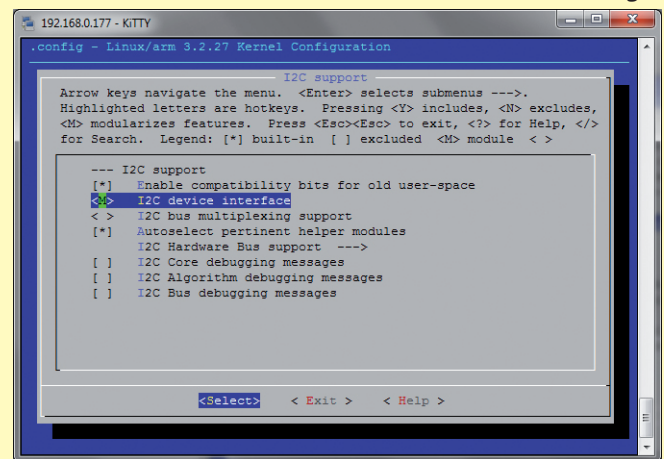
Questi numeri sono identificativi univoci per i Device Driver che sono definiti nella documentazione del kernel e sono cablati all'interno del codice dei driver per ciascun tipo di dispositivo.

Per avere un'idea dei driver disponibili nel

**Fig. H**



**Fig. I**



```

192.168.0.177 - KTTY
root@raspberrypi:~# cat /proc/devices
Character devices:
1 mem
4 /dev/vc/0
4 tty
5 /dev/tty
5 /dev/console
5 /dev/ptmx
5 ctyprintk
7 vcs
10 misc
13 input
14 sound
29 fb
89 i2c
116 alsa
128 ptm
136 pts
153 spi
162 raw
180 usb
189 usb device
204 ttyAMA
216 rfcomm
252 vchiq
253 vc-mem
254 rtc

```

Fig. J

```

192.168.0.177 - KTTY
root@raspberrypi:~# cd /dev
root@raspberrypi:/dev# ls -l
total 0
crw-rw-rw- 1 root root 10, 235 Jan 1 1970 autofs
drwxr-xr-x 2 root root 600 Jan 1 1970 block
crw-rw-rw- 1 root root 10, 234 Jan 1 1970 btrfs-control
drwxr-xr-x 3 root root 60 Jan 1 1970 bus
crw-rw-rw- 1 root root 10, 63 Jan 1 1970 cachefiles
drwxr-xr-x 2 root root 2300 Jan 18 10:33 char
crw-rw-rw- 1 root root 5, 1 Jan 18 08:59 console
crw-rw-rw- 1 root root 10, 62 Jan 1 1970 cpu_dma_latency
drwxr-xr-x 5 root root 100 Jan 1 1970 disk
crw-rw-rw- 1 root root 29, 0 Jan 1 1970 fb0
lrwxrwxrwx 1 root root 13 Jan 1 1970 fd -> /proc/self/fd
crw-rw-rw- 1 root root 1, 7 Jan 1 1970 full
crw-rw-rw- 1 root fuse 10, 229 Jan 1 1970 fuse
crw-rw-rw- 1 root root 89, 0 Jan 18 10:33 i2c-0
crw-rw-rw- 1 root root 89, 1 Jan 18 10:33 i2c-1
drwxr-xr-x 2 root root 60 Jan 1 1970 input
crw-rw-rw- 1 root root 1, 11 Jan 1 1970 kmsg
srw-rw-rw- 1 root root 0 Jan 18 08:59 log
brw-rw-rw- 1 root disk 7, 0 Jan 1 1970 loop0
brw-rw-rw- 1 root disk 7, 1 Jan 1 1970 loop1
brw-rw-rw- 1 root disk 7, 2 Jan 1 1970 loop2
brw-rw-rw- 1 root disk 7, 3 Jan 1 1970 loop3
brw-rw-rw- 1 root disk 7, 4 Jan 1 1970 loop4
brw-rw-rw- 1 root disk 7, 5 Jan 1 1970 loop5

```

Fig. K

kernel di RaspberryPi possiamo dare il comando

`cat /proc/devices`

con il quale otteniamo la lista di **Fig. J** che mostra i driver disponibili con il loro numero identificativo major. Potete notare che la lista è divisa tra dispositivi a carattere e dispositivi a blocco. L'associazione dei Device Driver del kernel con i file di device presenti nella cartella /dev avviene utilizzando proprio i numeri di device. Possiamo costatarlo andando nella cartella /dev e digitando il comando

`ls -l`

Otterremo il risultato visibile in **Fig. K**. Per ciascuna riga che identifica un device troviamo la prima lettera "c" o "b" che identifica se il dispositivo è a carattere o a blocco, seguono le autorizzazioni, poi ancora il "proprietario" del file. Seguono ancora due numeri separati da

una virgola; il primo è il numero di device major ed il secondo il minor. Dopo la data troviamo il nome del file che identifica il device per le applicazioni. In questo modo quando una applicazione richiede l'utilizzo di un certo device richiamandolo con il nome corrispondente in /dev, il sistema operativo è in grado, risalendo a ritroso la catena descritta, di associare il Device Driver corretto per la sua gestione. Quando si compila un modulo per un nuovo Device Driver l'associazione tra numeri di device ed ingressi in /dev viene creata utilizzando il comando `mknod`, come per esempio potrebbe essere nel caso del driver del bus i2C.

`mknod /dev/i2c-0 c 89 0`

Nel corpo dell'articolo è descritto come caricare nel kernel i Device Driver compilati come moduli, specificatamente per i moduli di gestione dei bus i2C ed SPI.

convertitore ADC. Dato che stiamo realizzando uno strumento di integrazione con gli shield di Arduino, per sperimentare il funzionamento del convertitore ADC, e del bus i2C in generale, abbiamo preparato un piccolo circuito, derivato dagli esempi presenti nel libro Arduino Uno - Programmazione avanzata e librerie di sistema e dall'eserciziario LabVIEW MyDAQ (entrambi reperibili presso Futura Elettronica), per misurare, rispettivamente, la temperatura con una resistenza NTC e la luminosità ambiente con una fotoresistenza.

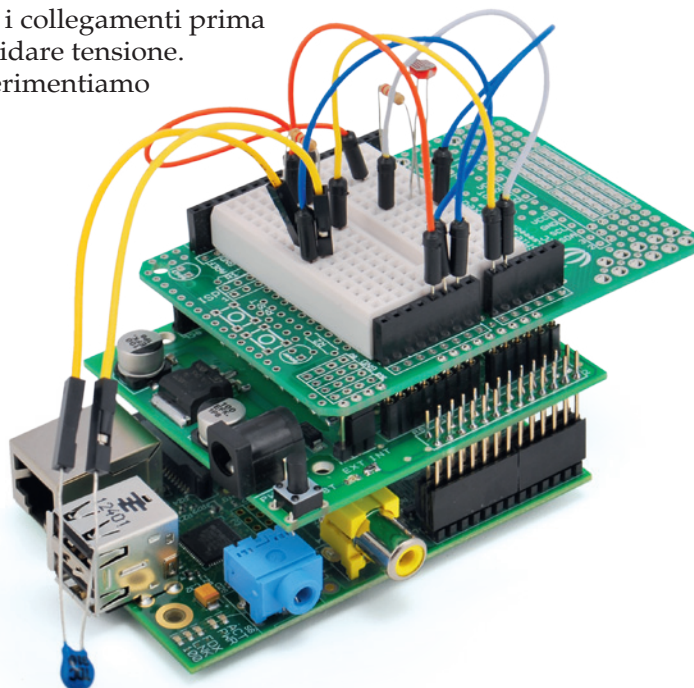
Lo schema del circuito è visibile in **Fig. 11**.

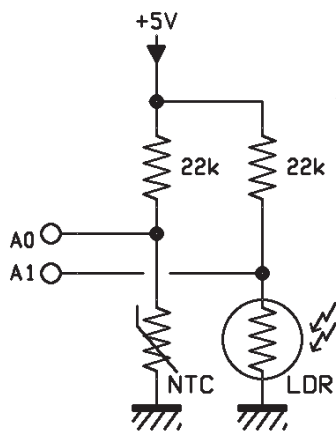
Le resistenze sono da 22kOhm.

La NTC è una Vishay NTCLE-100E-3103 da 10kOhm.

La foto resistenza è una LDR04 o una GL5528 da 10kOhm.

Il punto centrale del partitore di tensione del quale fa parte la NTC è collegato al canale CH1+ (A0) del convertitore ADC, il punto centrale del partitore di tensione del quale fa parte la fotoresistenza è collegato al canale CH2+ (A1). I terminali CH1- e CH2- sono collegati a massa. Il circuito è realizzato con uno shield di prototipazione per Arduino al quale abbiamo incollato una bread board mille fori con nastro biadesivo. Lo shield di prototipazione è montato a pacchetto sullo shield di integrazione, che a sua volta è innestata sul connettore GPIO di RaspberryPi. Spegnete sempre RaspberryPi prima di eseguire interventi e verificate sempre i collegamenti prima di ridare tensione. Sperimentiamo





**Fig. 11**

ora il funzionamento del convertitore utilizzando la linea di comando da terminale. Usiamo il comando

`i2cget -y 0 0x68 0x88 w` per RaspberryPi rev. 1  
oppure  
`i2cget -y 1 0x68 0x88 w` per RaspberryPi rev. 2

per leggere il valore esadecimale (**Fig. 12**) dei due byte restituiti dal canale 1 del convertitore ADC che rappresenta il valore digitale della tensione misurata ai capi della resistenza NTC.

Il secondo parametro indica il bus i2C da interrogare, il terzo il registro che contiene l'indirizzo del dispositivo da leggere (0x68), il quarto il registro di comando che indica l'ingresso del quale desideriamo conoscere la lettura digitale e le modalità di campionamento, l'ultimo, il parametro w, che indica che desideriamo di ritorno entrambi i byte della misura. A seconda della temperatura otteniamo un valore di quattro

caratteri, che rappresentano la conversione in valore digitale della tensione misurata ai capi della resistenza NTC espressa da due byte esadecimali, nel nostro caso "5769" (**Fig. 12**). Ricordiamo sempre che il primo byte (low byte) è il meno significativo ed il secondo è il più significativo (high byte). Prima di utilizzarli, per esempio per convertirli in un valore decimale, dobbiamo invertirli, ottenendo, nel nostro caso, "6957". Per convertire il valore esadecimale nel corrispondente valore decimale possiamo usare il comando:

`echo 'ibase=16;obase=A;6957' | bc`

che ci restituirà il valore 26967 (**Fig. 13**).

Il comando stampa su console il risultato della conversione da base 16 a base 10 (A in esadecimale) del valore 6957 (è obbligatorio utilizzare le maiuscole per le lettere) e lo passa al programma "bc". "bc" è un generico programma di calcolo a riga di comando, che permette calcoli e conversioni in tutte le basi, val la pena di provarlo perché può risolvere molte situazioni. Per esempio per le conversioni da binario a esadecimale utilizzate il comando:

`echo 'ibase=2;obase=10000;bbbbbbbb' | bc`

sostituendo bbbbbbbb con il

## per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono di facile reperibilità. Il master del circuito stampato può essere scaricato dal sito della rivista così come il programma di esempio che verrà presentato nel prossimo numero. Di questo shield Futura Elettronica ha realizzato la versione montata e collaudata (cod. FT1041M) disponibile al prezzo di 27 Euro, IVA compresa.

Il materiale va richiesto a:  
Futura Elettronica, Via Adige 11,  
21013 Gallarate (VA)  
Tel: 0331-799775 • Fax: 0331-792287  
<http://www.futurashop.it>

valore che volete convertire. Per convertire da esadecimale a binario utilizzate il comando:

`echo 'ibase=16;obase=2;hh' | bc`

sostituendo hh con il valore che volete convertire. Possiamo eseguire lo stesso test sul canale CH2 del convertitore ADC per sincerarci del corretto funzionamento del circuito della foto resistenza. Nel prossimo numero approfondiremo i calcoli di conversione dalla lettura digitale ai valori di temperatura e luminosità, con il relativo programma in python. ■

```
192.168.0.177 - KITTY
root@raspberrypi:~# i2cget -y 1 0x68 0x88 w
0x5769
root@raspberrypi:~#
```

**Fig. 12**

```
192.168.0.177 - KITTY
root@raspberrypi:~# i2cget -y 1 0x68 0x88 w
0x5769
root@raspberrypi:~# echo 'ibase=16;obase=A;6957' | bc
26967
root@raspberrypi:~#
```

**Fig. 13**



Dopo ZigBee, Aurel torna con un nuovo sistema di rete wireless questa volta proprietario e funzionante ad 868 MHz, dedicato alla domotica ed alla home-security.

# RFTIDE: LA WIRELESS PAN DI AUREL

di DAVIDE SCULLINO

**D**opo l'esperienza fatta nella ZigBee Alliance, Aurel esordisce con un proprio sistema wireless PAN destinato alle applicazioni domotiche, che, secondo l'annuncio fatto dalla Casa di Modigliana (FC) è più economico, semplice e meno dispendioso, sul piano energetico, dello ZigBee. Il nuovo sistema si chiama RFTide e consiste in una rete PAN mesh, costituita perciò da dispositivi Master che inviano i comandi (i quali possono essere comandi di acquisizione dati o esecuzione di operazioni), da altri che ne formano i nodi o End Device; i comandi possono essere inviati dall'apposito telecomando Aurel, ovvero da altri sistemi di trasmissione interfacciati con smartphone o PC. Per l'esattezza, la rete RFTide prevede due tipi di trasmissione: quella dove un Master (tipicamente un telecomando) invia nella rete comandi che devono essere eseguiti dagli Slave, i quali a seconda del tipo di comando possono anche rispondere comunicandone, per esempio, l'avvenuta esecuzione (ad esempio l'impostazione di un relé); quella in cui a trasmettere è un Master collegato ad esempio ad un computer e quindi il comando può contenere un'interrogazione dello stato dello Slave o di un sensore ad esso collegato. In quest'ultimo caso il Master non trasmette comandi di attivazione di utilizzatori, ma diviene veicolo di tra-

sporto di dati da e verso l'eventuale computer o altro sistema collegato ad esso.

Gli Slave o End Device sono unità di vario genere che possono attivare utilizzatori elettrici mediante relé, ovvero acquisire dati riguardanti la misura di temperatura o altre informazioni da sensori.

La rete RFTide opera ad 868 MHz, banda di frequenze in cui Aurel ha una consolidata esperienza e nella quale al momento operano meno sistemi che nella classica ISM a 2,4 GHz; ciò permette una comunicazione meno disturbata. Inoltre, operando ad una frequenza più bassa le perdite sono minori che a 2,4 GHz, quindi il consumo di elettricità da parte dei dispositivi è ridotto rispetto a quelli standard ZigBee.

Come nello ZigBee, nell'RFTide si può realizzare la comunicazione tra nodi che non si "vedono" direttamente, sfruttando il mesh networking: in pratica i dati trasmessi dai Master verso le unità Slave possono essere ripetuti da altri Slave o da appositi dispositivi chiamati Wall Repeater. Inoltre l'RFTide ammette la presenza, nella rete, di nodi mobili.

L'elemento principale di una rete RFTide è un dispositivo Master, cui possono essere associati tanti nodi (Slave); la procedura di abbinamento è semplicissima e simile a quella di apprendimento dei dispositivi basati sull'encoder rolling-code HCS creato da Microchip per



l'impiego nel comando a distanza via radio.

Ripetendo questa procedura per tutti i nodi, si crea la rete.

Questo modo di funzionamento semplifica il ruolo dell'installatore e risolve il problema dell'installazione iniziale. Inoltre, per modificare la rete o aggiungere un nodo basta semplicemente ripetere il processo di abbinamento.

Una funzionalità interessante è la copia dell'unità Master Remote Control.

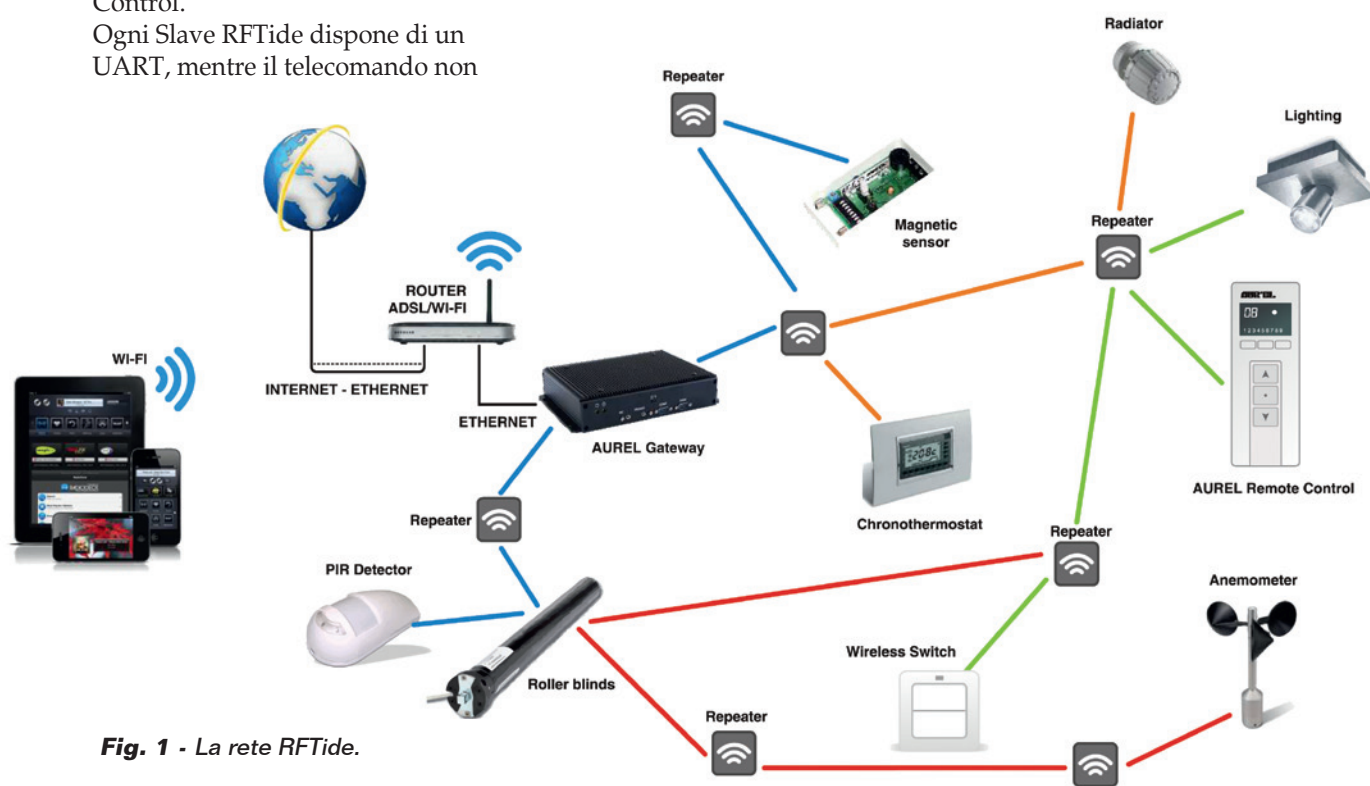
Ogni Slave RFTide dispone di un UART, mentre il telecomando non

volatile di questi ultimi. Il codice è univoco per ciascun Remote Control e viene copiato negli Slave durante la procedura di abbinamento. Per aumentare l'immunità alle interferenze e la sicurezza d'uso, Aurel ha previsto la possibilità, per i clienti, di customizzare il codice dell'unità Master.

I dispositivi che compongono una rete RFTide sono:

- Master;
- Slave;

quindi pensato per la domotica o il comando di tapparelle motorizzate, illuminazione, elettroserrature ecc. Master può anche essere un Personal Computer o uno Smartphone, purché "introdotto" nella rete mediante un Gateway: quest'ultimo dispositivo è una sorta di traslatore dei comandi sulla rete RFTide e dei dati in arrivo verso PC o Smartphone. Il Gateway Aurel RFTide può interfacciarsi con dispositivi esterni alla rete, trasferendo



**Fig. 1 - La rete RFTide.**

l'ha, o meglio, ce l'ha ma non ne fa uso, perché il suo scopo è inviare comandi specifici e non ricetrasmettere dati, compito, questo, del Gateway e delle unità Slave.

RFTide permette di far operare più reti nello stesso ambiente senza che esse si disturbino a vicenda; ciò perché ciascuna rete è identificata da un codice univoco (NetworkID) di 4 byte, che durante l'abbinamento dei dispositivi Slave viene memorizzato nella memoria non

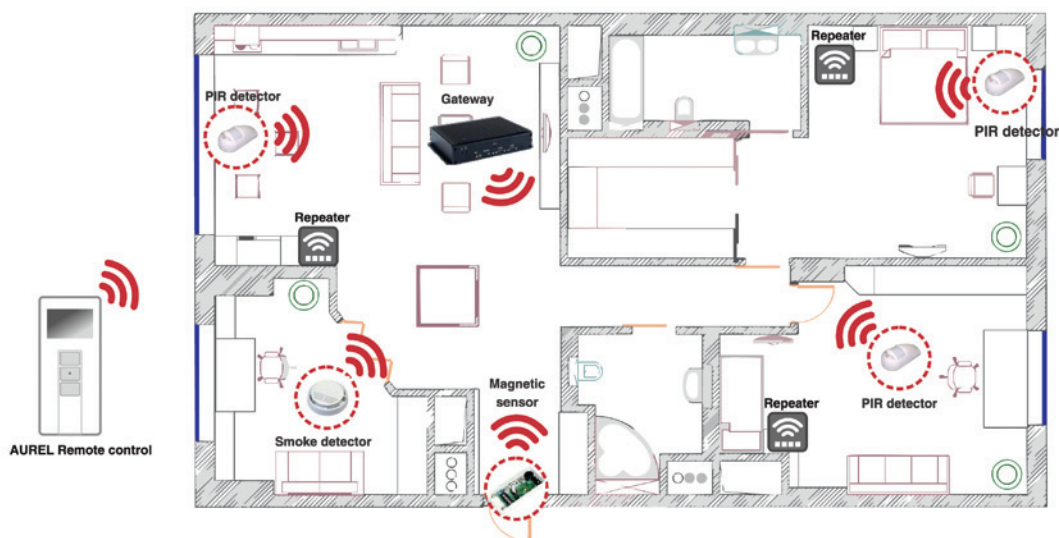
- Ripetitori;
- Gateway.

Descriviamoli brevemente qui di seguito.

### UNITÀ MASTER

I Master sono quelli che impartiscono i comandi; il principale è il telecomando Aurel (Remote Control) progettato per impartire comandi di attivazione di vari utilizzatori diretti ad unità Slave e

eventualmente i dati su ethernet o Wi-Fi, perciò consente di interagire con la rete usando appunto PC, tablet, smartphone ed altri dispositivi dotati di tale connettività. Il Gateway trova applicazione in sistemi decisamente più articolati e complessi del semplice comando a distanza: ad esempio nella telemetria e nel monitoraggio di ambienti, nel controllo dei consumi energetici (in questo caso permette di acquisire i consumi e inserire



**Fig. 2**  
Applicazione  
dell'RFTide in  
un sistema di  
sicurezza.

o disinserire utilizzatori collegati alla rete elettrica) nella gestione del condizionamento e del riscaldamento (permette di acquisire i dati su temperatura e umidità nell'ambiente e governare l'attività di caldaie, pompe di calore ecc.) nella sicurezza (consente di verificare l'attività di sensori PIR, radar, contatti ecc. e gestire gli attuatori di allarme) nella domotica (permette di governare illuminazione, tapparelle e tende da esterno motorizzate sulla base dei dati forniti da sensori crepuscolari e di pioggia) ed altro ancora.

Il Gateway va, insomma, usato quando nella RFTide si voglia

inserire un computer o uno smartphone o altro dispositivo a microprocessore; consente altresì di affacciare una rete RFTide sul web e quindi di realizzare soluzioni di telegestione e web-based.

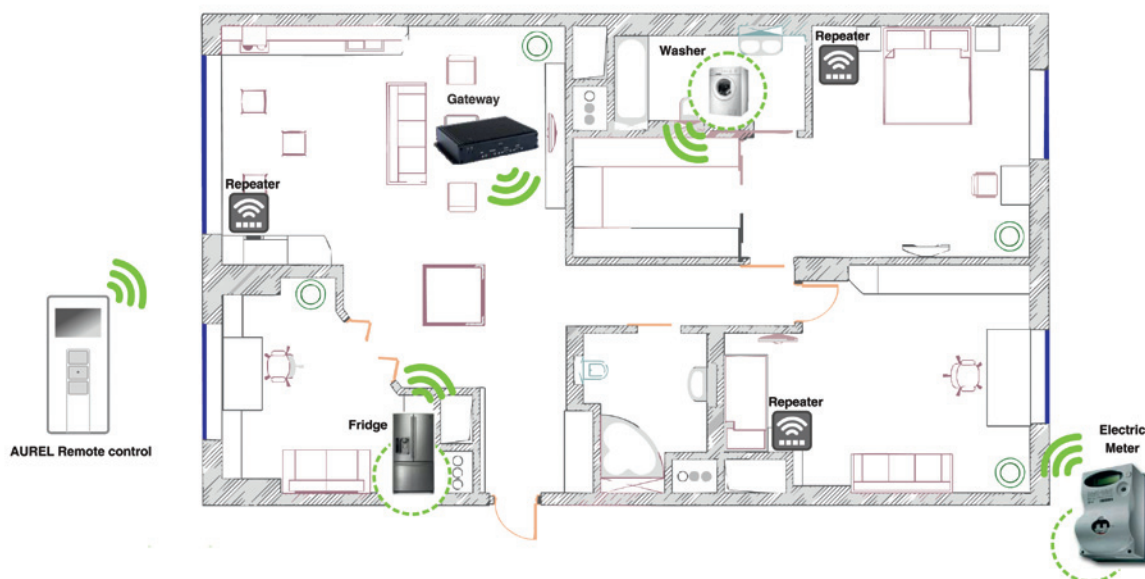
Finora abbiamo parlato di Gateway in senso generale, però a questo punto va precisato che esistono due tipi di Gateway: uno è un dongle USB, mentre l'altro è una RaspberryPi opportunamente programmata per funzionare da interfaccia di rete ethernet.

Il Gateway USB non è altro che un transceiver, ossia un'unità Slave abbinata ad un convertitore seriale/USB e programmata per fun-

zionare semplicemente da UART via radio; permette di affacciare direttamente sulla rete RFTide qualsiasi dispositivo (computer, ecc.) in grado di inviare e ricevere dati dalla rete stessa.

Quanto al Gateway ethernet, è una Raspberry abbinata ad un Gateway USB: in pratica è programmata per funzionare da ponte da e verso l'ethernet, ma, non avendo alcuna interfaccia RF, necessita del dongle USB per introdursi ed operare via radio nella rete RFTide.

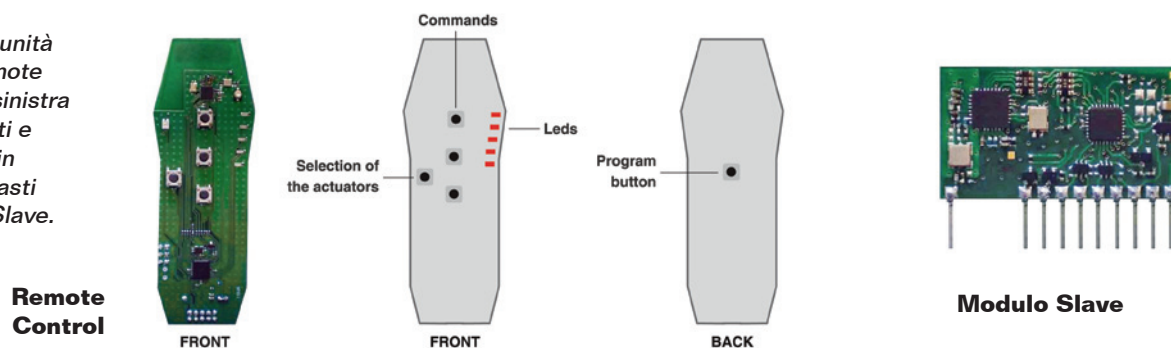
L'unità Master Remote Control è un vero e proprio telecomando dotato di pulsanti per inviare comandi e per avviare la procedura di



**Fig. 3**  
Monitoraggio  
dei consumi di  
elettricità  
mediante  
RFTide.



**Fig. 4** - Le unità Master Remote Control (a sinistra visto davanti e dietro, con in evidenza i tasti e i LED) e Slave.



abbinamento delle unità Slave: sul pannello frontale riporta un tasto per selezionare l'attuatore e tre per impartire i comandi, oltre ad una fila di LED che segnalano lo stato degli attuatori; sul retro c'è il pulsante con cui si avvia la procedura di abbinamento agli Slave.

Quanto ai LED, la fila segnala qual è il canale selezionato cui si trasmette il comando, mentre esiste anche un LED a più colori in grado di segnalare lo stato assunto dall'attuatore collegato allo Slave cui è stato inviato un comando.

In una rete RFTide ci possono essere più dispositivi Remote Control; ogni Remote Control può entrare in una rete RFTide semplicemente apprendendo il codice del telecomando esistente: basta avviare l'apposita procedura sui due Remote Control. Questo tipo di unità Master può trasmettere in modo Broadcast: ciò significa che il comando è diretto a tutti gli Slave indistintamente.

Il telecomando è un dispositivo di comando bidirezionale, nel senso

che quando invia un segnale di comando verso uno Slave può ricevere da esso e mostrare tramite LED se il comando è giunto ed è stato ricevuto.

### UNITÀ SLAVE

Lo Slave è un transceiver, quindi può sia ricevere comandi che deve trasformare in azioni locali, sia, dietro comando di richiesta di acquisizione da parte di un'unità Master, inviare dati di stato di sensori eventualmente collegati ad esso. In realtà l'unità Slave è l'insieme di un UART e di un microcontrollore che gestisce direttamente I/O digitali; l'UART serve quando il modulo viene usato per acquisire dati da un dispositivo ad esso collegato che li invia serialmente, sia per inviarli ad un dispositivo locale dotato di interfaccia seriale. L'UART consente un data-rate di 15 kbps.

Ciascun modulo Slave dispone di un canale dati seriale bidirezionale (pin TX per trasmettere ed RX per ricevere) TTL-compatibile (sia a livello 0/5V, sia compatibile 0/3V) utilizzabile per interagire con dispositivi locali; oltre a ciò, ha tre linee di GPIO (General Purpose I/O) utilizzabili per comandare carichi mediante transistor o line-driver e per acquisire livelli logici.

Ogni unità Slave dispone di uno stadio RF sintonizzato sugli 868 MHz capace di una sensibilità in ricezione di ben -100 dB ed una potenza in trasmissione di +7 dB. Il modulo Slave è un End Device, ossia un terminale destinatario di

comandi provenienti dal Master, tuttavia va notata una particolarità: lo Slave può funzionare anche da ponte. In breve, la rete Aurel funziona così: quando uno Slave intercetta un segnale radio contenente dati secondo il protocollo RFTide, se questi dati contengono un comando diretto ad esso lo Slave lo esegue (se il comando prevede una risposta o la lettura di dati locali, lo Slave li trasmette al Master) altrimenti lo ritrasmette. Analogamente, se uno Slave intercetta dati trasmessi da un'unità analoga e diretti ad un Master, li ritrasmette verso quest'ultimo.

Gli Slave, oltre ad essere dei ricetrasmittitori fanno da ponte e quindi permettono di realizzare reti ben più estese della massima distanza coperta dal segnale radio trasmesso dai Master o dagli Slave stessi; gli Slave formano quindi una rete mesh.

Quando sulla rete viaggia un comando, solo il nodo Slave destinatario risponderà eseguendo l'operazione richiesta ed eventualmente trasmettendo i dati richiesti.

Lo Slave si alimenta tipicamente a 3,3 V (ma esiste anche in versione a 5 V) ed assorbe pochissimo: appena 33 mA in trasmissione e 10 mA in ricezione; prevede una modalità di Standby nel quale si colloca automaticamente una volta eseguito il comando e si risveglia da essa quando riceve l'apposito comando di wake-up. In Standby il modulo Slave assorbe un massimo di 4  $\mu$ A. Queste caratteristiche consentono di alimentare a pila



**Fig. 5**  
L'unità  
Wall Repeater.

**Fig. 6** - Le unità Master Gateway: a sinistra la chiavetta wireless e a destra la RaspberryPi.



gli Slave, garantendo lunghissima autonomia.

Per funzionare in una rete e ricevere i comandi in arrivo da un Gateway o da un Remote Control, ciascuno Slave va abbinato mediante una semplice procedura: basta premere l'apposito pulsante di apprendimento collegato al suo piedino 7 ed il tasto Program sul Remote Control (dal Gateway occorre impartire il comando di Learning).

Una particolarità dei moduli Slave è che prevedono già tre differenti profili di funzionamento grazie ai quali possono essere facilmente usati in determinate applicazioni standard:

- **MOTOR profile;** è il profilo che permette il controllo di un motore bidirezionale e l'acquisizione, tramite gli input digitali, dei segnali di due eventuali fine corsa;
- **PWM profile;** con questo profilo un'uscita dello Slave genera un segnale PWM grazie al quale può gestire (tramite adeguati driver di potenza) lampade o motori elettrici in continua;
- **Wireless metering profile;** in questa modalità il nodo periodicamente invia al Master dei dati letti dalla porta seriale, i quali possono essere l'acquisizione di sensori o rilevatori di grandezze elettriche.

Per impostazione predefinita in fabbrica, ciascun nodo esce per uso generico.

### RIPETITORI

Anche detti Wall-Repeater, sono dei nodi che ricevono il segnale radio e lo ritrasmettono; in altre parole sono dei nodi non intelligenti e si distinguono dagli Slave per il fatto che ritrasmettono i segnali che ricevono senza fare distinzioni. I Wall Repeater sono quindi dei ponti radio in miniatura

che consentono ai Master di raggiungere gli Slave destinatari dei loro comandi ma che sono troppo lontani per essere raggiunti direttamente dal segnale RF; viceversa, permettono ai Master di ricevere eventuali dati che hanno richiesto agli Slave troppo lontani. In una rete RFTide si possono installare quanti ripetitori servono, senza limite. Un malfunzionamento di uno o più ripetitori non pregiudica il funzionamento di tutta la rete, ma può renderne inutilizzabile la parte più lontana, che non può più ricevere il segnale dal Master.

Il ripetitore non va chiuso in contenitori metallici o messo a ridosso di ampi piani metallici, che ne influenzerebbero la diffusione dei segnali radio; ciò vale anche per le unità Slave. Le dimensioni di appena 43x43x25 mm e l'apposita scatola ne favoriscono l'incasso. Anche lo Slave è in teoria un ripetitore, ma usarlo come tale sarebbe improprio, perché il ripetitore è fatto per questa funzione ed allo scopo funziona a 220 V in modo da operare in stand-alone.

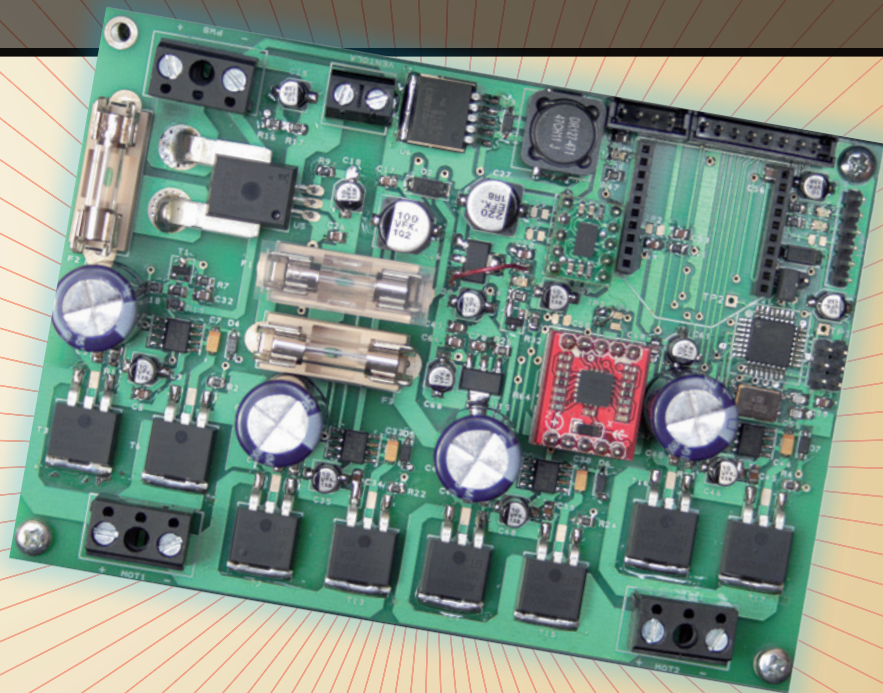
### APPLICAZIONI

I dispositivi che realizzano la rete wireless di Aurel trovano applicazione in svariati campi, sebbene il sistema sia nato essenzialmente per essere integrato nella domotica, quindi per realizzare controlli globali degli utilizzatori elettrici di casa (illuminazione ecc.) delle tapparelle motorizzate, delle tende da esterno motorizzate e via di seguito, fino ad arrivare all'integrazione con l'impianto di riscaldamento e condizionamento.

Trova impiego anche nella sicurezza, giacché consente di realizzare sistemi di monitoraggio dei locali; anzi, si integra perfettamente con gli appositi sensori Aurel e consente di realizzare sofisticati impianti d'allarme wireless eventualmente affacciati sul web per effettuare il monitoraggio a distanza e per gestire da remoto le varie funzionalità. Questo grazie all'unità Gateway, che si interfaccia con tutti gli elementi della rete e trasferisce i dati ad un computer, sul quale è possibile sia l'elaborazione locale degli eventi, sia la connessione ad Internet con piattaforme remote. Sempre il Gateway, permette di realizzare una rete wireless PAN che monitorizza l'attività ed il consumo dei vari utilizzatori elettrici e che può interagire con gli elettrodomestici intelligenti, ritrasmettendone i dati ed eventualmente collegandosi ad interruttori digitali che possono governarne l'attività in funzione delle fasce orarie che garantiscono il maggior risparmio. Comunque nulla vieta di impiegare l'RFTide come semplice (si fa per dire) comando a distanza per gestire utilizzatori domestici; in questo caso si avrà a disposizione un radiocomando un po' più flessibile ed evoluto del classico sistema TX/RX chiuso su se stesso. **I prodotti RFTide di Aurel possono essere acquistati presso Futura Elettronica srl (per disponibilità e prezzi collegarsi al sito [www.futurashop.it](http://www.futurashop.it)).**







# OpenWheels

## SENSORI ED ELETTRONICA

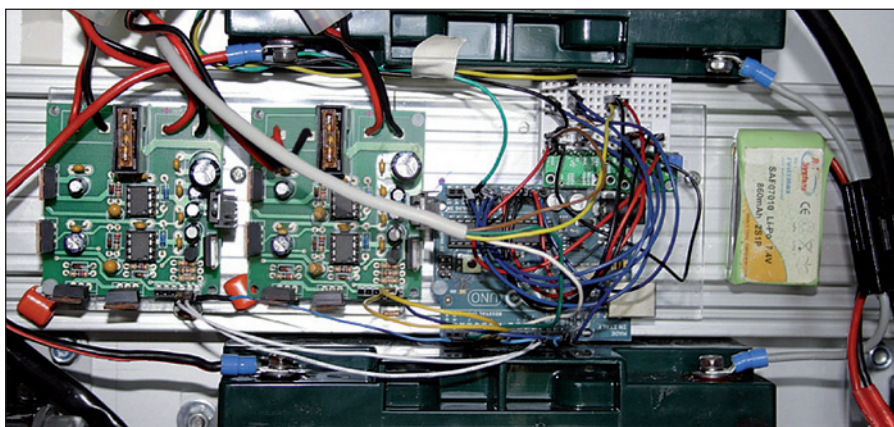
Dopo la descrizione della struttura meccanica, entriamo nel vivo del principio di funzionamento e dello schema elettrico da noi messo a punto per controllare OpenWheels.

di MIRCO SEGATELLO

**Q**uesto mese, ultimata la descrizione della meccanica di OpenWheels, iniziamo ad occuparci dell'elettronica di controllo, prendendo in esame innanzitutto i meccanismi adottati per mantenere in equilibrio il nostro veicolo. Per rendervi partecipi della nostra esperienza vi diciamo subito che la soluzione da noi adottata (ancora non definitiva, peraltro, e sicuramente migliorabile), ha seguito un percorso di continui piccoli miglioramenti, non



esente da difficoltà. A dire il vero ce lo aspettavamo essendo questo un progetto piuttosto complesso che comprende sia una parte meccanica molto "particolare" che una sezione elettronica di controllo che non poteva che utilizzare un microcontrollore (oltre a tutta una serie di sensori) e che doveva essere in grado di svolgere un compito piuttosto complesso, proprio per rispondere alle specifiche di questo veicolo. La prima che abbiamo dovuto fare riguardava



**Fig. 1** - Il primo prototipo del controller, realizzato con una board Arduino Uno e con molti componenti volanti, ci ha consentito di valutare le scelte progettuali e di verificarne le prestazioni.

il sistema di sviluppo da adottare, scelta non difficile perchè, essendo il nostro un progetto open source, la scheda Arduino è apparsa sin da subito la soluzione più appropriata. Qualche dubbio iniziale ha fatto capolino nella nostra mente considerando che la scheda Arduino non è proprio il massimo in fatto di velocità di elaborazione ed il calcolo matematico complesso non è il suo forte ma, come vedremo, con un sapiente utilizzo delle risorse disponibili, siamo riusciti ad ottenere ottimi risultati.

Per fuggire sin dall'inizio qualsiasi dubbio, abbiamo realizzato una prima versione dell'elettronica di OpenWheels con una board Arduino Uno, una breadboard filata e tutti gli altri componenti collegati in maniera provvisoria per verificarne (e nel caso sostituirli facilmente) le prestazioni; pur con collegamenti precari, questa prima soluzione ci ha permesso di acquisire una certa esperienza e ci ha consentito di verificare le prestazioni delle soluzioni adottate. La prima verifica l'abbiamo fatta col modulo inerziale (IMU) dai cui segnali, opportunamente elaborati, dipende la stabilità del veicolo. Come sempre accade in questi casi, si inizia scegliendo e testando i vari componenti disponibili sul mercato. Il modulo IMU ha essenzialmente la funzione di rilevare,

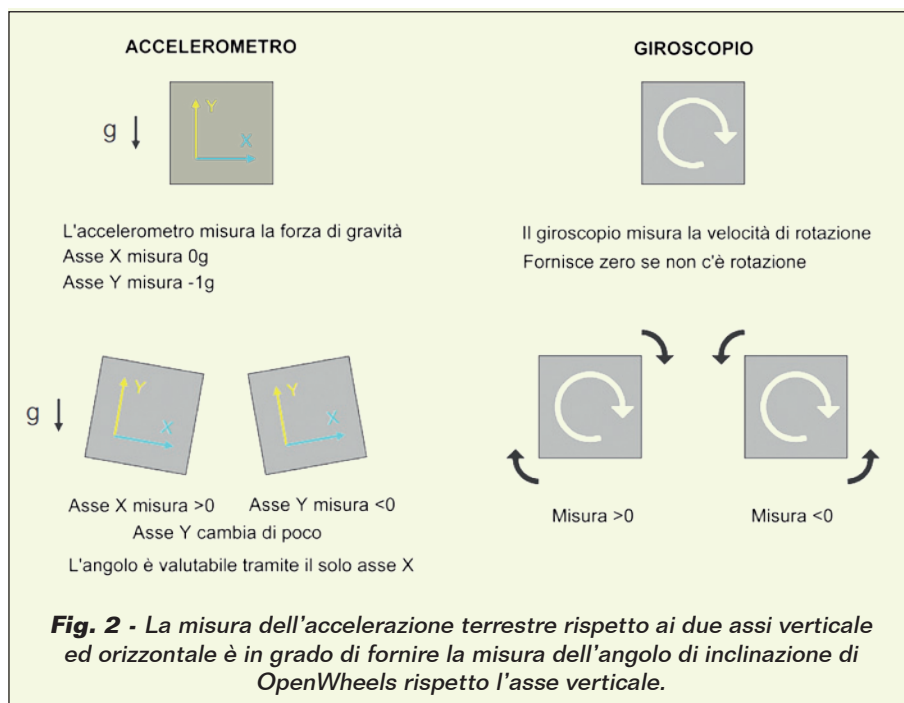
con la massima accuratezza, l'inclinazione di OpenWheels rispetto all'asse trasversale, coincidente con l'asse delle ruote, e la velocità con la quale la piattaforma si inclina in avanti o indietro, ovvero la velocità di rotazione rispetto l'asse delle ruote. Il primo componente preso in considerazione è stato l'accelerometro MMA7260QT fornito in breakout dalla Pololu e quindi facilmente interfacciabile con Arduino.

La misura dell'accelerazione terrestre rispetto i due assi verticale ed

orizzontale è in grado di fornire la misura dell'angolo di inclinazione di OpenWheels rispetto l'asse verticale (Fig. 2).

