

Mensile di progettazione elettronica, attualità scientifica, novità tecnologiche.

€ 6,00

Elettronica In

oltre l'elettronica

www.elettronica.in.it

Anno XIX - n. 176
Maggio 2013

Robot Area

L'Italia alla
Shell Eco-marathon

Zerorobotics:
per due anni
in finale

mOway
il piccolo robot
superaccessoriato



- Scheda USB per stepper motor
- Sistema Home Theater video light
- Gestire i moduli GSM con Arduino
- Dotklok, il montaggio
- Alimentatore a corrente costante da laboratorio
- myKIT, amplificazione audio e LabVIEW

Controlla da
remoto la tua

3DAG

tramite
Raspberry Pi



- Analizzatore di spettro, parliamo di FFT e analizziamo il firmware
- Termostato Web con shield di espansione ADC per RaspberryPi
- OpenWheels, il firmware

Gonfiatutto programmabile



ALIMENTATORE A CORRENTE COSTANTE DA LABORATORIO



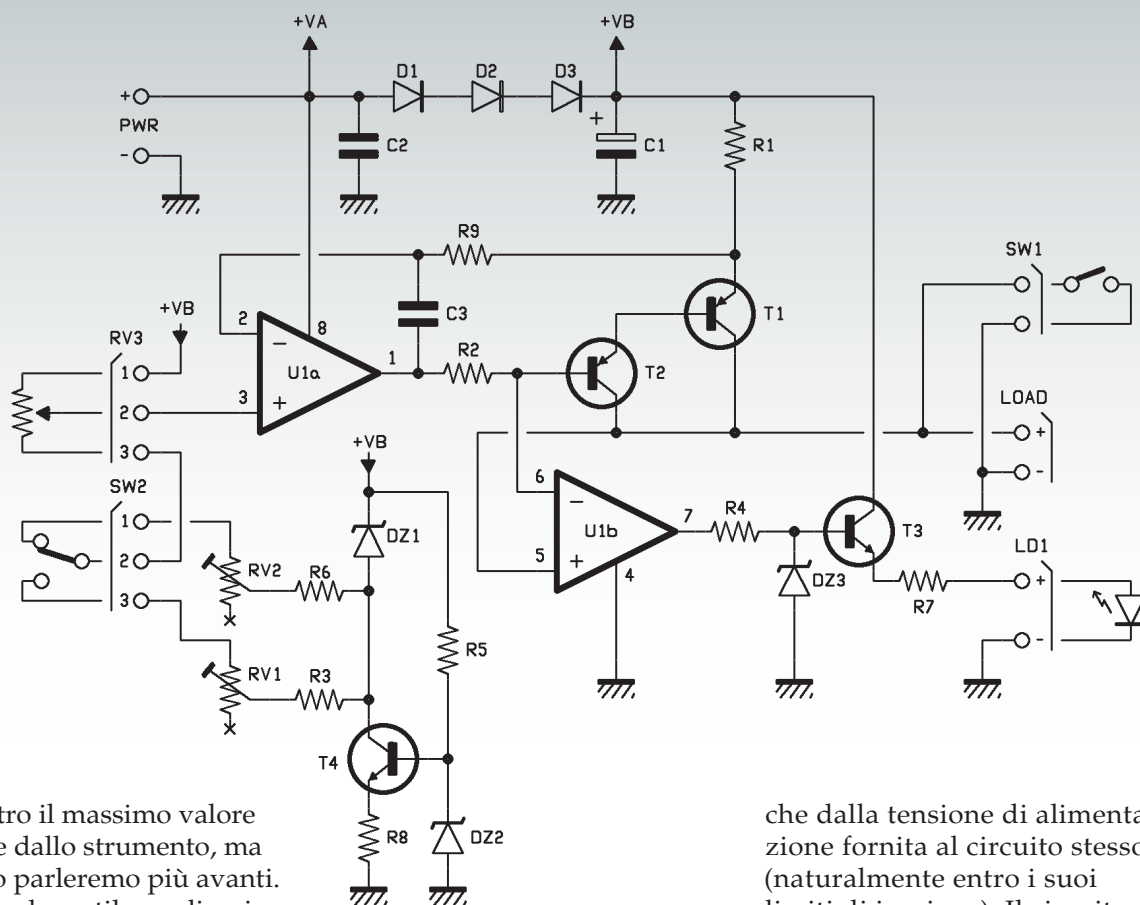
Un utile strumento di laboratorio per alimentare e provare componenti elettronici in modo controllato e senza rischiare di danneggiarli.