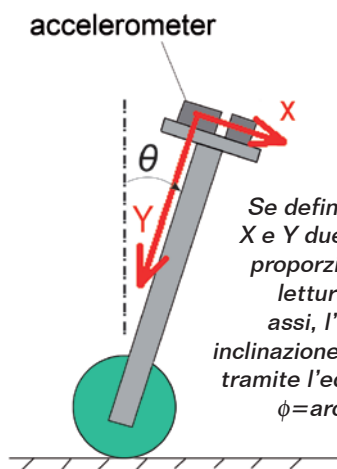
Se definiamo con X e Y due variabili proporzionali alla lettura dei due assi ed opportunamente scalate (Fig. 3) possiamo ricavare l'angolo di inclinazione tramite l'equazione:  $\phi = \arctg(X/Y)$ .

Arduino mette a disposizione varie funzioni matematiche, compreso l'inverso della tangente, che però impegna non poco la CPU; abbiamo quindi utilizzato un'equazione semplificata applicabile in considerazione del fatto che l'inclinazione di OpenWheels non supererà mai i 30°; la semplice equazione  $\phi = K \cdot X$  fornisce la misura dell'angolo con un errore non superiore all'1% nel range da -30° a +30°. Aver escluso la misura rispetto all'asse verticale (Y) permette di eliminare tutte le interferenze dovute ai sobbalzi che necessariamente avrebbero interessato il veicolo durante la marcia. La costante K è determinata considerando la sensibilità



**Fig. 2** - La misura dell'accelerazione terrestre rispetto ai due assi verticale ed orizzontale è in grado di fornire la misura dell'angolo di inclinazione di OpenWheels rispetto l'asse verticale.





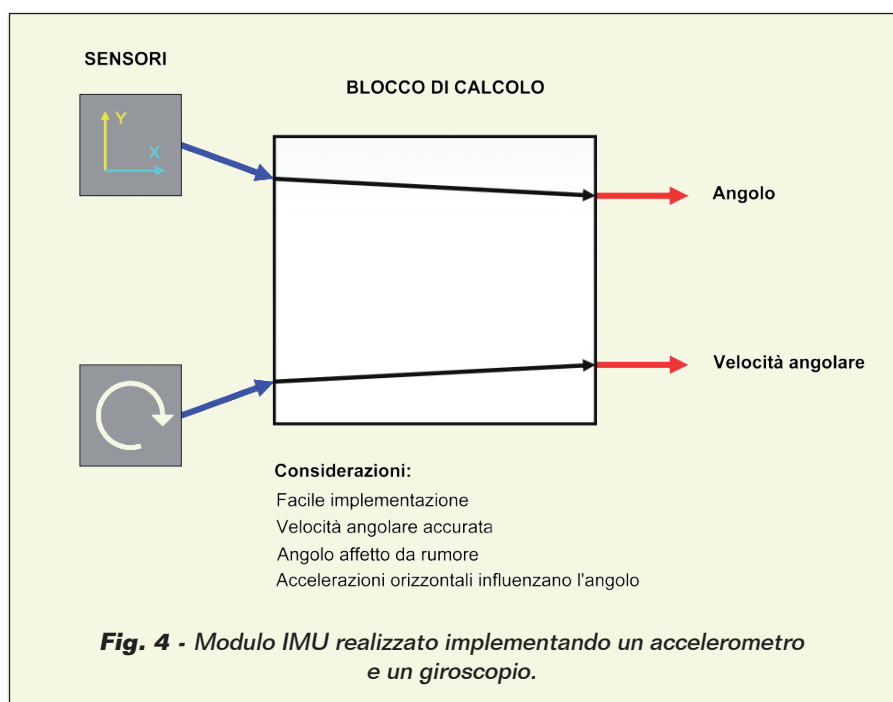
**Fig. 3**

Se definiamo con  $X$  e  $Y$  due variabili proporzionali alla lettura dei due assi, l'angolo di inclinazione si ricava tramite l'equazione:  $\phi = \arctg(X/Y)$ .

dell'accelerometro ed il valore fornito dal modulo ADC interno di Arduino nella misura della tensione analogica dell'accelerometro. Definendo con  $V_{ref}$  [mV] la tensione di riferimento per l'ADC di Arduino e con sensitivity la sensibilità dell'accelerometro [mV/g] il valore di  $K$  è determinabile tramite:

$$K = 360 * V_{ref} / (sensitivity * 2 * \pi * 1024)$$

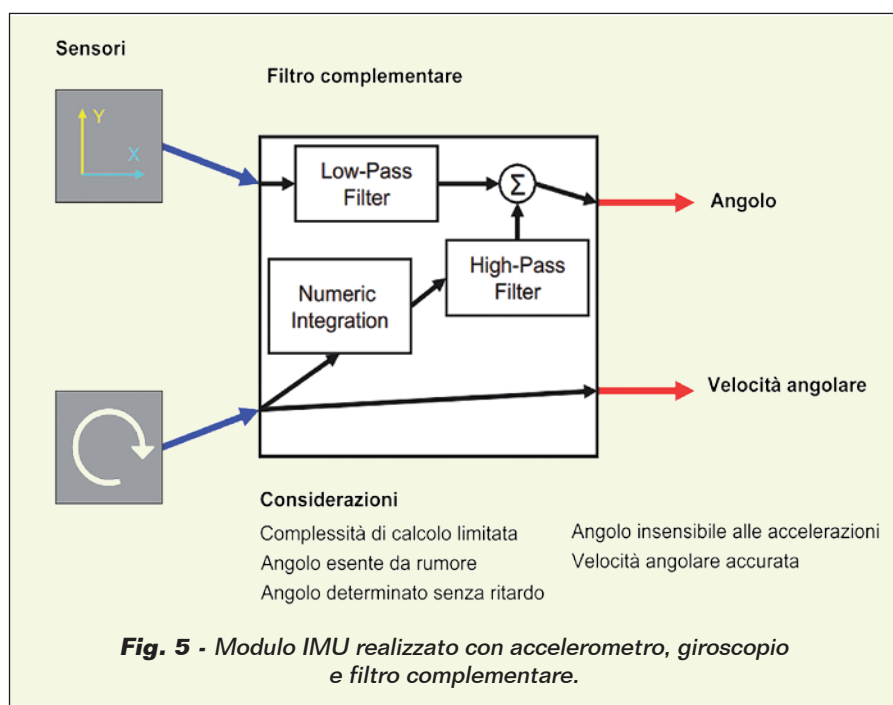
Il valore 1024 è dovuto al fatto che il convertitore ADC di Arduino lavora a 10 bit fornendo un valore numerico compreso tra 0 e 1024. Da sola, tuttavia, questa prima misura dell'angolo tramite la sola lettura dell'accelerometro è però inutilizzabile per il nostro scopo, mentre, ad esempio, va benissimo per individuare l'inclinazione di un tablet o di uno smartphone. Un primo motivo di questa affermazione risiede nell'elevata quantità di rumore che affligge la tensione generata dall'accelerometro; una semplice media dei valori letti porterebbe sicuramente un grande beneficio nell'eliminare il rumore ma, inevitabilmente, comporterebbe un ritardo nella determinazione del valore corretto. Un altro punto a svuotare è dato dalle inevitabili vibrazioni e sobbalzi ai quali è sottoposto OpenWheels che prontamente sarebbero misurati dall'accelerometro e riportati alla sua uscita



**Fig. 4 - Modulo IMU realizzato implementando un accelerometro e un giroscopio.**

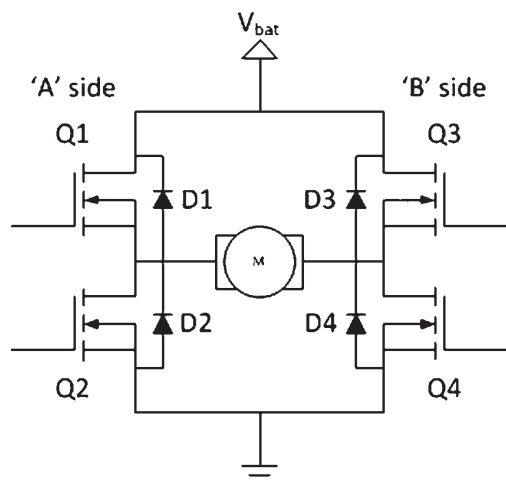
come impulsi di disturbo. Per la misura della velocità di rotazione della nostra piattaforma esiste un'altra categoria di sensori che possono fare al caso nostro; ci riferiamo ovviamente ai giroscopi (vedi Fig. 2). Questa famiglia di componenti permette di misurare la velocità di rotazione attorno ad

uno o più assi con estrema precisione e con un limitatissimo rumore elettrico; inoltre sono immuni alle vibrazioni esterne. Misurare la velocità di rotazione non permette però di ricavare direttamente l'angolo assoluto di rotazione che invece si può determinare con una operazione matematica detta



**Fig. 5 - Modulo IMU realizzato con accelerometro, giroscopio e filtro complementare.**





**Fig. 6** - Classica configurazione a ponte H di un driver di potenza per motori a spazzole in CC.

calcolo integrale. Purtroppo il giroscopio è affetto dalla presenza di un offset sulla sua uscita che a causa delle caratteristiche costruttive non rimane stabile nel tempo; in altri termini ciò significa che, ad esempio, ad una velocità di rotazione nulla corrisponda un'uscita di valore zero. Questo impedisce una corretta implementazione del calcolo integrale se non per un brevissimo tempo di funzionamento. Per ottenere un'accurata misura sia dell'angolo che della velocità angolare è necessario sfruttare contemporaneamente le caratteristiche di accelerometro e giroscopio; si tratta di sfruttare la misura dell'accelerometro nel lungo periodo per conoscere l'inclinazione assoluta di OpenWheels, mentre l'accelerometro ci fornisce la variazione istantanea dell'inclinazione. Tralasciamo in queste pagine la descrizione delle possibili tecniche matematiche per la fusione della lettura dei due sensori e descriviamo brevemente la tecnica implementata da noi. Si tratta, in sintesi, di implementare un filtro complementare nel quale la misura dell'accelerometro viene processata da un filtro passa basso e fusa con l'integrale della lettura del giroscopio dopo essere stata filtrata con un circuito passa alto. Il filtro complementare

sarà implementato digitalmente tramite elaborazione matematica dai campioni della lettura dell'accelerometro e del giroscopio. Vista la sua semplicità è sufficiente calcolare la frequenza di taglio dei due filtri passa alto e passa basso, valore determinato sperimentalmente tramite l'analisi del grafico relativo all'acquisizione dei dati. Anche queste informazioni, grazie al sistema di telemetria implementato a bordo della scheda, potranno essere visualizzati a video insieme a tutti gli altri parametri di funzionamento.

Nella versione definitiva l'accelerometro impiegato è il modello MMA7361L fornito in breakout dalla Pololu e distribuito in Italia da Futura Elettronica col codice 8218-MMA7361L. Per l'accelerometro la scelta è caduta sul modello LPY503AL fornito in breakout dalla ditta Sparkfun e distribuito in Italia da Futura Elettronica col codice 8218-LPY-510AL. La scheda definitiva da noi messa a punto per OpenWheels accetta sia l'installazione di queste due breakout sia la saldatura diretta dei due chip, soluzione che permette di risparmiare sull'acquisto dei componenti ma implica una saldatura in SMD realizzabile solo da personale molto esperto, fornito di adeguata attrezzatura.

Descriviamo ora la modalità utilizzata per misurare

l'inclinazione del manubrio, dato che sarà usato per implementare la sterzata durante la marcia. Grazie alla particolare struttura meccanica utilizzata, l'intero manubrio può inclinarsi lateralmente ruotando attorno ad un perno fissato alla pedana sul quale è calettato un potenziometro. Il potenziometro, utilizzato come partitore di tensione, fornisce una misura proporzionale all'angolo di inclinazione del manubrio, valore facilmente letto dal modulo ADC di Arduino. Per quanto riguarda la telemetria, il circuito da noi messo a punto dispone di un sistema wireless basato su moduli XBee. Questi moduli implementano al loro interno tutto il necessario per rendere una trasmissione wireless seriale del tutto simile ad una trasmissione via cavo, assolutamente "trasparente" rispetto al supporto trasmissivo.

Per la nostra applicazione sono sufficienti due moduli di bassa potenza, uno presente sulla scheda

OpenWheels ed uno che sarà collegato opportunamente al PC destinato alla visualizzazione della telemetria. Saranno necessari due moduli XBee 1 mW con antenna a chip (codice XBEE1MWCHIP) di cui uno installato sulla scheda OpenWheels ed uno interfacciato al PC tramite un apposito adattatore (codice EXPLORERXBEE oppure codice XBEE Dongle); descriveremo la programmazione dei due moduli quando ci occuperemo del software del progetto.

Dopo i primi test sul modulo IMU e sulla telemetria abbiamo cominciato a sviluppare i

driver di potenza destinati al controllo dei due motori; la prima scelta è caduta su due moduli commerciali che però non si sono



rivelati all'altezza della situazione, soprattutto quando abbiamo cominciato ad sfruttare OpenWheels alla massima potenza: in breve tempo gli elevati assorbimenti in corrente hanno messo fuori uso i moduli impiegati. Scartate altre soluzioni commerciali che avrebbero fatto lievitare il costo finale della scheda, abbiamo optato per realizzare un circuito di potenza ad hoc per il controllo dei motori. La primissima versione prevedeva due schede separate, una per ciascun motore, connesse ad una board Arduino tramite opportune connessioni; questa prima soluzione ci ha consentito di mettere a punto i circuiti di potenza che successivamente sono stati integrati in un'unica basetta con tutti gli altri componenti necessari al funzionamento di OpenWheels.

Il driver doveva essere alimentato a 24 volt e capace di gestire con facilità gli oltre 10 ampere assorbiti da ciascun motore a piena potenza; ovviamente occorreva tenere conto anche delle sovracorrenti in fase di accelerazione e dei picchi di corrente dovuti agli aggiustamenti di velocità. Alcune considerazioni sono state fatte circa la modalità di funzionamento del driver; quando OpenWheels è in posizione verticale, i motori possono subire repentini cambi di direzione il che ci ha fatto escudere la modalità di funzionamento a due comandi, PWM e direzione (sign-magnitude mode) che normalmente si usa sui vari Motor Shield per Arduino. In questa modalità, quando OpenWheels è fermo nella posizione verticale, il segnale PWM di controllo della potenza dei motori presenta un duty-cycle molto basso, mentre il segnale di direzione cambia spesso di stato. Questa è una situazione assai stressante per un driver ed il fatto di generare segnali impulsivi di così breve durata, implica sempre problemati-

Nella puntata precedente abbiamo descritto tutti i dettagli della piattaforma meccanica da noi realizzata e che a breve metteremo in produzione per dare a tutti l'effettiva possibilità di realizzare un vero veicolo personale. Tutti i dettagli costruttivi e le modifiche apportate al progetto al fine di migliorarne le prestazioni sono disponibili sul sito <http://openwheels.elettronica.in.it>; sullo stesso sito presentiamo anche i suggerimenti e le soluzioni riguardanti l'elettronica inviate dai nostri lettori. A partire da questo numero, con la presentazione del circuito di controllo, invitiamo i nostri lettori a suggerire modifiche e miglioramenti sia all'hardware che al software da noi implementati. Tornando alla meccanica, la soluzione da noi adottata è molto semplice e sicuramente alla portata anche di quegli hobbysti che non hanno particolare dimestichezza con cacciaviti, chiavi inglesi, brugole e quant'altro.

Per il montaggio della struttura meccanica occorre partire da un profilato in alluminio ai lati del quale vanno montate due staffe piuttosto robuste utilizzate per fissare le ruote. Nonostante la loro robustezza, le staffe non sarebbero mai in grado di sostenere lo sforzo che grava sul perno delle ruote. Per questo motivo, all'interno di ciascuna staffa è montato un collare piuttosto robusto con foro centrale da 20 mm che può essere stretto mediante una vite a brugola. Il nostro veicolo utilizza pneumatici tipo 120/70 R12 montati su cerchi da 12 pollici completi di cuscinetti da 15x25x11 mm; all'interno di ciascun cerchio è montata una piastra porta corona dello spessore di 5 millimetri ed una corona a 76 denti per mini moto, utilizzando 5 viti tipo M5 (**Fig. A**). Per il fissaggio delle ruote vengono utilizzati due perni in acciaio del diametro di 20 millimetri con la parte finale filettata per consentire il bloccaggio delle ruote mediante dadi autobloccanti. La pedana in alluminio va fissata centralmente sopra il profilato.



I due motori sono montati sotto la pedana, lateralmente, utilizzando due piastre di supporto; il moto viene trasmesso mediante due catene che collegano i pignoni dei motori alle rispettive corone dentate.

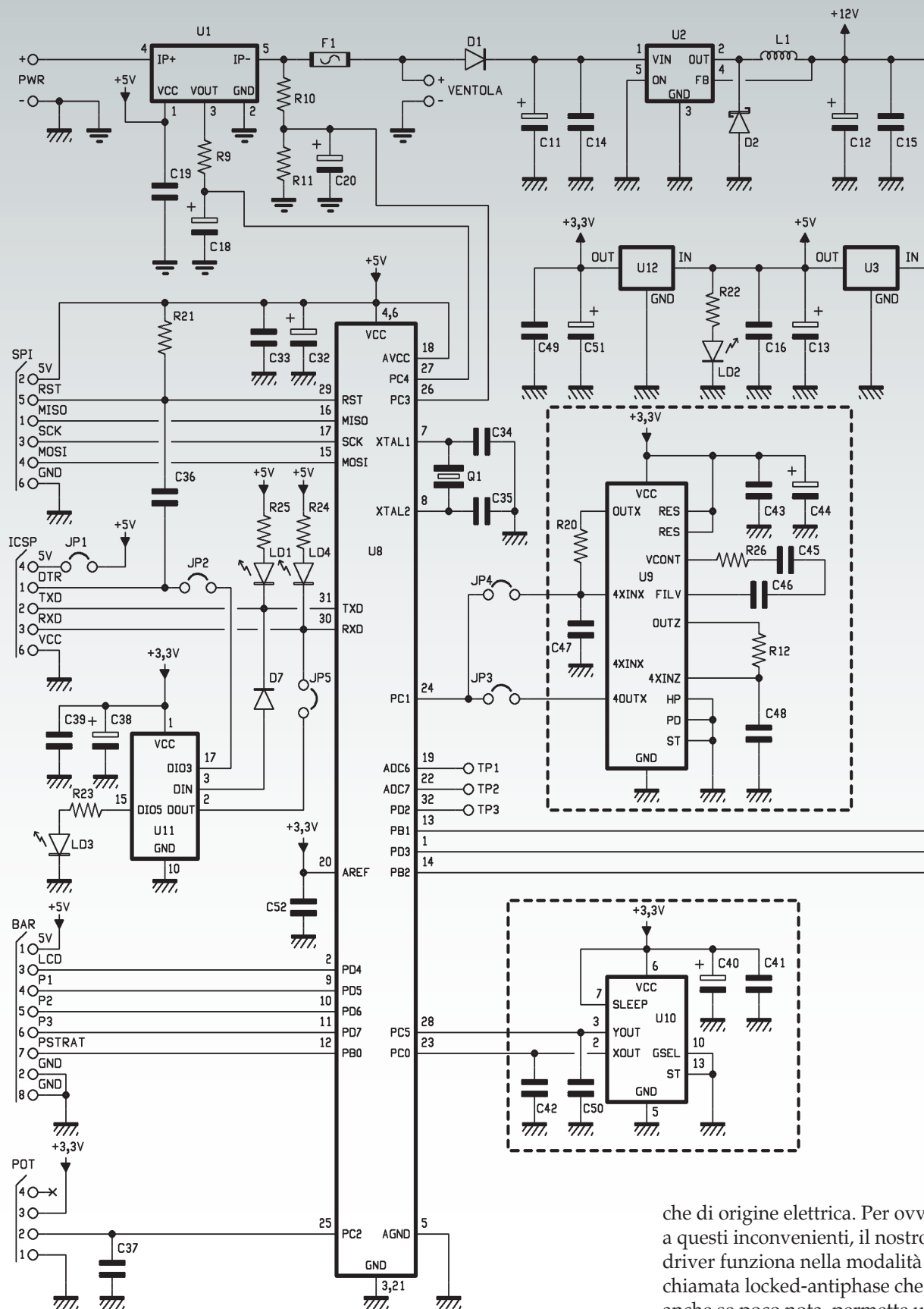
Il nostro veicolo dispone di un manubrio costituito da un tubolare d'alluminio che può inclinarsi lateralmente; per far sì che lo spostamento del manubrio produca un segnale in grado di controllare la centralina elettronica di gestione, viene utilizzato un potenziometro da 10 kohm il cui perno è messo in rotazione dal movimento laterale del manubrio.

In testa all'asta è fissato il manubrio vero e proprio che non è altro che un pezzo di tubolare di alluminio dello stesso tipo dell'asta, lungo circa 44 centimetri. Il nostro veicolo Sotto A questo punto la struttura meccanica è praticamente finita, proviamo a tenerla in posizione verticale e vediamo cosa succede.

Il nostro veicolo utilizza due batterie da 12V 20Ah collegate in serie in modo da ottenere la tensione di 24 volt adatta ad alimentare i due motori. Ciascuna batteria (del peso di circa 7 kg) è fissata mediante due piattine sotto la pedana, in posizione centrale, a partire dal profilato d'alluminio centrale.

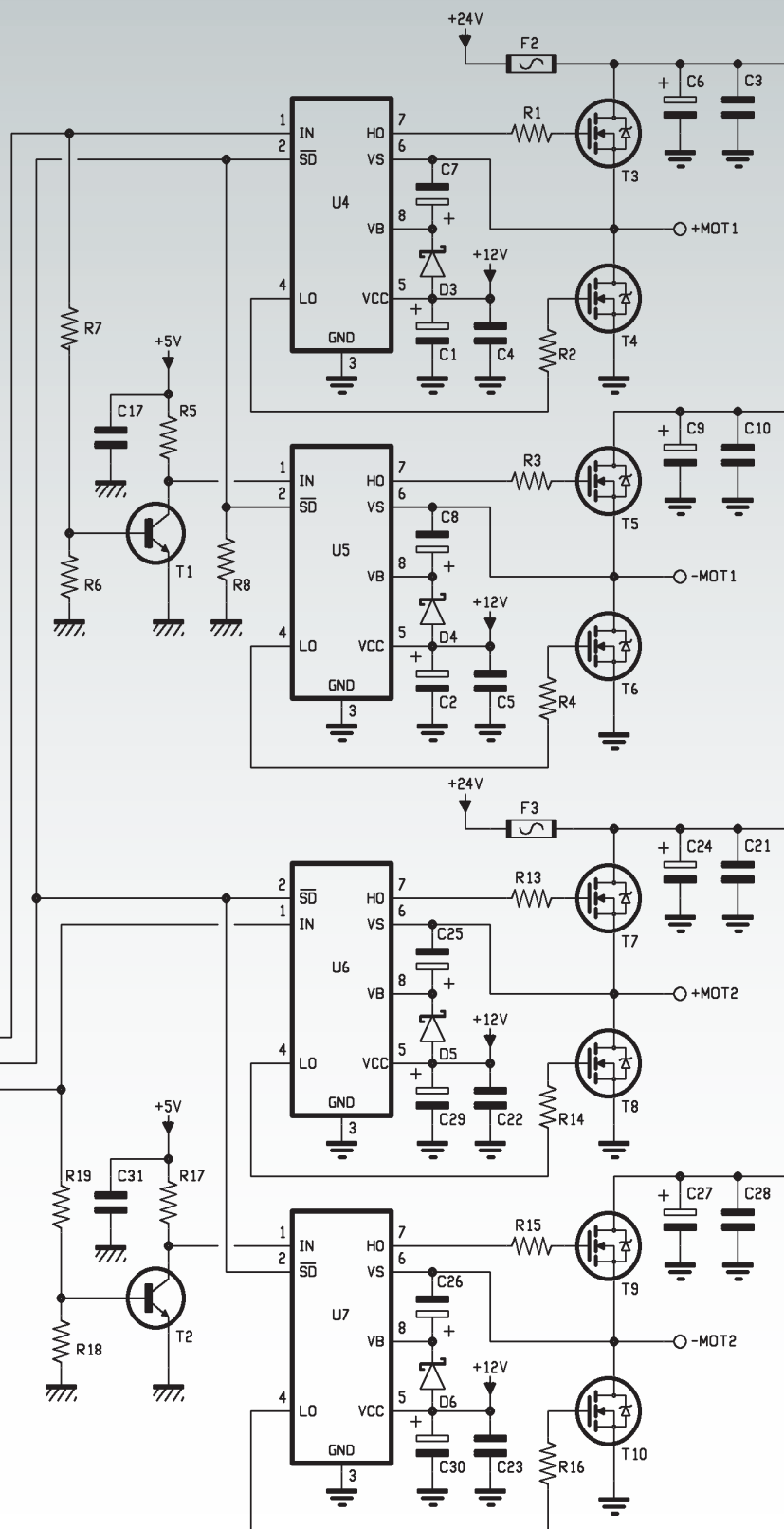
Montate le batterie il veicolo rimane magicamente "in piedi" per effetto del baricentro che si trova sotto al centro di rotazione delle ruote.

Se proviamo a salire sul veicolo rischiamo di farci del male: il nostro peso, infatti, riporta sopra il centro di rotazione il baricentro e pertanto il sistema non risulta più in equilibrio.



che di origine elettrica. Per ovviare a questi inconvenienti, il nostro driver funziona nella modalità chiamata locked-antiphase che, anche se poco nota, permette un funzionamento ottimale in questo genere di applicazioni. Per capire





## Elenco Componenti:

|                                         |                                         |
|-----------------------------------------|-----------------------------------------|
| R1 ÷ R4: 4,7 ohm (0805)                 | C37: 220 nF ceramico (0805)             |
| R5, R6: 1 kohm (0805)                   | C38, C40: 10 µF 35 VL elettrolitico (B) |
| R7: 4,7 kohm (0805)                     | C39, C41: 100 nF ceramico (0805)        |
| R8, R9: 2,2 kohm (0805)                 | C42: 3,3 nF ceramico (0805)             |
| R10: 47 kohm (0805)                     | C43, C47 ÷ C49: 100 nF ceramico (0805)  |
| R11, R13 ÷ R16: 4,7 kohm (0805)         | C44: 10 µF 35 VL elettrolitico (B)      |
| R12: 33 kohm (0805)                     | C45: 470 nF ceramico (0805)             |
| R17, R18: 1 kohm (0805)                 | C46: 10 nF ceramico (0805)              |
| R19: 4,7 kohm (0805)                    | C50: 3,3 nF ceramico (0805)             |
| R20: 33 kohm (0805)                     | C51: 10 µF 35 VL elettrolitico (B)      |
| R21: 10 kohm (0805)                     | C52: 100 nF ceramico (0805)             |
| R22 ÷ R25: 1 kohm (0805)                | D1: GF1M                                |
| R26: 10 kohm (0805)                     | D2 ÷ D6: BAT42W                         |
| C1, C2: 10 µF 35 VL elettrolitico (B)   | D7: GF1M                                |
| C3 ÷ C5: 100 nF ceramico (0805)         | T1, T2: BC817                           |
| C6: 1000 µF 35 VL elettrolitico         | T3 ÷ T10: AUIRF2804S                    |
| C7, C8: 10 µF 25 VL tantalio (A)        | Q1: Quarzo 16 MHz                       |
| C9: 1000 µF 35 VL elettrolitico         | U1: ACS758LCB-050U-PFF                  |
| C10: 100 nF ceramico (0805)             | U2: LM2576HVS-12/NOPB                   |
| C11: 100 µF 35 VL elettrolitico (F)     | U3: LM2937IMP-5.0/NOPB                  |
| C12: 220 µF 25 VL elettrolitico (F)     | U4 ÷ U7: IRS2184S                       |
| C13: 10 µF 35 VL elettrolitico (B)      | U8: ATMEGA328-AU                        |
| C14 ÷ C17: 100 nF ceramico (0805)       | U9: LPR403 o WRL-10415                  |
| C18: 10 µF 50 VL elettrolitico (B)      | U10: MMA7361                            |
| C19, C21 ÷ C23: 100 nF ceramico (0805)  | U11: XBEE1MWCHIP                        |
| C20: 10 µF 35 VL elettrolitico (B)      | U12: TC1262-3.3VDB                      |
| C24, C27: 1000 µF 35 VL elettrolitico   | L1: Induttanza 470 µH 850 mA            |
| C25, C26: 10 µF 25 VL tantalio (A)      | F1: Fusibile 1A                         |
| C28: 100 nF ceramico (0805)             | F2, F3: Fusibile 10A                    |
| C29, C30: 10 µF 35 VL elettrolitico (B) | LD1, LD2: LED verde (0805)              |
| C31: 100 nF ceramico (0805)             | LD3, LD4: LED rosso (0805)              |
| C32: 10 µF 35 VL elettrolitico (B)      |                                         |
| C33, C36: 100 nF ceramico (0805)        |                                         |
| C34, C35: 22 pF ceramico (0805)         |                                         |

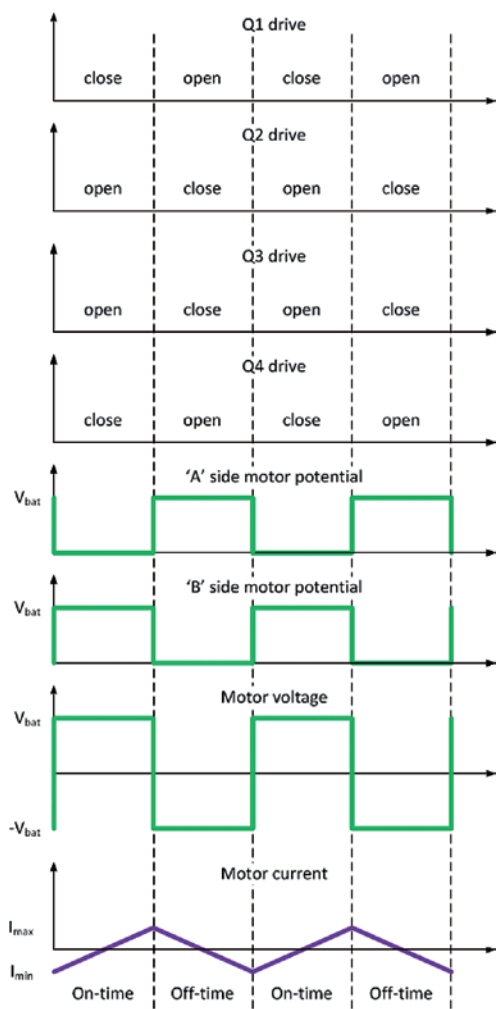
### Varie:

- Porta fusibile da CS
- Morsetteria 2 poli passo 10 mm (3 pz.)
- Morsetteria 2 poli passo 5 mm (1 pz.)
- Strip maschio 2 poli
- Strip maschio 3 poli (2 pz.)
- Strip maschio 4 poli
- Strip maschio 6 poli
- Strip maschio 8 poli
- Strip femmina 5 poli (4 pz.)
- Strip femmina 10 poli passo 2 mm (2 pz.)
- Jumper
- Circuito stampato

in cosa consiste questa modalità, osserviamo la Fig. 6 che riporta la classica configurazione a ponte H di un driver di potenza per motori

a spazzole in CC. Durante il funzionamento la coppia di MOSFET Q1 e Q4 sono ON alternativamente alla coppia Q2 e Q3: in pratica, è

come se alimentassimo alla massima tensione prima in un verso e poi nell'altro. Se il carico fosse costituito da una lampadina essa sa-



**Fig. 7** - Tensioni e correnti con duty-cycle=0.

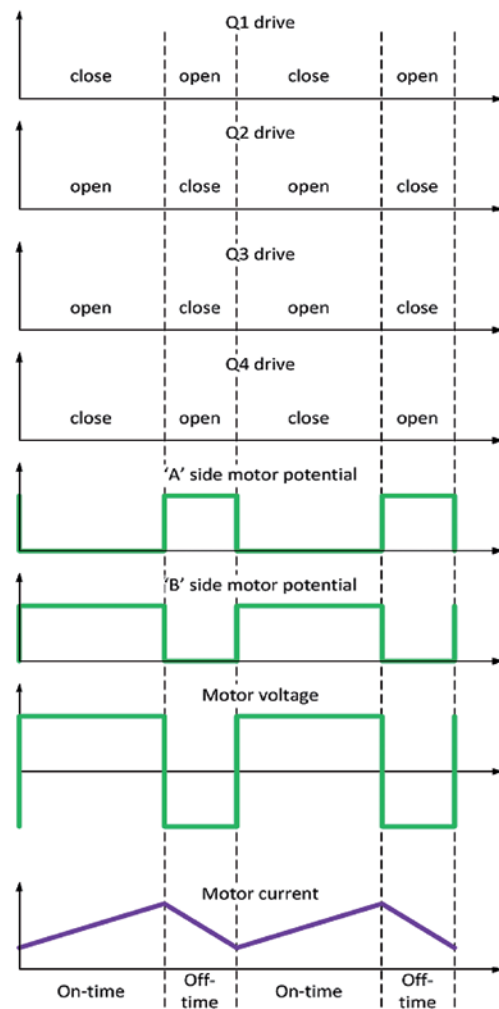
rebbe sempre accesa alla massima potenza indipendentemente dal duty cycle con il quale comandiamo le due coppie di MOSFET. Un motore elettrico si presenta elettricamente come un carico composto da una componente resistiva più una componente induttiva (R-L serie) e quindi anche alimentandolo istantaneamente in una direzione la sua corrente impiega del tempo ad aumentare secondo la costante di tempo determinata dal rapporto  $L/R$ ; l'andamento delle tensioni e delle correnti è riportata in Fig. 7. Con un duty cycle del 50% il motore viene alternativamente alimentato con  $+V_{cc}$  e  $-V_{cc}$ , con un valore medio nullo, mentre la corrente assume un andamento ad

onda triangolare anch'essa con valore medio nullo, di conseguenza il motore non gira in alcun senso. Con un duty cycle diverso da zero la tensione media ai capi del motore risulta ovviamente diversa da zero, così come la corrente, ed il motore risulta alimentato come indicato in Fig. 8.

La velocità di rotazione dipende dalla tensione di alimentazione che a sua volta è direttamente proporzionale al duty-cycle con il quale comandiamo il driver.

Come potete vedere il comando dei MOSFET non presenta un carattere impulsivo e nel caso di motore fermo è semplicemente un'onda quadra.

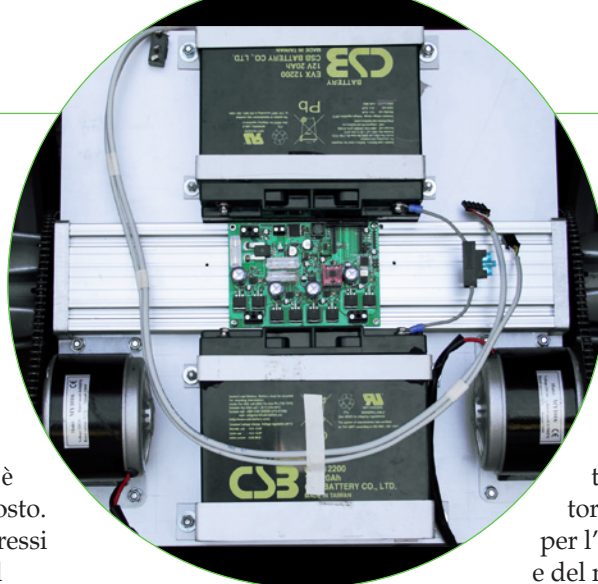
Per realizzare un driver con queste caratteristiche abbiamo utilizzato un circuito integrato specifico per il controllo di MOSFET di potenza, il modello IRS2184, al quale abbiamo abbinato quattro MOSFET per impieghi automotive codice AUIRF2804. I MOSFET scelti supportano tensioni sino a 40 volt con correnti di oltre 200 ampere, più che sufficienti per il nostro scopo; non fatevi ingannare da un valore così elevato nella massima corrente gestibile perchè il limite nel funzionamento sarà determinato dalla dissipazione in potenza sul MOSFET, dovuto alla sua resistenza di conduzione ed alle perdite durante la fase di commutazione. La potenza dissipata durante la fase di conduzione (ON) è determinata dalla formula  $P = R_{DS(on)} \cdot I^2$  e quindi dipende in maniera significativa dal valore della corrente ma anche dalla resistenza interna che per il nostro componente vale tipicamente 1,5 mΩ. L'integrato IRS2184 è progettato per gestire i due MOSFET di ciascun ramo del ponte H, con l'introduzione anche di un brevissimo ritardo durante la commutazione dal MOSFET sopra a quello sotto. La gestione del MOSFET posto sopra avviene



**Fig. 8** - Tensioni e correnti con duty-cycle>0.

in modalità flottante con una tecnica chiamata bootstrap. Questa particolare tecnica consente di usare entrambi MOSFET a canale N con il MOSFET in alto che viene gestito con una tensione di gate più alta di quella di alimentazione, ottenuta tramite un sistema a pompa di carica tramite un diodo e un condensatore. Questa tecnica consente l'impiego di MOSFET più performanti e meno costosi ma per contro impedisce il corretto funzionamento per valori di duty-cycle inferiori al 5% e superiori al 95% in quanto in queste condizioni il circuito di bootstrap non funziona correttamente. Questa configurazione per il driver è perfetta per un controllo in modalità locked-

antiphase. Con due integrati IRS2184 e quattro MOSFET si realizza un completo ponte H idoneo alla gestione in quattro quadranti di un motore; un driver è comandato direttamente con il segnale PWM mentre l'altro è comandato con il segnale opposto. I due transistor posti sugli ingressi dei driver servono appunto ad ottenere l'inversione del segnale PWM di ingresso, come richiesto nella modalità locked-antiphase. In totale, con soli due segnali PWM gestiamo entrambi i driver e conseguentemente i due motori; un terzo segnale indicato con il nome di shutdown viene usato per disabilitare i driver quando OpenWheels è in standby. Infatti, utilizzando la modalità locked-antiphase, anche a motori fermi vi è comunque un minimo assorbimento di corrente, inoltre vi è la necessità di lasciare alimentata l'intera sezione di controllo mentre, ad esempio, si sposta manualmente OpenWheels. Con i driver in shutdown tutti i MOSFET sono nello stato di OFF e le ruote possono ruotare liberamente permettendo di trascinare con semplicità OpenWheels. Particolare attenzione è stata posta nella corretta scelta della frequenza del segnale PWM usato per gestire i driver, in considerazione di molteplici fattori che ora andiamo ad elencare. Come descritto sopra, nella modalità locked-antiphase la corrente nei motori presenta una certa oscillazione (varia linearmente tra un valore massimo ed uno minimo) tanto maggiore quanto più bassa è la frequenza del PWM e quanto più bassa è l'induttanza del motore. Per evitare che la corrente presenti un'ondulazione eccessiva è necessario che la frequenza del PWM sia abbastanza elevata rispetto alla costante di tempo elettrica del motore; per impieghi sperimentali è bene rispettare la seguente equa-



*Come si vede nell'immagine, la scheda di controllo è fissata nella parte inferiore del telaio di OpenWheels.*

zione:  $f_{PWM} > 10 \cdot R_m / L_m$  dove  $R_m$  e  $L_m$  rappresentano le componenti resistive e induttive del motore. La gestione di un motore con un segnale PWM introduce anche delle vibrazioni sugli avvolgimenti interni che si traducono in un sibilo udibile in modo fastidioso se stiamo lavorando a frequenze audio. Entrambe queste considerazioni ci portano a scegliere una frequenza per il PWM superiore ai 10-12 kHz, limite oltre il quale la sensibilità uditiva umana cala rapidamente. Non possiamo però salire troppo con la frequenza a causa delle inevitabili perdite che si inducono all'interno del nucleo magnetico del motore, quando è interessato da campi magnetici fortemente variabili. Lo stesso driver presenta delle perdite specifiche dovute alla fase di commutazione, durante la quale i MOSFET passano dallo stato di OFF allo stato di ON e viceversa.

I quattro integrati IRS2184 necessitano di un'alimentazione a 12 volt che ricaviamo direttamente dalla tensione della batteria principale a 24 volt tramite un alimentatore switching di tipo step-down basato sull'integrato LM2576. L'elevato rendimento di questo tipo di alimentatori ci risparmia la necessità di un voluminoso dissipatore di calore e limita il consumo di corrente. Questo integrato funziona alla frequenza fissa di 50 kHz e con i valori da noi scelti il ripple in uscita risulta inferiore

ai 4mVrms. Da questa prima tensione regolata, tramite uno stabilizzatore LDO, ricaviamo i 5 volt per l'alimentazione della logica e successivamente tramite un secondo regolatore LDO ricaviamo i 3,3 volt per l'alimentazione dei sensori e del modulo Xbee. Sulla linea di alimentazione a 5 volt è presente un LED usato come indicazione di corretta alimentazione. A completare l'hardware abbiamo previsto un partitore di tensione necessario per la misura della tensione della batteria di alimentazione e un sensore di corrente basato sull'integrato ACS758. Quest'ultimo sfrutta l'effetto Hall per misurare la corrente senza apprezzabili cadute di tensioni, come invece necessario nella tecnica con shunt. In uscita avremo una tensione proporzionale alla corrente secondo la scala interna dell'integrato, permettendo misure sino a 50 ampere. L'indicazione della corrente assorbita verrà successivamente monitorata e resa disponibile per l'analisi dei consumi ma sarà usata anche quale sistema di protezione, permettendo di spegnere il driver di potenza non appena si rilevi un assorbimento anomalo di corrente. Questo sistema presenta però un certo ritardo e quindi sullo schema sono presenti due fusibili posti sull'alimentazione di ciascun driver pronti ad intervenire nel caso di brutali corto circuiti sui motori. Un terzo fusibile è a protezione della sezione di comando, sempre per evitare danni dovuti ad eventuali corto circuiti sulla scheda. Dopo una prima versione del circuito OpenWheels basata su una scheda Arduino e schedine separate per sensori e driver, siamo passati alla realizzazione finale tutta con componenti in SMD, in modo da ridurre costi ed ingombri.



La scheda definitiva è ancora basata sul chip di Arduino in versione SMD ovvero l'integrato ATMEGA 328A-U, sfruttato appieno per quanto riguarda l'hardware ma anche nel software. Abbiamo previsto nella scheda il connettore standard di Arduino per la programmazione in-circuit necessaria a caricare il bootloader mentre la programmazione avverrà tramite la porta seriale del chip utilizzando un convertitore USB-seriale (codice Futura 7300-FTDI5V) per la connessione lato PC; sarà a tutti gli effetti come programmare una scheda Arduino duemilanove. Anche il modulo XBee impegna la porta seriale per cui, per evitare conflitti, abbiamo previsto un piccolo jumper da inserire quando si desidera abilitare il modulo XBee; il jumper andrà rimosso durante la fase di programmazione del chip Atmel. Abbiamo inserito anche un piccolo jumper a saldare che connette il pin DIO3 di XBee con la linea di RESET dell'Atmel con lo scopo, per chi lo desidera, di abilitare la programmazione wireless (da remoto) del chip. Questa modalità è molto interessante perchè permette di programmare l'Atmel creando una comunicazione trasparente tra il PC ed il chip tramite due moduli XBee; in questo caso la linea DIO3 svolge la

funzione di reset automatico per l'Atmel. Questa modalità implica però una programmazione specifica per i due moduli XBee.

Un apposito connettore permette di collegare il potenziometro per la misura dell'inclinazione del manubrio, mentre un secondo connettore porta i segnali verso l'asta del manubrio nel quale sono presenti alcuni pulsanti ed un display LCD. Sul manubrio è stata prevista la presenza di un pulsante di start necessario per mettere in funzione OpenWheels e rendere operativi i motori che altrimenti rimangono in shutdown; questo circuito ha anche lo scopo di spegnere il veicolo a seguito di un'eventuale caduta; togliendo le mani dal manubrio, infatti, i motori vengono disabilitati; spiegheremo più in dettaglio questa funzione quando descriveremo il firmware. Sul manubrio è presente un display LCD di tipo seriale che visualizza tutte le informazioni di funzionamento ed eventuali stati di anomalia. OpenWheels è configurabile da remoto, ma chi volesse risparmiare sui moduli XBee preposti alla telemetria potrà usare il display LCD ed i pulsanti del manubrio per impostare i parametri di funzionamento. Sullo stampato sono presenti anche i jumper JP3 e JP4 che permettono di selezionare la

portata del giroscopio. Il jumper JP2 - se cortocircuitato - permette di abilitare la programmazione del chip Atmel da remoto tramite il modulo XBee. Infatti l'uscita DIO3 del modulo XBee può essere programmata per far da reset al chip Atmel durante la programmazione, funzione normalmente svolta dalla linea DTR del convertitore USB/seriale. La programmazione da remoto è possibile solo dopo aver adeguatamente programmato i due moduli XBee preposti alla comunicazione seriale.

Sui prossimi fascicoli della rivista continueremo la descrizione dell'hardware e presenteremo il firmware da noi messo a punto. **Per questo progetto abbiamo realizzato un minisito** (<http://openwheels.elettronica.in.it>) nel quale vengono descritti in maniera molto più dettagliata i particolari costruttivi del nostro veicolo, le modifiche apportate nel tempo e quelle suggerite dai lettori, nonché i circuiti di controllo proposti dai lettori. Lo scopo ultimo è quello di realizzare un veicolo perfettamente funzionante ad un costo di gran lunga inferiore rispetto a quello di analoghi prodotti disponibili in commercio nonché quello di poter personalizzare il veicolo in relazione alle proprie esigenze. ■

# RM ELETTRONICA

## Forniture per hobbisti, scuole e industria

Distributore autorizzato:

**FUTURA  
ELETTRONICA**

**ELSE  
Kit**

**NUOVA  
ELETTRONICA**

Il tuo punto  
di riferimento  
per l'elettronica  
a ROMA

Via Val Sillaro 38 • 00141 Roma • Tel: 06-8104753

Figgo è progettato per essere un semplice specchio, ma con un po' di elettronica diventa un affascinante orologio luminoso.



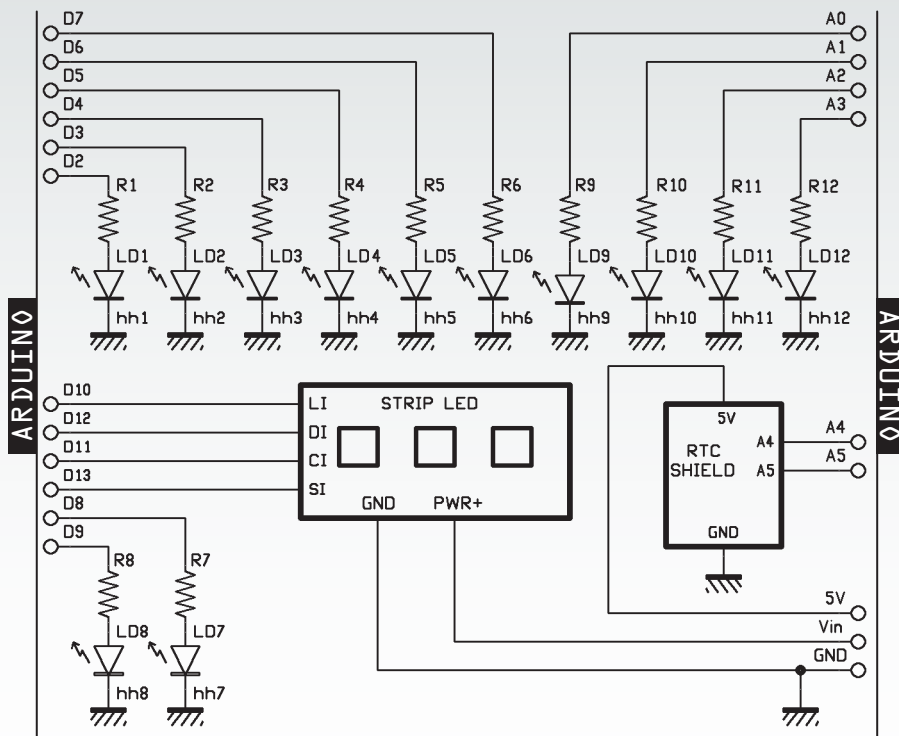
## STAR CLOCK: DESIGN IKEA, TECNOLOGIA ARDUINO

..... di BORIS LANDONI

**O**gni volta che si fa un giro in un negozio Ikea, è difficile non essere colpiti da qualche oggetto, suppellettile o accessorio che, per la sua forma o le sue caratteristiche, fa scattare qualche impulso creativo. Un vecchio adagio recita "la bellezza è negli occhi di chi guarda" e noi vogliamo parafrasarlo in "l'applicazione è nella testa di chi osserva". Figgo è uno di quegli specchi che dicono Ikea lontano un miglio: design funziona-

le, estetica ricercata ma non troppo, forme pulite combinate a materiali resistenti, montaggio solido, ma spazi e strutture che si prestano ad accogliere nuovi componenti. Insomma: lo abbiamo guardato e lui ci ha fatto intravedere una sua applicazione in abbinamento con una striscia LED RGB e altri led singoli. Cosa vi viene in mente guardando uno specchio tondo dal quale partono dodici raggi che terminano in altrettanti piccoli

## [schema **ELETTRICO**]

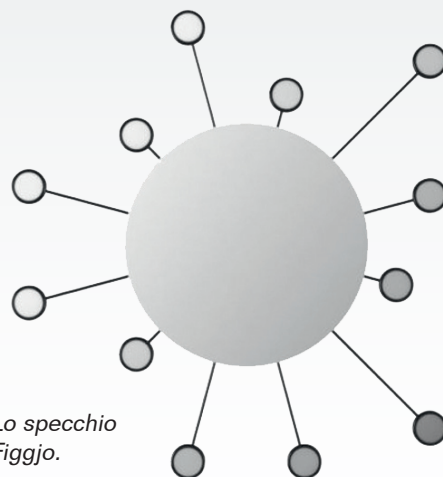


### Elenco Componenti:

R1 ÷ R12: 100 ohm  
LD1 ÷ LD12: LED alta luminosità 5 mm rosso

Varie:

- Shield RTC (cod. RTC\_SHIELD)
- Arduino Uno (cod. ARDUINOUNOREV3)
- Strip indirizzato (cod. STRIPRGBHL1606)
- Alimentatore 5Vdc/2A



cerchi, spazati in modo omogeneo? Esatto! Un orologio. I dodici raggi con i piccoli specchietti tondi saranno le nostre 12 ore, mentre la circonferenza del nostro specchio diventerà una corona su cui far scorrere i secondi e i minuti, sotto forma di LED di diverso colore illuminati come striscia continua e come elemento singolo (secondi e minuti).

### LED E STRISCE

Circa un anno fa vi abbiamo presentato le strisce LED basate su controller HL1606, con due RGB SMD di tipo 5050 per ciascun chip e 32 LED a metro. Facendo due conti, 30 LED sono quindi poco meno di un metro di striscia che si può "arrotolare" su una circonferenza di circa 30 centimetri di diametro e questo è proprio quello che serve a noi per avere la striscia nascosta dietro allo specchio, ma non troppo. Se volessimo 60 LED avremmo bisogno di quasi due metri di striscia che si arrotolano su una circonferenza

di quasi 60 centimetri: troppo per il diametro del nostro specchio Figgjo.

Dovremo quindi usare solo 30 LED per indicare i minuti e i secondi, giocando su punto e riga per differenziare fra i due; per il colore, invece, possiamo anche lasciare che sia il caso a definirlo ad ogni ora. Vedremo come realizzare tutto questo nello sketch di Arduino.

### LO SHIELD RTC

Ci sono vari modi per avere un orologio con Arduino e fra i più semplici c'è lo shield di Futura Elettronica denominato RTCSHIELDKIT. Questo è basato sull'integrato DS1307 di Maxim-Dallas, che è un Real Time Clock in grado di liberare la CPU dal calcolo e dalla gestione dei dati orari. L'integrato DS1307 è un contatore BCD (Binary Coded Decimal) a basso consumo, che conta secondi, minuti, ore, giorni, mesi e anni, provvisto di 56 byte di RAM statica non volatile. Può operare

nelle modalità 12 o 24 ore, con indicazione delle ore antimeridiane (AM) e di quelle pomeridiane (PM). Le informazioni sull'ora e la data vengono collocate in un apposito registro e trasferite al microcontrollore di Arduino mediante l'I<sup>2</sup>C-bus. Il bus fa capo ai piedini 5 (SDA) e 6 (SCL) che combaciano con i pin SCL e SDA di Arduino (UnoRev3 e Mega-Rev3), il quale fa da unità master dell'I<sup>2</sup>C-Bus, mentre il DS1307 è lo slave.

Per le versioni di Arduino precedenti ad Arduino Uno Rev3 (UnoRev2 o Arduino2009), i segnali SCL e SDA sono disponibili sui pin A4 e A5, per rendere lo shield compatibile anche con queste versioni, sul lato rame sono presenti delle piazzole a saldare.

### IL PROGETTO PRENDE FORMA

Ricapitolando, abbiamo bisogno del nostro specchio, un metro di striscia LED RGB intelligente con 30 LED, da pilotare con tre linee dati secondo lo standard SPI, uno



shield che ci fornisce il valore di data e ora – anche se a noi serve solo l'ora – da pilotare con l'PC, e una dozzina di LED per segnare le ore illuminando a rotazione ciascuno dei dodici piccoli specchi che orbitano attorno allo specchio principale.

Per mantenere il tutto alla sua massima semplicità, non faremo alcun multiplexing dei LED delle ore e li collegheremo direttamente ai PIN di Arduino da D2 a D9, ricorrendo anche a quattro pin solitamente dedicati all'ingresso analogico (da A0 ad A3) e che verranno riconfigurati come uscite digitali tramite l'istruzione pinMode.

Lo shield RTC dialoga su SDA (A4) ed SCL (A5) come indicato sopra, mentre per pilotare la striscia LED in SPI usiamo i pin D10, D11 e D12.

Abbiamo praticamente occupato quasi tutti i pin disponibili, ma questo non è un problema dato che non ci serve nient'altro. Non aggiungiamo controlli a pulsante perché non ce n'è bisogno grazie al sistema di regolazione automatica di data e ora del RTC nel caso ce ne sia bisogno, altrimenti la sua precisione è tale da garantire un funzionamento senza problemi per parecchio tempo.

Vediamo ora con maggiore dettaglio la struttura elettrica ed elettronica del progetto.

Arduino, nelle sue varie versioni compatibili con l'attuale versione UNO, non ha bisogno di commenti e basta che ve ne procuriate uno, magari recuperandolo da un progetto precedente di cui vi siete ormai stufati (Arduino è bello anche perché può essere facilmente recuperato dai progetti precedenti senza fatica).

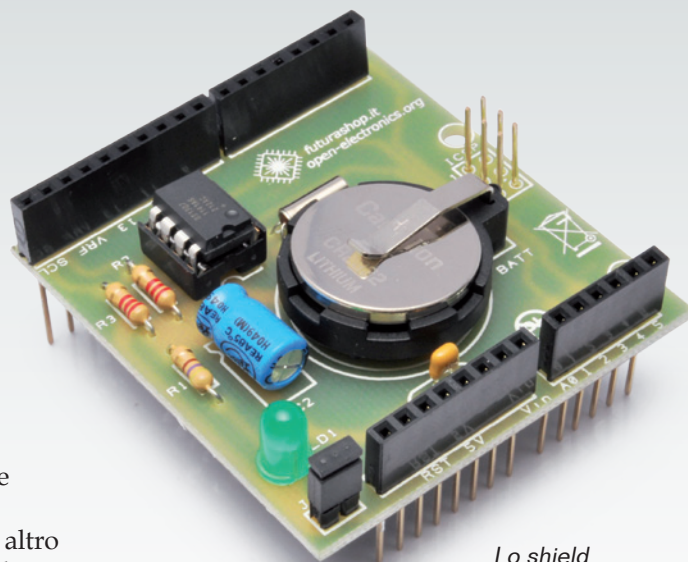
Lo shield RTC va inserito sopra Arduino dopo aver provveduto al suo montaggio se non l'avete già fatto in precedenza.

Per questa applicazione sono necessari quasi tutti i vari collegamenti, dai connettori inferiori ai pin strip superiori, dove dovremo attestare una serie di fili. Se l'RTC è recuperato da un altro progetto, probabilmente sarà già regolato su data e ora, sempre che non lo abbiate privato della sua batteria tampone.

La striscia LED va tagliata a 30 chip RGB: se avete acquistato un metro vi avanzeranno due LED – che potreste anche lasciare collegati – mentre se avete la versione da 5 metri, partite dal capo a cui sono già collegati i fili e vi eviterete anche la fatica di doverne saldare di nuovi. Il codice colore dei singoli fili vi servirà per individuare le linee da collegare ad Arduino e all'alimentazione.

I LED delle ore sono con il catodo a massa e l'anodo collegato ciascuno a un pin di Arduino come segue:

- D2 = ore 1
- D3 = ore 2
- D4 = ore 3
- D5 = ore 4



Lo shield  
RTCShieldKIT.

- D6 = ore 5
- D7 = ore 6
- D8 = ore 7
- D9 = ore 8
- A0 = ore 9
- A1 = ore 10
- A2 = ore 11
- A3 = ore 12

Ciascun diodo LED è collegato quindi con il catodo a massa e l'anodo al proprio pin dedicato su Arduino; in teoria dobbiamo portare due fili per ciascun LED, ricordandoci anche di inserire una resistenza di limitazione della corrente, ma la struttura di Figgio ci viene in aiuto. La descrizione specifica infatti che è realizzato in acciaio, con strato di

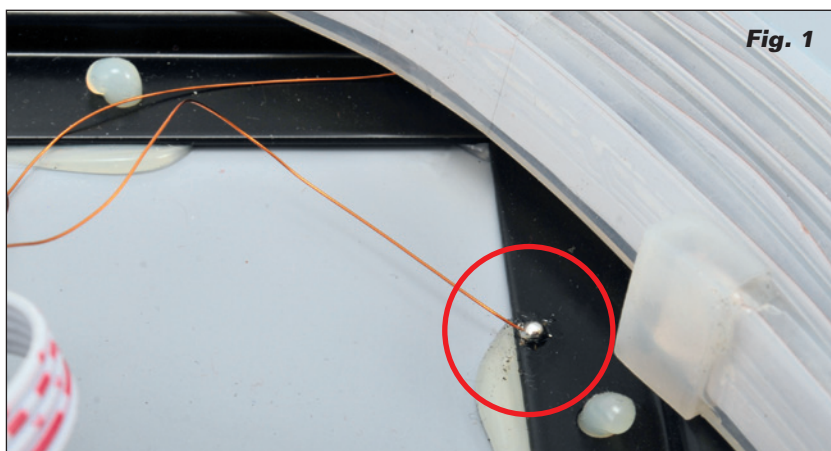


Fig. 1



fosfato antiruggine e rivestimento epossidico/poliestere a polvere, pigmentato. In pratica la struttura è metallica e conduttiva, quindi possiamo usarla come massa e portare ai led solo un filo, dato che il catodo potrà essere saldato direttamente alla struttura stessa. Nella Figura 1 si nota il filo che abbiamo saldato alla struttura metallica di Figgjo e che la mette a massa, essendo il filo collegato al pin GND di Arduino.

Dalla strip arrivano invece cinque fili come da tabellina seguente:

L in  
C in  
D in  
S in  
GND

Osservando la striscia noterete anche la presenza di un sesto collegamento, ovvero +5V, a cui è però saldato un filo di alimentazione attestato a una presa di tipo tondo, con positivo centrale.

Questo tipo di strip ha un proprio verso per il collegamento: a ogni piazzola si affaccia l'uscita dal blocco di due LED precedenti e nel caso di una strip tagliata da un pezzo più lungo, dovete collegare Arduino ai pin contrassegnati con la "i" che sta appunto per in, mentre dal lato opposto si trovano le piazzole con la stessa prima lettera, seguita da "O" per indicare che sono uscite. Ogni chip funziona da registro a scorrimento che invia al chip seguente i dati contenuti nella sua memoria ogni volta che riceve dei nuovi

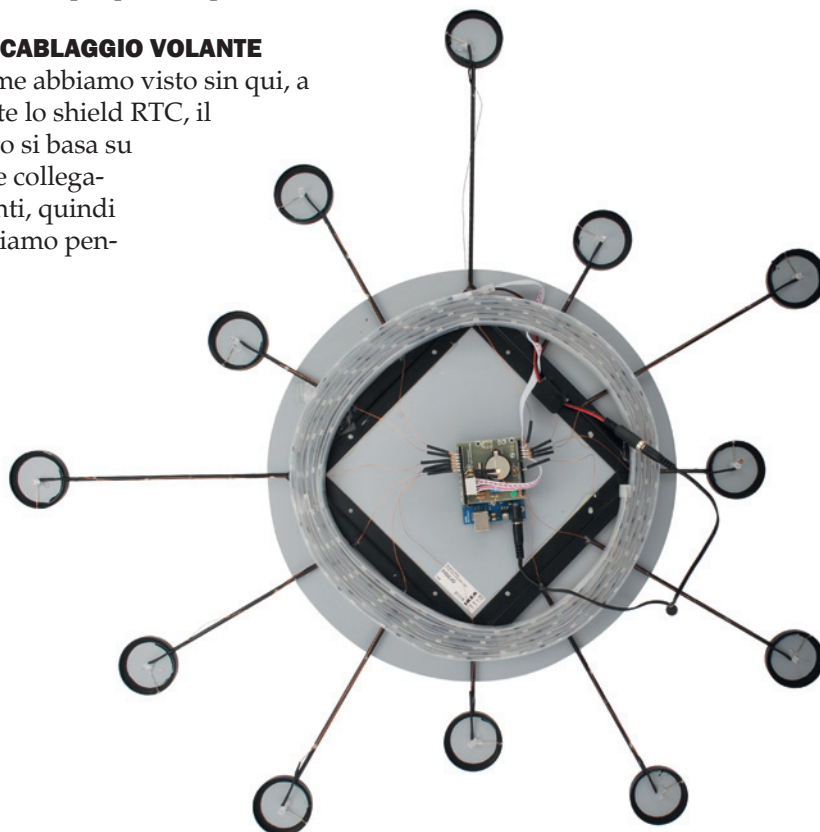
dati. Il segnale di clock provvede a gestire la temporizzazione dei trasferimenti, mentre il pin non collegato "S" sarebbe da utilizzare come segnale di step per attivare la funzione di fade in e fade out prevista dal chip HL 1606 e basata su PWM, dove a ogni impulso su S si ha l'incremento o il decremento dell'intensità del PWM con cui è attivato il LED. Questo sistema permette di avere la sincronizzazione delle sfumature sull'intera striscia in modo certo e ripetibile, ma rende la programmazione della striscia stessa un po' più complessa.

#### UN CABLAGGIO VOLANTE

Come abbiamo visto sin qui, a parte lo shield RTC, il resto si basa su fili e collegamenti, quindi abbiamo pen-

sato di proporre un montaggio basato sul collegamento diretto delle resistenze al filo isolato che porta a ciascun LED, mentre l'altro capo può essere inserito direttamente nel foro del pin strip corrispondente al segnale da utilizzare. Dalla foto si può vedere come questa soluzione sia, alla fine, molto pratica e veloce.

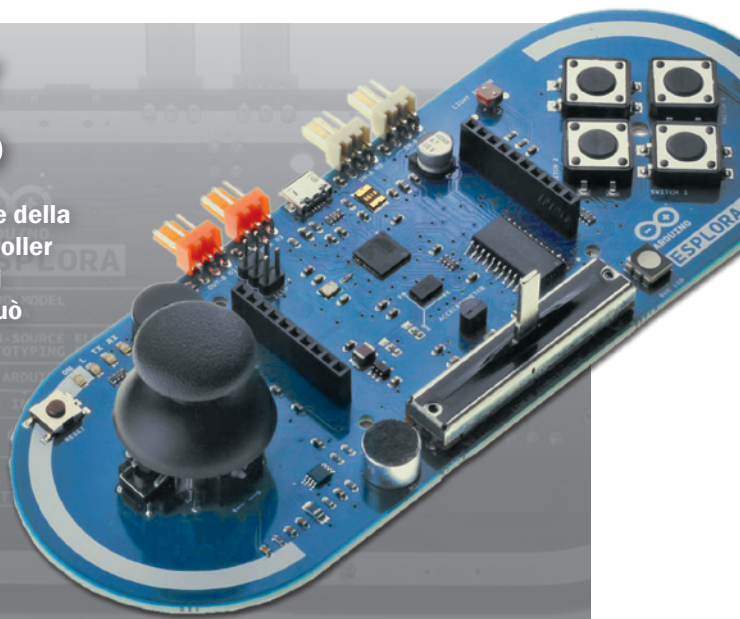
Ciascun filo (noi abbiamo usato del rame smaltato e isolato) va fatto passare nascondendolo dietro al tubetto che collega la struttura tonda principale allo specchio "orbitante" e il terminale





# Esplora: controller per giochi targato Arduino

Basato sulla board Leonardo, questa nuovo componente della famiglia Arduino consente di creare un sofisticato controller personalizzato per giochi, sistemi musicali, strumenti di modellazione 3D e tanto altro ancora. La scheda, che può emulare un mouse o una tastiera, ha la forma di una controller per giochi e dispone di un joystick, di un accelerometro a tre assi, di una serie di pulsanti, di un sensore di luce, uno di temperatura, di un LED RGB e di un buzzer. Il microcontrollore è lo stesso di Leonardo, ovvero un ATmega32u4 a 16 MHz con 32 KB di memoria FLASH. Nato per le sperimentazioni più ardite nel campo dei game controller, viene fornito con uno script pre-programmato che consente di giocare subito con Super Tux Cart. [www.arduino.cc](http://www.arduino.cc)



corto va saldato direttamente alla struttura, come visibile in figura 2 e 3. Per riuscire a saldare il catodo dietro a ciascuno specchietto è necessario dare una buona grattata alla superficie per rimuovere la polvere antiruggine e l'eventuale resina epossidica, fino a scoprire il metallo. Data la massa metallica da scaldare, dovrete ricorrere a un saldatore da almeno 40 watt, cercando comunque di essere veloci e precisi per non mettere sotto stress termico tutti gli elementi nei dintorni, LED incluso. Dopo aver fissato ciascun LED dietro al proprio specchietto, potete fissare i fili di collegamento sul tubetto metallico, eventualmente aiutandovi con qualche goccia di colla a caldo o pezzetto di nastro adesivo (figura 3). La resistenza di limitazione della corrente che abbiamo scelto si basa sull'impiego di LED blu; rispetto ai 5V forniti sulle uscite di Arduino, per limitare a circa 15 mA la corrente, la resistenza da utilizzare è da 82 ohm. Se usate LED di altri colori, ricordate di ricalcolare la resistenza per non superare i 15 mA di assorbimento. Ora che i 12 LED delle ore sono collegati, possiamo procedere con il cablaggio della striscia LED, già

tagliata a lunghezza. Se non è già presente, saldate una piattina a 5 fili dal lato della striscia partendo da GND e arrivando a LI. Dal lato di Arduino potete usare o un pin strip maschio o uno femmina al quale dovrete aggiungere cinque pin maschio/maschio in strip. Questo strip maschio va inserito a partire dal pin D10 e arriva fino al GND dello strip di Arduino. Il positivo dell'alimentazione fornita al circuito, rigorosamente di 5V regolati e stabilizzati, va inviata anche alla scheda Arduino usando uno spezzone di

filo isolato che parte dalla presa tonda della striscia LED (pin centrale, che corrisponde al positivo) e arriva sul pin strip di Arduino (tramite RTC SHIELD) dal lato degli ingressi analogici, in corrispondenza di 5V (tra GND e 3,3V). In questo caso state alimentando il microcontrollore direttamente, saltando i regolatori di tensione, quindi fate attenzione all'alimentatore che sceglierete. In alternativa collegate un secondo alimentatore alla scheda Arduino sfruttando la regolazione della scheda.

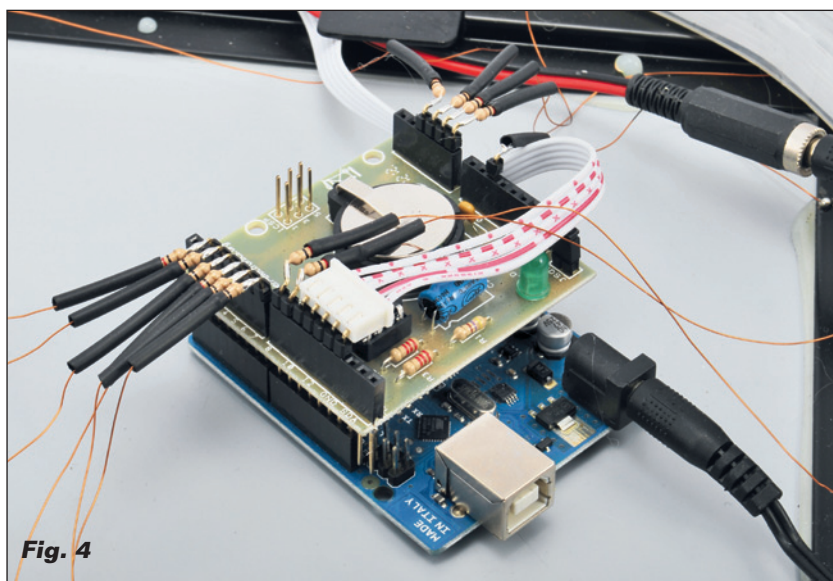


Fig. 4



## Sketch

```
#include <Wire.h>
#include "RTClib.h"

RTC_DS1307 RTC;

// HL1606strip is an adaptation of LEDstrip from http://code.google.com/p/ledstrip/
#include "HL1606strip.h"

#define STRIP_D 12    // i tre pin a cui è collegata la strip
#define STRIP_C 11
#define STRIP_L 10

// Per creare l'istanza della strip LED vanno indicati i PIN e il numero di LED
HL1606strip strip = HL1606strip(STRIP_D, STRIP_L, STRIP_C, 30);

const int h1 = 2;      // hour 1
const int h2 = 3;      // hour 2
const int h3 = 4;      // hour 3
const int h4 = 5;      // hour 4
const int h5 = 6;      // hour 5
const int h6 = 7;      // hour 6
const int h7 = 8;      // hour 7
const int h8 = 9;      // hour 8
const int h9 = A0;     // hour 9
const int h10 = A1;    // hour 10
const int h11 = A2;    // hour 11
const int h12 = A3;    // hour 12

int hours=0;
int minutes=0;
int seconds=0;
int ledcolor=0;
int mcolor=3;
int scolor=2;
boolean change=0;

boolean randomcolor=1;
boolean sbar=0;
boolean chghour=1; //Cambia la barra/punto ogni ora

void setup(void) {
  pinMode(h1, OUTPUT);
  pinMode(h2, OUTPUT);
  pinMode(h3, OUTPUT);
  pinMode(h4, OUTPUT);
  pinMode(h5, OUTPUT);
  pinMode(h6, OUTPUT);
  pinMode(h7, OUTPUT);
  pinMode(h8, OUTPUT);
  pinMode(h9, OUTPUT);
  pinMode(h10, OUTPUT);
  pinMode(h11, OUTPUT);
  pinMode(h12, OUTPUT);

  Serial.begin(9600);
  Wire.begin();
  RTC.begin();
  RTC.sqw(1); //0 Led off - 1 Freq 1Hz - 2 Freq 4096kHz - 3 Freq 8192kHz - 4 Freq 32768kHz
  if (! RTC.isrunning()) {
    Serial.println("Clock regulated");
    // Se l'RTC non sta funzionando, viene impostato
    // con data e ora di compilazione dello sketch
    RTC.adjust(DateTime(__DATE__, __TIME__));
  }
}

void loop(void) {
  readclock();
  stripOff();
  writeseconds();
  writeminutes();
  writehours();

  delay(200);
}

/***** Funzioni *****/

void readclock() {
```

(Continua)

## LO SKETCH

Dal punto di vista concettuale, il programma che pilota il nostro orologio deve fare ben poco: acquisire dall'RTC ore, minuti e secondi, quindi dividere minuti e secondi per due, e procedere alla visualizzazione dei tre valori. Per le ore va acceso il LED corrispondente, per i secondi la striscia si accende progressivamente, mentre per i minuti si accende un LED singolo sovrapposto ai secondi, se necessario.

La maggior parte della complessità risiede nella lettura di RTC tramite I<sup>2</sup>C e nella scrittura dei dati sulla striscia LED secondo il protocollo SPI.

Fortunatamente per entrambe le problematiche ci sono delle librerie dedicate che ci risolvono i problemi in modo pressoché completo.

La prima è dedicata al RTC clock e si chiama RTCLib.h e può essere scaricata da questo link: <http://www.futurashop.it/download/RTCLib.zip>. Si tratta di una versione modificata della libreria inizialmente sviluppata da Adafruit per la propria scheda RTC, visibile a questo indirizzo: <http://www.ladyada.net/learn/breakoutplus/ds1307rtc.html>. Grazie alla RTCLib sapere la data o l'ora diventa semplicissimo: una volta creata l'istanza del nostro orologio con "RTC\_DS1307 RTC", potremo sapere ore, minuti e secondi correnti passando dalla funzione "DateTime now = RTC.now()" in cui viene predisposta la lettura dei valori correnti ottenuti dal nostro shield inizializzato precedentemente. Con "now.hours", "now.minutes" e "now.seconds" otteniamo direttamente i valori per le elaborazioni successive, mentre merita una nota di commento la speciale routine che sfrutta il risultato della funzione "RTC.isrunning" per capire se è

## Sketch (seguito)

```
DateTime now = RTC.now();
hours=now.hour();

if (hours>12) hours=hours-12;

minutes=now.minute();
minutes = (minutes/2);

seconds=now.second();
seconds = (seconds/2);

Serial.print(now.hour(), DEC);
Serial.print(':');
Serial.print(now.minute(), DEC);
Serial.print(':');
Serial.print(now.second(), DEC);
Serial.println();

Serial.print(hours, DEC);
Serial.print(':');
Serial.print(minutes, DEC);
Serial.print(':');
Serial.print(seconds, DEC);
Serial.println();
Serial.println();

if ((now.second()==59)&&(now.minute()==59)&&(chghour==1)) {
  if (sbar==1) {
    sbar=0;
  }
  else
  {
    sbar=1;
  }
}

if ((now.second()==00)&&(randomcolor==1)&&(change==0)) {

  mcolor=color();
  scolor=color();
  while (mcolor==scolor){
    scolor=color();
  }
  change=1;
}

if ((now.second())==02) {
  change=0;
}

if ((now.second())%2==0) {
  strip.writeStrip();
  // Ogni due secondi viene aggiornata la strip
}

}

void writehours() {
  digitalWrite(h1, LOW);
  digitalWrite(h2, LOW);
  digitalWrite(h3, LOW);
  digitalWrite(h4, LOW);
  digitalWrite(h5, LOW);
  digitalWrite(h6, LOW);
  digitalWrite(h7, LOW);
  digitalWrite(h8, LOW);
  digitalWrite(h9, LOW);
  digitalWrite(h10, LOW);
  digitalWrite(h11, LOW);
  digitalWrite(h12, LOW);
  switch (hours) {
    case 1:
      digitalWrite(h1, HIGH);
      break;
    case 2:
      digitalWrite(h2, HIGH);
      break;
    case 3:
```

(Continua)

## Sketch (seguito)

```
digitalWrite(h3, HIGH);
break;
case 4:
digitalWrite(h4, HIGH);
break;
case 5:
digitalWrite(h5, HIGH);
break;
case 6:
digitalWrite(h6, HIGH);
break;
case 7:
digitalWrite(h7, HIGH);
break;
case 8:
digitalWrite(h8, HIGH);
break;
case 9:
digitalWrite(h9, HIGH);
break;
case 10:
digitalWrite(h10, HIGH);
break;
case 11:
digitalWrite(h11, HIGH);
break;
case 12:
digitalWrite(h12, HIGH);
break;
}
}

void writeminutes(){
strip.setLEDcolor(minutes, mcolor);
}

void writeseconds(){
if (sbar==1){
for (int i=0; i <= seconds; i++) {
strip.setLEDcolor(i, scolor);
}
}
strip.setLEDcolor(seconds, scolor);
}

void stripOff(void) {
// Imposta a 0 il colore di tutti i LED
for (uint8_t i=0; i < strip.numLEDs(); i++) {
strip.setLEDcolor(i, BLACK);
}
}

int color(){
int tmp= random(1,7);
switch (tmp){
case 1:
ledcolor= WHITE;
break;
case 2:
ledcolor= RED;
break;
case 3:
ledcolor= YELLOW;
break;
case 4:
ledcolor= GREEN;
break;
case 5:
ledcolor= TEAL;
break;
case 6:
ledcolor= BLUE;
break;
case 7:
ledcolor= VIOLET;
break;
}
return(ledcolor);
}
```

necessario regolare o meno l'ora. Se l'RTC non è ancora stato regolato con la funzione "adjust" ed è stato avviato, la funzione "isrunning" risulta falsa (viene letto dalla libreria il bit 7 "Clock Halt" del registro 0x00 del DS1307) e questo manda in esecuzione l'if che utilizza il timestamp dell'istante di compilazione per scrivere i corrispondenti valori numerici - sostituiti dall'IDE in fase di compilazione alle etichette - sul chip RTC come valori di partenza. Con questo accorgimento, avremo una regolazione dell'orologio molto vicina a quella del computer ospite su cui gira l'IDE. Per quanto riguarda la strip, utilizziamo la libreria HL1606strip.h, disponibile su <http://code.google.com/p/ledstrip/>, dove basta definire i pin collegati alle linee di Arduino e il numero di LED attivi nella striscia per disporre di semplici funzioni per la sua gestione. In questa libreria, la funzione setLEDcolor (posizione, colore) permette di definire il colore di ciascun LED fra gli otto disponibili (nero, bianco, rosso, giallo, verde, azzurro, blu e viola); la scrittura del colore di ciascun LED e la trasmissione delle informazioni alla striscia sono due funzioni separate e grazie a questo il software può essere strutturato per definire in momenti diversi il colore di ciascuno dei LED per poi scrivere e visualizzare il risultato finale in un solo istante. La velocità con cui si gestisce la scrittura dei vari colori per i LED sarà poi quella che determina la frequenza con cui si può rinfrescare l'intera striscia. Per come funziona il sistema con HL1606, è necessario inviare i dati a tutti i chip attivi della striscia per poter cambiare anche un solo colore di un singolo LED, ma di questo si occupa la libreria che mantiene in memoria un array

corrispondente all'intera striscia e che viene inviata ogni volta che si usa la funzione "writeStrip". Con queste informazioni, possiamo quindi passare alla lettura del nostro sketch per comprenderne le funzionalità; ovviamente si tratta di una nostra interpretazione che si presta a interventi creativi da parte vostra. Intenzionalmente non abbiamo utilizzato la funzione PWM, disponibile nella libreria, ma il pin di clock per il PWM è già collegato (D13) e basterebbe aggiungere poco codice per usufruire della sfumatura da un colore a un altro o dal nero al colore e viceversa. Allo stesso modo, senza ricorrere al PWM, potete trovare la soluzione di programmazione per avere l'indicazione di tutti i minuti e non di due minuti alla volta (come indizio vi possiamo dire che i minuti mancanti sono a cavallo di due LED...). Lo sketch inizia con l'inclusione delle librerie che servono al funzionamento dello shield RTC (wire e RTCLib), seguita dalla creazione dell'istanza RTC; si prosegue con l'include per la libreria HL1606strip e la definizione dei soli tre pin L, C e D a cui potrebbe seguire anche il pin S per cadenzare le variazioni di intensità (il PWM con fade in e fade out). Il blocco successivo definisce come costanti gli abbinamenti fra pin e ore, riducendo l'occupazione della memoria RAM rispetto all'uso di altrettante variabili. Come variabili abbiamo invece ore, minuti e secondi, quindi il colore dei LED di minuti e secondi e una variabile per gestire la variazione casuale di colore a ogni cambio di ora. Altre variabili di servizio ci permettono di attivare il cambio casuale e la modalità a punto o barra per i secondi. La sezione di setup () si occupa



di definire tutti i pin delle ore – dichiarati come costanti – come uscite digitali con l'istruzione `pinMode`, applicata anche ai pin da A0 ad A3; la porta seriale viene attivata per consentire il controllo del funzionamento del sistema senza dover “tirare a indovinare” in base all'accensione dei LED se c'è qualche problema. Sempre in fase di setup c'è il controllo dello stato dell'RTC descritto in precedenza: se il bit `Clock Halt` è a 1 allora viene effettuato l'`adjust` con i valori di minuti e secondi scritti in fase di compilazione. Per aggiornare l'orologio è necessario staccare la batteria tampone per una trentina di secondi, dopodiché lo sketch va ricompilato e ricaricato per avere dei valori “freschi” da scrivere con questa funzione. La sezione di `loop()` richiama in sequenza delle funzioni definite più avanti nello sketch e nell'ordine legge l'orario dall'RTC, quindi spegne virtualmente la striscia scrivendo per tutti i LED il colore nero (ma non invocando l'effettiva scrittura della striscia); seguono le funzioni di scrittura di secondi, minuti e ore, quindi 0,2 secondi di pausa e poi si ricomincia. Da notare che la scrittura effettiva della striscia, modificando la configurazione di colori e stato dei LED avviene alla fine della funzione `readclock()` e solo ogni due secondi; i valori di minuti e secondi letti dall'RTC sono divisi per due, così da avere una semplice corrispondenza fra numero del LED (da 0 a 30) e valore da indicare. La funzione di scrittura delle ore avviene invece in modo diretto, prima spegnendo tutte le uscite e poi utilizzando una struttura `switch case` per attivare l'uscita corrispondente all'ora restituita dalla funzione `readclock()`. Il valore è gestito sulle 12 ore, senza indicazione di mattina e pomeriggio e anche qui c'è spazio per ag-

giungere una soluzione che renda esplicito il passaggio da mattino a pomeriggio.

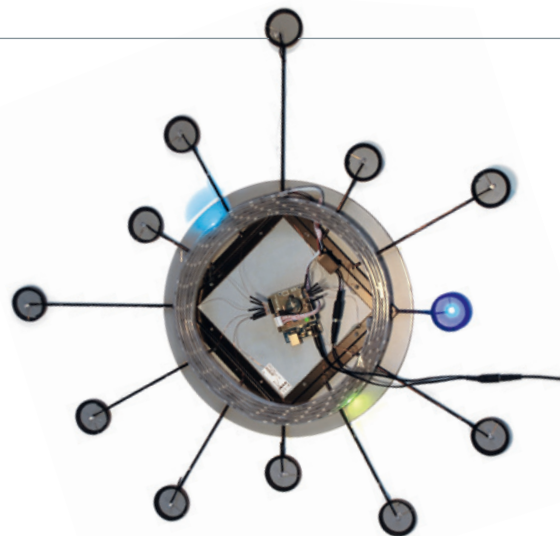
La scrittura dei minuti è semplice in quanto accende il LED corrispondente al valore della variabile, mentre quella dei secondi utilizza un ciclo che accende i LED dal primo fino al valore corrente dei secondi (diviso per due, come spiegato prima).

L'ultima funzione serve a definire il colore casuale fra i sette disponibili, evitando che venga assegnato il nero.

### IL COLLAUDO

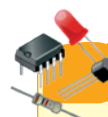
Dato che il cablaggio del progetto si basa sull'integrazione dei componenti all'interno della struttura dello specchio Figgjo, potete scegliere se procedere con un solo collaudo a fine montaggio, oppure se fare un primo step in cui sono collegati i componenti elettronici principali (Arduino, Striscia LED, RTC e un paio di LED delle ore) in modo del tutto volante e solo per verificare che i collegamenti e lo sketch siano funzionanti come previsto.

Se intendete modificare il nostro sketch per aggiungere funzionalità ed effetti di colore, vi suggeriamo di farlo operando su un setup ridotto ad Arduino, RTC e striscia. Quando il codice è funzionante come da vostre specifiche, potete procedere con la fase successiva di montaggio permanente. Ricordate inoltre che l'alimentazione della striscia a 5V è collegata all'alimentazione della scheda Arduino e questo potrebbe creare problemi in fase di programmazione tramite porta USB (che è alimentata) da PC. In questo caso potete tenere la striscia alimentata, rimuovere il collegamento di quell'alimentazione al pin 5V sul pin strip di Arduino e alimentare Arduino tramite USB. Le due alimentazioni, in questo caso,



avranno solo la massa in comune e la striscia si accenderà come previsto.

Se non alimentate la striscia indipendentemente, ma lasciate il collegamento fra il positivo della striscia e il pin 5V di Arduino, allora l'alimentazione sarà quella proveniente dalla porta USB del PC e questo potrebbe portare a superare l'assorbimento consentito dall'hardware del PC in quelle situazioni dove sono accesi numerosi LED.



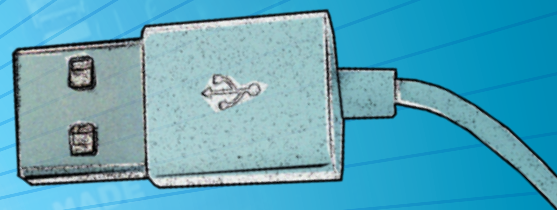
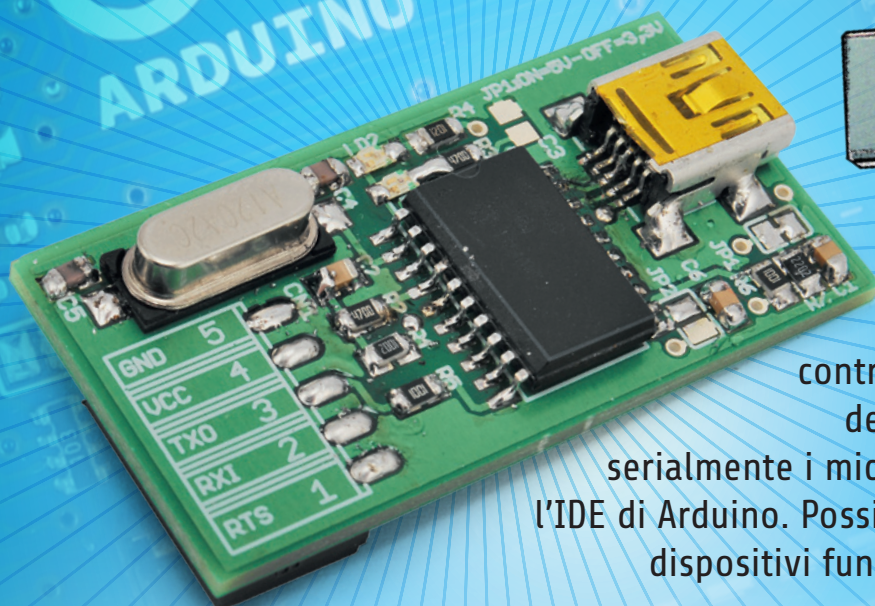
### per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono facilmente reperibili in commercio. Lo specchio Figgjo è disponibile presso i negozi della catena IKEA a 25,50 Euro. Lo shield RTC (cod. RTCSHIELDKIT, Euro 11,00), la strip luminosa con controllo a singolo LED (cod. STRIPRGBHL1606, Euro 180,00), Arduino Uno (Cod. ARDUINOUNOREV3, Euro 24,50) e tutte le altre minuterie sono disponibili presso Futura Elettronica. Tutti i prezzi si intendono IVA compresa.

Il materiale va richiesto a:  
Futura Elettronica, Via Adige 11,  
21013 Gallarate (VA)  
Tel: 0331-799775 • Fax: 0331-792287  
<http://www.futurashop.it>



# CONVERTITORE USB PER PROGRAMMAZIONE SERIALE a 5 e 3,3V



Semplice circuito controllato dalla porta USB del PC per programmare serialmente i micro Atmega utilizzando l'IDE di Arduino. Possibilità di programmare dispositivi funzionanti a 3,3 e 5 volt.

..... del Prof. MICHELE MENNITI

**L'**affermazione del fenomeno Arduino ha avvicinato molti appassionati di informatica all'elettronica, ma ha anche risvegliato la passione ormai sopita in tanti appassionati di elettronica, che non avevano accettato di buon grado l'avvento dei microcontrollori. Tutto ciò grazie alla semplicità con cui è possibile programmare questa scheda ed interfacciarla col mondo esterno.

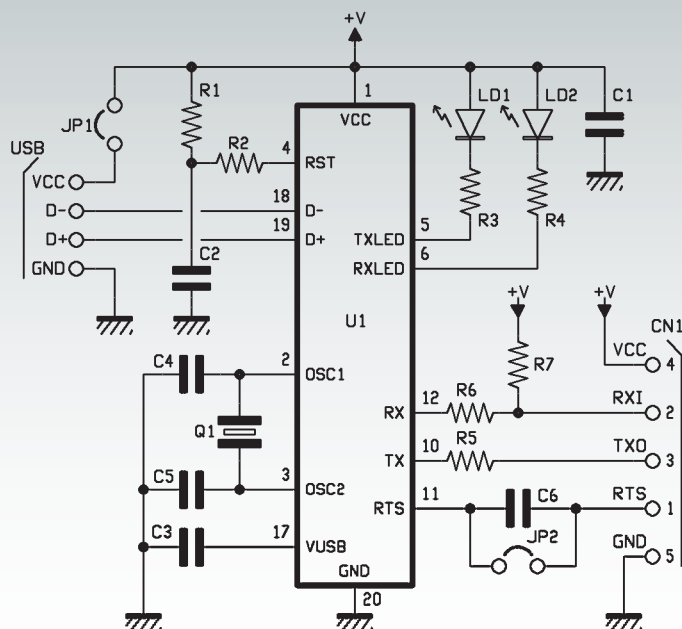
Come abbiamo più volte spiegato nei mesi precedenti, nelle applicazioni definitive è possibile recuperare la scheda Arduino e sostituirla col solo microcontrollore, l'ATmega328P (o addirittura con altri microcontrollori di casa ATMEL) ed altri pochi componenti esterni, creando un circuito stand-alone. Oggi proliferano quindi i circuiti basati su microcontrollori ATMEL usati nella modalità stand-alone. Anche a circuito completato, però, si può avere la

necessità di riprogrammare questi chip, per aggiungere nuove funzionalità al circuito, oppure per eliminare qualche bug scoperto nel tempo, ed allora ecco che bisogna ricorrere nuovamente ad Arduino per la ri-programmazione.