di MARCO MOROCUTTI

Quello che qui presentiamo è un semplice, ma utile, strumento di laboratorio che consente di fornire una corrente costante (e regolabile) ad un componente elettronico che si trova sottoposto a prova. Il circuito ha diverse applicazioni: può servire per verificare empiricamente la quantità di luce emessa da un LED quando viene alimentato con una data corrente, senza dover scegliere la giusta resistenza di caduta e collegarla in serie al LED. Poiché, a differenza di un normale alimentatore stabilizzato, questo

strumento non fornisce una tensione costante, ma una corrente costante, è anche possibile collegare in serie più LED di tipo diverso, o anche di diverso colore, per valutarne contemporaneamente la diversa emissione luminosa. Che siano uno o più LED saremo comunque certi che la corrente di prova sarà sempre la stessa, con l'ulteriore vantaggio che non sarà mai possibile danneggiare i LED in prova.

Questa regola vale naturalmente fino a che la caduta di tensione data dalla serie dei LED è com-



presa entro il massimo valore erogabile dallo strumento, ma di questo parleremo più avanti. Una seconda e utile applicazione dello strumento è verificare rapidamente la tensione caratteristica di un diodo Zener, anche in questo caso senza il rischio di danneggiarlo a causa di un collegamento errato. Ci basterà impostare la corrente ad un valore adeguato, che in questo caso potrebbe essere 10 mA, collegare all'uscita il nostro Zener in prova e misurare la tensione ai suoi capi.

LE CARATTERISTICHE

Quali dovrebbero essere le caratteristiche di un circuito da laboratorio come quello qui descritto? Innanzitutto la corrente erogata, che deve essere regolabile fra zero ed un valore utile ad eseguire diversi tipi di prova. Il nostro circuito possiede due diversi valori di fondo scala, selezionabili con un deviatore, che consentono di erogare corrente in due intervalli: da 0 a 20 mA e da 0 a

100 mA. La portata che impiegheremo più di frequente sarà probabilmente la prima, ma in laboratorio sarà certamente utile poter arrivare anche a 100mA.

Un'altra caratteristica è quella di essere avvisati quando lo strumento non riesce ad erogare la corrente richiesta. Questo accade se la caduta di tensione sul carico è troppo elevata (ad esempio: troppi LED in serie), oppure quando la tensione di alimentazione fornita al nostro strumento è troppo bassa per il tipo di carico che abbiamo collegato. Se questa condizione si verifica, il circuito lo indicherà accendendo un LED di segnalazione. Infine, la stabilità. Un alimentatore in corrente costante deve erogare sempre la corrente impostata, indipendentemente sia dal carico collegato

che dalla tensione di alimentazione fornita al circuito stesso (naturalmente entro i suoi limiti di impiego). Il circuito che presentiamo possiede tutte queste caratteristiche, ed in più è stato volutamente progettato utilizzando solo componenti elettronici di uso comune.

LO SCHEMA ELETTRICO

Lo stadio di uscita del circuito è un inseguitore di emettitore che sfrutta la coppia di transistor T1 e T2, collegati in configurazione Darlington per aumentarne il guadagno in corrente. In virtù dell'elevato guadagno, la corrente che attraversa il carico collegato all'uscita LOAD sarà quasi identica a quella che attraversa la resistenza di emettitore R1. Se questa corrente è mantenuta costante, anche la corrente nel carico lo sarà. U1 è un comune doppio amplificatore operazionale LM358, dove la metà indicata con U1a è collegata in configurazione