Nei tanti articoli dedicati alla Programmazione ISP abbiamo visto che in realtà è possibile predisporre sul circuito stand-alone un connettore a 3x2 pin per poter programmare il micro "sul posto", ma non sempre questa è una cosa fattibile, in quanto la circuiteria esterna potrebbe bloccare qualcuno dei pin coinvolti nella Programmazione ISP.

In questi casi è possibile allora predisporre la normale Programmazione Seriale - quella tipica di Arduino - che richiede però la presenza nel micro del bootloader di Arduino, a differenza della Programmazione ISP. Il circuito che vi proponiamo si presta egregiamente





a tale scopo, essendo assolutamente autonomo ed indipendente da Arduino, quindi direttamente collegabile alla porta USB di un qualunque PC fisso o portatile, ed utilizzabile mediante l'IDE in dotazione ad Arduino. Inoltre, rispetto a molti altri circuiti simili presenti sul mercato, abbiamo previsto la doppia possibilità di impostare i segnali di comunicazione con il circuito da programmare a livello logico 5V o 3,3V. Quanto appena detto significa che il nostro circuito può facilmente programmare le schede Arduino della famiglia "mini", cioè quelle che sono sprovviste del Convertitore USB-Seriale e ne richiedono espressamente uno esterno. L'altro grande vantaggio di questo nostro progetto è che abbiamo previsto due diversi circuiti stampati: uno in doppia faccia, più piccolo, per una realizzazione più professionale, e l'altro in monofaccia, senza alcun ponticello, per chi volesse realizzarlo in casa mediante fotoincisione o toner transfer (**Fig. 1**).

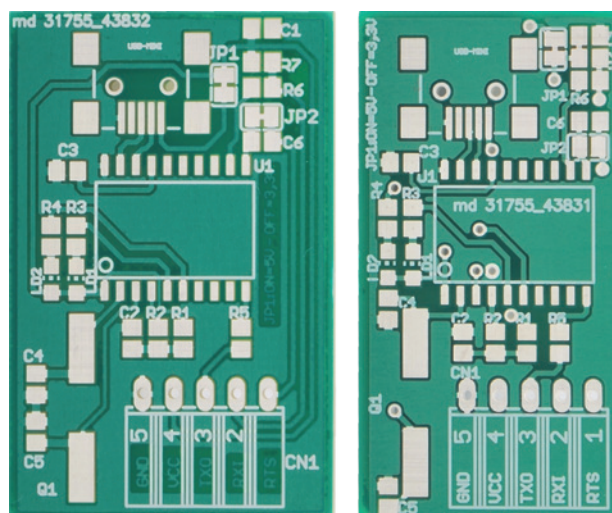
#### LO SCHEMA ELETTRICO

Il cuore del nostro circuito è il microcontrollore U1 siglato

MCP2200 che appartiene alla famiglia PIC e che Microchip ha programmato espressamente per svolgere la funzione di convertitore USB-Seriale; i lettori più assidui ricorderanno che abbiamo usato questo integrato nel progetto del Programmatore HV per microcontrollori ATMEL. Oltre allo schema applicativo di base, che abbiamo modificato per renderlo più efficiente ed aumentarne le funzionalità, Microchip mette a disposizione i driver ed un semplice programma di configurazione, che vi spiegheremo più avanti.

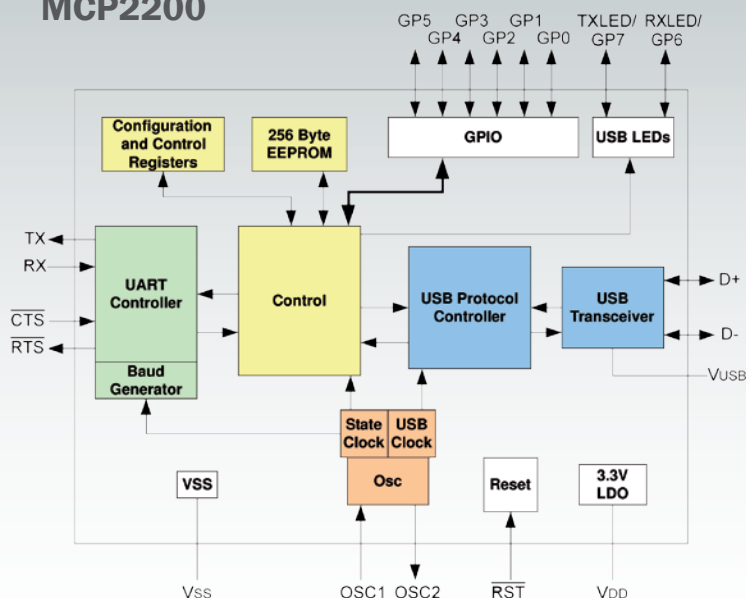
Osservando lo schema elettrico si vede la porta USB-mini che, tramite un normale cavo, permette di collegare il Convertitore ad una porta USB del PC. La porta USB-mini porta al circuito l'alimentazione a 5V ed i segnali D+ e D-.

Come tutti gli integrati che svolgono questa importante funzione di conversione, anche l'MCP2200 lavora con un clock a 12 MHz; avremmo potuto usare un normale risuonatore ceramico alla stessa frequenza, ma abbiamo preferito un quarzo in quanto la tipologia di segnali coinvolti nella comunicazione seriale richiede sincronizzazioni estremamente precise, e un quarzo in tal senso offre l'affidabilità richiesta. Il quarzo Q1 ed i relativi condensatori di disaccoppiamento C4 e C5 sono collegati ai pin 2 e 3 del micro. Il circuito costituito da R1 e C2 serve per resettare il micro ogni volta che il circuito viene alimentato; infatti inizialmente il condensatore C2 è scarico e si comporta come se fosse un cortocircuito verso GND, il pin 4 del micro riceve quindi un livello logico LOW e questo provoca il reset del micro; subito dopo C2 si carica in virtù della corrente che scorre attraverso R1 e resta

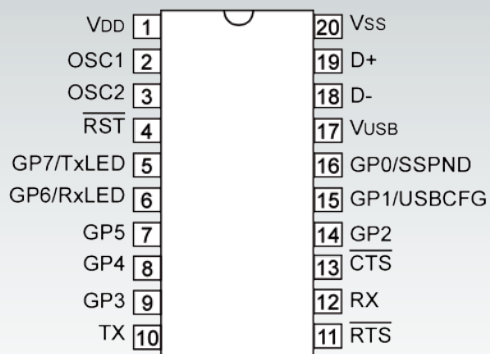


**Fig. 1** - Per realizzare il convertitore USB/seriale è possibile fare ricorso ad un PCB a singola faccia (a sinistra) o ad un più compatto PCB a doppia faccia (a destra).

## MCP2200



## MCP2200 SOIC, SSOP



Disposizione dei piedini e schema funzionale dell'integrato Microchip MCP2200 utilizzato in questo progetto. Il chip è in formato SOIC con passo di 1,27 mm.

in questa condizione fintanto che il circuito resta sotto tensione, garantendo che non avvengano altri reset; R2 serve per limitare la corrente in ingresso al pin 4 (RST) quando si toglie alimentazione al circuito e C2 inizia a scaricarsi proprio su questo pin.

I pin 19 e 18 sono collegati direttamente alla porta USB (rispettivamente D+ e D-) e permettono il dialogo tra il PC ed il micro; tale scambio di dati lato USB, corrisponde sul fronte seriale ai pin 10 (TX), 12 (RX) e 11 (RTS).

Sui pin 5 e 6 sono collegati due LED che servono per monitorare, rispettivamente, lo stato dei segnali TX e RX del micro. La comunicazione durante la fase di programmazione avviene mediante i tre pin TXO, RXI e RTS: i primi due servono per il normale "dialogo" in trasmissione e ricezione, mentre l'ultimo svolge il compito di controllo del RESET del circuito da programmare, e va collegato (direttamente o mediante un condensatore da 100nF, secondo i casi) al pin di RESET del microcontrollore da programmare. Abbiamo aggiunto la lettera O (Output) a TX e la lettera I (Input) ad RX per rappresentare

la direzione dei due segnali. L'uscita del nostro Convertitore è rappresentata dal connettore CN1, un header a 5 poli, che porta all'esterno della scheda, nell'ordine: !RTS, RXI, TXO, VCC, GND; il simbolo "!" posto davanti a RTS significa che tale segnale opera in modalità attiva LOW, cioè normalmente è su HIGH e, solo nel momento in cui deve resettare il micro del circuito da programmare, va per un istante su LOW. Il nostro Convertitore USB-Seriale è di tipo bidirezionale, cioè la sua funzione non si limita alla sola programmazione di un microcontrollore target, ma può essere anche quella di invio dati al PC. Infatti, aprendo un programma di emulazione terminale (anche il semplice Serial Monitor in dotazione all'IDE di Arduino), sarete in grado di dialogare con il microcontrollore target proprio per via seriale, ricevendo dati a video e inviandoli mediante tastiera. In virtù di tale funzione, unita alle sue ridottissime dimensioni, il nostro Convertitore si presta benissimo ad essere integrato in circuiti stand-alone che richiedano la possibilità, costante o periodica, di un interfacciamento

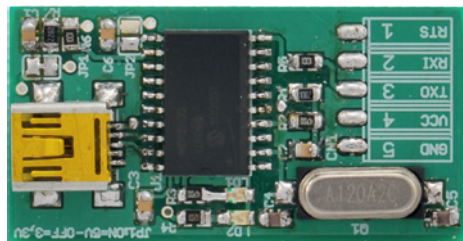
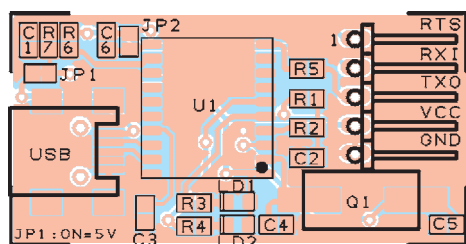
seriale con un PC.

Come anticipato nell'introduzione del presente articolo, il nostro Convertitore è in grado di operare sia con segnali a 5V, sia con segnali a 3,3V, ma su questa opzione dobbiamo fornire alcuni necessari chiarimenti.

Sullo schema elettrico è visibile il jumper JP1: non è un normale jumper con cappuccio, ma è di tipo a saldare, costituito da una coppia di piazzole presenti sul circuito stampato, normalmente in cortocircuito (tramite una sottile pista in rame) e quindi in posizione "ON".

Il compito di questo jumper è quello di collegare o meno al circuito i +5V provenienti dalla porta USB. Nella condizione di default si presuppone che il circuito target funzioni anch'esso a 5V e quindi la tensione proveniente dalla porta USB alimenterà sia il nostro Convertitore, sia il circuito da programmare mediante il pin VCC dell'header CN1.

Nel caso in cui si dovesse interfacciare il Convertitore con un circuito alimentato a 3,3V, diventa **obbligatorio** tagliare la piccola pista che unisce le due piazzole di JP1, scollegando in tal modo



## Elenco Componenti:

R1: 10 kohm (0805)  
 R2÷R4: 470 ohm (0805)  
 R5, R6: 1 kohm (0805)  
 R7: 22 kohm (0805)  
 C1, C2: 100 nF ceramico (0805)  
 C3: 470 nF ceramico (0805)  
 C4, C5: 18 pF ceramico (0805)  
 C6: 100 nF ceramico (0805)  
 U1: MCP2200-I/SO  
 LD1: LED rosso (0805)  
 LD2: LED verde (0805)  
 USB: Connettore mini USB  
 Q1: Quarzo 12 MHz (HC49/4H SMX)

Varie:

- Strip Femmina 5 vie 90°
- Circuito stampato

la tensione di 5V proveniente dall'USB. A questo punto la situazione si invertirà: infatti ora sarà il Convertitore ad essere alimentato a 3,3V, sempre tramite il pin VCC dell'header CN1, con la tensione proveniente da circuito target.

Nello schema è presente un altro jumper a saldare, denominato JP2, normalmente aperto ed in parallelo a C6; potrebbe capitare, in alcuni casi, che il circuito target non sia compatibile con la temporizzazione del segnale RTS definita da tale condensatore. In questo caso sarà sufficiente mettere in cortocircuito questo jumper (posizione ON) mediante una goccia di stagno, per bypassare C6.

**Una nota conclusiva importantissima:** come abbiamo visto, il nostro Convertitore si può interfacciare sia con circuiti alimentati a 5V che con circuiti alimentati a 3,3V, semplicemente agendo sul jumper JP1. Come abbiamo detto la funzione del jumper JP1 non è quella di ridurre l'alimentazione da 5V a 3,3V, bensì solo quella di collegare o scollegare i 5V provenienti dalla porta USB. È ovvio, ma è bene rimarcarlo, che se il circuito target è alimentato

a 3,3V e voi non ponete in OFF (interrompendolo) tale jumper, gli invierete i 5V tramite VCC ed anche il segnale TXO avrà un livello logico HIGH pari a 5V, quindi è molto probabile che possiate danneggiare il microcontrollore da programmare. Fate molta attenzione a questo aspetto e verificate ogni volta le caratteristiche del micro da programmare o da interfacciare in seriale!

## IL MONTAGGIO DELLA SCHEDA

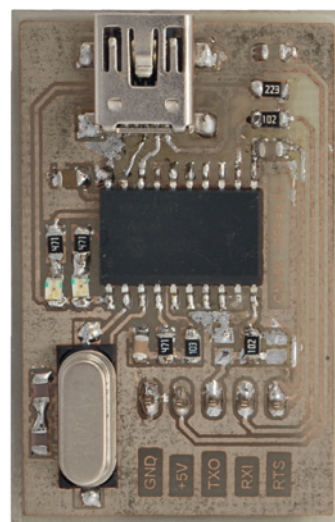
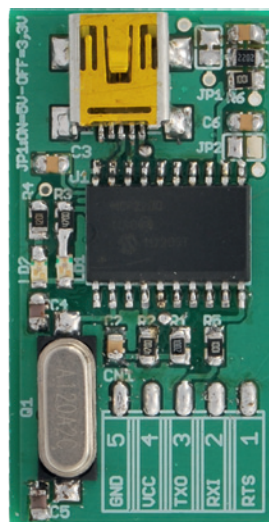
Una volta in possesso del PCB che avete scelto di usare, fate

riferimento al relativo piano di montaggio. Tutti i componenti usati in questo progetto sono in tecnologia SMD, una scelta indispensabile per poter contenere le dimensioni dei PCB. Il montaggio naturalmente richiede un po' più di attenzione di quella necessaria per la componentistica a foro passante ma non ci sono difficoltà insormontabili.

Una semplice tecnica che abbiamo sperimentato con successo consiste nel depositare preventivamente una piccola goccia di stagno su una delle due piazzole destinate ad ospitare il componente da saldare, quindi si prende il pezzo con la pinzetta e lo si adagia delicatamente in posizione e, sempre tenendolo con la pinzetta, si salda uno dei due capi, facendo nuovamente fondere la goccia di stagno depositata in precedenza; a questo punto si può facilmente e rapidamente saldare anche l'altro capo. Con lo stesso sistema si può saldare l'integrato, iniziando da uno dei quattro pin di spigolo (1-10-11-20).

L'integrato è in formato SOIC, con i pin distanziati di 1,27mm al posto dei canonici 2,54mm degli integrati PDIP; tutti gli altri componenti (ad eccezione del quarzo

**Fig. 2 - La versione definitiva con PCB a doppia faccia (a sinistra) confrontata con il prototipo originale monofaccia (a destra).**





che ha le stesse dimensioni di quelli a foro passante), cioè resistenze e condensatori, sono nel formato standard 0805 (2,1mm x 1,3mm circa): dimensioni piccole, ma accettabili, che permettono una facile manipolazione usando le comuni pinzette da elettronica; l'unico componente che può creare qualche difficoltà è il connettore USB-mini, non tanto e non solo per la distanza tra i pin, ma anche per il fatto che essi sono semi-nascosti dal contenitore stesso. Ovviamente il saldatore deve avere una punta molto fine, l'ideale è quella da 0,2mm, e nel caso del connettore sarebbe preferibile usare il tipo ad aria calda. L'ordine di montaggio dei componenti è quello solito: prima i componenti passivi (resistenze e condensatori), poi i due LED, poi l'integrato, il quarzo, il connettore USB-mini e infine il connettore header.

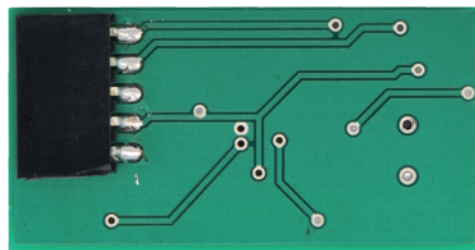
A proposito dei LED, gli SMD non hanno una differenza visibile ad occhio nudo tra anodo e catodo, come quelli per montaggio a foro passante, aventi i reofori di diversa lunghezza; bisogna osservarli attentamente e far riferimento al data-sheet per individuarne i due poli e saldarli in modo corretto; i LED che abbiamo usato noi hanno un riferimento di color verde in corrispondenza del catodo.

In Fig. 2 potete vedere le due schede montate: a sinistra il PCB in doppia faccia, a destra il prototipo originale del PCB monofaccia realizzato artigianalmente con il metodo del toner transfer; in questo caso abbiamo pensato di incidere direttamente sul rame le informazioni riguardo l'uso del JP1 ed il nome dei cinque pin dell'header. Nella sezione download del sito [www.elettronica.in](http://www.elettronica.in) troverete le versioni definitive di entrambi i PCB, dotate di serigrafia.

Per quanto riguarda il connettore header CN1, non dimenticate che questo è l'unico componente che va montato (in ogni caso e a prescindere dal tipo di PCB usato) sul lato inferiore del circuito stampato, saldando poi i pin sul lato componenti (Fig. 3); inoltre bisogna fare in modo che il connettore sia perfettamente aderente al PCB, in caso contrario durante l'uso potrebbe tendere a piegarsi creando difficoltà nell'infilare su uno strip maschio per la connessione.

Una volta completate le saldature è importante controllarle con una buona lente di ingrandimento, al fine di localizzare e rimuovere eventuali cortocircuiti; i rischi maggiori si hanno tra i pin dell'integrato e soprattutto tra i pin del connettore USB-mini; nel dubbio usate sempre un tester per verificare l'effettiva presenza del cortocircuito; fate inoltre attenzione alle micro-gocce di stagno che a volte si formano durante la saldatura e si attaccano in un punto qualsiasi del PCB, nel tempo potrebbero staccarsi e rotolare incastrandosi tra i pin dell'integrato o tra due componenti, creando così un cortocircuito inaspettato; quindi se ne localizzate qualcuna rimuovetela immediatamente. A proposito dei jumper a saldare, vi ricordiamo che JP1 (Fig. 4) è di default in condizione ON, in conseguenza a una piccola pista che unisce le due piazzole; nel caso in cui aveste necessità di posizionarlo su OFF, sarà sufficiente interrompere con un taglierino tale pista.

Ovviamente a questo punto, per rimettere il jumper su ON, non sarà possibile ripristinare la pista originale, però le due piazzole sono così vicine che basterà unirle con una goccia di stagno. Il discorso opposto vale per JP2, che di default è in condizione OFF.



**Fig. 3** - Il connettore header montato sul lato inferiore del PCB.

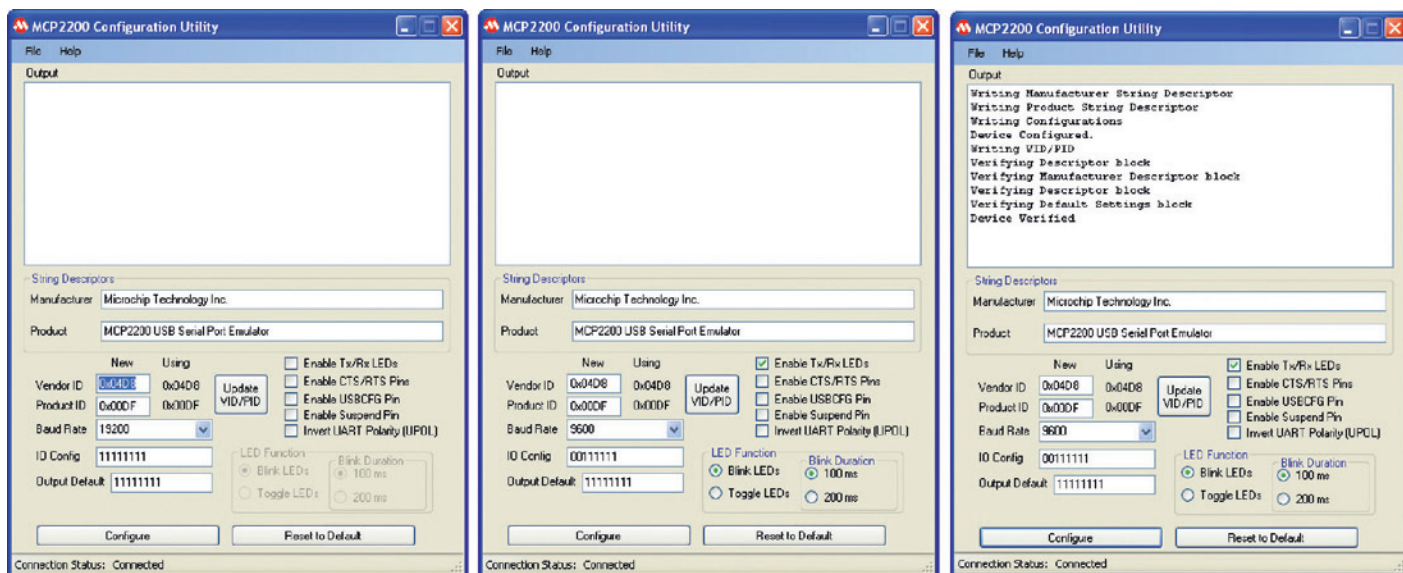
## LA PREPARAZIONE DEL SOFTWARE

L'integrato MCP2200 non può funzionare appena montato in quanto richiede una procedura di programmazione tramite un'utilità dedicata, eseguibile solo dopo aver installato il driver USB. Nella sezione download del sito [www.elettronica.in](http://www.elettronica.in) troverete una cartella con due sottocartelle, contenente appunto i driver e l'utilità di configurazione; una volta che le avrete scaricate e messe in un percorso noto, potrete seguire le prossime semplici istruzioni. Il primo passaggio da fare è eseguire il file "MCP2200ConfigUtility-v1.3-windows-installer" che si trova nella cartella "MCP2200 Configuration Utility v1.3"; questa operazione avrà il duplice scopo di installare l'utilità di configurazione ed i driver necessari per la gestione della nuova seriale; al termine dell'installazione – e solo allora – collegate la scheda ad una porta USB del PC, mediante un normale cavo USB.

La scheda sarà riconosciuta come periferica seriale e vi sarà chiesto di fornire il driver: dovrete semplicemente confermare l'installazione automatica. Nel caso in cui questa procedura non andasse a buon fine, dovrete indicare il percorso della cartella "MCP2200

**Fig. 4**  
Il jumper JP1 a saldare posto direttamente sul PCB.





**Fig. 5** - La configurazione dell'MCP2200 è molto semplice come illustrato in queste tre schermate: il Baud Rate va impostato a 9600 baud e va selezionata l'opzione "Enable Tx/Rx LEDs".

Windows Driver" e confermare le successive richieste; dopo alcuni istanti la nuova porta seriale sarà installata tra le periferiche del vostro PC, come "USB serial port (COMxx)".

A questo punto individuate fra i programmi "MCP2200 Configuration Utility", eseguitelo e vi apparirà la maschera visibile a sinistra in Fig. 5.

Dovete solo effettuare le due semplici modifiche visibili nell'imma-

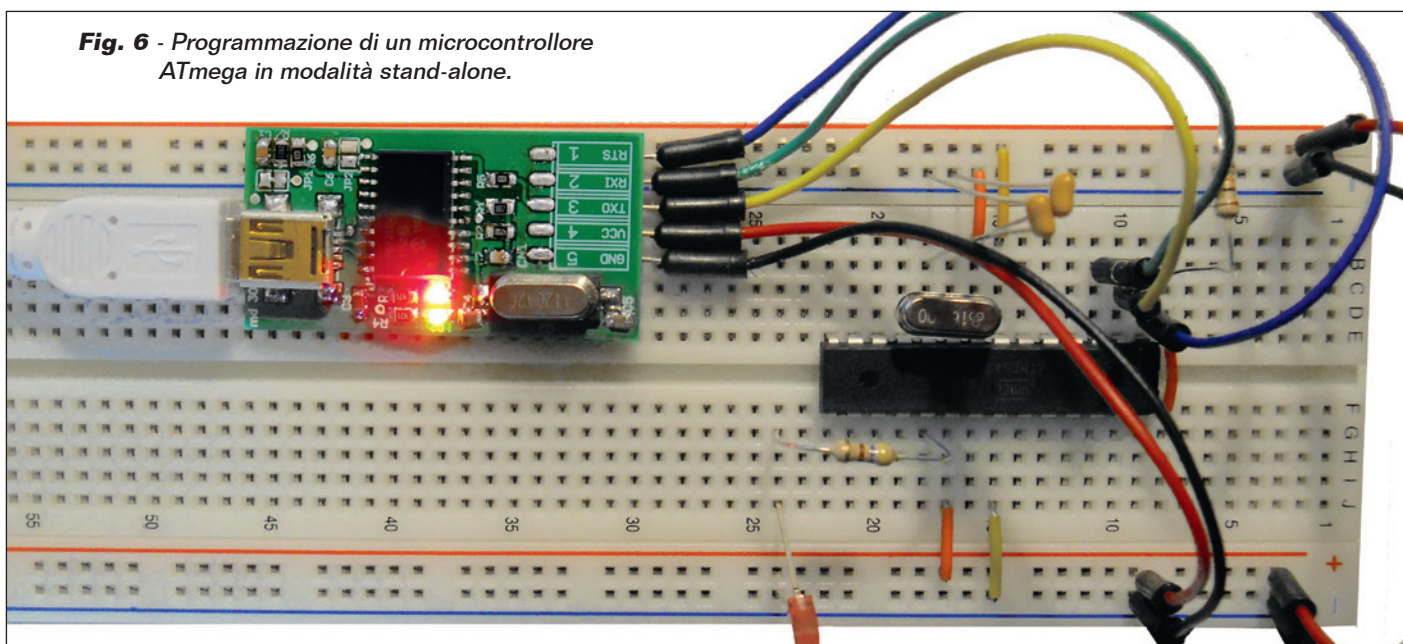
gine centrale e cioè: impostare il Baud Rate a 9600 e attivare il campo "Enable Tx/Rx LEDs", quindi cliccate sul pulsante "Configure"; i messaggi visibili nell'immagine a destra vi indicheranno che tutto è andato a buon fine, a questo punto potete chiudere la maschera. Vi sconsigliamo decisamente di fare esperimenti con questa utility dato che alcune combinazioni bloccano il chip e possiamo assicurarvi che è un'impresa

riuscire a sbloccarlo; limitatevi a quanto vi abbiamo consigliato e tutto andrà nel migliore dei modi.

## L'USO DEL CONVERTITORE USB-SERIALE

Ecco qualche consiglio per chi non avesse ancora dimestichezza con questi dispositivi. Il Convertitore USB-Seriale necessita in genere di tre segnali per dialogare con un microcontrollore: TX, RX e RTS (o DTR), deve inoltre avere la

**Fig. 6** - Programmazione di un microcontrollore ATmega in modalità stand-alone.



massa in comune con quella del circuito target.

Il dialogo avviene nella modalità a segnali incrociati (collegamento *cross-over*):

- TXO Convertitore → RX dispositivo target
- RXI Convertitore ← TX dispositivo target
- RTS Convertitore → RESET dispositivo target
- GND Convertitore ↔ GND dispositivo target
- VCC Convertitore ↔ +V dispositivo target

Quindi dovete fare ben attenzione a collegare correttamente, questi segnali, altrimenti la programmazione non andrà a buon fine. Riguardo l'alimentazione (VCC e +V) abbiamo già trattato la questione in modo esaustivo nella parte dedicata alla descrizione dello schema elettrico, fate quindi riferimento ad essa in base alla tensione richiesta per l'alimentazione del dispositivo target. In **Fig. 6** potete vedere appunto questi collegamenti effettuati tra il nostro Convertitore ed un microcontrollore ATmega328P configurato in modalità standalone con bootloader precaricato. Se usate l'IDE di Arduino dovete selezionare come board quella corrispondente al bootloader caricato sul microcontrollore, e come porta seriale quella che il sistema ha assegnato al Convertitore. A questo punto caricate il vostro sketch ed eseguite la normale operazione di upload; dopo qualche istante vedrete i due LED LD1 e LD2 accendersi - come visibile nella figura - segno che la programmazione sta procedendo regolarmente.

Una possibile applicazione di questo Convertitore riguarda i casi in cui quello integrato nella scheda Arduino si fosse danneggiato irrimediabilmente, impen-

dendo di fatto la riprogrammazione del microcontrollore. In questo caso basta effettuare questi semplici collegamenti (modalità 5V):

- TXO Convertitore → pin 0 Arduino
- RXI Convertitore ← pin 1 Arduino
- RTS Convertitore → pin RESET Arduino
- GND Convertitore ↔ pin GND Arduino
- VCC Convertitore → pin 5V Arduino
- Collegare il Convertitore alla porta USB del PC

Da questo momento in poi potrete riutilizzare la scheda Arduino come sempre, soltanto cambiando la porta COM originale con quella assegnata al Convertitore. Naturalmente l'operazione è eseguibile anche nel caso in cui il Convertitore USB di Arduino funzionasse alla perfezione.

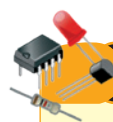
Se invece dovete programmare una scheda **Arduino Pro Mini**, versione 8 MHz e alimentazione a 3,3 V, i pin che servono sono già contrassegnati con i nomi dei segnali; dovete procedere come segue:

- Mettete su OFF il jumper JP1 del Convertitore
- TXO Convertitore → pin RXI Arduino Pro Mini
- RXI Convertitore ← pin TXO Arduino Pro Mini
- RTS Convertitore → pin RST Arduino Pro Mini
- GND Convertitore ↔ pin GND Arduino Pro Mini
- VCC Convertitore → pin VCC Arduino Pro Mini
- Collegare la vostra alimentazione al pin RAW dell'Arduino Pro Mini (automaticamente sul pin VCC si verranno a trovare i 3,3V che alimentano sia Arduino Pro Mini che il Convertitore USB-Seriale)

- Collegare il Convertitore alla porta USB del PC

A questo punto selezionate il modello Arduino Pro Mini e la porta COM assegnata al Convertitore e sarete pronti per la programmazione.

A prescindere dalla modalità di collegamento usata, potrebbe verificarsi la condizione di errore in cui il solo LED LD1 (TX) si accende per tre volte di seguito, mentre il LED LD2 (RX) resta spento; in questo caso quasi certamente c'è un problema legato al controllo del RESET; se siete sicuri che i collegamenti siano stati eseguiti correttamente e che il microcontrollore da programmare contenga effettivamente il bootloader della scheda che avete selezionato, probabilmente la durata del segnale RTS non è sufficiente a resettare correttamente il microcontrollore. In questo caso spegnete tutto e mettetelo in cortocircuito il jumper JP2, per bypassare il condensatore C6, quindi riprovate: questa volta la programmazione dovrebbe andare a buon fine. ■



## per il MATERIALE

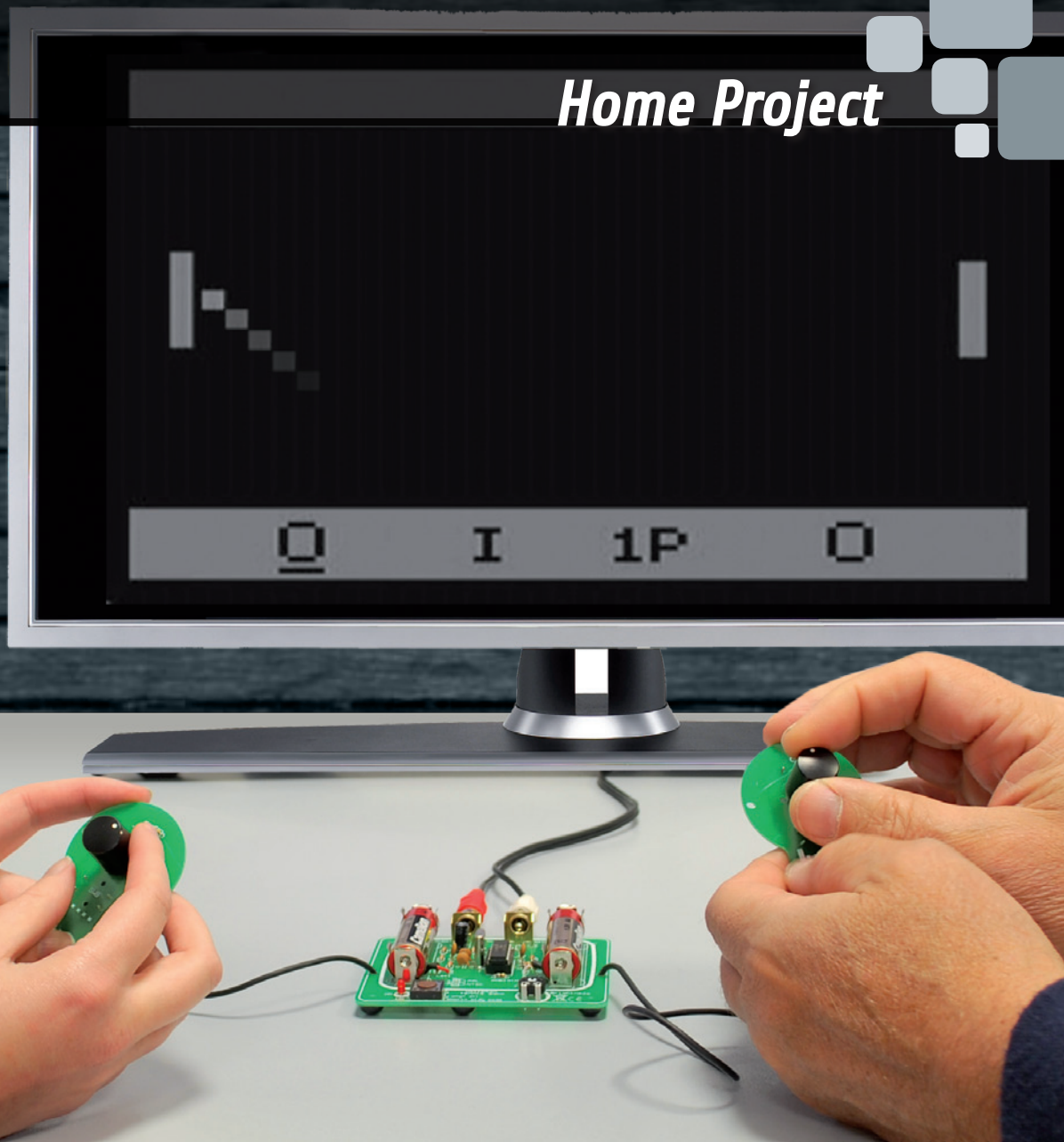
Tutti i componenti utilizzati in questo progetto sono facilmente reperibili in commercio; i master degli stampati ed il software di configurazione dell'integrato sono scaricabili gratuitamente dal sito della rivista ([www.elettronica.in.it](http://www.elettronica.in.it)). L'integrato in versione SIOC cod. MCP2200-I/SS) è disponibile al prezzo di 3,80 Euro IVA compresa.

Il materiale va richiesto a:  
Futura Elettronica, Via Adige 11,  
21013 Gallarate (VA)  
Tel: 0331-799775 • Fax: 0331-792287  
<http://www.futurashop.it>





Classico videogioco del tennis per uno o due giocatori, con 5 livelli di difficoltà, funzionante su qualsiasi TV.



di SANDRO REIS

## TENNIS GAME

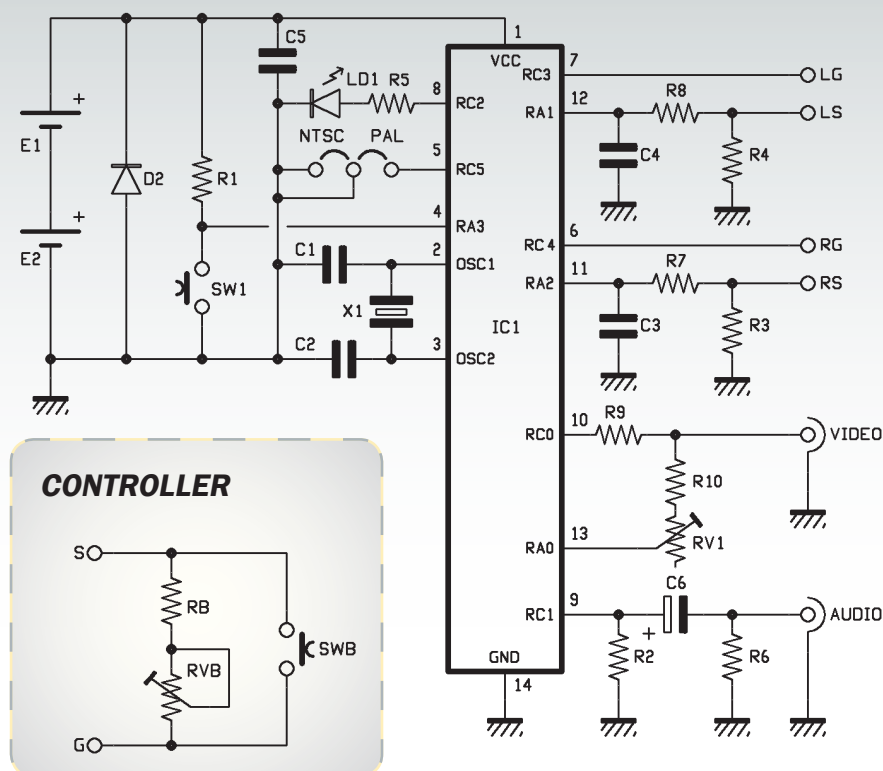
**S**iamo ormai in inverno e si stanno avvicinando le festività di fine d'anno, momenti in cui, complice il clima rigido, si trascorre più tempo in casa che fuori, magari con i propri passatempi preferiti e i giochi elettronici; sebbene i videogame si siano ormai da tempo

trasferiti sul computer, il fascino di quelli davanti alla TV non è mai tramontato, neppure oggi che i supertecnologici TV LCD, OLED e a plasma si sono fatti largo nelle nostre case soppiantando i pesanti e ingombranti televisori con schermo CRT. Ecco perché abbiamo pensato

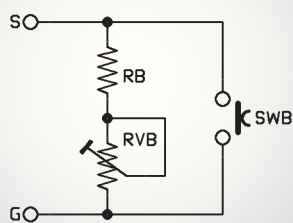
di proporvi la realizzazione di un circuito che fa tanto "vintage": il classico gioco del tennis da collegare al televisore, al quale giocare in due o da soli mediante comodi controller a manopola facilmente autocostruibili; il gioco consiste in un circuito molto semplice adatta-

bile a televisori sia PAL (lo standard televisivo che abbiamo in Italia) sia NTSC, da collegare all'ingresso video o alla presa SCART. Il tutto si alimenta con due semplici pile stilo (AA). Una volta messo in funzione, il circuito compone sullo schermo del televisore il classico pattern del

## [schema elettrico **SCHEDA BASE**]



### CONTROLLER



campo da tennis in bianco e nero, con ai lati i due giocatori simboleggiati da rettangoli: tramite i controller possiamo spostare su e giù questi giocatori (ogni controller ne governa uno) in modo da far loro incrociare la palla da tennis, rappresentata da un pallino bianco, e ribatterla. In ciascun controller il movimento dei giocatori si comanda mediante una manopola, mentre un pulsante attiva la funzione SMASH.

### SCHEMA ELETTRICO

Dell'uso e delle funzioni ci occuperemo più avanti; ora diamo uno sguardo all'elettronica del circuito, partendo dall'analisi dello schema elettrico: l'intero videogame viene gestito all'interno di un microcontrollore PIC16F1823-I/P della Microchip, programmato con il firmware VMK191; questo PIC ha un'architettura RISC ad 8 bit con memoria di programma di tipo Flash,

ed è incapsulato in un contenitore plastico dip a 7+7 piedini. Il micro è uno dei più recenti della produzione Microchip e può funzionare a tensioni anche di soli 3 volt, quindi i suoi I/O sono TTL-compatibili sia nei confronti dei livelli 0/3 volt, sia riguardo a quelli 0/3 V. Chiaramente, funzionando a 3 volt di alimentazione gli input devono essere interfacciati con livelli al massimo di 3 volt. Il clock dell'oscillatore principale è deciso dal quarzo esterno X1, ausiliato dai condensatori ceramici C1 e C2; la frequenza è stabilizzata a 16 MHz, un valore che sembra alto ma che in realtà è indispensabile a garantire l'esecuzione della routine di costruzione e refresh del quadro televisivo, che deve essere molto veloce onde evitare che l'immagine sfarfalli. Infatti utilizzando il sistema televisivo NTSC bisogna comporre 30 frame completi (60 semiquadri)

al secondo, mentre optando per il PAL servono 25 quadri completi (50 semiquadri) al secondo.

Dopo il power-on-reset, il micro inizializza le proprie linee di I/O impostando RC2 come uscita per il comando del LED di segnalazione D1, RA3 come input senza pull-up (perché sul registro RA non si può impostare quello interno; a riposo, il piedino 4 viene mantenuto a livello alto dal resistore R1) per leggere lo stato del pulsante SW1 ed RC5 da input con pull-up interno, usato per acquisire lo stato del jumper che seleziona la modalità video televisiva tra NTSC (standard americano) e PAL (standard europeo). RC0 ed RC1 vengono inizializzate come uscite rispettivamente per il segnale televisivo e audio, generati all'interno del microcontrollore mediante il modulo PWM dedicato. Sempre con l'istruzione e il modulo PWM, il PIC gestisce la lettura dei controller dei giocatori, ognuno dei quali contiene un trimmer configurato come reostato semifisso e una resistenza serie, oltre ad un pulsante impiegato per la funzione SMASH; per l'esattezza, il modulo PWM interno al PIC16F1823-I/P genera un segnale di frequenza costante

## La Wii incontra il touch-screen

Non paga di tutte le innovazioni che ci ha proposto in questi anni, Nintendo ha sviluppato, per la sua nuova



che viene inviato dalle linee LG (per il giocatore di sinistra ed RG per quello di destra) quindi la tensione uscente dal controller raggiunge i punti LS ed RS, dove viene filtrata da due celle passabasso ad RC, che sono formate da R8/C4 per il controller del giocatore di sinistra e da R7/C3 per quello del giocatore di destra. Siccome la frequenza di taglio del filtro passa-basso RC varia in maniera inversamente proporzionale al valore della resistenza complessiva, minore è la resistenza inserita dal potenziometro del controller, più alta sarà la frequenza di taglio del filtro e viceversa; applicando il discorso al controller del giocatore di destra, inserendo con RVB una maggiore resistenza, la tensione PWM determinerà tra la linea RA2 del micro e massa una differenza di potenziale livellata minore di quella che si otterrebbe se la resistenza del potenziometro RVB fosse più bassa. Con questa tecnica, che vale pari-pari per il controller del giocatore sinistro (tensione continua ci capi del C4), il firmware del microcontrollore è in grado di rilevare la posizione del cursore del potenziometro e quindi la posizione che il giocatore richiede alla relativa racchetta.

In maniera analoga, viene letta la funzione SMASH, che corrisponde alla "schiacciata" data dalla racchetta alla palla da tennis e che si attiva premendo il pulsante SWB mentre la palla a video arriva sulla racchetta; la pressione del pulsante SWB (che comanda lo SMASH) determina il cortocircuito della resistenza del controller e quindi un brusco elevamento della tensione continua riportata al corrispondente input di lettura del micro (RA1 o RA2). Notate che per come funziona il firmware, premendo questo pulsante prima che la palla tocchi la racchetta, la brusca variazione di resistenza che determina viene ignorata dal microcontrollore, il quale la rileva solo quando la palla batte sulla racchetta. Il segnale video viene sintetizzato dal firmware tramite un'apposita routine, che opera con temporizzazioni differenti a seconda di come, all'avvio, il microcontrollore trova impostato il piedino 5; in altre parole, se la linea RC5 è a livello alto (jumper

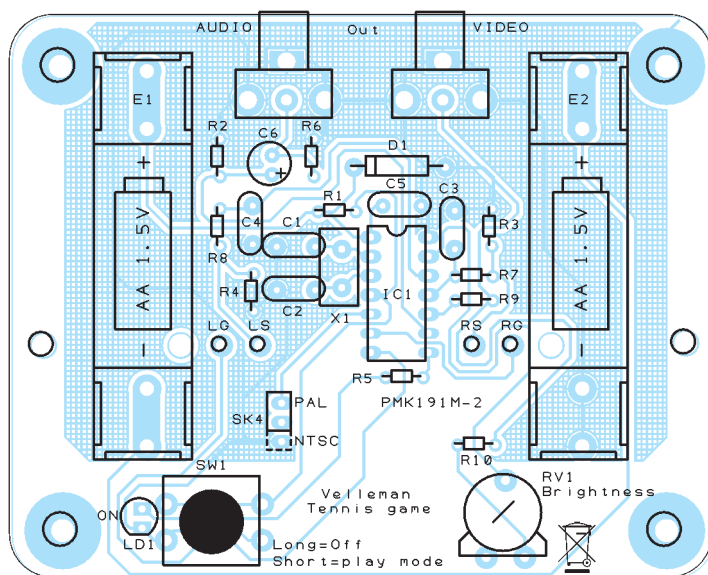
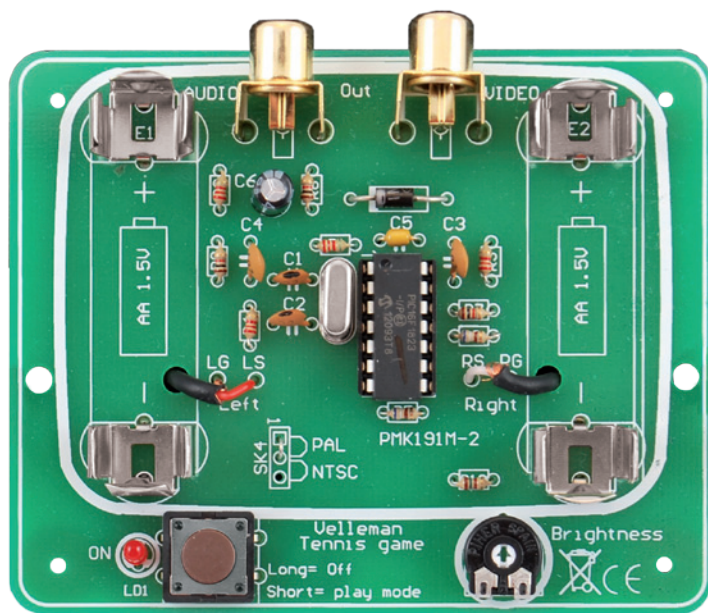
PAL aperto ed NTSC chiuso o aperto) il firmware genera le temporizzazioni NTSC, mentre se RC viene trovata a zero logico (jumper PAL chiuso ed NTSC chiuso o aperto) vengono generate le temporizzazioni a standard PAL. Nel nostro caso, presupponendo l'uso in Italia e in Europa, dobbiamo sempre chiudere il ponticello PAL. Ma allora, a cosa serve la funzione NTSC? Semplice: se abbiamo un TV o monitor proveniente dagli USA o dal Giappone, ci torna utile perché questi funzionano secondo lo standard americano NTSC. Nella modalità PAL, la relativa routine firmware imposta una frequenza di quadro di 50 Hz, quindi compone ogni secondo 50 fotogrammi corrispondenti alla schermata del gioco, aggiornandoli continuamente per adeguare la visualizzazione al movimento della pallina e della racchetta; in realtà, per sgravare dal lavoro il microcontrollore, il firmware prevede la composizione di 50 semiquadri al secondo,

console Wii U, un controller dotato di touch-screen. Presentata di recente all'E3 di Los Angeles, la console dispone di un innovativo controller che centralmente riporta uno schermo touch-screen e lateralmente ha due mini-joystick analogici "cliccabili" posti in una posizione agevole quando si tiene il controller fra le mani. Oltre al touch-screen, il nuovo controller integra la funzionalità di telecomando universale ad infrarossi per controllare il televisore, e sarà dotato di un chip NFC (Near Field Communication) per consentire l'acquisizione di dati da altri dispositivi. Accanto al



controller touch-screen, la nuova console dispone del Wii U Pro, un joypad di tipo classico per i giochi di tipo tradizionale.

## [piano di montaggio **SCHEMA BASE**]



### Elenco Componenti:

R1 ÷ R4: 2,2 kohm  
 R5: 680 ohm  
 R6 ÷ R8: 1 kohm  
 R9: 680 ohm  
 R10: 150 ohm  
 RV1: Trimmer 1 kohm MO  
 C1: 22 pF ceramico  
 C2: 22 pF ceramico  
 C3, C4: 10 nF multistrato  
 C5: 100 nF multistrato  
 C6: 10 µF 63 VL elettrolitico  
 D1: 1N4007

IC1: PIC16F1823-I/P (VMK191)  
 LD1: LED 3 mm Rosso  
 X1: Quarzo 16 MHz

#### Varie:

- Manopola per potenziometro
- Zoccolo 7+7
- Connettore RCA da CS (2 pz.)
- Clip porta batterie AA (4 pz.)
- Filo 30 cm. bifilare (2 pz.)
- Circuito stampato base
- Circuito stampato racchetta (2 pz.)

ossia 25 contenenti le righe pari dell'immagine ed altrettante, intercalate ad esse, contenenti le righe dispari. In tal modo il microcontrollore deve svolgere, in un secondo, metà dei calcoli e quindi è richiesta una velocità di elaborazione tutto sommato bassa, compatibile con le prestazioni del PIC16F1823-I/P; questa tecnica (chiamata *interlacciamento* o *interallacciamento*) è la stessa che nella televisione analogica ha permesso per decenni di trasmettere utilizzando una larghezza di banda del canale video di appena 5 MHz, contro gli oltre 10 richiesti dalla trasmissione di 50 quadri interi al secondo. Peraltro la scomposizione e proiezione alternata non è percepibile dall'occhio umano, in quanto la composizione dell'immagine a trame pari e dispari alternate non si vede, stando a una distanza sufficiente (già una o due volte la diagonale dello schermo) ad occhio nudo.

Per comporre questi quadri nel tempo previsto, considerato che nello standard PAL ogni quadro intero è composto da 625 righe e utilizzando il rapporto base/altezza dello schermo originario (cioè il 4/3) la frequenza di riga, ossia quella con cui viene acceso ciascun pixel (punto) di ogni riga dello schermo, è 15.635 Hz. In realtà l'altezza dell'immagine del PAL è stata standardizzata in sole 576 linee effettive, mentre le restanti 49 vengono utilizzate per il ritorno della traccia nello spegnimento del cannone elettronico dei vecchi TV e monitor a tubo catodico, nonché per inserire nell'immagine informazioni numeriche quali i dati del televideo e in generale per i servizi di teletext.

Dunque, in PAL lo schermo rende visibili effettivamente 576 linee, il che permette di

risparmiare altro lavoro al microcontrollore, dato che ogni venticinquesimo di secondo deve comporre 576 invece che 625 linee d'immagine.

Per la composizione delle immagini secondo lo standard NTSC, il firmware opera con una frequenza di riga di 15.750 Hz e le righe da comporre per ciascun quadro sono 525, con refresh verticale di 30 Hz; in realtà anche nell'NTSC parte delle righe non serve perché quelle effettivamente visualizzate dallo schermo sono 480, con evidente beneficio per il microcontrollore che deve ricostruire l'immagine.

Indipendentemente dalla modalità video scelta, il PIC genera una componente videocomposita standard, tutta positiva, dell'ampiezza di 1 volt; in realtà, siccome il videocomposito è bidirezionale e composto da una parte positi-

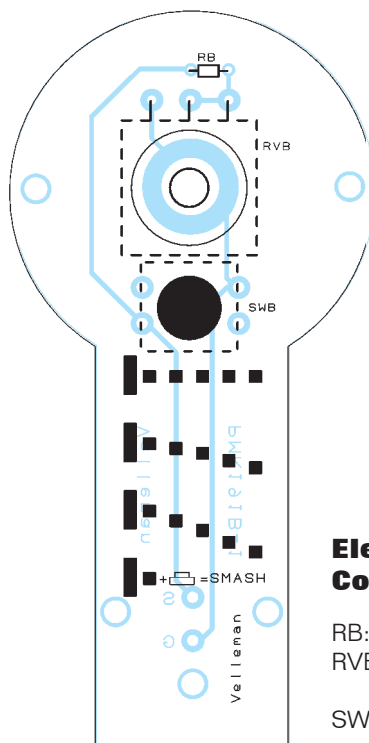
va ( $0 \div 0,7$  volt) ed una negativa (sotto il livello del nero, ossia  $-0,3$  V) per poter inviare il segnale video al televisore viene traslata la tensione di zero, che è pari a  $0,3$  V positivi: quindi sotto gli  $0,3$  volt il piedino RC0 produce gli impulsi di sincronismo di riga, mentre sopra mette la luminosità dell'immagine punto per punto. Più esattamente, a zero volt corrisponde il nero e a 1 volt ( $0,7$  V sopra il riferimento) corrisponde il bianco. Visto che l'immagine è in bianco e nero, in realtà il micro potrebbe limitarsi a generare solo impulsi da 1 V o pause a 0 V (impulsi di riga) e di quadro. Per stabilire il livello del segnale prodotto dal piedino RC0, nel circuito è presente una sorta di rete di retroazione realizzata dalla serie di R10 ed RV1; essendo quest'ultimo un trimmer, possiamo regolare agevolmente



**Schermate iniziali del gioco per le partite ad 1 (1P, sopra) e 2 (2P, sotto) giocatori; il livello di difficoltà è il primo (I).**

*Tutte le videate del gioco sono in bianco e nero.*

## [piano di montaggio **RACCHETTA**]

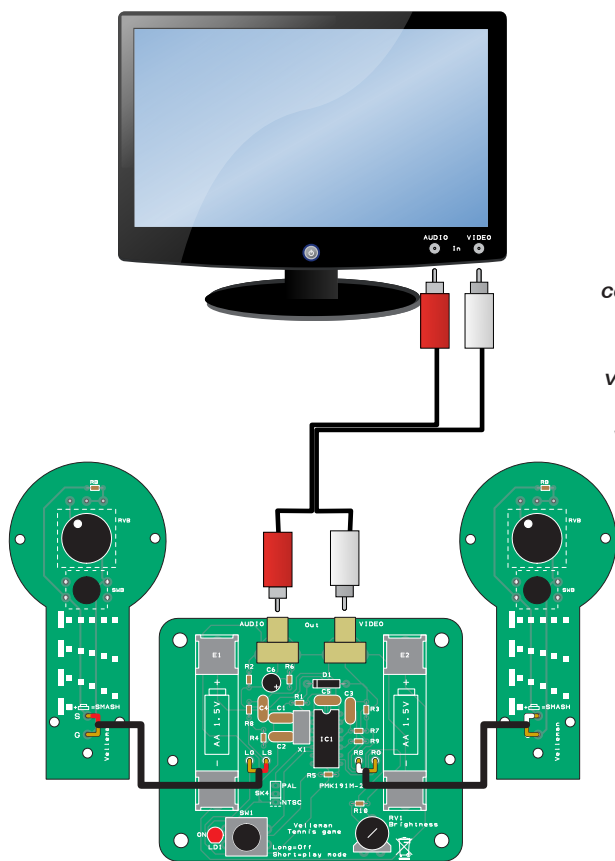


### **Elenco Componenti:**

RB: 470 ohm (2 pz.)  
RVB: Potenziometro  
1 kohm (2 pz.)  
SWB: Microswitch.



## I collegamenti



*Per collegare il controller al circuito base usate indifferentemente cavo coassiale o piattina, ma rispettate le connessioni indicate. Il collegamento al TV o monitor videocomposito si fa con un cavo audio/video RCA o con un RCA più un adattatore SCART bidirezionale o per solo input. Una volta acceso il TV, regolate il trimmer per ottenere la luminosità ottimale dell'immagine.*

ed entro certi limiti l'ampiezza. Il controllo del PWM che produce il segnale video viene ottenuto leggendo con la linea RA0 il segnale presente ai capi del connettore VIDEO, quindi assegnandolo all'A/D converter interno al microcontrollore: la lettura dell'ADC viene usata per correggere il PWM. Il trimmer RV1, variando l'ampiezza del segnale videocomposito, permette di determinare la luminosità dell'immagine, giacché più ampio è il segnale, maggiore è la luminosità. Notate che il segnale prodotto dal PIC è simile a quello videocomposito inviato agli stadi modulatori TV o estratto dai demodulatori: unidirezionale. Il televisore lo accetta sulla SCART o sull'ingresso videocomposito grazie al fatto che il condensatore di disaccoppiamento normalmente presente all'input video permette di ottenere nel TV la

componente videocomposita alternata, quindi con livelli positivi e negativi come dovrebbe essere. Quanto al segnale audio, viene anch'esso sintetizzato dal modulo PWM interno al microcontrollore e riprodotto dalla linea RC1; il condensatore C6 provvede al necessario disaccoppiamento in continua dallo stadio d'ingresso del televisore. Il livello di uscita dell'audio è costante e impostato ad un valore che il TV tollera: circa un volt di picco. Con la regolazione del volume potete tranquillamente adattarlo alle vostre esigenze. L'audio del gioco è composto dai suoni di conferma del contatto con la palla, di inizio e fine partita ecc. Concludiamo la descrizione dello schema elettrico del circuito con l'alimentazione, che si preleva semplicemente da una serie di pile stilo (formato AA) da 1,5 V, per complessivi 3 V; notate

il diodo D2, che cortocircuita le batterie nel caso le montiate al contrario, proteggendo così il microcontrollore. Nel normale esercizio, il sistema assorbe appena 7 mA, il che garantisce lunga durata alle pile; l'assorbimento da spento è trascurabile.

### COME FUNZIONA

Bene, spiegato lo schema elettrico vediamo la funzione dei comandi e più in generale come si usa il videogame del tennis. Il primo comando da osservare è il pulsante multifunzione SW1, che serve sia ad accendere e spegnere il circuito, sia a scorrere tra le modalità di gioco; più esattamente, premendo a lungo il pulsante si accende il videogame (se è spento) o lo si spegne (se è acceso) mentre brevi pressioni quando il circuito è acceso permettono di scorrere tra le modalità previste. Quando il circuito è spento, il microcontrollore si porta in modalità sleep ed assorbe praticamente nulla (poche decine di microampere); notate inoltre che quando non utilizzato per un certo tempo, il circuito si spegne automaticamente.

Per quanto riguarda la scelta dei giochi, è possibile disputare gare in due o da soli (si gioca contro il "cervello" del videogame) e sono possibili 5 livelli di difficoltà per la gara a singolo giocatore ed altrettanti per quella a due giocatori. Per ogni livello di gioco si possono scegliere tre diverse angolazioni della pallina. La selezione della modalità di gioco si effettua a video nella schermata introduttiva, dopo aver acceso la console con una breve pressione del pulsante SW1: ogni pressione breve fa cambiare la modalità di gioco, che viene indicata nella sezione in basso allo schermo; qui trovate agli estremi il punteggio della

manche, mentre in mezzo vengono indicati un numero (a sinistra) e il numero dei giocatori (a destra) sotto forma di 1P per indicare che si gioca da soli e 2P che la partita è tra due giocatori. Dunque, dopo l'accensione (confermata dall'illuminazione del LED presente nel circuito) premendo consecutivamente SW1 per un breve istante vedrete alternarsi le seguenti combinazioni: I – 1P, II – 1P, III – 1P, IV – 1P, V – 1P, I – 2P, II – 2P, III – 2P, IV – 2P, V – 2P e poi si ricomincia da capo. La Fig. 1 e la Fig. 2 mostrano la schermata iniziale rispettivamente della modalità ad un giocatore (con livello di difficoltà I) e di quella a due giocatori (sempre con livello di difficoltà I).

Una volta scelta la modalità di gioco, per iniziare a giocare basta premere il pulsante del controller di sinistra (quello collegato ad LG ed LS). Ogni manche dura 10 punti, quindi la vince il giocatore che raggiunge quota 10: dopo i nove punti, sul lato del giocatore che segna i 10 appare la lettera W (winner, ossia vincitore) e un avviso acustico segna la conclusione della partita. Notate che

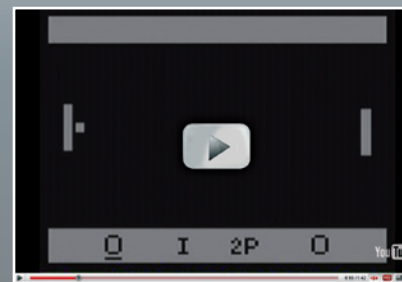
l'impostazione di gioco fatta si conserva anche spegnendo e riaccendendo il circuito con il pulsante SW; viene invece persa sconnettendo le pile.

## REALIZZAZIONE E COLLAUDO

Spiegato il funzionamento del videogame, passiamo alla costruzione: non spaventatevi, perché se un tempo un gioco del genere richiedeva uno stampato di grandi dimensioni, doppia faccia e pieno di integrati, nel nostro caso abbiamo a che fare con qualcosa di estremamente semplice e alla portata di chiunque sappia tenere in mano un saldatore; infatti la basetta è piccola, a semplice ramatura e richiede solamente il microcontrollore e pochi componenti discreti. Questo per la base, ossia la scheda che si connette al televisore; i controller (sono due, quindi due stampati) richiedono una basetta ciascuno, di semplice costruzione e nella quale montare soltanto un potenziometro rotativo ed un pulsante.

In queste pagine trovate i piani di montaggio dei componenti; dopo aver scaricato dal nostro sito web [www.elettronica.in](http://www.elettronica.in) le rispettive tracce, preparare lo stampato base e i due dei controller, quindi iniziate il montaggio dei componenti dalla base. Per primi inserite le resistenze e il diodo al silicio, per il quale raccomandiamo di rispettare la polarità indicata, quindi il pulsante SW1, lo zoccolo per il microcontrollore, il trimmer e i condensatori (prestate attenzione alla polarità del C6). Completate la realizzazione inserendo e saldando il quarzo, le due prese RCA per l'uscita audio e quella video, quindi le quattro clip portatile. Infine, realizzate, con uno spezzone di terminale tagliato da una resistenza o

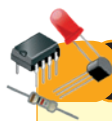
**Guarda il Tennis Game  
in azione su YouTube!**



**Vuoi vedere una demo della console per il gioco del tennis? Collegati all'indirizzo web [http://youtu.be/slf1\\_F5B2tk](http://youtu.be/slf1_F5B2tk) e troverai le schermate principali del gioco, oltre ad assistere ad una partita. Sul nostro canale YouTube sono disponibili i video di moltissimi altri nostri progetti pubblicati in passato.**

un condensatore, il ponticello per decidere lo standard TV: se optate per l'NTSC potete anche non fare alcun ponticello. Inserite nel rispettivo zoccolo il microcontrollore già programmato con l'apposito software ed avrete completato il circuito base (console).

Ora montate, sui due circuiti stampati dei controller i potenziometri e, sul lato opposto, i pulsanti; prendete poi due spezzoni di cavetto bipolare (va bene anche il coassiale, nel qual caso la calza schermo di ciascuno va sul contatto G, quindi LG ed RG) e connettete, rispettando le indicazioni dello schema elettrico e del piano di montaggio, ciascun controller alle rispettive piazzole dello stampato base. In breve, fate in modo che RG dello stampato base vada su G del controller destro ed RS su S dello stesso, quindi connettete LG ad L del controller sinistro ed LS ad S dello stesso controller. Se avete dubbi, riferitevi al disegno di cablaggio che trovate in queste pagine. Sarebbe buona cosa "carrozzare" i controller cercando dei contenitori che abbiano una sagoma compatibile o delle



## per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono di facile reperibilità. Il progetto descritto è anche disponibile in kit. La scatola di montaggio - di produzione Velleman - comprende tutti i componenti, le minuterie ed il micro già programmato. Il kit (cod. MK191) costa 25,00 Euro IVA compresa.

Il materiale va richiesto a:  
Futura Elettronica, Via Adige 11,  
21013 Gallarate (VA)  
Tel: 0331-799775 • Fax: 0331-792287  
<http://www.futurashop.it>



scatole in plastica per elettronica, dalla più diffusa forma rettangolare. Procacciatevi anche un contenitore dove inserire il circuito base, che lavorerete ricavando i tre fori occorrenti a far uscire il LED, le due prese RCA e le due feritoie da cui far passare i cavetti che connettono i controller; se preferite, potete prevedere la possibilità di staccare i controller, nel qual caso vi basta collegare le relative piazzole del circuito stampato a due prese jack montate su un pannello del contenitore. Alle estremità dei cavetti dei controller salderete dunque uno spinotto jack adatto alle prese (suggeriamo prese e spinotti mono da 3,5 mm). Per meglio gestire l'unità, sarebbe opportuno montare il pulsante di accensione e selezione dei giochi su un pannello del contenitore; in questo caso è conveniente usare, invece di un pulsante da c.s. come previsto, uno di quelli a pannello, con dado o clip di fissaggio. Anche il LED dovrà es-

sere prolungato o collegato allo stampato mediante due spezzoni di filo, allo scopo di renderlo visibile dall'esterno. Completato l'assemblaggio della console e dei controller, inserite le pile stilo nei rispettivi alloggiamenti, badando di rispettare la polarità indicata, altrimenti il videogioco non si accenderà; fatto ciò, procuratevi un cavetto audio/video con connettori RCA (in linea di massima potreste usare anche un cavetto audio stereo di buona qualità, che permette il transito della banda video) e con esso collegate il circuito all'ingresso AV del televisore o monitor videocomposito, badando di connettere l'audio alla RCA audio e il video alla presa RCA siglata VIDEO. Se il TV non dispone di prese RCA d'ingresso, procuratevi un cavo con la SCART da un lato e gli RCA dall'altro, ovvero abbinare al cavetto AV con gli RCA un adattatore SCART di ingresso (SCART IN) o completo (ingres-

so e uscita) dotato di prese RCA; la scelta dell'adattatore è importante in quanto la connessione SCART supporta sia gli ingressi sia le uscite audio e video ed in commercio se ne trovano di completi, di provvisti delle sole connessioni di uscita e di forniti delle sole connessioni di ingresso. Quindi se ne comperate uno con le sole connessioni di uscita, non potete far funzionare il videogioco.

Bene, realizzate le connessioni e acceso il TV o monitor videocomposito, potete accendere il circuito premendo il pulsante SW1, fino a veder illuminare il LED; contemporaneamente, lo schermo deve mostrarvi il quadro del gioco del tennis. Ruotando in un verso e nell'altro il cursore del trimmer presente sul circuito stampato base, regolate la luminosità dell'immagine fino ad ottenere il livello ottimale. Adesso siete pronti per giocare. ■

**Tutti i nostri progetti,  
i master, i firmware  
e tanto altro ancora su**  
**www.elettronica.in.it**  
**Collegati subito!**



# PER UNA SETTIMANA PESCARA CAPITALE DELLA ROBOTICA EDUCATIVA

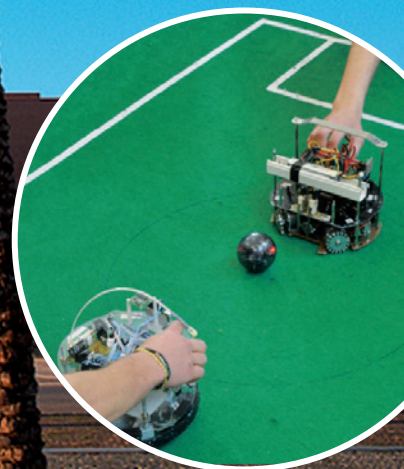
**Fervono i preparativi per la finale nazionale della RoboCup Junior Italia che quest'anno si svolgerà nella città abruzzese e che vedrà la partecipazione di un numero record di squadre.**

di ARSENIO SPADONI

**D**opo Torino, Vicenza, Catania e Riva del Garda(TN), quest'anno sarà Pescara – dal 17 al 20 di aprile - ad ospitare la quinta edizione della Finale Nazionale della RoboCup Jr Italia, manifestazione valida anche per selezionare i team (di alcune categorie) che prenderanno parte alla finale mondiale che quest'anno si svolgerà a Eindhoven (Olanda) dal 26 al 30 giugno.

La RoboCup e la RoboCup Jr sono tra le più importanti e diffuse manifestazioni a carattere internazionale ed hanno come scopo la diffusione della Robotica ed in particolare della Robotica Educativa tra i giovani. Originariamente la manifestazione era riservata a team universitari i cui robot erano (e sono) particolarmente complessi e costosi ed il cui scopo era quello di raggiungere, attra-

verso delle partite di calcio (soccer), le capacità necessarie per sfidare (e magari battere) un team di umani; in seguito, per avvicinare alla Robotica il maggior numero di studenti, sono state istituite altre specialità (Rescue, Dance, Theatre) ed è nata la RoboCup Junior riservata agli studenti con meno di 19 anni (Under 19, Secondary) e con meno di 14 anni (Under 14, Primary). Da quando, poi,





## First Lego League Italia, buona la prima

È iniziata anche in Italia l'avventura della First Lego League, la manifestazione di scienza e robotica che vede in gara squadre di ragazzi tra i 10 e i 16 anni (dalla quarta elementare alla seconda superiore) che – sfruttando i robot Lego Mindstorms opportunamente configurati e programmati – realizzano applicazioni in grado di risolvere piccoli e grandi problemi di natura sociale, economica ed ambientale.

Nel nostro paese l'organizzazione della First Lego League è stata affidata al Museo Civico di Rovereto, ed è proprio al Palasport di Rovereto che sabato 15 dicembre si è svolta la prima gara del Campionato 2012-2013, la qualificazione riservata alle squadre del Nord/Est. C'era molta attesa per la prima gara italiana, nonostante il successo del test che si era svolto in primavera a Riva del Garda in occasione della finale nazionale della RoboCup Jr Italia.

Diciamo subito che l'organizzazione è stata sicuramente all'altezza, sia sotto l'aspetto logistico che per quanto riguarda il rispetto del format americano. Ma soprattutto – ed è questa la cosa che conta maggiormente – le 18 squadre in gara hanno dimostrato di aver raggiunto un ottimo livello, sia nella gara robotica vera e propria che nella presentazione delle strategie adottate, come nella soluzione del tema scientifico assegnato, comune (a livello mondiale) a tutti i team iscritti al Campionato 2012-2013, e che quest'anno riguarda lo studio e la realizzazione di soluzioni in grado di migliorare la qualità di vita degli anziani.

Certo, chi non ha mai assistito ad una qualsiasi gara della First Lego League, anche solo in video, entrando al Palasport di Rovereto, magari a gara iniziata, avrà avuto in un primo momento l'impressione di assistere ad un evento di tutt'altro genere: musica rock “a palla”, megaschermi, DJ in azione, tecnologia a volontà. Insomma, un tipico spettacolo all'americana, con ritmi incalzanti, arricchito con contributi dal vivo e filmati. Questo grande dispiego di tecnologia ha comunque consentito al folto pubblico presente sugli spalti del Palasport di partecipare più da vicino alle gare, grazie in particolare alle immagini proiettate sul maxi schermo. Appuntamento per la finale nazionale l'8 e 9 di marzo, sempre a Rovereto (TN).



la valenza didattica dell'uso della Robotica nell'apprendimento delle materie curriculari (Robotica Educativa) è stata ampiamente sperimentata e dimostrata, anche i bambini delle scuole elementari e addirittura delle scuole d'infanzia sono stati coinvolti in queste manifestazioni; non a caso, alla finale di Pescara è stata istituita una categoria (assolutamente non competitiva) riservata ai bambini con meno di 8 anni (Under 8). Alla finale nazionale di Pescara prenderanno parte quasi 200 team provenienti da tutta Italia che si sfideranno nelle seguenti categorie:

### Under 19

- Rescue A
- Rescue B
- Dance e Theatre
- Soccer A Light

### - Soccer A e B Open

Le prime tre categorie sono valide per le qualificazioni alla finale mondiale di Eindhoven.

### Under 14

- Rescue
  - Dance e Theatre
- Entrambe valide per la finale mondiale.

### Under 8

- Rescue
- Theatre

Solo prove dimostrative di cui non sarà stilata alcuna classifica.

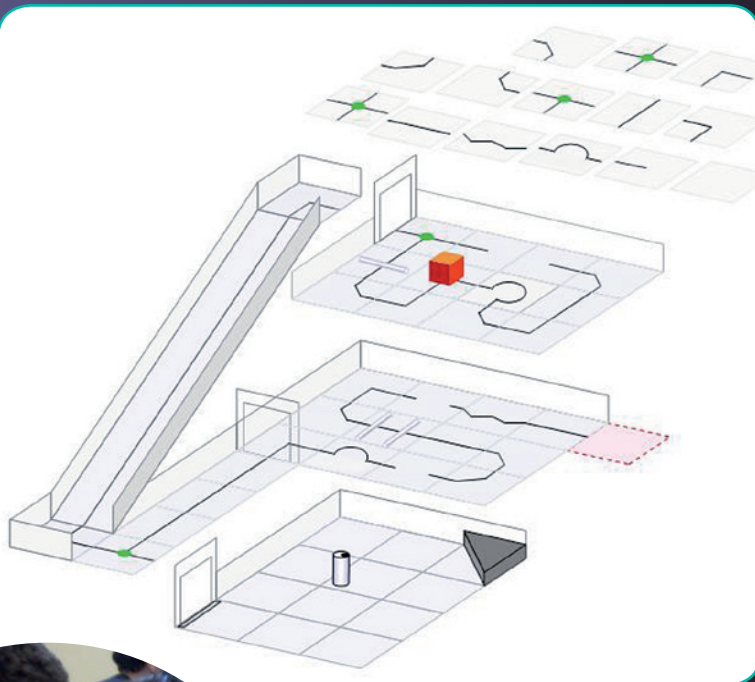
La RoboCup Junior Italia è promossa dalla “Rete di scuole per la Robocup Jr ITALIA” che ha come obiettivo la “Diffusione della Robotica Educativa come metodologia didattica innovativa, trasversale e multidisciplinare, a ca-





rattere fortemente laboratoriale, capace di avvicinare, coniugare e fondere i saperi umanistici e scientifici". Ogni anno l'organizzazione vera e propria della manifestazione nazionale viene affidata ad uno degli Istituti scolastici affiliati; quest'anno il testimone è passato all'IIS "Volta" di Pescara che si avvarrà della collaborazione di altri Istituti scolastici presenti in zona, specie per quanto riguarda la gestione dell'ospitalità dei team che prenderanno parte alle gare.

Come location è stata scelta Montesilvano, una cittadina balneare ai confini di Pescara; le gare di Rescue e Soccer si svolgeranno presso il nuovo polo congressuale Palco, il Palazzo Congressi d'Abruzzo, in un'area che concentra nel raggio di soli 500 metri 9 hotel a 4 e 3 stelle per un totale di 4.000 posti letto e ristoranti per 5.000 coperti. È sicuramente il luogo ideale per questo genere di manifestazioni perché tutto è a distanza pedonale, senza problemi di logistica e trasporti.



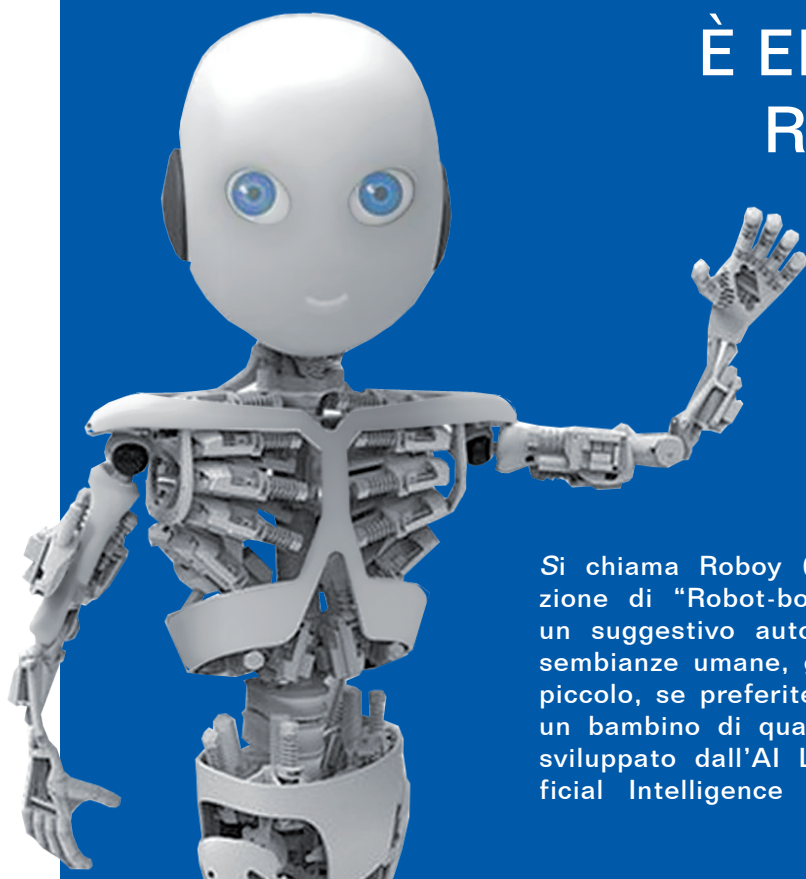
Il Centro Congressi d'Abruzzo, progettato da Aldo Rossi, dispone di una sala congressi plenaria da 3.000 posti, sale congressi minori da 10 a 300 posti e ampi spazi espositivi.

Le gare di Dance e Theatre si svolgeranno invece nella vicina multisala The Space, luogo più adatto per questo genere di competizioni, che fa parte del complesso Porto Allegro, distante qualche centinaio di metri dal Palacongressi.

Durante i giorni delle gare si terrà anche il convegno Roboscuola 2013 con la presentazione da parte degli insegnanti che ne sono stati protagonisti, di numerose esperienze di Robotica Educativa nella scuola dell'obbligo e nella scuola secondaria tecnica e professionale. Maggiori informazioni sulla manifestazione e sulla "Rete di scuole per la RoboCup Jr Italia" sono disponibili sul sito [www.robocupjr.it](http://www.robocupjr.it) dove sono anche disponibili i nuovi regolamenti di gara 2013 che per alcune categorie prevedono sostanziali modifiche rispetto alle precedenti regole. □







## È ELVETICO L'ULTIMO ROBOT UMANOIDE

Si chiama Roboy (abbreviazione di "Robot-boy") ed è un suggestivo automa dalle sembianze umane, grande (o piccolo, se preferite...) come un bambino di quattro anni, sviluppato dall'AI Lab (Artificial Intelligence Laborato-

ry) dell'Università di Zurigo (Svizzera). Roboy vedrà il suo debutto ufficiale durante una presentazione nel corso dell'evento "Robots on Tour" volta ad illustrare le qualità del progetto, che è stato coordinato dal professore Rolf Pfeifer.

La caratteristica principale che distingue Roboy dal resto dei robot umanoidi costruiti fino ad oggi è la sua capacità di replicare con precisione i movimenti degli umani, grazie alla sua particolare struttura: ogni movimento è infatti ottenuto da tendini, giunti, ossa

## EV3: COSÌ LEGO MINDSTORMS DIVENTA OPEN

Tra le novità del CES 2013 di Las Vegas, spicca quella di LEGO, che ha presentato la nuova versione della sua piattaforma Mindstorms, creata principalmente per l'ambiente educational, ma molto apprezzata anche dagli utenti "comuni" che si diletano creando con i popolari mattoncini colorati. La novità si chiama LEGO Mindstorms EV3 (EV3 sta per Evolution 3) e rispetto alla tradizionale Mindstorms NXT è nata per essere "hackerata"; in altre parole nasce come piattaforma open, compatibile anche con iPhone e iPad. LEGO Mindstorms EV3 è un kit che permette di creare fino a diciassette diversi robot con funzionalità meccaniche di vario genere, anche in grado di replicare i movimenti degli arti umani. LEGO

Mindstorms EV3 comprende centinaia di mattoncini LEGO, quattro motori elettrici e cinque sensori, tra cui una nuova unità ad infrarossi che può essere usata per dare la vista ai robot o ricevere comandi a distanza da un telecomando IR.

La terza evoluzione della serie Mindstorms presenta inoltre la compatibilità totale con iOS, quindi i robot creati con essa potranno interagire con l'iPhone attraverso una connessione wireless Bluetooth in modalità protetta; non



a caso, per la prima volta LEGO fornirà un manuale utente con le istruzioni di montaggio 3D in formato App per iPad, realizzato in collaborazione con Autodesk.

Altra novità rilevante è il nuovo mattone EV3 programmabile, che è il cuore di ogni progetto Mindstorms: rinnovato internamente ed esternamente, si basa su un microprocessore con core ARM9 dotato di 64 MB di RAM e 16 MB di Flash per i programmi, e dispo-

e muscoli riprodotti artificialmente, che, secondo gli scienziati svizzeri che li hanno sviluppati, dovrebbero permettere una flessibilità e un'abilità che molto si avvicinano a quella degli esseri umani. Questi tendini artificiali funzionano allo stesso modo dei tendini organici che connettono ossa e muscoli naturali nei vertebrati. Roboy è alto 1,20 metri ed ha una testa con un viso i cui tratti sono quelli di un bambino; questo aspetto lo rende familiare e ne favorirà l'integrazione in un ambiente domestico. Infatti i robot umanoidi e concepiti per l'assistenza alle persone devono avere in primis un aspetto rassicurante.

con la quale LEGO promette di aprire il sistema, per quanto possibile, agli hacker (cioè gli utenti che personalizzano prodotti standard) con il rilascio di una documentazione dettagliata e un SDK che sarà reso disponibile entro la fine dell'anno; la sola parte che resterà proprietaria sarà il chip che supporta l'accesso Bluetooth da iOS.

Per programmare le creazioni robotiche, è possibile immettere i comandi direttamente nel mattone EV3 attraverso il suo display LCD oppure utilizzare il software per PC in dotazione.

In aggiunta a quella standard, LEGO sta lanciando una versione Educational, che contiene una raccolta di kit progettati per l'uso nelle scuole e presenta alcuni componenti in più rispetto alla versione standard, come ad esempio un giroscopio, che consente la costruzione di un balancing robot tipo Segway. La versione Educational verrà fornita con batterie ricaricabili agli ioni di litio per



ne di uno slot per SD-Card che permette di ampliare la memoria disponibile.

C'è anche una porta USB che consente di collegare qualsiasi dongle Wi-Fi.

Il tutto viene governato con una distribuzione di Linux, il sistema operativo open source per eccellenza, prima mossa della strategia

il mattone programmabile.

LEGO Mindstorms EV3

sarà in vendita dalla seconda metà di quest'anno ad un prezzo di circa 350 dollari. Per quanto riguarda la versione Educational, il

costo parte da 5.000 dollari ma il set contiene elementi per far lavorare, in teoria, un'intera classe di 24 studenti a gruppi di due.

# L'AUTO ROBOTICA VIAGGIA SUL SICURO

Si chiama *Active Safety Research Vehicle* ed è una Lexus (marchio detenuto dalla Toyota) presentata all'ultimo CES di Las Vegas e descritta nel sito web della casa giapponese (alla pagina <http://pressroom.lexus.com/releases/toyota+lexus+advanced+active+safety+research+vehicle+ces+jan7.htm>). Questa particolare autovettura è equipaggiata con un sistema di guida autonoma o semiautonoma in grado sia di dare assistenza al conducente nella prevenzione degli incidenti, sia di sostituirlo, in un prossimo futuro.

Già le Lexus attualmente in circolazione dispongono di telecamere, radar, dispositivi di rilevamento laser e GPS per esplorare i punti ciechi (le zone che il conducente normalmente non vede) rilevare l'invasione di corsia e regolare automaticamente la velocità in base alla distanza dal veicolo che precede (sistema ACC=Adaptive Cruise Control); questa nuova versione gestisce tutte le funzioni esistenti e le integra con altri sensori, riportando tutto ad un potente computer, nel quale un apposito software consente di interagire con i comandi impartiti dal conducente, correggendoli quando occorre.

Il computer rileva la presenza di altri veicoli nelle vicinanze e ne analizza il comportamento, ma è anche capace di distinguere la luce proiettata dai semafori, rimediando alle distrazioni del conducente mediante attuatori che interagiscono con la meccanica e l'elettronica che governano la vettura.

Tra i sensori di cui questa vettura sperimentale è dotata, ci sono:

- un'unità LIDAR a 360° (il LIDAR è un misuratore di distanza e velocità a laser) sul tetto del veicolo che rileva gli oggetti intorno alla

macchina fino a circa 70 metri;

- tre telecamere ad alta definizione a colori in grado di rilevare oggetti a circa 150 metri di distanza, usate anche per "leggere" i semafori;
- radar sul frontale e sui lati del veicolo per misurare la posizione e la velocità di oggetti circostanti, così da creare un campo di visione globale agli incroci;
- encoder rotativo situato su una ruota posteriore per rilevare i parametri di viaggio (distanza percorsa e velocità del veicolo);
- IMU (Inertial Measurement Unit), ossia un'unità di misura inerziale posta sul tetto, che misura le variazioni di accelerazione e di imbardata per determinare il comportamento del veicolo;
- antenne GPS sul tetto per determinare direzione e orientamento ancora prima che il veicolo si metta in movimento.

Simile alle vetture a guida autonoma concepite per Google Street View, la Lexus è una piattaforma di sviluppo finalizzata alla creazione, in un futuro non molto lontano,

di una vettura capace di muoversi autonomamente sulla strada e nel traffico, sebbene nell'immediato il fine ultimo del progetto sia testare e migliorare i sistemi di assistenza alla guida che già esistono. Il progetto della Toyota è rivolto alla creazione di un sistema di trasporto intelligente (ITS), integrando l'auto con l'ambiente di guida, e nella prospettiva della comunicazione Veichle To Veichle. Per accelerare lo sviluppo di infrastrutture di comunicazione V2V, Toyota ha iniziato nel novembre scorso la costruzione di un ambiente di prova all'interno dello Higashi-Fuji Technical Center a Susono City (Giappone) che replica strade e semafori e simula situazioni di vita reale del traffico con altri veicoli, pedoni e dispositivi di controllo del traffico.

Nella simulazione, i veicoli sono collegati tra loro e con l'infrastruttura stradale mediante collegamenti radio a corto raggio, allo scopo di verificare e prevenire le collisioni imminenti con veicoli agli incroci, con quelli che cambiano corsia, nonché i tamponamenti. □





# CENTRALE ANTIFURTO 24 ZONE

di ALESSANDRO SOTTOCORNOLA

Sistema d'allarme adatto a sensori sia cablati che wireless, è facile da programmare e può essere dotato di modulo GSM per il teleallarme e la configurazione da remoto. 2<sup>a</sup> e ultima puntata.



**D**opo avervi descritto il circuito elettronico nonché le fasi di realizzazione del nostro antifurto a 24 zone, è giunto il momento di concludere il discorso spiegando la fase di collaudo della centralina e analizzando l'uso della tastiera locale, la programmazione proprio mediante tale tastiera e la configurazione da remoto, effettuabile se decide-

te di dotare l'impianto del modulo GSM da montare sul retro del circuito stampato.

## IL COLLAUDO

Una volta assemblata la centralina e verificato di non avere commesso errori, potete dare l'alimentazione. Inizialmente consigliamo di alimentare la centralina con 15 V dalla morset-



**Tabella 1 - I comandi della tastiera.**

| Tasto | Funzione                                                                                                                                                           |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1     | Carattere 1 sul display.                                                                                                                                           |
| 2     | Carattere 2 sul display.                                                                                                                                           |
| 3     | Carattere 3 sul display.                                                                                                                                           |
| 4     | Carattere 4 sul display.                                                                                                                                           |
| 5     | Carattere 5 sul display.                                                                                                                                           |
| 6     | Carattere 6 sul display.                                                                                                                                           |
| 7     | Carattere 7 sul display.                                                                                                                                           |
| 8     | Carattere 8 sul display.                                                                                                                                           |
| 9     | Carattere 9 sul display.                                                                                                                                           |
| 0     | Carattere 0 sul display.                                                                                                                                           |
| *     | Dopo il codice utente, attiva la protezione perimetrale.                                                                                                           |
| #     | Dopo il codice utente, attiva la protezione full.                                                                                                                  |
| A     | Scorrimento verso l'alto nel menu. Incremento del valore visualizzato.                                                                                             |
| B     | Scorrimento verso il basso nel menu. Decremento del valore visualizzato.                                                                                           |
| C     | Ritorna al menu precedente. Cancella l'ultimo carattere immesso.                                                                                                   |
| D     | Dopo il codice utente visualizza il menu utente. Dopo il codice installatore visualizza il menu installatore. Conferma il dato immesso. Accede al menu successivo. |

tiera "POWER", lasciando per il momento scollegati la batteria, la sirena, i sensori, ecc. Con un tester, verificate che tutte le tensioni siano presenti.

Fornita l'alimentazione alla centralina, verificate che LD1 emetta una serie di lampeggi e che successivamente LD2 faccia altrettanto; se ciò non accade, rimuovete subito l'alimentazione ed effettuate di nuovo tutte le verifiche.

Nel caso, invece, tutto sembri funzionare, si può agire sul trimmer multigiri R7 per regolare il contrasto del display; a regolazione avvenuta, rimuovete l'alimentazione, attendete qualche secondo per fare in modo che tutti i condensatori si scarichino e quindi applicate di nuovo l'alimentazione. Adesso guardate il display, nel quale oltre alla versione firmware verrà mostrato il risultato della fase di test che avviene ad ogni accensione; in questa fase viene verificata l'integrità dei componenti fondamentali del sistema, ovvero il micro principale accerta che il PIC dedicato alla sezione ingressi sia operativo,

che l'RTC funzioni regolarmente e che il modulo GSM sia o meno presente. Nel caso venga mostrata la scritta "ERRORE" accanto al nome di uno di questi elementi, significa che non è stato montato oppure risulta difettoso; la scritta "OK" indica invece che il componente ha passato il test. Notate che il test avanza anche in caso di errore, quindi durante la verifica non dovete perdere d'occhio il display.