“buffer” in modo da mantenere costante la tensione ai capi di R1. Per farlo, il circuito utilizza come punto di riferimento il lato più positivo di R1, quello connesso al catodo di D3 ed indicato sullo schema con “+VB”, che è anche l'estremo (positivo) del potenziometro RV3. Grazie alla retroazione presente nel gruppo U1a / T2 / T1, la tensione sul cursore di RV3 (rispetto al punto “+VB”), che è applicata all'ingresso non invertente di U1a, porterà ad avere una tensione identica anche su R1. Spostando il cursore di RV3 da un estremo all'altro della corsa, avremo quindi una tensione su R1 che andrà da zero fino al valore di fondo scala, e lo stesso farà la corrente nel carico, che è esattamente ciò che vogliamo ottenere.

Per avere due diversi valori di corrente massima, cioè 20 mA e 100 mA, occorre che l'altro estremo di RV3 possa assumere due diversi (e costanti) valori di tensione. A questo provvede il deviatore SW2, che commutando in serie ad RV3 due distinti rami, costituiti da R3 / RV1 o da R6 / RV2, fornisce due possibili valori di tensione ai capi di RV3. La tensione complessiva applicata a questo partitore “variabile” è la tensione presente sul diodo Zener DZ1; per fare in modo che tale tensione sia stabile, la corrente che lo attraversa è a sua volta regolata da uno stadio in corrente costante, costruito attorno al transistor T4. I valori di fondo scala si potranno regolare una volta per tutte, in fase di messa a punto, tramite i trimmer RV1 e RV2. Il condensatore C3 riduce la velocità complessiva dell'insieme, in modo da evitare auto-

oscillazioni dovute all'elevato guadagno del Darlington. Da notare che l'alimentazione dell'operazionale LM358 è mantenuta superiore di circa 2V rispetto alla tensione “+VB”, e ciò grazie alla presenza dei diodi D1 / D2 / D3. Tale espediente è necessario perché questo operazionale non può fornire in uscita una tensione che arriva fino alla sua alimentazione positiva, mentre per ridurre fino a zero la tensione su R1 (e quindi la corrente sul carico) occorre che l'uscita di U1 possa salire a sufficienza, per avvicinarsi a “+VB”.

La seconda metà di U1 svolge

il compito di segnalare quando il circuito non è in grado di erogare la corrente impostata. Esso sorveglia la tensione presente sul collettore di T1 e la confronta con quella in base a T2. Quando la prima sale fino a raggiungere la seconda, significa che la coppia T1 / T2 è vicina alla saturazione e il circuito non può erogare la corrente prevista. In tal caso si accenderà il LED collegato all'emettitore di T3, a cui il

Le lampadine a LED di CREE

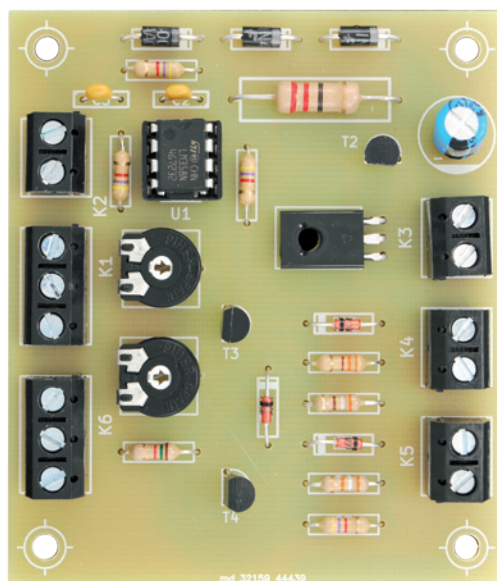
Da quando è stata inventata la lampadina, si sono susseguiti innumerevoli tentativi di trovare un'alternativa che fosse meno esigente in termini di consumi energetici e che al tempo stesso non fosse una “bomba” in termini di sostanze e materiali da riciclare. I LED negli ultimi anni hanno assunto il ruolo principale come alternativa a tutte le altre soluzioni per la realizzazione della lampadina del 21mo secolo, ma il prezzo restava un ostacolo significativo alla sostituzione del classico e onnipresente bulbo domestico da 60 watt. Da marzo Cree ha reso disponibile quello che potrebbe essere considerato come il vero primo passo verso la diffusione di massa delle