Se la parte digitale passa il test, procuratevi un tester disposto sulla misura di tensione e per prima cosa verificate che su "BATTERY" sia presente una tensione compresa tra 13,5 e 14,5 V, dopo di che passate al morsetto "SIREN", su cui non dovrebbe esserci alcuna tensione; portate i puntali su "12V\_AUX" e verificate che vi sia una tensione compresa tra 12 e 13V. Concludete il test verificando che sul terminale "+12V" sia presente la stessa tensione rilevata su "12V\_AUX". Se dopo aver effettuato queste verifiche non avete riscontrato anomalie, potete completare i collegamenti inserendo la batte-

ria, la sirena, il LED di stato e i sensori che desiderate.

## LA TASTIERA E IL DISPLAY

La centralina si interfaccia con l'utente mediante la tastiera e il display.

Nella tastiera ci sono 16 tasti, le cui funzioni sono descritte nella **Tabella 1**; i numeri servono per digitare il codice di accesso, oppure per immettere dei valori numerici durante la configurazione. I tasti rimanenti gestiscono le varie funzioni: "\*" permette, se digitato dopo il codice di accesso da utente, di attivare la centralina in modalità perimetrale, mentre il "#" di attivarla in modalità controllo completo (anche detto "Full"). Nel caso si voglia disattivare la centralina, è sufficiente digitare il codice utente seguito da uno dei due tasti "\*" o "#".

Il tasto "A" può essere tradotto per comodità con "Alto": ciò perché permette di spostarsi in alto nel menu, oppure di incrementare un valore visualizzato durante una configurazione.

Funzione opposta è quella del tasto "B", che sempre per comodità tradurremo con "Basso", ad indicare in decremento oppure uno spostamento verso il basso nei menu.

Il tasto "C" è utilizzato per la cancellazione di un valore a display o per tornare indietro nei menu. Infine il tasto "D" ha anch'esso più funzioni: consente di accedere al menu, oppure all'interno di quest'ultimo consente di dare conferma per le varie configurazioni. Inoltre, se dopo aver inserito il codice utente/installatore si preme "D", si accede al menu di configurazione utente oppure a quello di configurazione installatore, a seconda del codice digitato.

Passiamo adesso al display, il quale può fornire diverse in-

formazioni, ovvero permette di conoscere lo stato della centralina quando si trova in standby, se è stato rilevato un allarme, se è stata avviata la procedura di allarme, oppure, nel caso del menu, indica in quale e a quale livello ci si trova.

```
01/09/2012 - 10.36
Centrale Disattivata
*****
ALLARME!!!
```

Quando la centralina si trova in standby viene sempre visualizzato in prima riga l'orario attuale, in seconda riga viene indicato se la centralina è attiva oppure no, mentre la terza riga è sempre vuota salvo quando si digita un numero servendosi della tastiera; in tal caso viene mostrato un "\*" per non rendere visibile il codice di accesso. In ultima riga invece viene mostrato un messaggio qualora vi sia un allarme in corso. Per quanto riguarda l'accesso al menu, tutte le informazioni appena citate non vengono mostrate, in quanto la priorità è la configurazione; pertanto verrà sempre visualizzato, sempre nella prima riga, il nome del menu in cui ci si trova, mentre nelle righe successive sarà mostrata la possibile scelta per tale menu, evidenziando con una freccia la scelta effettuata.

```
***** MENU *****
->Attiva/Disattiva
Setup
LOG Eventi
```

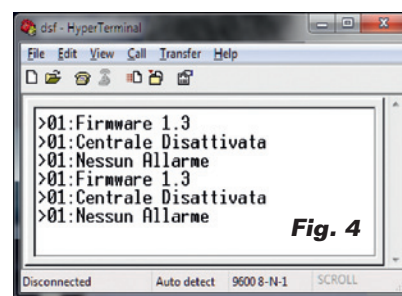
### LE INFORMAZIONI VIA RS485

La centralina può essere collegata ad una postazione remota per monitorarne lo stato; le informazioni da essa fornite possono essere sfruttate per creare

un'interfaccia che le pubblichi in tempo reale su una pagina web. Ciò permetterebbe di conoscerne lo stato del sistema d'allarme accedendo ad Internet da una qualsiasi postazione. La comunicazione RS485 per il momento è unidirezionale ed è utilizzata solo come monitor di stato, ma in futuro non escludiamo di modificare il firmware del micro principale affinché offra ulteriori funzionalità. Periodicamente la centralina, inviata sul bus dati come la versione del firmware, lo stato del sistema d'allarme (ovvero se è attivata oppure no) la modalità in cui è attivata, ma soprattutto se è stato riscontrato un allarme in una zona. Notate che viene indicata solo l'ultima zona allarmata e non l'elenco completo delle zone in cui si è verificato un allarme. Per chiarire meglio alcune situazioni, guardate le schermate del PC ritratte in Fig. 4 e Fig. 5; si riferiscono a due videate di Hyper Terminal che mostrano dati forniti dalla centralina via RS485.

Come spiegato in precedenza, ogni informazione fornita serialmente è composta da un header, il cui carattere è ">" seguito dall'ID della centralina e quindi dall'informazione.

Tra le tante informazioni che la centralina può fornire, è stata prevista, se avete montato il modulo GSM, la possibilità di monitorare ogni SMS ricevuto dall'unità. Questa soluzione viene utile per due motivi: in primo luogo perché è possibile dal lato PC conservare tutto lo storico degli SMS; in secondo luogo, per sapere quali utenti hanno cerca-



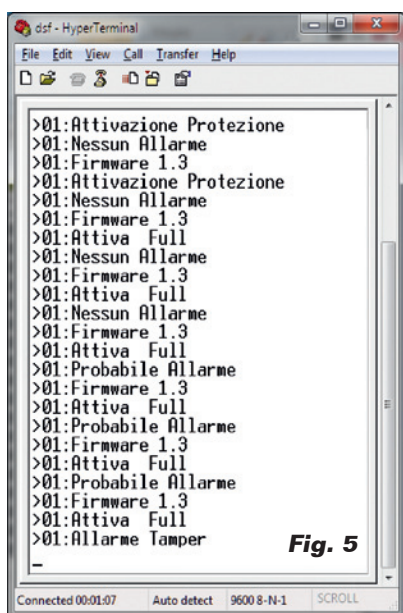
to di accedere all'unità. Infatti viene inviata sul bus una serie di informazioni riguardante l'SMS, ovvero il numero di telefono del mittente, se questo utente era autorizzato (la presenza della password utente o installatore considera l'utente autorizzato), la posizione in memoria dell'SMS ricevuto e il testo dell'SMS. Pertanto le informazioni saranno riassunte come mostrato in Fig. 6. Se siete in grado di realizzare un software di monitor di queste informazioni, potete venire avvisati a video in caso di intrusione, ad esempio, in una zona perimetrale.

### IL MENU

La centralina prevede due codici differenti: il primo è dedicato all'installatore e permette di gestire interamente i sensori, dalla memorizzazione, alla cancellazio-







**Fig. 5**

ne del singolo sensore o di tutti i sensori associati alla zona; il codice di accesso (per impostazione predefinita è 123456) può essere variato a proprio piacimento e deve essere obbligatoriamente di 6 caratteri. L'importante è che non sia uguale a quello dell'utilizzatore finale.

Per quanto concerne l'utente, o meglio l'utilizzatore della centralina, è previsto un codice sempre a 6 cifre (quello predefinito è 000000) che permette sia di attivare/disattivare la centralina, sia di operare la riconfigurazione di alcuni parametri, ma non consente in alcun modo di intervenire sulla memoria sensori o di effettuare reset di alcun tipo. L'unico reset fattibile è a livello hardware e permette di azzerare l'intera memoria (vedere il riquadro "Parametri predefiniti"). L'utente, come si vede dalla struttura ad albero del menu, può visionare il Log degli Eventi, parzializzare le zone desiderate e intervenire sulla configurazione, ovvero può attivare o meno il buzzer, gestire la retroilluminazione del display, variare i tempi di attesa nel caso venga rilevato

un allarme o il tempo che la sirena può suonare. Inoltre gli è consentito di cambiare la password: quella utente, s'intende. Notate che se viene montato il modulo GSM (cod. TDGGS-M\_900), si può pensare di gestire anche la lista dei numeri di telefono (massimo 6 numeri) da avvisare in caso di allarme; a questi numeri verrà inviato un SMS di segnalazione della condizione d'allarme.

Il sistema prevede una funzione di test dei componenti, la stessa che è visibile anche all'utente durante la fase di avvio del sistema e che permette di identificare se uno dei componenti principali è difettoso o mancante.

Bene, adesso spieghiamo i singoli menu, con la premessa che i tasti "A" e "B" servono per scorrere nei menu o incrementare e decrementare un valore visualizzato a display, mentre il tasto "C" permette di cancellare il valore immesso o tornare al menu precedente; invece "D" permette di memorizzare il dato che si è variato. Ogni qualvolta si accede ad una sezione che necessita una possibile configurazione, viene automaticamente mostrato il valore attualmente impostato nella centralina.

Ma vediamo, ora, i singoli menu della centralina, spiegandoli nel dettaglio. Innanzitutto, appena fornita l'alimentazione è possibile visionare i dettagli della centralina, ovvero il modello e la versione del firmware, come mostrato qui di seguito.

**CENTRALINA ALLARME**  
mod. ACU2012

Ver:1.3 23-09-2012

Trascorsi circa due secondi verrà avviata la fase di test del siste-

ma, durante la quale il display mostrerà lo stato di avanzamento e l'esito delle prove; ciò vi permette di capire se qualcosa non funziona come deve. Ricordiamo che qualora non sia stato montato il modulo GSM, è normale che il display segnali un errore sulla parte GSM. È invece importante assicurarsi che gli altri due componenti superino il test; in caso contrario provate a spegnere e riaccendere la centralina.

\*\*\*\*\* TEST \*\*\*\*\*  
PIC I/O...OK  
RTC...OK  
GSM...ERRORE

Terminata la fase di test verrà mostrata la data e l'ora del sistema, e nel caso fosse il primo avvio, tale data verrà automaticamente impostata al primo gennaio 2012. Inoltre, nella seconda riga del display verrà mostrato lo stato della centralina: se non attiva, l'indicazione sarà "Centralina Disattivata"; se in fase di attivazione, uscirà "Attivazione Protez...", mentre se attiva in modalità perimetrale il messaggio mostrato sarà "Attivata Perimetro", oppure "Attiva Full" se è attiva su tutte le zone scelte.

24/09/2012 - 19:27  
Centrale Disattivata

La terza riga viene utilizzata per mostrare il codice inserito, rappresentato da asterischi, mentre nella quarta riga troverete una serie di informazioni; ad esempio può comparire la scritta generica "ALLARME!!!" ad indicare l'avvenuto allarme in una delle 24 zone, oppure un messaggio più significativo, ad esempio di indicazione dell'allarme di mancanza rete 220V, oppure di

un allarme di manomissione o, ancora, dell'allarme riguardante la funzione "panico".

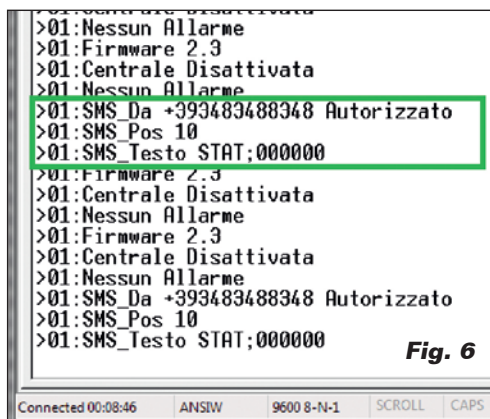
```
24/09/2012 - 19:27
Centrale Disattivata

ALLARME 24h!
```

Per quanto concerne i menu, partiamo ora con il menu utente, accessibile digitando "000000" + "D". Notate che da adesso in poi mostriamo anche il display esteso, ovvero l'elenco completo delle scelte possibili del menu, senza badare alla reale dimensione del display. Nel caso il menu sia esteso oltre la prima videata del display, sarà sufficiente premere il tasto "A" o "B" per spostarsi all'interno di esso.

```
***** MENU *****
->Attiva/Disattiva
Setup
LOG Eventi
Zone ON/OFF
```

Come si vede, dal menu è possibile attivare o disattivare la centralina in modalità Full, confermando con il tasto "D" la scelta "Attiva/Disattiva". Se invece si vuole accedere al menu di configurazione, bisogna scegliere "Setup". Da "LOG



Eventi" è possibile visualizzare o cancellare la lista di eventi, ovvero l'indicazione degli avvenuti allarmi, mentre in "Zone ON/OFF" si possono parzializzare le 24 zone. Dato che la prima voce del menu è già stata spiegata, passiamo al menu "Setup", che ci appare sul display come segue.

```
***** SETUP *****
->Data/Ora
Buzzer tastiera
Tempo attesa
Tempo suono sirena
Retroilluminazione
Password User
Rileva alim. Rete
Numeri di telefono
```

In "Data/Ora" è possibile variare le relative impostazioni; in particolare, si potrà sfruttare i tasti "A" e "B" per scegliere il parametro da impostare e "D" per modificarlo. Dopo aver scelto il parametro da modificare, si potranno usare sempre i tasti "A" e "B" per incrementare o decrementare data e ora. In questo caso è stato fatto un controllo sui mesi da 30 e 31 giorni, ma non sull'anno bisestile, quindi fate attenzione ai dati immessi.

```
***** DATA/ORA *****
24/09/2012 20:26:06

Sposta(A,B) Ok(D)
```

Dal menu "Buzzer tastiera" si potrà decidere se attivare o meno il buzzer ad ogni pressione di un tasto; questa funzione è utile per sapere se il sistema ha rilevato la pressione dei tasti quando li abbiamo premuti. La tastiera è dotata di funzione antirimbombo, quindi anche se un tasto viene mantenuto premuto il coman-

do corrispondente non viene ripetuto.

```
***** BUZZER *****
->On
Off
```

La funzione "Tempo attesa" permette di impostare il tempo durante il quale l'utente ha la possibilità di disattivare la centralina o il tempo che la centralina attende prima di attivarsi; in questo modo, anche se un sensore va in allarme perché l'utente sta uscendo di casa, viene ignorato. Il tempo è impostabile a scaglioni di 10 secondi, da un minimo di 20 ad un massimo di 250 secondi. Anche in questo caso la pressione dei tasti "A" e "B" incrementa o decrementa il valore visualizzato, mentre "D" memorizza l'impostazione fatta.

```
*** TEMPO ATTESA ***

030 secondi
```

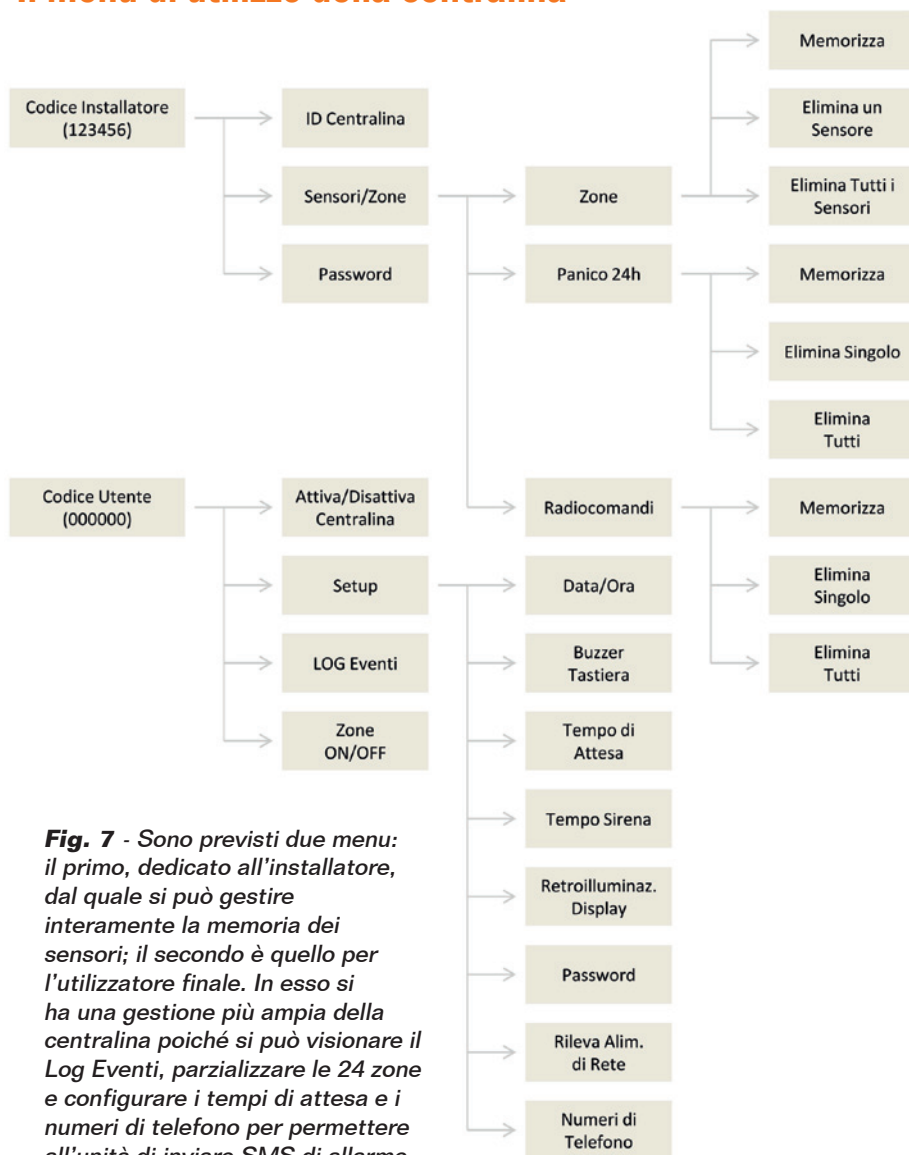
Riguardo al "Tempo sirena", il discorso è simile al precedente, però in questo caso si varia il tempo per cui la sirena deve suonare.

Notate che il suono non sarà intermittente, ma continuo per tutto il periodo impostato.

A tal proposito va detto che la normativa italiana in materia di sistemi di antifurto, che fa riferimento alla norma CEI79-2 (cui rimandiamo per verificare tutti i dettagli inerenti al tipo di sirena da utilizzare) prevede che la sirena non possa suonare per più di 4 minuti.

Per questa ragione il tempo di attivazione della sirena non può andare oltre 4 minuti.

## Il menu di utilizzo della centralina



**Fig. 7** - Sono previsti due menu: il primo, dedicato all'installatore, dal quale si può gestire interamente la memoria dei sensori; il secondo è quello per l'utilizzatore finale. In esso si ha una gestione più ampia della centralina poiché si può visionare il Log Eventi, parzializzare le 24 zone e configurare i tempi di attesa e i numeri di telefono per permettere all'unità di inviare SMS di allarme se il modulo GSM è installato.

\*\*\* TEMPO SIRENA \*\*\*  
030 secondi

Per la retroilluminazione del display è stata prevista la possibilità di renderla sempre attiva oppure di disattivarla, pertanto non è ad esempio possibile gestirla automaticamente o controllarne il livello di luminosità. In questo caso la retroilluminazione è stata calibrata per rendere ben visibile il display e ridurre i consumi di corrente al minimo.

\*\*\* LUCE DISPLAY \*\*\*  
→Attivata  
Disattivata

Il menu "Password User" permette di variare la password attualmente impostata per accedere alla configurazione, che poi è la stessa utilizzata per attivare/disattivare la centralina. Alla digitazione della password, questa verrà mostrata in chiaro nel display per fare in modo che possiate verificare i caratteri introdotti.

\*\*\*\*\* PASSWORD \*\*\*\*\*  
Nuova Password  
—

In qualsiasi momento è possibile decidere se attivare o meno il rilevamento della tensione di rete: basta accedere al menu "Rileva alim. rete".

Questa funzione, per quanto sia utile a proteggere la centralina dal sabotaggio (in quanto permette di dare l'allarme quando i ladri tagliano i cavi della corrente per cercare di neutralizzarla) in certi casi è meglio disinserirla: ad esempio se ci si trova in una zona soggetta a frequenti blackout o se c'è in corso un temporale. In questo caso la disattivazione della funzione potrebbe evitare di disturbare inutilmente il vicinato.

\*\*\* RILEVA ALIM. \*\*\*  
→Attivata  
Disattivata

Per concludere l'analisi del menu di setup utente, vediamo il sottomenu "Numeri di telefono", dal quale è possibile visualizzare i numeri memorizzati (massimo 6 numeri) ed eventualmente modificarli. Se si vuole modificare o inserire un numero è sufficiente posizionarsi sulla posizione desiderata e premere "D", quindi inserire il numero di telefono e confermarlo con "D" o annullare l'immissione premendo "C". Suggeriamo di inserire il numero con il prefisso internazionale, quindi immettere ad esempio 00393491234567 dove "0039" è il prefisso internazionale per l'Italia.



\*\*\* NUMERI TEL. \*\*\*

→1:  
2:  
3:  
4:  
5:  
6:

\*\*\*\*\* NUMERO \*\*\*\*\*

Numero di telefono

—

Tornando indietro di un passo nel menu utente, dopo avere visto le configurazioni possibili, ora possiamo citare il menu "LOG Eventi" che è di notevole importanza perché permette di verificare eventuali allarmi avvenuti e la relativa data e ora. Accedendo al menu in questione si dovrà come prima cosa scegliere se visualizzare o eliminare gli eventi.

Nel caso dell'eliminazione verrà richiesta conferma, questo perché il Log è molto importante.

Se invece si è scelto di visualizzare gli eventi, si potrà accedere ad una lista di 18 eventi.

Nel caso non vi siano dati da mostrare il messaggio "nessun evento registrato" verrà mostrato, invece se vi è almeno un elemento verrà mostrata la lista completa.

Ogni evento sarà accompagnato dalla data e ora e gli eventi possibili saranno i seguenti:

- 24h: Allarme Panico
- I05: Allarme zona interna numero 5
- P06: Allarme zona perimetrale numero 6
- PWR: Allarme mancanza tensione di rete
- MAN: Allarme manomissione

\*\*\*\* LOG EVENTI \*\*\*\*

→24/09/12-21:18>24h  
25/09/12-09:10>I05  
25/09/12-09:10>P06  
25/09/12-09:12>PWR  
25/09/12-09:13>MAN  
-----

Per concludere il menu utente, accedendo al menu "ZONE ON/OFF" è possibile parzializzare la zona desiderata, ovvero fare in modo che al momento dell'attivazione una zona venga esclusa dalla verifica, magari perché il sensore si è danneggiato, oppure perché in quel momento in quel locale c'è qualcuno o un animale. Per modificare la zona è sufficiente selezionare la zona scelta e premere "D" quindi, scegliere se attivarla o meno. Questa impostazione sarà sempre valida salvo che non si cambi di nuovo l'impostazione.

\*\*\* ZONE ON/OFF \*\*\*

→Zona 01 Attivata  
Zona 02 Disattivata  
Zona 03 Attivata

Passiamo ora al menu dedicato all'installatore, in questo caso accessibile con la password "123456" seguita dal tasto "D". Questo menu sarà dedicato principalmente alla gestione dei sensori.

\*\*\* INSTALLATORE \*\*\*

→ID centralina  
Sensori/Zone  
Password

Nel caso si scelga "ID centralina" sarà possibile impostare un identificativo da 001 a 255 della centralina. Questo dato è

utile nel caso si colleghino più centraline e si voglia tenere sotto controllo la situazione mediante comunicazione RS485. Ogni dato informativo sul bus RS485 è accompagnato proprio da questo indirizzo, in modo da identificare la centralina cui appartiene l'informazione.

\*\*\* ID CENTRALE \*\*\*

Indirizzo: 001

Qualora si scelga "Sensori/Zone" è possibile accedere a tutta l'area di configurazione dei sensori. Come prima cosa bisognerà scegliere quale tipo di sensore o radiocomando si vuole memorizzare; proprio a seconda della scelta fatta, si avrà accesso ad un menu differente.

\*\*\* SENSORI/ZONE \*\*\*

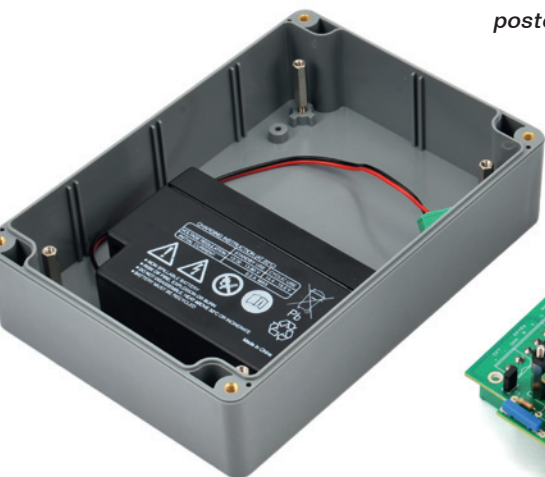
→Zone  
Panico 24h  
Radiocomandi

Scegliendo "Zone" è possibile configurare le zone con sensori a filo oppure gestire i sensori associati alla zona scelta. Nel caso al passo successivo si scelga una zona compresa tra 1 e 4, essendo queste zone associate a sensori collegati via filo, sarà possibile scegliere se configurarle come normalmente aperte (NO) oppure normalmente chiuse (NC); invece, scegliendo una delle altre 20 zone, si accederà alla gestione dei sensori wireless.

\*\*\*\*\* ZONA \*\*\*\*\*

ID Zona:01

*La batteria per la centralina deve prendere posto sul fondo del contenitore plastico; la scheda, distanziata da colonnine, le verrà sovrapposta.*



Come anticipato, scegliendo una delle prime quattro zone si potrà indicare se i sensori devono essere di tipo NO oppure NC.

```
***** ZONA01 *****
→Ingresso NO
Ingresso NC
```

Scegliendo una delle zone comprese tra 5 e 24, si potrà memorizzare ed eliminare un sensore per volta, o tutti quelli della zona.

```
***** ZONA05 *****
→Memorizza
Elimina un sensore
Elimina tutti sens
```

Nel caso si scelga di eliminare tutti i sensori, verrà avviata senza ulteriore conferma la procedura di eliminazione completa; se invece verrà scelto di memorizzare o eliminare un singolo sensore, una successiva schermata attenderà che venga trasmesso un segnale in radiofrequenza. Nel caso in cui durante tale procedura non venga ricevuto alcun segnale, la centralina rimarrà sempre in attesa, quindi si consiglia di premere "C" per uscire dalla procedura. Va precisato che per la fase di memorizzazione sarà l'utente a scegliere se il sensore da memo-

rizzare sarà di tipo "Perimetrale" o ad uso "Interno"; ad esempio un sensore volumetrico.

```
** MEMORIZZA SENS **
Attesa segnale RF...
```

Lo stesso discorso vale nel caso si scelga "Panico 24h" dal menu. In tal caso si potrà scegliere se memorizzare o eliminare i sensori, in questo caso senza alcuna distinzione tra interna e perimetrale perché il sensore in questione è attivo tutto il giorno indipendentemente dal fatto che la centralina sia armata oppure no. Può far parte di questo gruppo un radiocomando di "panico". Fa parte di questa zona anche l'ingresso via filo siglato "24h", che deve obbligatoriamente essere di tipo NC; se non utilizzato, questo va cortocircuitato verso "+12V". Per i radiocomandi, anche in questo caso vale lo stesso discorso sopra riportato, ovvero accedendo alla sezione "Radiocomandi" sarà possibile memorizzare un segnale RF che potrà essere utilizzato per attivare o meno il sistema antifurto. Per concludere il menu in questio-

La centralina può accettare qualsiasi sensore filare dotato di uscita a relé di tipo NO/NC, mentre per quanto riguarda i sensori wireless bisogna obbligatoriamente scegliere tra quelli che hanno una codifica compatibile; a tale riguardo notate che i sistemi Rolling Code e Motorola non sono accettati.

Le codifiche accettate sono la MM53200 (incluse le codifiche compatibili), la HT12 e la Velleman. Abbiamo incluso quest'ultima categoria perché la Casa belga produce un'ampia gamma di sensori a buon prezzo ed alta affidabilità, dato che sono dotati di sistema anti manomissione ed hanno una potenza in trasmissione di 10 mW, che garantisce la copertura di un appartamento da circa 100 m<sup>2</sup> disposto su un piano, oppure anche due livelli uguali. Il prospetto qui sotto riporta la gamma completa di sensori e radiocomandi compatibili con la nostra centralina, distinti per codifica; tutti sono disponibili presso Futura Elettronica ([www.futurashop.it](http://www.futurashop.it)) e tutti compatibili con la centralina antifurto.

L'attivazione/disattivazione della centralina mediante radiocomando prevede l'utilizzo di un solo pulsante, pertanto se usate un trasmettitore che dispone di 2 pulsanti, potete abbinare entrambi alla centralina come se si trattasse di due radiocomandi diversi; in tal modo sarete certi che qualsiasi sarà il tasto premuto, la centrale antifurto risponderà.

Va però tenuto in considerazione che mediante telecomando non è possibile decidere se la centralina deve attivarsi in modalità perimetrale o full; questo per semplificarne l'utilizzo e per impedire che si faccia confusione con il tasto, ma soprattutto perché se ci si trova all'interno della casa, si ha la possibilità di scegliere la modalità di attivazione, mentre se si opta per l'utilizzo del telecomando, si presume che la persona sia fuori casa e voglia quindi attivare la centralina in modalità full (monitoraggio di tutte le zone, interne e perimetrali).

Tutti i sensori e radiocomandi qui proposti, ad esclusione di uno di essi, prevedono la possibi-

ne, se si sceglie "Password" è possibile variare la password installatore con una a piacere, sempre di 6 caratteri.

```
***** PASSWORD *****
Nuova Password
—
```

## UTILIZZO DELLA CENTRALE

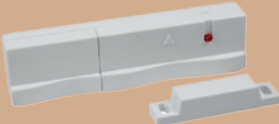
Ora che vi abbiamo spiegato la struttura del menu e l'uso della

lità di impostare manualmente il codice identificativo del sensore. Tale caratteristica può essere utile ad esempio per la zona perimetrale, poiché se si utilizza un contatto magnetico a dip-switch è possibile impostare tutti i sensori, ad esempio quelli del soggiorno, con lo stesso codice. In questo modo sarà necessario memorizzare un solo sensore, ma soprattutto si potranno avere infiniti sensori in quanto hanno tutti lo stesso codice. Nel caso invece il sensore preveda un pulsante di generazione del codice, come il prodotto cod. HAM06WS/DET, risulterà prati-

camente impossibile avere tanti sensori con lo stesso codice; per ovviare a ciò, la centralina prevede la possibilità di memorizzare più sensori per la stessa zona. Per i radiocomandi vale in linea di massima lo stesso discorso, ovvero sono disponibili sistemi con i dip-switch, così se il telecomando si guasta è sufficiente acquistarne uno nuovo e impostare lo stesso codice senza doverlo memorizzare nuovamente nella centralina. Esistono però anche telecomandi con generazione del codice casuale, oppure come per il modello HAM06WS/REMOTE, in

cui non si può impostare nulla perché il produttore memorizza un suo codice che non può in alcun modo essere cambiato. Se si desidera impostare il codice identificativo del sistema di trasmissione, bisogna come prima cosa aprire il guscio del dispositivo, quindi se l'unità dispone di un dip-switch è sufficiente intervenire su di esso anche a dispositivo alimentato; invece se il sensore ha un pulsante di generazione del codice, allora il discorso è diverso. Di norma tutti i sistemi dotati di generazione casuale del codice funzionano nello stesso modo,

pertanto bisogna rimuovere la batteria che li alimenta, quindi va premuto il pulsante e sempre mantenendolo premuto, va collegata di nuovo la batteria attendendo circa 5 secondi che il sistema si inizializzi. Fatto ciò si potrà rilasciare il pulsante. Questa è in linea di massima la procedura standard, ma in taluni casi vanno premuti due tasti in contemporanea; in ogni caso per comprendere la procedura più adatta consigliamo di consultare il manuale fornito a corredo del sensore e se non presente, contattare il produttore o eventuale fornitore.

| CODIFICA | TRASMETTITORI                                                                       |                                                                                     |                                                                                      |                                                                                       |
|----------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
|          | TX3750-2CS                                                                          | TX3750-4CS                                                                          |                                                                                      |                                                                                       |
| MM53200  |    |    |                                                                                      |                                                                                       |
| HT12     | TX12E-4C-SAW433                                                                     | SIR2008HCS-HT12                                                                     |                                                                                      |                                                                                       |
|          |   |   |                                                                                      |                                                                                       |
| VELLEMAN | HAM06WS/DET                                                                         | HAM06WS/MEDICAL                                                                     | HAM06WS/MAGNET                                                                       | HAM06WS/REMOTE                                                                        |
|          |  |  |  |  |

tastiera, facciamo un esempio di come procedere per memorizzare un sensore e fare un primo test; ciò vi aiuterà a capire come configurare il resto dell'antifurto. Come prima cosa memorizziamo un radiocomando e un sensore; in questa fase dovete collegare anche la batteria, così da simulare anche la mancanza di rete. Per verificare anche il sistema di attivazione si consiglia di collegare un LED a 12V al morsetto "LED". Digitate da tastiera "123456" seguita

dal tasto "D", quindi accedete a "Sensori/Zone > Zone" quindi scegliete la zona "05 > Interna" e quando richiesto mandate in allarme un sensore e attendete che questo venga memorizzato. Un messaggio a display ne confermerà l'avvenuta memorizzazione.

**\*\* MEMORIZZA SENS \*\***  
Impostazione  
OK

Ora premete tre volte consecutive il tasto "C" e dal menu "Sensori/Zone" scegliete "Radiocomandi > Memorizza" e, quando richiesto, premete il tasto di un telecomando; nel fare ciò state attenti che in quell'istante non trasmetta alcun altro radiocomando. Quindi accertatevi che eventuali altri TX o sensori wireless siano disattivati. Terminata la memorizzazione del radiocomando, premete più



# Parametri predefiniti

In qualsiasi momento è possibile ripristinare i parametri predefiniti del sistema: basta alimentare la centralina con il tasto P1 premuto.

Quando il display mostra la scritta "RESET..." dovete rilasciare il pulsante. I parametri sono quelli illustrati di seguito.

| Parametro                     | Stato          | Configurazione Possibile        |
|-------------------------------|----------------|---------------------------------|
| Codice Utente                 | 000000         | Qualsiasi purché di 6 caratteri |
| Codice Installatore           | 123456         | Qualsiasi purché di 6 caratteri |
| Monitor alimentazione di rete | Attivo         | Attivato o Disattivato          |
| Buzzer                        | Attivo         | Attivato o Disattivato          |
| Retroilluminazione display    | Attiva         | Attivato o Disattivato          |
| Tempo attesa allarme          | 20             | 020 ÷ 180 secondi               |
| Durata suono sirena           | 30             | 020 ÷ 240 secondi               |
| LOG eventi                    | Nessun Dato    | -                               |
| Memoria sensori               | Nessun Sensore | 4 sensori per ogni zona         |
| Zona 1                        | N.O.           | N.O. oppure N.C.                |
| Zona 2                        | N.O.           | N.O. oppure N.C.                |
| Zona 3                        | N.O.           | N.O. oppure N.C.                |
| Zona 4                        | N.O.           | N.O. oppure N.C.                |
| Zone ON/OFF                   | Tutte Attive   | Attivabili a piacimento         |

volte il tasto "C" fino a tornare in standby.

Ora che nel display compare la scritta "Centrale Disattivata" pigiate il tasto del radiocomando memorizzato e verificate che sul morsetto "LED" il dispositivo luminoso inizia a lampeggiare, segno che ha riconosciuto il segnale del telecomando e sta procedendo all'attivazione della centralina; infatti un messaggio sulla centralina informerà che è in corso l'attivazione mediante messaggio "Attivazione Protez.". Attendete quindi il tempo di attesa impostato e la

centralina sarà attiva; a riprova di ciò, il display visualizzerà il messaggio "Attiva Full".

Ora passate a simulare l'allarme; allo scopo, agite sul sensore memorizzato. Quando il segnale corrispondente verrà rilevato, il display visualizzerà il messaggio "Probabile allarme!". Trascorso il tempo di attesa impostato, dovrete udire il relé scattare; a questo punto, se all'uscita corrispondente avrete collegato una sirena, questa inizierà a suonare per il tempo impostato.

A conferma dell'allarme in

corso, la scritta "ALLARME!!!" apparirà sulla quarta riga del display. Ora, dato che tutto funziona, potete disattivare la centralina, questa volta provando a digitare il codice utente seguito da "#". La centralina si disattiverà ed il relé tornerà nello stato di riposo e la spia luminosa collegata su "LED" si spegnerà.

Durante la fase di attivazione/disattivazione con radiocomando, la spia luminosa lampeggerà, mentre se si digita il codice utente, verrà subito avviata la procedura di attivazione/disattivazione, che nell'altro caso verrà avviata solo dopo 5 secondi circa.

Effettuate ora la medesima procedura per simulare questa volta un allarme di mancanza rete, quindi digitate il codice utente, attendete l'attivazione della centralina e, quando essa si attiverà, rimuovete il morsetto ad innesto "POWER". Fatto ciò, verificate l'avvio della procedura di allarme, che si concluderà con il relativo messaggio di allarme, quindi disattivate la centralina.

Ora potete verificare che gli eventi accaduti siano stati correttamente memorizzati:

**Tabella 2**

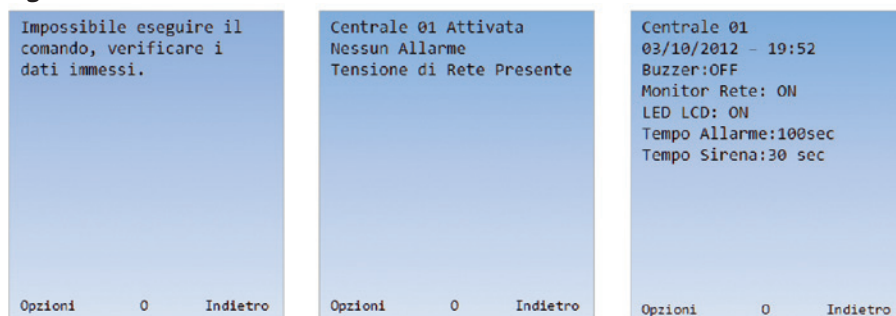
| COMANDO | PARAMETRI                                                                  | DESCRIZIONE                                                                   |
|---------|----------------------------------------------------------------------------|-------------------------------------------------------------------------------|
| ATT=xxx | xxx > Valore compreso tra 020 e 180 secondi                                | Cambia il tempo di attesa per attivazione e/o disattivazione della centralina |
| SIR=xxx | xxx > Valore compreso tra 020 e 240 secondi                                | Cambia il tempo per il quale la sirena deve suonare in caso di allarme        |
| LCD=x   | x > ON: Attiva retroilluminazione<br>x > OFF: Disattiva retroilluminazione | Permette di gestire la retroilluminazione del display                         |
| BUZ=x   | x > ON: Attiva il buzzer<br>x > OFF: Disattiva il buzzer                   | Permette di attivare oppure no il buzzer della tastiera                       |
| STA?    |                                                                            | Richiede lo stato attuale della centralina                                    |
| SET?    |                                                                            | Richiede i parametri impostati nella centralina                               |
| FW?     |                                                                            | Richiede la versione firmware                                                 |

digitate la password utente, ma questa volta premete "D" e accedete quindi al menu "LOG Eventi > Visualizza". Il display dovrebbe mostrarvi gli eventi che sono stati registrati. Bene, se tutto è andato a buon fine si può procedere a memorizzare tutti i sensori desiderati; a riguardo vi consigliamo di verificare che il sensore REED funzioni correttamente, simulando anche in questo caso la procedura di allarme. Per un corretto funzionamento dovete posizionare, a centralina spenta, un magnete quasi a contatto con il sensore; il magnete dovrà essere fissato al coperchio. Ora rimuovete il jumper J1 e chiudete il contenitore.

### PROGRAMMAZIONE VIA SMS

La centralina antifurto, come anticipato nei precedenti paragrafi, può essere dotata di un modulo GSM che, se presente, permette l'invio di un semplice SMS ai numeri abilitati per avvisare dell'avvenuto allarme; ma questa è solo una delle funzioni previste, poiché è stata implementata la possibilità di effettuare anche la programmazione dell'unità. Per ovvi motivi di protezione del sistema, è consentito intervenire solo sui parametri non vitali per la centralina: ad esempio, si può attivare la retroilluminazione del display, gestire il buzzer della tastiera, variare le tempistiche, conoscere lo stato della centralina, ma non è in alcun modo possibile variare i numeri di telefono oppure attivare/disattivare la centralina da remoto. Questa soluzione consente di evitare a chiunque conosca una password di accesso, di eliminare i numeri di telefono o disatti-

**Fig. 8**



vare la centralina per impedire l'invio degli SMS.

Questa non è l'unica forma di protezione; infatti ogni SMS inviato deve obbligatoriamente contenere la password installatore o utente e ogni messaggio deve essere composto da soli caratteri maiuscoli.

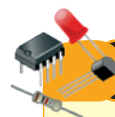
Nella **Tabella 2** è riportato l'elenco dei comandi utilizzabili, con la relativa descrizione. Ogni singolo comando, come anticipato, deve contenere la password di accesso alla centralina, pertanto, facendo un chiaro esempio, se la password utente è "000000" e si vuole conoscere lo stato della centralina, sarà sufficiente comporre un SMS contenente il testo **STA?;000000**.

Come si vede dall'esempio, il carattere ";" viene utilizzato per separare la parte di comando, dalla parte di codice, ed è molto importante che questo carattere sia presente; in caso contrario l'SMS verrà ignorato. Per tutti e quattro i comandi di configurazione, l'unità invierà una risposta contenente la configurazione della centralina, ma solo per i parametri non sensibili.

Qualora l'SMS abbia il formato corretto, si otterrà una risposta certa (credito permettendo); se invece ci sarà un errore (ad esempio se si compone il messaggio con caratteri minuscoli o si omette un carattere del comando) l'unità analizzerà

l'SMS solo se questo conterrà la password utente/installatore e ci invierà in tal caso un SMS informando dell'errore. Se la password non sarà stata inserita, l'unità eliminerà il messaggio senza inviare alcuna risposta.

Nella **Fig. 8** è possibile vedere tre tipologie di risposta: la prima (sinistra) contenente un messaggio di errore, una seconda (in mezzo) con lo stato della centralina ed una terza (a destra) contenente i parametri attualmente impostati.



### per il MATERIALE

La scatola di montaggio dell'antifurto è disponibile al prezzo di Euro 126,00 (cod. FT1015K). Il kit comprende tutti i componenti, i micro programmati, il PCB forato e serigrafato nonché tutte le minuterie. Gli unici elementi non inclusi (ma disponibili separatamente) sono: Modulo GSM completo di antenna (TDGGS\_900, Euro 51,00), batteria ricaricabile cod. NP08-12 (Euro 21,00), alimentatore AL08-12N (Euro 8,00), contenitore G313 (Euro 11,00), telecomando TX3750-2CS (Euro 19,00). Tutti i prezzi si intendono IVA compresa.

Il materiale va richiesto a:  
Futura Elettronica, Via Adige 11,  
21013 Gallarate (VA)  
Tel: 0331-799775 - Fax: 0331-792287  
<http://www.futurashop.it>





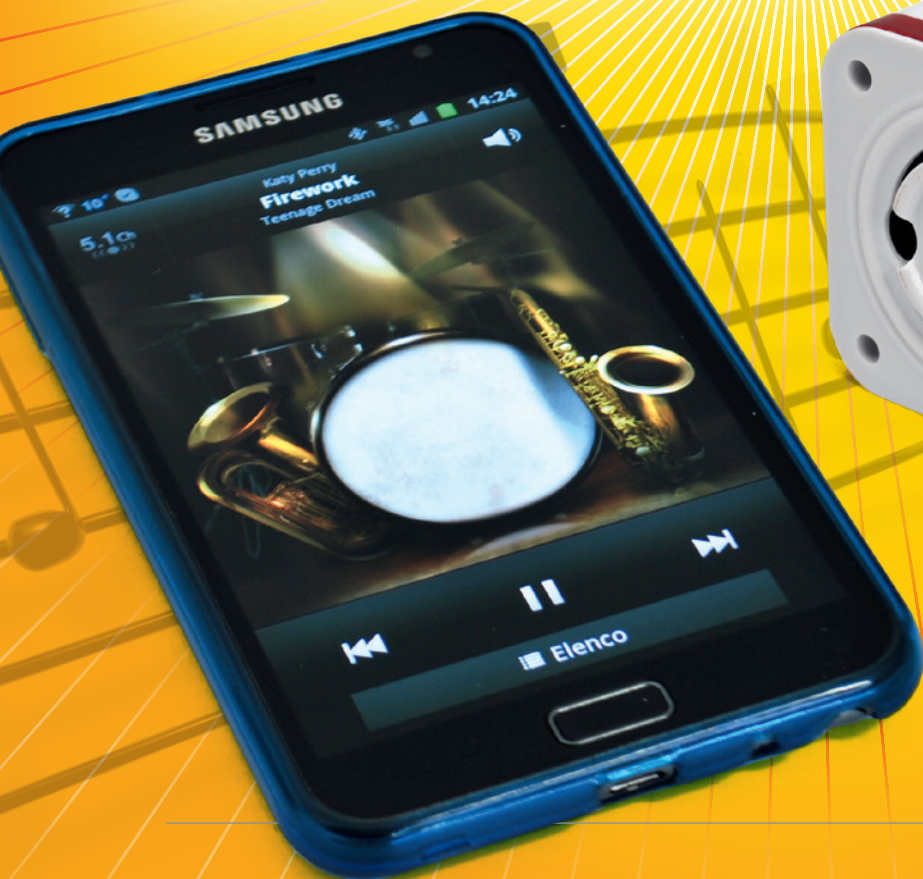
# STREAMING AUDIO BLUETOOTH E CASSE PORTATILI

di SIMONE MAJOCCHI

**L**a presenza del bluetooth in smartphone e tablet è ormai data per scontata, sia perché consente il collegamento a una serie di dispositivi accessori quali vivavoce e GPS, sia perché dal punto di vista costruttivo sono gli stessi produttori di chip di comunicazione che offrono i combo WiFi/Bluetooth con poca spesa aggiuntiva rispetto al solo WiFi. Le possibilità che uno smartphone abbia il Bluetooth sono quindi molto elevate, anche sui modelli di tipo più semplice. Ma cosa si può fare con questa interfaccia oltre a collegare l'auricolare? La risposta non è immediata in quanto dipende dai profili supportati dal telefonino o dal tablet.

## I PROFILI

Quando è stato definito lo standard Bluetooth, lo si è fatto pensando a una soluzione che fosse l'opposto di USB. Con USB, l'interfaccia è molto semplice a livello di protocollo e ogni dispositivo va gestito tramite appositi driver, da scaricare di volta in volta e che sono anche specifici per sistema operativo ospite. Senza driver, la periferica è inutilizzabile, mentre se questa soluzione per un telefonino non era certamente ottimale e quindi è nata la soluzione a "profili", ovvero una definizione puntuale ed esaustiva del protocollo di comunicazione dal livello più basso a quello delle funzionalità sovrastanti. Con



Mettete assieme un embedded dedicato alla decodifica audio MP3, un amplificatore digitale e un modulo bluetooth e avrete un diffusore wireless per il vostro smartphone o tablet.

# Usare l'uCOM A5

Il catalogo di Futura Elettronica si è arricchito recentemente di questo piccolo ma potente box. Il dispositivo ha le seguenti funzioni:

- 1) Bluetooth 2.0, EDR, multimedia audio device
- 2) Riproduttore di musica in formato MP3 memorizzata su supporto microSD,
- 3) Amplificatore audio

Al suo interno si trova un amplificatore da 3W RMS alimentato da una batteria ricaricabile Li-Ion e un microcontrollore multimediale dedicato per la riproduzione dei file MP3 e

per la gestione dell'interfaccia Bluetooth, basata sul chip OVC3860 di OmniVision che si prende carico di tutta la parte di ricezione dello stream secondo il profilo A2DP e gestisce il player del trasmettitore comandandolo tramite il profilo AVRCP.

Su un lato di A5 – da sinistra a destra - si trovano l'ingresso audio stereo con presa 3mm standard audio stereofonica, la presa microUSB per l'alimentazione e la ricarica della batteria, lo slot per schede di memoria microSD (**Fig. 1**). Sul lato opposto di A5 (**Fig. 2**)

si trovano – da sinistra a destra – il deviatore di accensione, avanti (Vol +), Pausa e Play (seleziona MODO), indietro (Vol -)

Modalità di funzionamento:

## REGOLAZIONE DEL VOLUME

In tutte le modalità di funzionamento, i due tasti Vol + e Vol - sono attivi. Vanno usati tenendoli premuti per almeno 3 secondi senza rilasciarli, altrimenti eseguono le funzioni Avanti e Indietro.

## SELEZIONE MODALITA'

Il tasto Play (il terzo da sini-



Fig. 1



Fig. 2

questa soluzione, ogni dispositivo, trasmittente o ricevente, poteva dialogare e usufruire delle funzionalità proprio grazie ai profili.

Niente driver quindi, ma un insieme di funzioni specifiche per ciascun profilo, permettendo così il collegamento immediato in ogni situazione in cui il profilo necessario è disponibile su chi trasmette e su chi riceve. Se invece manca uno specifico profilo a uno dei due dispositivi, non è possibile realizzare il "collegamento senza fili".

Dai primi semplici sistemi di sostituzione del filo, tipicamente con un collegamento seriale, lo standard si è arricchito di una serie di profili dedicati alla stampa, al trasferimento dell'audio, allo scambio di dati (rubriche e contatti), fino alla musica stereofonica, alla condivisione delle risorse di rete e molto altro.

Nel tempo il gruppo di standardizzazione di Bluetooth ha anche migliorato la velocità di trasferimento dei dati (Enhanced Data Rate – EDR) così come ha creato delle versioni successive fino ad arrivare all'attuale 4.0, definita a luglio 2010. Nella sostanza, la versione 2.1 con EDR è quella che sotto l'aspetto della connettività contempla e aggiunge pratica-

mente tutti i profili significativi, dagli originali ai più recenti, mentre le due release successive sono nella direzione dell'integrazione con WiFi e del miglioramento dei consumi.

## I RICEVITORI

Se dal lato dei telefonini, smartphone e tablet il Bluetooth incide in modo marginale sul costo dell'apparecchio e quindi fa parte del corredo quasi senza essere messo in discussione, i dispositivi e gli accessori compatibili con questo standard per offrire funzionalità aggiuntive sono ancora in fase di affermazione. Ci sono casi eclatanti dove autovetture da decine di migliaia di euro supportano malamente pochi profili, causando non poco disappunto negli utenti, mentre altri tendono a limitare nel telefonino le prestazioni ancora nella speranza di vendere accessori dedicati. Quello che però è certo è che se manca il profilo richiesto da una parte o dall'altra, non c'è nulla da fare, così come una implementazione errata del profilo può causare una serie di incompatibilità che si manifesta con il funzionamento con alcuni dispositivi, ma non con tutti – a

parità di profilo disponibile. Fortunatamente la tecnologia è arrivata a un elevato livello di maturità e ormai chi produce chip e moduli sa bene come offrire ai propri clienti il necessario per arrivare al risultato cercato. I moduli sono sostanzialmente delle soluzioni complete a cui basta collegare poco altro per ottenere il risultato, mentre con integrazioni più complesse si possono realizzare dispositivi



*Un "dongle" che preleva l'audio dall'uscita cuffia di un dispositivo senza Bluetooth e lo invia in streaming a delle casse o a un ricevitore stereo con profilo A2DP.*

stra) serve a selezionare la modalità di funzionamento. All'accensione viene scelta per default quella Bluetooth. Premendo per almeno 3 secondi il tasto play si passa in rotazione ad Audio MP3, (Amplificatore) e Bluetooth Audio. Se è collegata una sorgente audio alla presa da jack 3mm il dispositivo attiva l'amplificazione di questo ingresso, altrimenti viene saltata l'opzione "Amplificatore" e si passa direttamente all'audio da Bluetooth a MP3 play. Se scollegate il jack da 3mm con A5 in modalità

Amplificatore, A5 passa alla modalità MP3 e premendo Play si avvia la riproduzione dell'ultimo brano selezionato. L'ultimo brano ascoltato resta anche memorizzato fra un'accensione e l'altra.

#### BLUETOOTH

Accendere il dispositivo con l'interruttore. Dopo circa tre secondi verrà emesso un segnale acustico per annunciare la modalità Bluetooth. La scritta superiore uCOM lampeggia in rosso rapidamente se il dispositivo è in modalità di accoppiamento, altrimenti

se è già stato accoppiato a un dispositivo e il collegamento è attivo, la scritta lampeggia in bianco. Durante la riproduzione audio la scritta resta illuminata in bianco.

Per effettuare l'accoppiamento (pairing) basta attivare la ricerca sul proprio smartphone o tablet e accendere il dispositivo (lampeggio rosso frequente). Troverete un dispositivo "Bluetooth - A5" in elenco: selezionatelo e usate il pin "0000". Al termine della procedura, cesserà il lampeggio rosso della scritta uCOM e i due dispositivi saranno ac-

coppiati. Il profilo del dispositivo A5 è Audio Multimediale e se il vostro player lo supporta, potete usare i tasti Avanti, Play, Pausa e Indietro per pilotare il riproduttore software. Il dispositivo memorizza l'accoppiamento a più dispositivi e quindi una volta abbinato ai vostri vari dispositivi che supportano il profilo Audio Multimediale, potrete utilizzarli semplicemente accendendo A5 e il loro Bluetooth: A5 ripristinerà in pochi istanti il collegamento. Il dispositivo A5 supporta ovviamente un solo collegamento attivo alla volta.

multifunzionali che combinano al wireless bluetooth anche altre caratteristiche.

#### DOPO I VIVAVOCE, ARRIVA LA MUSICA

La comunicazione con l'auricolare bluetooth o il vivavoce in auto è obbligatoria e necessaria, per non perdere la concentrazione necessaria alla guida. Il tipo di audio trasportato con l'headset e l'hands free (i due profili usati per far parlare l'utente durante una telefonata) ha però delle caratteristiche tali da rendere i due profili inadatti alla trasmissione di musica. I profili HSP e HFP sono infatti stati pensati per il trasporto di voce già codificata secondo gli standard GSM, quindi in mono e con una banda ristretta, pur utilizzando uno stream di dati che può anche essere di 64Kbit/s. Questo non è stato ritenuto un problema per un po' di anni, dato che i telefonini non erano ancora dotati di riproduttori musicali. La cosa inizia a cambiare quando il player audio diventa una caratteristica presente negli smartphone che, oltre a consentire di telefonare, acquistano appunto una certa ricchezza funzionale. In termini

tecnologici, la reazione a questa nuova funzionalità richiede un tempo significativo e solo da qualche anno la diffusione di audio di qualità tramite bluetooth è diventata tanto normale, quanto richiesta dagli utenti.

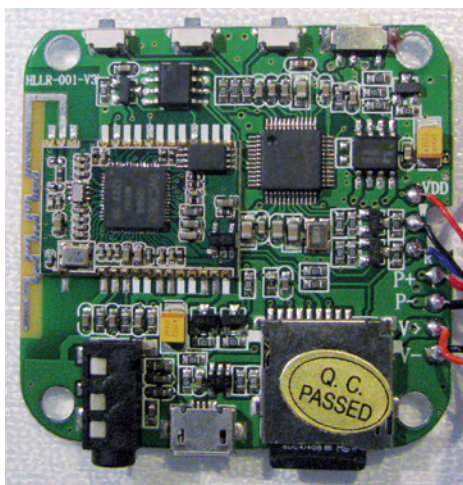
Per soddisfare questa esigenza, nascono due profili specifici: A2DP e AVRCP. Il primo è l'Advanced Audio Distribution Profile, mentre il secondo è l'Audio/Video Remote Control Profile.

Con A2DP viene stabilito tutto il necessario per lo streaming dal trasmettitore (smartphone o tablet o addirittura trasmettitore dedicato con ingresso audio) al ricevitore che ha un ruolo passivo e di pura riproduzione di quanto ricevuto. Le codifiche che il trasmettitore e il ricevitore devono supportare sono varie e lo standard prevede che alcune siano necessariamente supportate per garantire una buona interoperabilità fra vari dispositivi. La codifica SBC (sub band coding) è quella base e obbligatoria, ma l'evoluzione tecnologica ha permesso l'aggiunta dei codec audio più diffusi nell'ambito musicale come MPEG-1, MPEG-2, MPEG-4, AAC e ATRAC. Grazie a questo, l'audio in streaming non



viene più ricodificato prima di essere trasmesso (cosa che facevano i piccoli trasmettitori bluetooth da collegare alla sorgente audio al posto della cuffia), ma viene inviato direttamente con la codifica originale in quanto il dispositivo ricevente è in grado di decodificarla correttamente. Il profilo AVRCP è invece necessario per impartire dalla ricevente dei comandi al player, più nello specifico, grazie a questo profilo si possono inviare Play, pausa, avanti e indietro. Questo permette di controllare il player





*La scheda di uCOM A5, un box amplificatore che supporta sia lo streaming Bluetooth, sia la riproduzione da microSD di file MP3.*

sul telefonino o sullo smartphone senza intervenire su di esso fisicamente e la comodità di questa soluzione è notevole.

### **I PRODOTTI SUL MERCATO**

Un po' per la diffusione degli smartphone, un po' perché il mercato è sempre affamato di novità, anche le soluzioni audio bluetooth hanno ricavato una propria nicchia nel mercato dell'audio. Si va dai sistemi di diffusione domestici di elevata qualità fino ai minuscoli diffusori autoalimentati e ricaricabili. In pratica ce n'è per tutte le tasche e per tutte le esigenze. Basta recarsi in un negozio di elettronica di consumo per trovare un certo assortimento di prodotti "importanti" per prezzo e prestazioni, ma la vera novità di questo periodo sono i diffusori micro, alimentati a batteria ricaricabile e basati sugli amplificatori digitali. Grazie alla diffusione di questa tipologia di amplificatori, 3 watt possono essere erogati partendo dai 3,7V delle batterie Li-Ion e quando l'altoparlante ha una buona efficienza, si riesce a generare un audio con volume decisamente "forte". Anche la distorsione non è più marcata come una volta e quindi si riesce ad avere una cassa piccola, alimentata a batteria ricaricabile, ma in grado di generare un livello

audio più che rispettabile prima che la distorsione renda il tutto sgradevole.

La produzione di questi dispositivi è ormai quasi interamente ad appannaggio della Cina, ma diversi prodotti riescono a combinare un design gradevole a una ottima qualità della componentistica, lasciando talvolta stupiti da quanto si possa ottenere con una spesa contenuta. L'abilità dei produttori del Far East sta nella reinterpretazione di piattaforme ben consolidate e di provata affidabilità: una volta messo a punto il software e l'architettura di base, le variazioni sono sia di tipo estetico, sia di tipo funzionale, facendo del medesimo diffusore Wireless la versione mono, quella stereo, da 3 watt e da 10 watt, con la radio FM, senza la radio, ma con il lettore di microSD, oppure con la porta USB host con cui leggere la musica sulle chiavette USB. Insomma, date un chip a un produttore e questo vi declinerà il prodotto in almeno una decina di versioni diverse. Anche il prezzo, ovviamente, si aggancia alle funzionalità con una certa linearità, così da accontentare il più ampio segmento di mercato. Fra i vari prodotti che abbiamo esaminato, uCOM A5 ha attirato la nostra attenzione per le sue caratteristiche e quindi abbiamo scelto di "guardarci dentro" per capire quali fossero i suoi segreti. Nella fotografia potete notare la scheda interna: è popolata con componenti ad elevata integrazione ed il modulo bluetooth è su una schedaina dedicata. Il jack a sinistra della presa microUSB accetta un segnale audio in ingresso e lo amplifica. Due led, uno rosso

e uno bianco sono collegati con dei fili e illuminano la scritta del dispositivo per segnalare le varie modalità operative. Nel box di queste pagine avete una descrizione approfondita del prodotto e di come funziona. Certamente l'A5 non è né il primo né l'ultimo box audio amplificato wireless che vedremo, ma intanto merita di essere "ascoltato" perché in poco spazio e con una spesa tutto sommato ragionevole permette di dar voce al nostro smartphone per condividere con gli amici una canzone o vedere assieme un video da YouTube. Gli schermi di tablet e smartphone sono infatti diventati via via più grandi, ma per l'audio la situazione è ancora molto limitata e quindi avere un dispositivo in tasca, capace di amplificare con queste caratteristiche e questo impatto acustico, potrebbe essere l'asso nella manica da calare alla prossima occasione con i vostri amici.

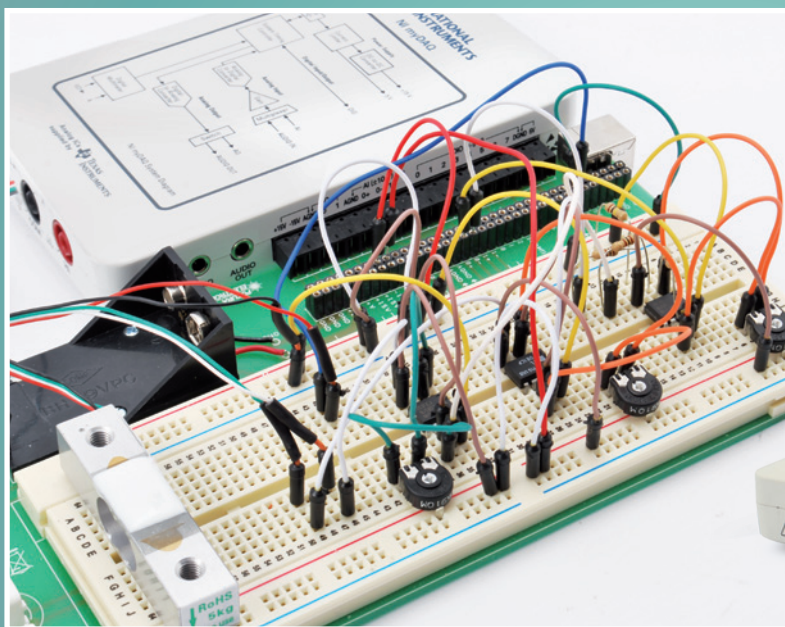
**L'audio box Bluetooth descritto in queste pagine (Codice prodotto 8100-SURE-A5) è disponibile presso Futura Elettronica srl ([www.futurashop.it](http://www.futurashop.it)) al prezzo di 29 Euro IVA compresa.** ■



# myKIT, MISURIAMO IL PESO CON LabVIEW E myDAQ

di DAVIDE SCULLINO

Proseguiamo le esercitazioni con myDAQ e myKIT, spiegando come pesare degli oggetti usando una cella di carico.

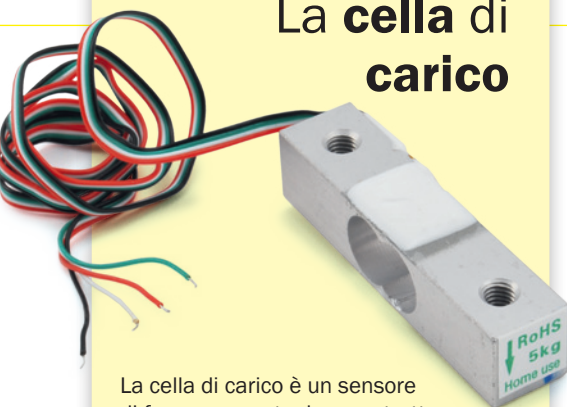


**E**ccoci giunti alla nostra seconda esercitazione con il set didattico myKit, durante la quale vedremo come sfruttare una cella di carico collegata alla scheda myDAQ per compiere delle misure di peso. La tensione in uscita dalla cella di carico sarà acquisita tramite il DAQ Assistant che viene installato in LabVIEW con il driver NI DAQmx e verrà quindi convertita in un valore di peso utilizzando la programmazione in LabVIEW; il tutto servendoci di un VI creato allo scopo. La cella di carico che utilizzeremo è siglata CZL635 e può misurare pesi da zero a 5 kg.

Ma prima di spiegare i collegamenti con myDAQ, la creazione dell'interfaccia di misura in LabVIEW e di entrare nel vivo della misura, è il caso di spiegare brevemente in cosa consiste il sensore che utilizzeremo; ebbene, una cella di carico è un trasduttore che rileva sostanzialmente la forza (quindi non la massa) ed è composto da una struttura in metallo che sostiene piccoli elementi chiamati estensimetri



## La cella di carico



La cella di carico è un sensore di forza composto da una struttura in metallo variamente conformata, che sostiene dei trasduttori di pressione chiamati *estensimetri*. Il segnale elettrico fornito dalla cella di carico è una piccolissima tensione proporzionale alla forza peso (si parla di qualcosa come alcuni mV).

Sebbene siano progettate per misurare la forza in una sola direzione ed in un solo verso, talvolta le celle di carico vengono utilizzate per misurare la forza anche nel verso opposto, ma in tal caso la sensibilità cambia, in quanto le parti mobili della cella di carico che normalmente funzionano in compressione si trovano a lavorare in estensione.

Fisicamente, la misurazione viene effettuata da reti di piccole resistenze in grado di cambiare il loro valore in base alla deformazione prodotta dall'applicazione della forza; gli estensimetri sono fissati su una barra o su un elemento strutturale che si deforma quando sottoposto ad una forza, deformando a sua volta gli estensimetri. Quando l'estensimetro subisce una deformazione, la sua resistenza elettrica varia in proporzione. Usare le celle di carico implica il superamento di certe difficoltà, dato che le variazioni di resistenza negli estensimetri dovute alla forza applicata sono comparabili con quelle causate in essi dalle variazioni di temperatura nell'ambiente. A ciò si può rimediare montando nella stessa cella di carico più estensimetri "di compensazione" collegati in modo che la variazione termica di resistenza di uno di essi annulli quella di un altro. Tipicamente per compensare la deriva termica si utilizzano estensimetri che non sono mai sottoposti ad alcuna forza, ma servono a bilanciare gli effetti della temperatura sugli estensimetri che effettivamente misurano la forza.

La maggior parte delle celle di carico di una certa qualità utilizza quattro estensimetri in configurazione a ponte di Wheatstone, che massimizza la sensibilità della cella di carico annullando l'effetto di deriva dovuto alla temperatura.

(in inglese, *strain gauges*) montati in punti opportuni sulla struttura stessa. Ogni estensimetro varia la propria resistenza in funzione della forza che riceve, tuttavia la cella di carico fornisce una tensione, perché normalmente viene alimentata.

Siccome la cella di carico fornisce segnali molto deboli (nel nostro caso si parla di qualche mV di ampiezza) occorre provvedere a un'adeguata amplificazione in tensione, prima di inviare il segnale al dispositivo di elaborazione. A ciò provvede il semplice circuito che dovrete realizzare, meglio descritto dallo schema elettrico che trovate nella **Fig. 1**. Si tratta di un amplificatore formato da tre operazionali TL081, provvisti ciascuno di trimmer per la regolazione dell'offset, taratura, questa, indispensabile perché abbiamo a che fare con segnali molto deboli e quindi un errore anche minimo potrebbe alterare significativamente le misure.

### CABLAGGIO DEL CIRCUITO DI TEST

La cella di carico è munita di quattro fili di collegamento: il rosso va connesso all'alimentazione +5V della scheda myDAQ, il filo nero va collegato al ground AGND, mentre il bianco ed il verde vanno collegati agli ingressi dei due amplificatori operazionali (**Fig. 1**). Inoltre, l'uscita dell'ultimo amplificatore operazionale va collegata all'ingresso AI0+ della scheda myDAQ e l'ingresso AI0- di quest'ultima deve essere connesso ad AGND.

### AMPLIFICAZIONE E CALIBRAZIONE

Per amplificare la tensione proveniente dalla cella di carico dovete implementare sulla breadboard il circuito di **Fig. 1**, composto da tre amplificatori operazionali (TL081) con rispettivi trimmer da 10 k $\Omega$  per regolarne il bilancia-

mento e alcune resistenze (due da 100  $\Omega$  e due da 1 k $\Omega$ ).

Dovrete utilizzare due amplificatori operazionali in configurazione inseguitore di tensione (buffer) e l'ultimo come amplificatore differenziale; ognuno di questi stadi va alimentato con le tensioni +15V e -15V fornite dalla scheda myDAQ.

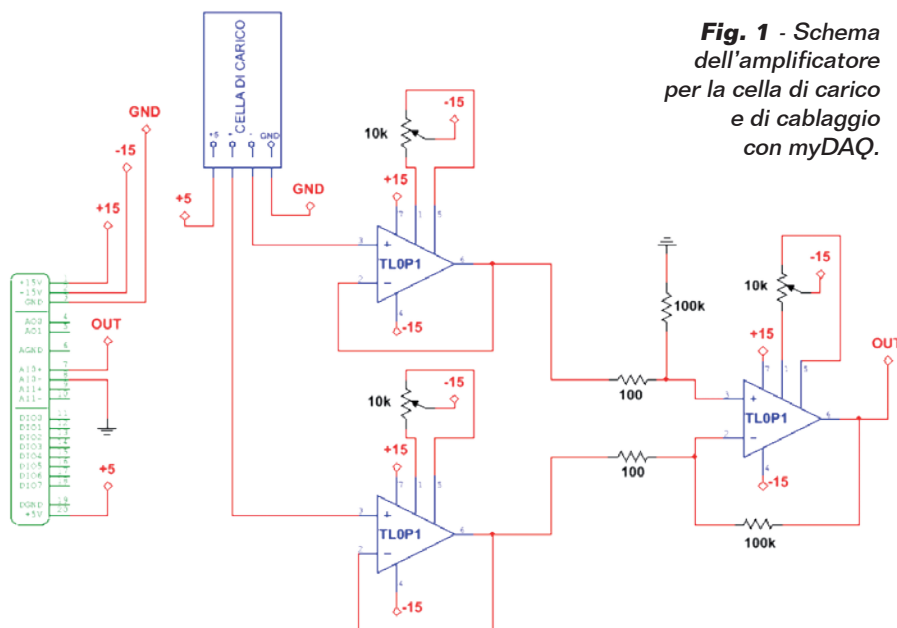
Affinché l'amplificazione del differenziale sia uguale sia per l'ingresso invertente, sia per quello non-invertente, dovete usare coppie di resistenze di valore il più possibile uguale, ragion per cui consigliamo di controllarne i valori utilizzando la scheda myDAQ e il Digital Multimeter (DMM) fornito da NI ELVISmx Instrument Launcher (**Fig. 2**).

Per regolare l'offset di ogni amplificatore operazionale dovete collegare una tensione nota (è consigliabile usare il ground della scheda myDAQ) all'ingresso non invertente dopo aver unito l'uscita con l'ingresso invertente, così da ottenere un guadagno unitario. Ora, misurando la tensione all'uscita dell'operazionale dovrete ottenere lo stesso valore applicato all'ingresso non invertente (zero volt, se avete utilizzato il ground come ingresso). Se ciò non avviene, agite sul trimmer per regolare l'offset dell'operazionale fino a ottenere il valore desiderato.

Per effettuare queste semplici misurazioni utilizzate il Digital Multimeter che trovate tra gli strumenti forniti da NI ELVISmx. Ripetete la precedente operazione per ognuno degli amplificatori e regolate tutti i rispettivi offset. Il circuito cui collegare le uscite della cella di carico garantisce un'amplificazione in tensione pari a 1.000 volte (dato dal rapporto delle resistenze 100k $\Omega$ /100 $\Omega$ ) il che significa che ogni mV all'ingresso determi-



**Fig. 1** - Schema dell'amplificatore per la cella di carico e di cablaggio con myDAQ.



nerà 1 volt all'uscita. Per quanto riguarda la calibrazione, dovete utilizzare la seguente formula:

$$\text{Peso applicato} = Kx (\text{misura mV} - \text{offset})$$

Da questa formula ricaverete il fattore di conversione  $K$  per calcolare i valori dei pesi che applicherete alla cella di carico. Il valore *offset* è la tensione acquisita senza che alla cella di carico venga applicato alcun peso (a riposo). Potete portare questo offset a un valore prossimo allo zero agendo sul trimmer dell'ultimo amplificatore, in modo da portare a zero il valore della tensione in uscita.

Se volete essere più precisi sulla conversione, dovete acquisire diverse misure di tensione associate a dei pesi noti e, aiutandovi con un foglio di calcolo, ricavare un'equazione ottenuta per interpolazione dei punti trovati. Utilizzerete poi questa equazione all'interno del software per ricavare i valori di peso esatti rispetto alla tensione acquisita.

#### PANNELLO FRONTALE LABVIEW

L'interfaccia utente LabVIEW che dovreste creare presenterà un controllo numerico per l'inserimento

del fattore ( $K$ ) di conversione da tensione acquisita a peso applicato alla cella di carico; avrà inoltre due indicatori a lancetta e due grafici per visualizzare, rispettivamente, la tensione acquisita ed il peso applicato. Nell'interfaccia utente sono previsti anche un LED che segnala all'utente quando il peso applicato è maggiore del massimo ammissibile e una barra formata da un insieme di dodici LED che si accenderanno in sequenza con colori diversi in base al peso applicato, oltre al pulsante Stop per fermare l'esecuzione del VI.

#### CODIFICA DELLA STRATEGIA

In LabVIEW avete bisogno di acquisire il segnale proveniente dalla cella di carico amplificato dal circuito descritto nella Fig. 1. Successivamente dovreste applicare alla cella di carico dei pesi noti e ricavarvi i corrispondenti valori di tensione quindi, utilizzando la formula:

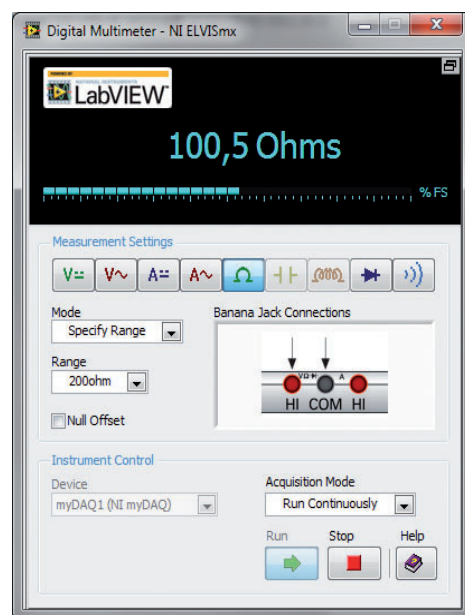
$$K = \frac{\text{Peso applicato [kg]}}{\text{Tensione acquisita [V]}}$$

calcolare il fattore di conversione  $K$ . Quest'ultimo sarà utilizzato nel software per ricavare

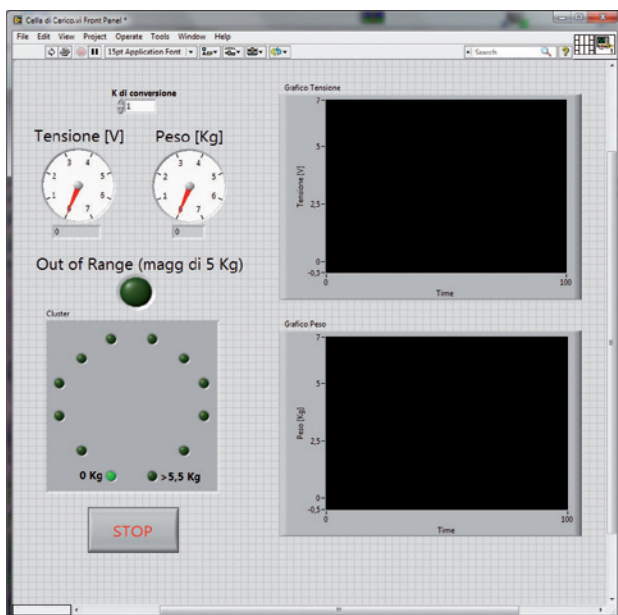
il valore del peso applicato in funzione della tensione acquisita. La misura di peso sarà successivamente visualizzata nel Pannello Frontale su un indicatore a lancetta e su un grafico; la stessa cosa avverrà per la tensione acquisita. Il diagramma a blocchi che dovreste implementare sarà quello di Fig. 4.

#### COME FUNZIONA

All'interno del ciclo While, sulla sinistra è situato DAQ Assistant, che è configurato per acquisire un singolo valore dai terminali d'ingresso analogici della scheda myDAQ ogni volta che viene eseguito. Il valore acquisito viene trasmesso all'indicatore ed al grafico della tensione. Lo stesso valore viene moltiplicato per il fattore di conversione  $K$  e trasformato in peso per essere visualizzato anch'esso su un indicatore e un grafico. Successivamente verrà verificato se il peso misurato è all'interno del range di misura del sensore (0÷5 kg) e in caso contrario si accenderà il



**Fig. 2** - Controllo dei valori delle resistenze con il Digital Multimeter del NI ELVISmx.



**Fig. 3 - Pannello Frontale in LabVIEW.**

LED siglato *out of range*. Per gestire i LED disposti in cerchio sul Pannello Frontale che si accendono in sequenza in base al peso applicato alla cella di carico, dovete confrontare quest'ultimo con un vettore di valori noti. Questo confronto genera un array di valori True o False che bisogna convertire in un cluster per gestire i LED di differenti colori (i cluster sono contenitori che raggruppano più elementi con tipi di dato diversi). Dovete utilizzare il cluster di indicatori booleani invece di un array di indicatori booleani, perché i LED sono di differenti colori ed è come se fossero oggetti diversi tra loro (anche se riguardano lo stesso tipo di dato). Tutto il codice presente nel ciclo While continua a funzionare fino a che sul Pannello Frontale non fate clic sul pulsante Stop. L'esecuzione del blocco Wait (in alto a sinistra nel diagramma a blocchi di Fig. 4), fa in modo che l'iterazione del ciclo avvenga ogni 100 ms, pertanto la frequenza di campionamento è di 10 campioni al secondo, ossia di 10 Hz.

### COSTRUZIONE DEL PANNELLO FRONTALE

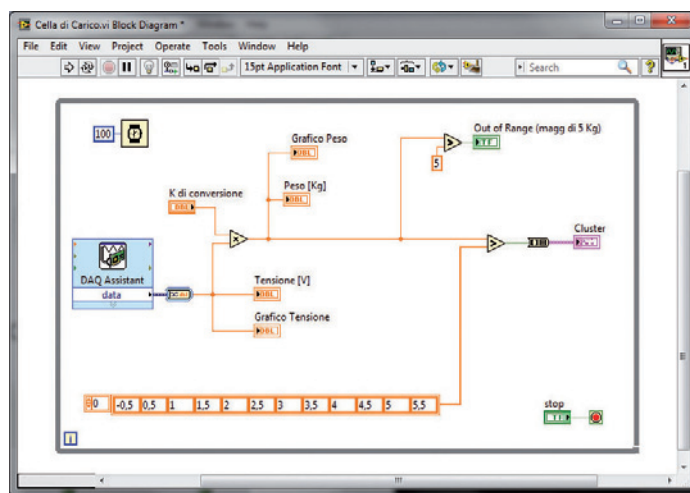
Sul Pannello Frontale, cliccate col tasto destro del mouse per aprire la palette *Controls*, dove sono elencati tutti i tipi di controllo e di indicatore, ma anche decorazioni, tabelle, ecc. Per comodità potete cliccare sulla puntina in alto a sinistra della palette *Controls* per bloccarla sullo schermo; ciò vi risparmierà di dover fare clic col pulsante destro del mouse ogni volta che vorrete farla ricomparire. Ora estendete la voce *Modern* (la prima) della palette *Controls*, selezionate *Boolean* e inserite un pulsante Stop sul Pannello Frontale. Sempre alla voce *Modern*, selezionate *Numeric* e inserite due indicato-

ri a lancetta tipo manometro (*Gauge*) e un controllo numerico. Infine, ancora da *Modern* selezionate *Graph* e inserite due *Waveform Chart*.

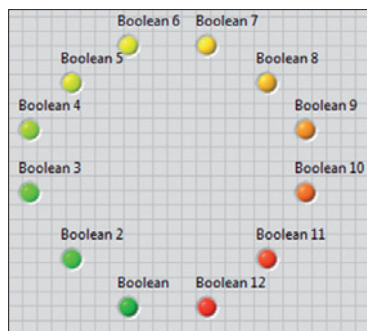
Ora, con gli strumenti *Positioning* e *Labeling* sistemateli e rinominateli come in Fig. 3. Se preferite cambiare le impostazioni di scrittura (font, dimensione, colore, ecc...), sia nel Pannello Frontale che nel diagramma a blocchi, nella barra del menu in alto cliccate su *Text Settings* (è il riquadro che riporta scritto la dimensione e il font che state utilizzando: **15pt Application Font**).

Per quanto riguarda i due grafici, sempre con lo strumento *Labeling*, rinominate entrambe gli assi Y, uno con *Tensione [V]* e l'altro con *Peso [kg]*. Successivamente cliccate col tasto destro del mouse in corrispondenza delle due etichette appena inserite sugli assi Y e, nel menu a tendina che comparirà, deselectionate *Autoscale Y* per entrambi i grafici. In seguito, con lo strumento *Labeling* inserite sull'asse Y di entrambi i grafici i valori massimo (7) e minimo (-0,5), come in Fig. 3. Inoltre rinominate anche i valori massimo (6) e minimo (0) presenti all'interno dei due indicatori a lancetta. Per far comparire i display sotto gli indicatori a lancetta, cliccate sopra di essi col tasto destro del mouse e selezionate *Visible Item»Digital Display*.

Adesso, alla voce *Modern* della palette *Controls*, selezionate *Boolean* e inserite dodici Round LED, cercando di disporli a forma di cerchio rispettando l'ordine d'inserimento, come nella Fig. 5. Per associare a ogni LED un colore diverso, che va dal verde (0 kg) al rosso acceso (peso fuori range, cioè maggiore di 5,5 kg, che può portare al danneggiamento permanente della cella di carico), dovete cliccare con il pulsante destro del mouse sopra ogni LED, selezionare *Properties* e,



**Fig. 4 - Diagramma a Blocchi LabVIEW.**

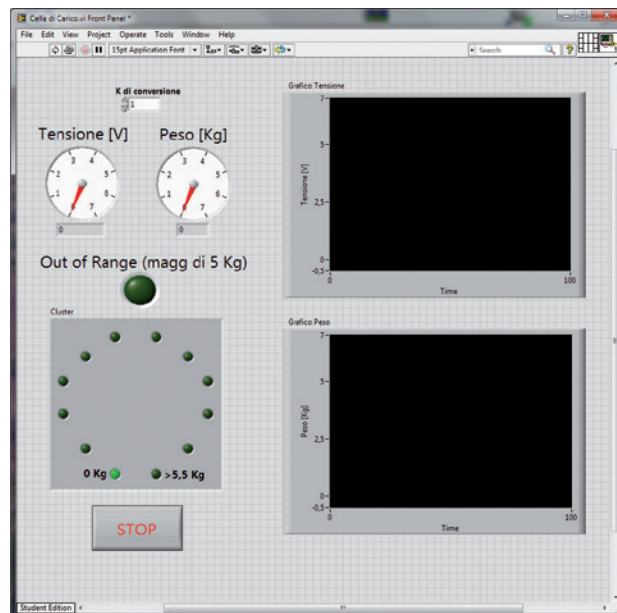
**Fig. 5**

nella schermata *Appearance* delle impostazioni che comparirà, andare su *Colors*; qui dovete cliccare sul quadratino colorato

*On* e, dalla palette che comparirà, scegliere il colore appropriato. Completate questa operazione per ogni LED fino a ottenere l'effetto desiderato. Tenete presente che per visualizzare i colori dei LED appena inseriti dovete, con lo strumento *Operating*, cliccare su ogni LED per portarlo nello stato *On* (perché altrimenti li visualizzerete tutti verde scuro, cioè del colore associato alla modalità *Off*).

Con lo strumento *Positioning* cliccate col tasto destro del mouse su un LED per volta e deselectionate *Label* alla voce *Visible Item*, tranne il primo e l'ultimo, che dovrete rinominare con lo strumento *Labeling* con 0 kg e >5,5 kg (**Fig. 6**). Così facendo renderete un po' più chiaro l'utilizzo di questi indicatori ed eliminerete le etichette inutili associate ai LED. Adesso dovete inserire questi dodici LED all'interno di un cluster. Per fare ciò, alla voce *Modern* della palette *Controls* selezionate *Array, Matrix & Cluster* e inserite un cluster (dovreste visualizzarlo sul Pannello Frontale sotto forma di riquadro vuoto grigio scuro). Con lo strumento *Positioning* ingrandite il cluster a sufficienza per alloggiarvi al suo interno il cerchio composto dai dodici LED.

Ora, sempre con il medesimo strumento, selezionate tutti i dodici LED e trascinateli dentro il cluster. Se provate ad aprire il diagramma a blocchi LabVIEW, noterete che al posto di vedere i dodici indicatori LED, vedrete solo un unico indicatore viola chiamato cluster. Per finire l'implementazione del Pannello Frontale, alla voce *Modern* della palette *Controls* selezionate *Boolean* e inserite un ultimo *Round LED*. Con lo strumento *Labeling* rinominate quest'ultimo in *Out of Range* (maggiore di 5 kg) e con lo strumento *Positioning* sistematelo come in **Fig. 6**. Se avete agito come appena descritto, il vostro Pannello Frontale si dovrà presentare all'incirca come in **Fig. 6**. Come ultima cosa inserite il valore del fattore di conversione *K* nell'apposito controllo numerico. Ora, dal menu *Edit* selezionate *Make Current Values Default*: così questo valore sarà predefinito ogni qualvolta aprirete e utilizzerete questo VI (ovviamente se utilizzerete una diversa cella di carico e calcolerete un nuovo fattore di conversione *K*, inserirete il valore corrispondente nell'apposito controllo e ripeterete questa operazione).

**Fig. 6 - Il Pannello Frontale terminato.**

Se nel circuito amplificatore non riuscite a portare l'offset a un valore prossimo allo zero, nel Pannello Frontale dovete inserire un ulteriore controllo numerico dove indicherete l'offset, così da poterlo eliminare via software. Per fare ciò andate alla voce *Modern* della palette *Controls*, selezionate *Numeric* e inserite un controllo numerico. Ora, con lo strumento *Labeling* rinominate in "offset" e con lo strumento *Positioning* disponetelo dove preferite sul Pannello Frontale. All'interno di questo controllo scriverete il valore di offset che potete leggere dal display sotto l'indicatore *Tensione [V]* senza applicare alcun peso alla cella di carico.

### COSTRUZIONE DEL DIAGRAMMA A BLOCCHI

Ora implementerete il diagramma a blocchi; se non lo visualizzate, potete riuscirci selezionando *Window»Show Block Diagram* o digitando <Ctrl+E>. Nel diagramma a blocchi gli oggetti del Pannello Frontale appaiono come terminali; i terminali sono porte d'ingresso e di uscita che scambiano informazioni tra il Pannello Frontale e il diagramma a blocchi.

Qui dovreste già visualizzare i controlli numerici e il pulsante di stop con le stesse etichette che avevate messo sul Pannello Frontale. Inoltre dovreste visualizzare i terminali, ad esempio quello del pulsante Stop, in questo modo: ; è tuttavia preferibile visualizzarlo così: . Questa visualizzazione riduce lo spazio che occupa sul diagramma a blocchi; per ottenerla, selezionate *Tools»Options..* e sulla sinistra della finestra che si aprirà selezionate *Block Diagram*, quindi dalla voce *General* disabilitate *Place front panel terminals as icons*.

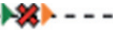


**Tabella 1**

| Tipo di collegamento | Scalare                                                                           | Array 1D                                                                          | Array 2D                                                                          | Colore                                   |
|----------------------|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|------------------------------------------|
| Numerico             |  |  |  | Arancione (virgola mobile), Blu (intero) |
| Booleano             |  |  |  | Verde                                    |
| Stringa              |  |  |  | Rosa                                     |

Ora, prima di iniziare ad implementare il diagramma a blocchi osservate che i rettangolini che rappresentano i controlli possono essere impostati sia come indicatori (cioè ricevono dati) sia come controlli (cioè producono dati).

I controlli si distinguono dagli indicatori per due caratteristiche: la prima è una freccia sul terminale che indica la direzione del flusso dei dati (i controlli hanno frecce che mostrano il dato partire dal terminale, mentre gli indicatori hanno una freccia che mostra il dato entrare nel terminale); la seconda è il bordo intorno al terminale (i controlli hanno un bordo più spesso e gli indicatori più sottile). Per trasformare un elemento da indicatore a controllo, dovete cliccare col pulsante destro del mouse sopra il rettangolino che lo rappresenta e impartire il comando *Change to Control*; questa operazione può essere effettuata anche dal Pannello Frontale. Notate che il pulsante Stop è già impostato come controllo.