lampadine LED in ambito domestico. Con soli 9,5 watt assorbiti, il modello indicato come sostitutivo dei tradizionali bulbi da 60 watt genera 800 lumen con temperatura di colore di 2700 o 5000K e può anche essere regolato con un dimmer. Il tutto a 13 o 14 dollari, a seconda che siano warm o cold white. La vendita al momento è limitata alla catena americana The Home Depot.



Elenco Componenti:

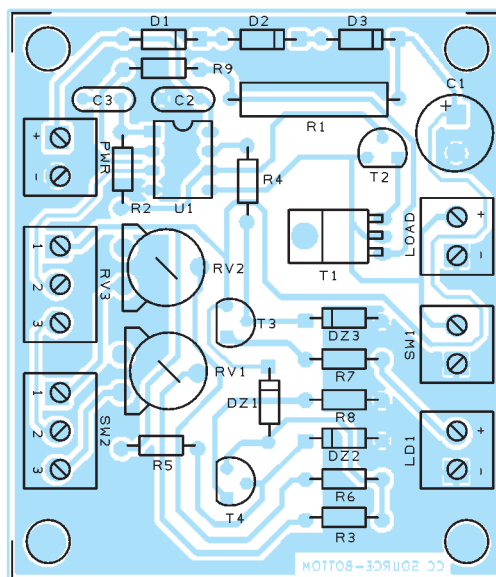
R1: 22 ohm 1W
 R2: 4,7 kohm
 R3: 4,7 kohm
 R4: 4,7 kohm
 R5: 1,5 kohm
 R6: 39 kohm
 R7: 390 ohm
 R8: 330 ohm
 R9: 4,7 kohm
 RV1: Trimmer 1 kohm
 RV2: Trimmer 10 kohm
 RV3: Potenzimetro lineare 4,7 kohm
 C1: 100 μ F 35 VL elettrolitico
 C2: 100 nF ceramico
 C3: 100 pF ceramico
 U1: LM358
 D1: 1N4004
 D2: 1N4004
 D3: 1N4004



LD1: LED 5 mm rosso
 DZ1: Zener 4,7 V 1/2W
 DZ2: Zener 4,7 V 1/2W
 DZ3: Zener 4,7 V 1/2W
 T1: BD140

T2: BC327
 T3: BC337
 T4: BC337
 SW1: Interruttore
 SW2: Deviatore

Varie:
 - Morsetto 2 poli (4 pz.)
 - Morsetto 3 poli (2 pz.)
 - Zoccolo 4+4
 - Circuito stampato



circuito costruito attorno al transistor T3 ed allo Zener DZ3 fornirà una corrente adeguata, indipendente dalla tensione di alimentazione generale. Una piccola raffinatezza finale. Il commutatore SW1, posto in parallelo al carico, consente di accendere o spegnere la corrente nel carico stesso mentre lo strumento continua ad erogare la corrente impostata. Si tratta di un accorgimento per evitare che, all'atto del collegamento del carico, in quest'ultimo scorra inizialmente (per qualche microsecondo) una corrente elevata, mentre il circuito si assesta fino a trovare il suo punto di funzionamento. L'uso di SW1 è quindi opzionale, ma se lo strumento viene usato per testare componenti delicati è consigliabile utilizzarlo, mantenendolo chiuso ed aprendolo solo quando il carico è già connesso.