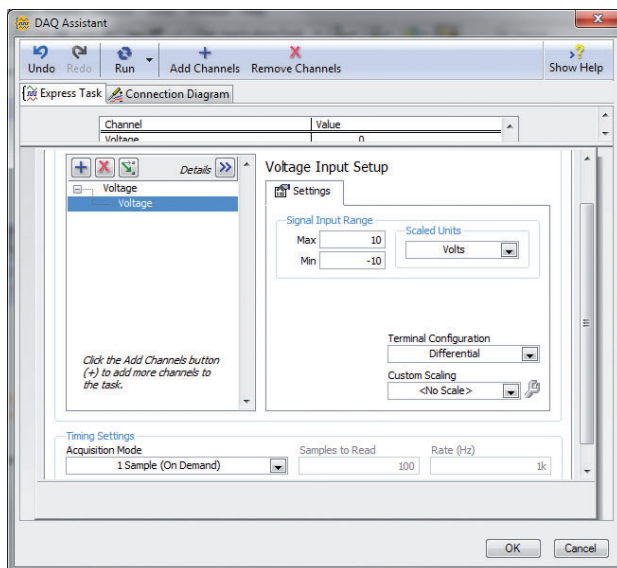
Prestate attenzione ai collegamenti che effettuerete sul diagramma a blocchi: essi permettono di trasferire dati tra oggetti del diagramma a blocchi; ogni collegamento parte da una sorgente di dati e la collega ad uno o più VI e funzioni che leggono i dati stessi. I collegamenti hanno differenti colori, stili e spessori, in funzione del tipo di dati che veicolano. Un collegamento spezzato  appare come una linea nera tratteggiata con una X rossa in mezzo. I collegamenti possono rompersi per vari motivi: ad esempio se si tenta di collegare due oggetti con tipi di dati tra loro incompatibili. I collegamenti possono essere realizzati solamente tra ingressi e uscite che veicolino dati compatibili tra loro. La **Tabella 1** illustra i tipi di collegamento più comuni.

Nel diagramma a blocchi, facendo clic con il pulsante destro del mouse compare la palette *Functions*, che contiene i VI, le funzioni e le costanti da usare per creare il diagramma a blocchi. Se volete bloccare la palette *Functions* sullo schermo, fate clic sulla puntina che si trova in alto a sinistra. Ora potete iniziare la costruzione del diagramma a blocchi. In questo VI il blocco DAQ Assistant è configurato per il campionamento del canale d'ingresso analogico. Le istruzioni seguenti seguono passo-passo la configurazione del DAQ Assistant.

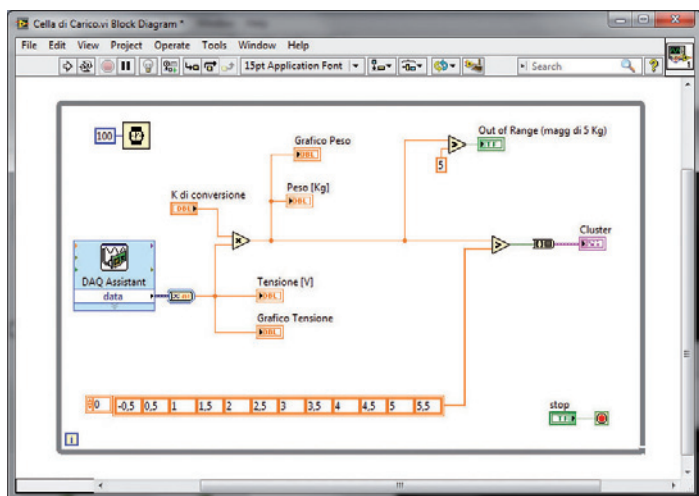
1. Assicuratevi che myDAQ sia collegato al PC;

2. Premete Ctrl-Spazio per aprire la finestra Quick Drop (al primo utilizzo, ciò richiederà circa un minuto);
3. Cercate la voce DAQ Assistant e fate doppio clic su di essa quando appare nella lista;
4. Trascinate il DAQ Assistant sul diagramma a blocchi;
5. Quando appare un nuovo riquadro per le attività di configurazione rapida, selezionate:
  - a. Acquire Signals;
  - b. Analog Input;
  - c. Voltage;
  - d. Dev 1 (NI myDAQ) \* Nota: Se si dispone di un altro hardware NI installato, myDAQ non sarà Dev1;
  - e. ai0;
  - f. Finish;
6. Impostate il corretto range di tensione di ingresso su Signal Input Range:
  - a. Max: 10V;
  - b. Min: -10V;
7. Cambiate il Timing Settings:
  - a. 1 Sample (On Demand);
8. Fate clic su OK.

Ora procedete inserendo alla destra del blocco DAQ Assistant (**Fig. 8**) il blocco *Convert from Dynamic Data Type* situato nella palette *Functions* alla voce *Express»Signal Manipulation*. Appena inserirete il blocco *Convert from Dynamic Data* si aprirà una finestra per la configurazione: alla voce *Conversion* selezionate *Single Scalar* e fate clic su OK. Ora, nella palette *Functions* alla voce *Programming»Numeric* selezionate e inserite nel



**Fig. 7 - Configurazione del DAQ Assistant.**



**Fig. 8**

diagramma a blocchi un blocco *Multiply* e alla voce *Programming»Comparison* selezionate e inserite due blocchi *Greater*. Successivamente, con gli strumenti *Positioning* e *Wiring* sistemate e collegate i blocchi come in **Fig. 8**. Per effettuare il confronto del peso con il valore del peso massimo ammissibile ed eventualmente accendere il LED *Out of Range* (peso < 5 kg), dovete inserire il valore massimo ammissibile (5 kg) al secondo ingresso del blocco *Greater*. Per fare ciò, cliccate col tasto destro del mouse in corrispondenza dell'ingresso, selezionate *Create»Constant* e inserite il valore "5" nel riquadro che comparirà. Adesso dovete creare l'array di costanti che verranno confrontate continuamente con i valori di peso per poter accendere in sequenza i rispettivi LED contenuti nel cluster. Nella palette *Functions*, alla voce *Programming»Array* selezionate *Array Constant* e trascinate all'interno di esso una *DBL Numeric Constant* che trovate alla voce *Programming»Numeric*. Con lo strumento *Positioning* portatevi sul bordo laterale o sull'angolo in basso a destra dell'array ed estendete quest'ultimo in modo da potervi inserire dodici elementi. Successivamente portate il puntatore del mouse tra un elemento a caso e il successivo ed estendete un po' le caselle degli elementi dell'array per visualizzare meglio i dati che inserirete. Infine, con lo strumento *Labeling* inserite i seguenti valori all'interno delle caselle:

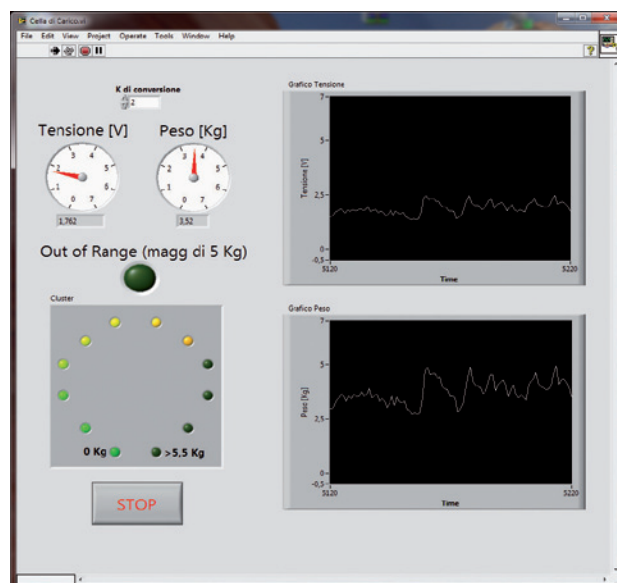
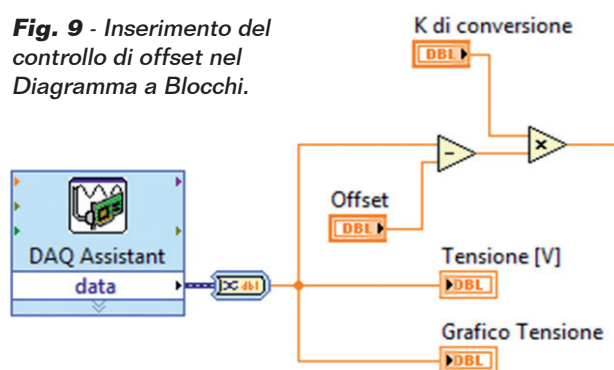
"-0,5\_0,5\_1\_1,5\_2\_2,5\_3\_3,5\_4\_4,5\_5\_5,5"

Ora, con lo strumento *Wiring* collegate al primo ingresso dell'ultimo blocco *Greater* rimasto il filo che trasmette il dato del peso, e al secondo ingresso collegate l'array di costanti. In uscita dal blocco *Greater* appena collegato avrete un array di dodici elementi di valori booleani in continuo aggiornamento. Per collegare quest'uscita al cluster contenente i LED

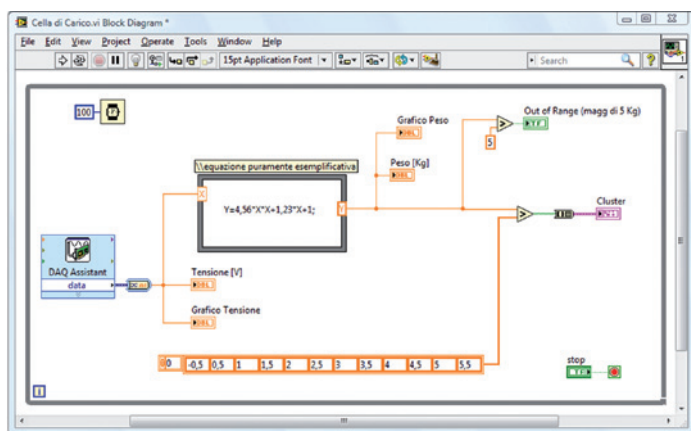
da controllare, dovete inserire il blocco *Array To Cluster* che trovate nella palette *Functions* alla voce *Programming»Cluster, Class, & Variant*; inoltre dovete fare clic con il pulsante destro del mouse su quest'ultimo blocco, selezionare *Cluster Size* e, nella finestra che si aprirà, specificare il numero di elementi che contiene il cluster, ovvero dodici. Tutto questo perché il software LabVIEW non sa da quanti elementi è composto il cluster da voi creato.

Adesso, con lo strumento *Wiring* potete collegare questi ultimi blocchi. Per finire, inglobate tutto il codice grafico fatto finora dentro un *While Loop* che troverete sulla palette *Functions* alla voce *Programming»Structures* e collegate con lo strumento *Wiring* il terminale del pulsante *Stop* al *Conditional Terminal* nell'angolo in basso a destra del ciclo *While*. Come ultima cosa inserite il blocco *Wait(ms)* situato nella palette *Functions* alla voce *Programming»Timing* e collegatelo a una costante con valore 100: fate ciò cliccando col tasto destro del

**Fig. 9 - Inserimento del controllo di offset nel Diagramma a Blocchi.**



**Fig. 10 - Il VI in esecuzione.**



**Fig. 11** - Diagramma a Blocchi con equazione di conversione tensione/peso.

mouse sul terminale d'ingresso del blocco *Wait(ms)*, selezionando *Create»Constant* e inserendovi il valore 100. Questo, per fare in modo che il ciclo *While* vada in esecuzione ogni 100 ms e perciò DAQ Assistant acquisisca 10 campioni al secondo.

Notate che se nel Pannello Frontale avete inserito il controllo offset, dovete collegarlo nel diagramma a blocchi. Per fare ciò avrete bisogno di un blocco *Divide* che trovate alla voce *Programming»Numeric*. Ora, con lo strumento *Positioning* e *Wiring* sistematelo e collegatelo tutto come in Fig. 9. Quando avrete finito di costruire il vostro VI, con il comando *File»Save as* scegliete un nome e una destinazione per il vostro VI, quindi cliccate sul pulsante *Run* nella barra degli strumenti per mettere il VI in esecuzione (Fig. 10).

Mentre il VI è in esecuzione, l'icona del pulsante *Run* si trasforma nella seguente: . Una volta terminata l'esecuzione, il pulsante *Run* torna al suo stato originale e gli indicatori del Pannello Frontale mantengono i dati visualizzati. Sempre durante l'esecuzione del VI, appare il pulsante *Abort Execution* ; facendovi clic potete fermare il VI immediatamente. Tale pulsante va usato soltanto se non riuscite ad arrestare il VI in altro modo.

Notate che se un VI non va in esecuzione, vuol dire che degli errori non lo rendono eseguibile. Se il VI che state creando o modificando contiene errori, il pulsante *Run* appare spezzato ; in questo caso, anche dopo aver finito i collegamenti nel diagramma a blocchi, il VI non è eseguibile.

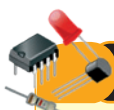
Generalmente l'errore può consistere in un ingresso richiesto non collegato o in un collegamento spezzato. Per risolvere il problema, fate clic sul pulsante *Run* spezzato in modo da accedere alla finestra di dialogo *Error List*, dove viene elencato e descritto ogni errore riscontrato; qui, facendo doppio clic su un errore andate direttamente alla posizione in cui esso si trova nel diagramma a blocchi.

## CONVERSIONE TENSIONE/PESO UTILIZZANDO UN METODO ALTERNATIVO

Come accennato prima, se volete essere più precisi sulla conversione dovete acquisire diverse misure di tensione associate a pesi noti e, aiutandovi con un foglio di calcolo, ricavare un'equazione ottenuta per interpolazione dei punti trovati. Dovrete poi utilizzare questa equazione all'interno del software per ricavarvi i valori di peso esatti dalla tensione acquisita da myDAQ.

Per prima cosa eliminate il controllo numerico *K* di conversione dal Pannello Frontale (ovviamente il controllo *K* scomparirà anche dal diagramma a blocchi). Ora andate sul diagramma a blocchi e inserite un *Formula Node*, che trovate sulla palette *Functions* alla voce *Programming»Structures*. All'interno del *Formula Node*, con lo strumento *Labeling*, inserite l'equazione che avete trovato utilizzando il foglio di calcolo (ad esempio  $Y=4,56*X*X+1,23*X+1;$ ) e cliccando il tasto destro del mouse sui bordi sinistro e destro del formula node inserite un ingresso nominandolo "X" (add input) e un'uscita nominandola "Y" (add output) rispettivamente. Ricordatevi di terminare l'equazione con un punto e virgola (;) e di rispettare le lettere maiuscole e minuscole (il *Formula Node* è case sensitive).

Per finire, con lo strumento *Positioning* riadattate un po' il diagramma a blocchi e con lo strumento *Wiring* effettuate i collegamenti necessari. Il vostro diagramma a blocchi dovrà presentarsi all'incirca come mostrato nella Fig. 11.



### per il MATERIALE

Questo progetto fa parte del set di esperimenti didattici realizzato da IRS-Elettronica per la board NI myDAQ con il supporto del software LabVIEW. Il set (cod. SETmyKIT, Euro 232,00) comprende tutti i componenti per realizzare i 22 esperimenti, i VI relativi e una protoboard. È possibile richiedere protoboard aggiuntive (cod. PROTOBOARDmyDAQ) al prezzo di 52,00 Euro. Tutti i prezzi sono comprensivi di IVA e non comprendono la board NI myDAQ né il software LabVIEW. Futura Elettronica distribuisce questi prodotti presso gli utenti privati mentre la distribuzione presso gli Istituti Scolastici è a cura di IRS srl (Via Vigonovese 81 - 35127 PADOVA - tel: +39 049 8705156 - info@irsweb.it).

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)  
Tel: 0331-799775 • Fax: 0331-792287  
www.futurashop.it



Riprendiamo lo shield Bluetooth per completare la nostra applicazione dotandola di ulteriori caratteristiche e funzionalità e approfittando per introdurre nuovi concetti Android. Quinta puntata.



## PROGRAMMIAMO CON ANDROID

**N**ella puntata precedente abbiamo analizzato il firmware per Android e per Arduino che permette di variare la luminosità ed il colore di una striscia a LED RGB ad interfaccia I<sup>2</sup>C-Bus. Immaginiamo ora di volere che la nostra striscia (o anche un singolo LED RGB) possa passare da un colore all'altro ad intervalli prestabiliti ed in tempi più o meno rapidi; questo potrebbe essere utile, ad esempio, per creare effetti alba-tramonto in presepi o modellini, senza dover ricorrere a più costose centraline dedicate. Dovremo perciò aggiungere alla nostra applicazione una modalità di programmazione tramite cui scegliere la sequenza di colori che invieremo ad Arduino tramite Bluetooth. Quest'ultimo memorizzerà i dati nella propria EEPROM, così sarà in grado di riprodurre la sequenza anche una volta che la connessione Bluetooth sarà terminata.

### CREARE IL MENU

Per non appesantire ulteriormente il layout grafico della schermata principale, gestiremo la possibilità di programmare questa sequenza in una *View* dedicata, che sarà pertanto richiamabile tramite una voce di menu. Come saprete, uno dei tasti "fisici" in uno smartphone Android è quello che permette di mostrare nella parte bassa dello schermo un menu contestuale con più opzioni, in modo da arricchire l'applicazione con maggiori informazioni o configurazioni. Esistono due modi per creare i menu: uno "statico", che consiste nel ricorrere a un file xml, e l'altro dinamico, in cui si possono aggiungere o togliere voci di menu direttamente dal codice. Sono modi equivalenti nel risultato, ma quello dinamico ci permette di modificare le opzioni che compariranno nel menu anche a run-time, a seconda di determinate azioni compiute dall'utente. Vedremo comunque entrambe le soluzioni.



Fig. 1

Innanzitutto dovremo aggiungere i seguenti *import* nel nostro codice java, per poter utilizzare le librerie interne adibite alla gestione del menu.

```
import android.view.Menu;
import android.view.MenuItem;
import android.view.MenuInflater;
```

A questo punto possiamo riprendere il codice della puntata precedente ed all'interno dell'Activity principale (class *BtClientActivity*) ridefinire la classe *onCreateOptionsMenu()* in modo da poter gestire un menu. Ogni volta che l'applicazione ridisegnerà la *View* principale verrà chiamata questa funzione; sarà proprio in questo punto che andremo a scrivere il nostro codice.

Nel caso come quello in oggetto, in cui sappiamo che le voci del menu saranno sempre le stesse, possiamo scrivere un file xml come quello visibile in Fig. 1, che salveremo nella cartella *layout*. Come si può notare, ogni tag *<item>* rappresenta una voce di menu e ad ogni *item* possiamo associare un *Id* univoco, una stringa di testo (nel campo *android:title*) ed il percorso della relativa immagine (nel campo *android:icon*) che deve essere presente nel progetto, normalmente all'interno della cartella *drawable*.

Ricordiamo inoltre, come già visto in precedenza nel caso di risorse grafiche, di non specificare l'estensione dell'icona (che nel nostro caso è *.png*) nel nome del percorso e di fare

uso solamente di lettere minuscole.

A questo punto abbiamo il nostro menu, ma è ancora "scollegato" e non visibile nella nostra applicazione. Per farlo comparire sullo schermo ogni qualvolta venga premuto il tasto "fisico" di menu, occorre scrivere ancora le poche righe di codice mostrate nel Listato 1, attraverso le quali istruiremo Android affinché possa utilizzare il file xml precedentemente creato, specificando il percorso della risorsa che sarà, appunto, *R.layout.menuseq*. Anche in questo caso *menuseq* è il nome del file xml privo della sua estensione e presente all'interno della cartella *layout*.

Il risultato che otterremo è visibile in Fig. 2. Possiamo anche ottenere lo stesso risultato senza scrivere alcun file xml, ma modificando la funzione *onOptionsItemSelected()* vista prima, come mostrato nel Listato 2.



Fig. 2

#### Listato 1

```
@Override
public boolean onCreateOptionsMenu(Menu menu)
{
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.layout.menuseq, menu);
    return true;
}
```

Sebbene i due metodi siano equivalenti, se utilizzate lo stesso menu anche in altre eventuali *View* vi consigliamo di usare il primo, perché vi risparmia di ripetere lo stesso codice più volte.

Inoltre, tramite il file xml è possibile aggiungere con semplicità maggiori caratteristiche, sfruttando il *code-assist* di Eclipse.

Qualunque metodo sia stato seguito, dobbiamo ora associare la pressione della voce di menu scelta (detta anche *MenuItem*) con la corrispondente azione da eseguire: questo viene realizzato ridefinendo la funzione *onOptionsItemSelected()* nella quale gestiremo appunto il parametro *item*.

Tale funzione, esplicitata nel Listato 3, effet-



**Listato 2**

```
@Override
public boolean onCreateOptionsMenu(Menu menu)
{
    menu.add(Menu.NONE, 0, 0, "Scan").setIcon(getResources().getDrawable(R.drawable.icon_scan));
    menu.add(Menu.NONE, 1, 1, "New Seq").setIcon(getResources().getDrawable(R.drawable.icon_seq));
    menu.add(Menu.NONE, 2, 2, "About").setIcon(getResources().getDrawable(R.drawable.icon_
information));
    return super.onCreateOptionsMenu(menu);
}
```

tua un controllo sull'*Id* della voce di menu selezionata (che viene restituito dall' API di sistema *getItemId()*); in base al suo valore effettueremo le nostre azioni, chiamando le rispettive funzioni.

Adesso bisogna fare una piccola precisazione a proposito della funzione *getItemId()*, la quale ritorna un valore intero: nel caso avessimo utilizzato il metodo dinamico per creare il nostro menu, tale valore corrisponderà a quello specificato nel metodo Add (**Listato 2**) e quindi, nel nostro caso, il valore 0 corrisponderà alla prima voce di menu ("Scan"), il valore 1 alla seconda e così via. Se invece avessimo creato il menu utilizzando il file xml *menuseq.xml*, allora la lettura del codice risulterebbe semplificata per via della stringa inserita nel campo *android:id* del relativo item (**Fig. 1**) che ci permette di riscrivere il codice del **Listato 3**, utilizzando il prefisso **R** seguito dal nome dell'*Id*.

Il tutto sarà più chiaro leggendo il codice così trasformato, nel **Listato 4**. In particolare, abbiamo utilizzato tre voci di menu: la prima per effettuare la scansione dei dispositivi Bluetooth circostanti, la seconda per creare le nostre sequenze di luce ed infine la terza per aggiungere ulteriori informazioni. È sempre buona norma, infatti, dotare le nostre applicazioni di un menu, sia per non appesantire la *View* principale con troppi pulsanti, sia per poter aggiungere, oltre alle eventuali finestre di opzione, anche una finestra di dialogo con le informazioni sulle nostre generalità, il nostro logo o semplicemente la versione dell'applicazione.

A riguardo rimandiamo al riquadro "Finestre di dialogo Android" che trovate nella pagina seguente, dove si utilizza la classe *AlertDialog* per visualizzare le informazioni aggiuntive accessibili con la terza voce del menu.

Andiamo ora ad implementare le altre due voci di menu che ci permetteranno di ac-

quisire nuovi concetti riguardanti il mondo Android.

**CREARE NUOVE ACTIVITY**

Finora in questa applicazione abbiamo utilizzato una singola Activity; in prima istanza, possiamo considerare un'Activity come la finestra grafica che permette all'utente di interagire con l'applicazione. Fisicamente un'Activity è una classe contenuta in un file sorgente con estensione **.java** (nel nostro

**Listato 3**

```
@Override
public boolean onOptionsItemSelected(MenuItem item)
{
    super.onOptionsItemSelected(item);
    switch(item.getItemId())
    {
        case 0:
            startScannerActivity();
            break;
        case 1:
            openSequencerView();
            break;
        case 2:
            openAboutDialog();
            break;
    }
    return true;
}
```

**Listato 4**

```
@Override
public boolean onOptionsItemSelected(MenuItem item)
{
    super.onOptionsItemSelected(item);
    switch(item.getItemId())
    {
        case R.id.scan:
            startScannerActivity();
            break;
        case R.id.seq:
            openSequencerView();
            break;
        case R.id.abt:
            openAboutDialog();
            break;
    }
    return true;
}
```



## FINESTRE DI DIALOGO ANDROID

Abbiamo già visto in precedenza come realizzare un messaggio a comparsa tramite la classe *MessageToast*, ma Android mette a disposizione anche finestre di dialogo più complesse e facilmente gestibili, che possiamo utilizzare per visualizzare informazioni veloci come quelle in una classica finestra di "About". Si tratta della classe *AlertDialog*, la quale con pochi metodi, come quelli elencati qui di seguito, permette di ottenere il risultato di **Fig. 3** senza costringerci ad utilizzare un'altra Activity.

```
AlertDialog dlg = new AlertDialog.Builder(this).create();
dlg.setTitle("RGB Controller");
dlg.setMessage("Elettronica In \n Copyright © 2012");
dlg.setIcon(R.drawable.icon_main);
dlg.show();
```



Fig. 3

Inoltre questa finestra verrà correttamente "distrutta" semplicemente premendo il tasto fisico "Back" sul dispositivo Android, senza doverci preoccupare di dotarla di un pulsante di uscita.

caso, *BtSequencerActivity.java*) che andremo a creare all'interno della cartella *src*, al pari degli altri file sorgente. Quando l'applicazione richiede particolari configurazioni o funzionalità aggiuntive, è meglio servirsi di nuove Activity per alleggerire la complessità, distribuendola su più finestre.

La principale differenza che troviamo rispetto al mondo Windows o Linux, al quale siamo più abituati, è che può essere attiva una sola Activity alla volta; nel caso di due o più Activity presenti nella nostra applicazione, se una di esse è in primo piano, le altre, oltre a non essere visibili, vengono configurate da Android in uno stato di secondo piano (o *background*) e, continuando a mantenere il loro ciclo di vita, potranno riprendere il loro stato di attività in futuro.

Senza scendere troppo nei dettagli, vediamo come poter lanciare una seconda Activity da quella principale. Occorre a questo punto introdurre il concetto di **Intent**, che può essere

### Listato 5

```
private void openSequencerView()
{
    Intent sequencerIntent = new Intent(this, BtSequencerActivity.class);
    startActivity(sequencerIntent);
}
```

visto come un messaggio col quale riusciamo ad ordinare ad Android l'esecuzione di una generica azione di sistema.

In questo caso dovremo creare un nuovo Intent passandogli l'informazione sulla nuova classe Activity ed utilizzare il metodo *startActivity()* al quale passeremo come argomento proprio l'Intent. Tutto questo è visibile nel **Listato 5**, in cui viene implementata la funzione chiamata dopo aver premuto la seconda voce del menu (*New Seq*).

L'Activity che andremo a creare sarà una classe estesa della super classe Activity e pertanto dovremo obbligatoriamente implementare i metodi ereditati da essa. In **Fig. 4** è possibile vedere un template minimale della nostra Activity in cui i metodi, che rappresentano il passaggio tra i suoi vari stati e quindi il suo ciclo di vita, non fanno altro (almeno per ora) che richiamare le funzionalità della sua super classe.



Fig. 4

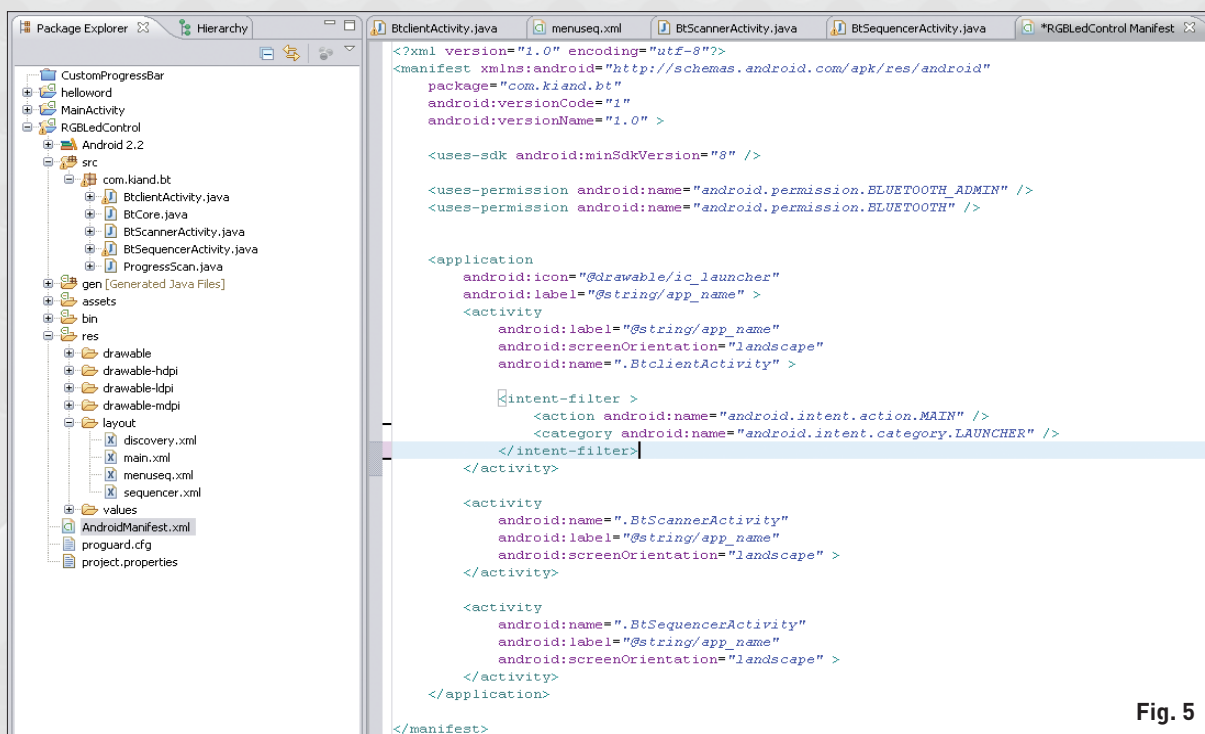


Fig. 5

Tutti questi metodi (*onCreate()*, *onStart()*, *onResume()*, *onPause()*, *onStop()*, *onDestroy()*) vengono chiamati direttamente dal sistema operativo e noi possiamo gestirli in modo attivo inserendo codice personalizzato a seconda delle nostre esigenze, oppure demandare tutto alla super classe tramite, ad esempio, la riga di codice *super.onStop()* valida nel caso del metodo *onStop()*.

Possiamo spiegare brevemente questi metodi dicendo che la *onCreate()* viene chiamata una sola volta, non appena viene eseguita la *startActivity()* vista precedentemente. È proprio qui che andremo ad impostare il layout grafico scelto per questa Activity (tramite la riga *setContentView(R.layout.sequencer.xml)* e ad aggiungere il nostro codice per associare le azioni agli eventi generati dalla pressione di tasti e controlli generici.

Il metodo *onStart()* viene chiamato anch'esso una sola volta e poco prima che l'Activity venga fisicamente disegnata. Il metodo *onResume()* viene chiamato prima che l'Activity passi in stato *running*, quindi verrà chiamato la prima volta, ma anche ogni volta che l'Activity esca dallo stato *stopped* e torni attiva. Il metodo *onPause()* viene chiamato quando l'Activity è ancora visibile, ma un'altra Ac-

tivity sta prendendo il suo posto per andare in *foreground*, ed il metodo *onStop()* quando proprio l'Activity non è più visibile. Infine, il metodo *onRestart()* è chiamato ogni qualvolta l'Activity venga ridisegnata ed il metodo *onDestroy()* appena prima che l'Activity venga distrutta.

Per poter utilizzare l'Activity appena creata, non dobbiamo dimenticarci di informare Android della sua esistenza; allo scopo la aggiungiamo nel file *AndroidManifest.xml*, già visto ed utilizzato nel corso delle puntate precedenti, che si presenterà come in Fig. 5. Da notare il tag *<intent-filter>* che deve essere presente solo nell' Activity principale, con la quale parte l'applicazione e dalla quale verranno lanciate le altre Activity. Spiegate le generalità, vediamo ora come useremo le Activity in questa puntata.

### LIGHT SEQUENCER

Premendo sulla voce di menu New Seq precedentemente creata, apparirà ora una nuova finestra con i controlli che abbiamo aggiunto nel nuovo file di layout *sequencer.xml*. e visibile in Fig. 6. Come noterete, sono gli stessi controlli usati nell'Activity principale, ma oltre a questi ci sono anche altri pulsanti



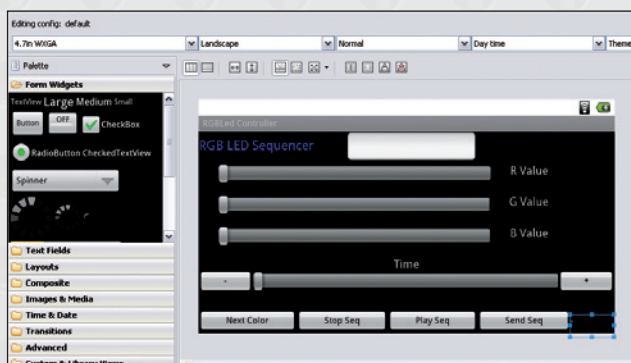


Fig. 6

per consentire maggiori funzionalità. L'idea è quella di creare una sequenza di colori direttamente sul nostro smartphone, quindi inviarla ad Arduino sotto forma di stringa Bluetooth. Abbiamo inserito anche una ulteriore *SeekBar* attraverso la quale settare il tempo (in secondi) in cui resterà attivo il colore scelto. Cliccando poi sul pulsante "Next Color" il colore attivo ed il relativo intervallo verranno memorizzati in una lista di stringhe e potremo scegliere un altro colore col suo relativo intervallo. Una volta terminata la nostra sequenza possiamo premere il pulsante "Stop Seq" e successivamente il pulsante "Play Seq" per riprodurla mediante l'RGB shield, oppure "Send Seq" per inviarla ad Arduino, che la memorizzerà sulla propria EEPROM. A questo punto, premendo "Play Seq" nella finestra principale la sequenza verrà rieseguita con un loop infinito anche dopo aver terminato l'applicazione sullo smartphone. Per l'invio della sequenza possiamo inventarci un protocollo di messaggi come preferiamo; sapendo che non abbiamo vincoli di tempo stringenti utilizzeremo semplicemente l'invio di stringhe. Sulla pressione del pulsante "Send Seq" abbiamo, così, scelto di inviare il carattere 'S' seguito dal numero totale di colori e successivamente le informazioni sulle componenti di colore e l'intervallo di tempo tramite i caratteri R, G, B e T.

Se, ad esempio, volessimo una sequenza che inizi con un colore rosso per poi passare dopo 30 secondi ad un verde e dopo 40 al blu, e desiderassimo rieseguire il loop dopo 50 secondi, invieremo un messaggio di questo tipo:

S3

R255 G000 B000 T30

### Listato 6

```
<SeekBar
    android:id="@+id/seekbarB"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:max="255"
    android:progress="0"
    android:progressDrawable="@drawable/custom_seek_bar"
    android:thumb="@drawable/custom_seek_bar_thumb"
    android:thumbOffset="0dip"
    android:maxHeight="10dip"
    android:layout_weight="80"
/
```

R000 G255 B000 T40

R000 G000 B255 T50

Dall'altra parte Arduino, dovrà essere in grado di parserizzare le stringhe correttamente, salvare i valori in un array, memorizzarli internamente e utilizzarli quando, dalla nostra applicazione Android, riceverà la stringa "P0".

### CONTROLLI PERSONALIZZATI

Non ci addentreremo nel codice dell'Activity *BtSequencerActivity* perché molto simile a quello dell'Activity principale già vista, ma possiamo approfittare dell'occasione per darvi alcuni suggerimenti su come personalizzare le vostre applicazioni creando controlli custom che presentino un'interfaccia grafica a vostro piacimento, un po' diversa da quelle classiche. Prendiamo, ad esempio, le quattro *SeekBar* inserite nel layout in Fig. 6 ed immaginiamo di volerle cambiare alcuni parametri estetici come la forma del cursore o il colore di riempimento della barra. Questa, per impostazione predefinita è arancione, ma nel nostro caso sarebbe interessante fosse rossa per la *SeekBar* relativa alla componente rossa, verde per la componente verde e blu per la componente blu. Android ci offre questa

### Listato 7

```
<selector xmlns:android="http://schemas.android.com/apk/res/android">

    <item android:state_pressed="true"
        android:drawable="@drawable/custom_thumb_state_pressed" />

    <item android:state_focused="true"
        android:drawable="@drawable/custom_thumb_state_selected" />
    <item android:state_selected="true"
        android:drawable="@drawable/custom_thumb_state_selected" />
    <item android:drawable="@drawable/custom_thumb_state_default" />
</selector>
```





Fig. 7

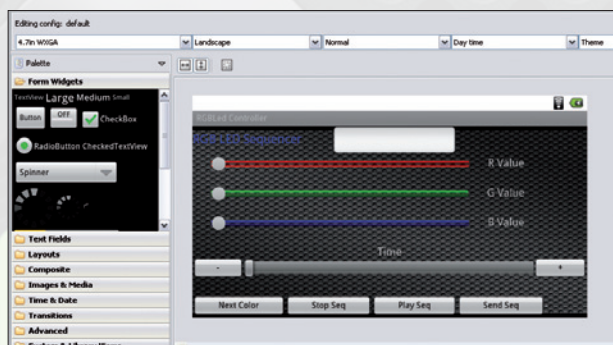


Fig. 8

### Listato 8

```

<?xml version="1.0" encoding="UTF-8"?>
<resources>
  <style name="CustomSeekBar" parent="android:Widget.SeekBar">
    <item name="android:progressDrawable">@drawable/custom_seek_bar_r</item>
    <item name="android:thumb">@drawable/custom_seek_bar_thumb</item>
    <item name="android:thumbOffset">0dip</item>
  </style>
</resources>

```

possibilità in modo tutto sommato semplice: riprendiamo il file di layout *sequencer.xml* ed aggiungiamo all'interno del tag *<SeekBar>* che ci interessa modificare, il campo *android:progressDrawable*, che imposteremo con un file di risorsa personalizzato (vedere il Listato 6).

Questo file di risorsa (*@drawable/custom\_seek\_bar\_R*) è un file xml (nulla, però, ci vieta di usare un'immagine) come quello visibile in Fig. 7 in cui sono presenti due item: uno per ridisegnare lo sfondo della barra di scorrimento (*background*) e l'altro per ridisegnare la

barra a sinistra del cursore (*progress*). Ognuno di questi item è personalizzabile tramite una serie di tag, come *corners* per definire gli angoli della barra, *solid* per definire un colore di sfondo pieno, piuttosto che *gradient* per definire l'effetto di sfumatura desiderato.

Allo stesso modo possiamo modificare la forma e il colore del cursore, aggiungendo il campo *android:thumb* ed impostandolo un file di risorsa come quello nel Listato 7, che a sua volta presenta collegamenti ad altri file di risorsa in base ai differenti stati del cursore (*pressed*, *focused*, *selected*).

Possiamo allora sbizzarrirci su forma, colore ed altri effetti, creando tanti file xml o immagini per costruire tre *SeekBar* di colori diversi e con caratteristiche estetiche differenti, come si può vedere in Fig. 8.

A questo punto, sulla base delle informazioni viste prima, potremmo anche creare un nostro controllo e dargli un nome a nostra scelta, semplicemente creando un nuovo file xml (non importa il nome del file, basta che sia presente il tag *<resources>*) e che andremo a mettere all'interno della cartella *values*. Un file di questo tipo è visibile nel Listato 8.

Ora nel nostro file di layout potremmo usare direttamente questo controllo usando il tag con lo stesso nome che gli abbiamo dato in

precedenza (nel nostro caso *<CustomSeekBar>*) preoccupandoci solo delle dimensioni e della sua posizione all'interno della *View*.

### SCANSIONE BLUETOOTH

Negli esempi fatti fino ad ora la comunicazione Bluetooth tra Android ed Arduino è avvenuta cablando nel codice l'indirizzo MAC del modulo RN-42; ciò rende l'applicazione poco flessibile e nel caso in cui volessimo riutilizzare questo stesso codice per gestire un altro modulo Bluetooth ci troveremmo fortemente vincolati.

Ora che abbiamo introdotto il menu, possiamo lanciare un'Activity dedicata che gestisca la ricerca dei dispositivi remoti nelle vicinanze e restituisca il MAC Address del dispositivo selezionato. La prima voce di menu, "Scan Device" fa proprio questo. Essendo un'Activity da cui vogliamo ottenere un risultato che utilizzeremo nell'Activity principale, dovremmo lanciarla in un modo diverso dal precedente e quindi, anziché utilizzare il metodo `startActivity()`, useremo `startActivityForResult()` (**Listato 9**).

Sempre nell'Activity principale, dovremo gestire questo risultato quando il sistema operativo chiamerà il metodo `onActivityResult()` (vedere ancora il **Listato 9**) in cui ricaviamo il MAC Address dal parametro `data`.

Analizziamo ora la classe `BtScannerActivity` che ha poche righe di codice ma è molto particolare.

Nel metodo `onCreate()` inseriremo il solito file di layout `discovery.xml`, che questa volta avrà un controllo nuovo: la lista `ListView`, in cui verranno elencati i dispositivi rilevati. Sempre nella `onCreate()`, troviamo la creazione dinamica di un controllo `ProgressScan` (il cerchietto rotante che in Android indica l'attesa dell'esecuzione di un'operazione) attraverso la quale viene lanciato un thread parallelo (`_discoveryWorker`) che effettua la scansione Bluetooth vera e propria.

Precisiamo che `_discoveryWorker` è un oggetto della classe base `Runnable` creato nello stesso file e che in Android viene utilizzato per gesti-

re appunto i thread, implementando il metodo `run()` di questa classe.

Infine troviamo i due **IntentFilter**, `discoveryFilter` e `foundFilter` (visibili nel **Listato 10**) che sono i responsabili della rilevazione dei dispositivi Bluetooth. Un `IntentFilter` può essere visto come una sorta di direttiva con cui informiamo il sistema operativo su quali Intent (messaggi provenienti dal sistema operativo) siamo in grado di gestire e con quale oggetto di classe.

Nel nostro caso l'Intent `foundFilter` sarà attivo sulla ricezione del messaggio Bluetooth `ACTION_FOUND`, e quindi ogni qual volta venga rilevato un nuovo dispositivo questo sarà gestito nel metodo `onReceive()` dell'oggetto `_foundReceiver`, dal momento che, tramite il metodo `registerReceiver()`, abbiamo associato l'Intent a questo oggetto. Nel metodo `onReceive()` inseriremo il nuovo dispositivo nella lista grafica, che così verrà continuamente aggiornata (`ShowDevices()`).

Per far terminare correttamente il thread e quindi la scansione, useremo proprio il secondo `IntentFilter` (`discoveryFilter`) che, invece, gestirà il messaggio Bluetooth di fine scansione (`ACTION_DISCOVERY_FINISHED`) facendo terminare il thread `_discoveryWorker` e lasciando all'utente la scelta del dispositivo.

A questo punto il controllo passerà alla `onListItemClick()` (sempre nel **Listato 10**) in cui saremo noi a creare un messaggio (Intent) con il MAC Address del dispositivo scelto, che restituiranno all'Activity principale.

#### Listato 9

```
public void startScannerActivity()
{
    Log.v(TAG, "Start Scan");
    Intent btScanIntent = new Intent(this, BtScannerActivity.class);
    startActivityForResult(btScanIntent, BtCore.REQUEST_DISCOVERY);
}
// after select, connect to device
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
    if (requestCode != BtCore.REQUEST_DISCOVERY)
    {
        return;
    }
    if (resultCode != RESULT_OK)
    {
        return;
    }
    BluetoothDevice device = data.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
    BtCore.serverAddressAfterScan = device.getAddress();
}
```



## Listato 10

```

protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    getWindow().setFlags(WindowManager.LayoutParams.FLAG_BLUR_BEHIND, WindowManager.LayoutParams.FLAG_BLUR_BEHIND);
    setContentView(R.layout.discovery);
    Log.w("RGB-CONTR-SCAN", "onCreate");

    // BT isEnabled
    if (!_bluetooth.isEnabled())
    {
        Log.w("RGB-CONTR-SCAN", "Disable!");
        finish();
        return;
    }

    // Register Receiver
    IntentFilter discoveryFilter = new IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_FINISHED);
    registerReceiver(_discoveryReceiver, discoveryFilter);
    IntentFilter foundFilter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
    registerReceiver(_foundReceiver, foundFilter);
    ProgressScan.indeterminateInternal(BtScannerActivity.this, _handler, "Scanning...", _discoveryWorker,
        new OnDismissListener()
        {
            public void onDismiss(DialogInterface dialog)
            {
                for (; _bluetooth.isDiscovering();)
                {
                    _bluetooth.cancelDiscovery();
                }
                _discoveryFinished = true;
            }
        }, true);
}

private Runnable _discoveryWorker = new Runnable()
{
    public void run()
    {
        // Start search device
        _bluetooth.startDiscovery();
        Log.d("RGB-CONTR-SCAN", "Starting Discovery");
        for (;;)
        {
            if (_discoveryFinished)
            {
                Log.d("RGB-CONTR-SCAN", "Finished");
                break;
            }
            try
            {
                Thread.sleep(100);
            }
            catch (InterruptedException e)
            {
            }
        }
    }
};

protected void onItemClick(ListView l, View v, int position, long id)
{
    Log.d("RGB-CONTR-SCAN", "Click device");
    Intent result = new Intent();
    result.putExtra(BluetoothDevice.EXTRA_DEVICE, _devices.get(position));
    setResult(RESULT_OK, result);
    finish();
}

```

## CONCLUSIONI

Abbiamo completato la nostra applicazione Bluetooth rendendola più completa e funzionale ed acquisendo alcuni concetti Android (*Intent*, *FilterIntent*, *Runnable*) che ci troveremo ad utiliz-

zare anche nei nostri prossimi progetti. Durante le puntate seguenti analizzeremo come gestire da un'applicazione Android, la ricezione e l'invio di messaggi e chiamate telefoniche verso dispositivi remoti dotati di modulo cellulare. ■