REALIZZAZIONE PRATICA

Per costruire questo strumento è disponibile la traccia di un circuito stampato monofaccia, scaricabile dal nostro sito www.elettronica.in.it; oppure, grazie alla sua semplicità, si può anche utilizzare una basetta forata. In ogni caso ci servirà un buon saldatore a punta fine,

T1 merita un discorso a parte, perché il calore generato nelle condizioni di funzionamento più sfavorevoli (tensione generale di alimentazione elevata, corrente elevata e carico a bassa caduta) richiede il montaggio di un semplice dissipatore, che si può realizzare con un lamierino sottile sagomato

Tabella 1

Tensione in ingresso	Corrente in uscita	Massima tensione sul carico	
		con alimentazione a 12V	con alimentazione a 24V
da 12 Vcc a 24 Vcc	due portate: 0-20mA e 0-100mA	8V a 20mA, 6V a 100mA	19V a 20mA, 18V a 100mA

possibilmente del tipo a temperatura regolata. Inizieremo come consueto dai componenti a più basso profilo, cioè le resistenze ed i diodi, seguite dallo zoccolo per U1 e dai due trimmer. Saldare poi i transistor T2, T3 e T4 ed i condensatori.

ad "U" e opportunamente forato con una punta da 3 mm. Inserire il dissipatore sotto al transistor, badando bene che non ne tocchi i piedini, e fissare il tutto al circuito stampato mediante una vite da 3 mm, un dado e due rondelle. Dopo

aver stretto la vite saldate il transistor, ed infine inserite l'LM358 sullo zoccolo ad 8 pin (attenzione al verso!).

Le connessioni con l'esterno si possono effettuare saldando dei fili direttamente al circuito, oppure montando dei morsetti a vite a passo 5 mm, o anche saldando dei pin-strip (cioè degli spezzoni di una striscia di pin) a cui saldare poi i fili di collegamento.

Il tutto potrà poi essere introdotto in un contenitore plastico o metallico, con il LED di segnalazione inserito in un portale ed il potenziometro di regolazione ed il deviatore fissati in posizione conveniente. Vicino al deviatore è utile incollare una etichetta che distingua fra le portate 0-20 mA e 0-100 mA. Il potenziometro dovrà essere collegato in modo che la corrente salga ruotandolo in senso orario; se avviene il contrario, basta scambiare fra loro i collegamenti dei due estremi.

TARATURA E COLLAUDO

Alimentare il circuito con 12V continui, rispettando la polarità e senza collegare nulla all'uscita, e verificare che lo

strumento assorba dall'alimentatore una corrente di circa 25mA. A questo punto selezionare la portata da 20 mA di fondo scala e collegare un milliamperometro (un multimetro sulla portata di misura di corrente continua) direttamente all'uscita. In questo modo si realizza un cortocircuito, ma poiché il nostro strumento è un alimentatore in corrente costante, la corrente che fluirà sarà comunque limitata e regolata.

Portare il potenziometro al valore di corrente massima e regolare RV2 per ottenere esattamente 20 mA. Spostare poi il commutatore sulla portata di 100 mA e regolare RV1 per ottenere esattamente 100 mA.

Si noterà che rimuovendo il carico (cioè il multimetro) si accende il LED di segnalazione, perché lo strumento non riesce (giustamente!) ad erogare al carico la corrente voluta.

Se volete verificare il buon funzionamento dello strumento utilizzate il milliamperometro per regolare il potenziometro su un valore a piacere, ad esempio 10 mA; collegate poi

un LED in serie al multimetro e noterete che la corrente resterà la stessa. Provando poi con un numero superiore di LED (senza raggiungere il limite di saturazione) potrete verificare che la corrente resterà praticamente la stessa, nonostante la caduta di tensione sul carico sia passata da zero (il solo multimetro) ad alcuni volt (la serie di LED). Lo strumento è ora pronto per essere impiegato nelle vostre prove di laboratorio. Ricordate di rispettare i limiti di funzionamento indicati in tabella, evitando quindi che durante l'uso il LED di errore si accenda per segnalare che lo stadio di uscita è in saturazione e la corrente voluta non può essere erogata.

LIMITI DI FUNZIONAMENTO

In **Tabella 1** sono riportati i limiti di funzionamento dello strumento, in particolare la massima tensione del carico nelle varie condizioni d'uso.

Tutti i componenti utilizzati in questo progetto sono di facile reperibilità mentre il master dello stampato può essere scaricato dal sito della rivista (www.elettronica.in.it).

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

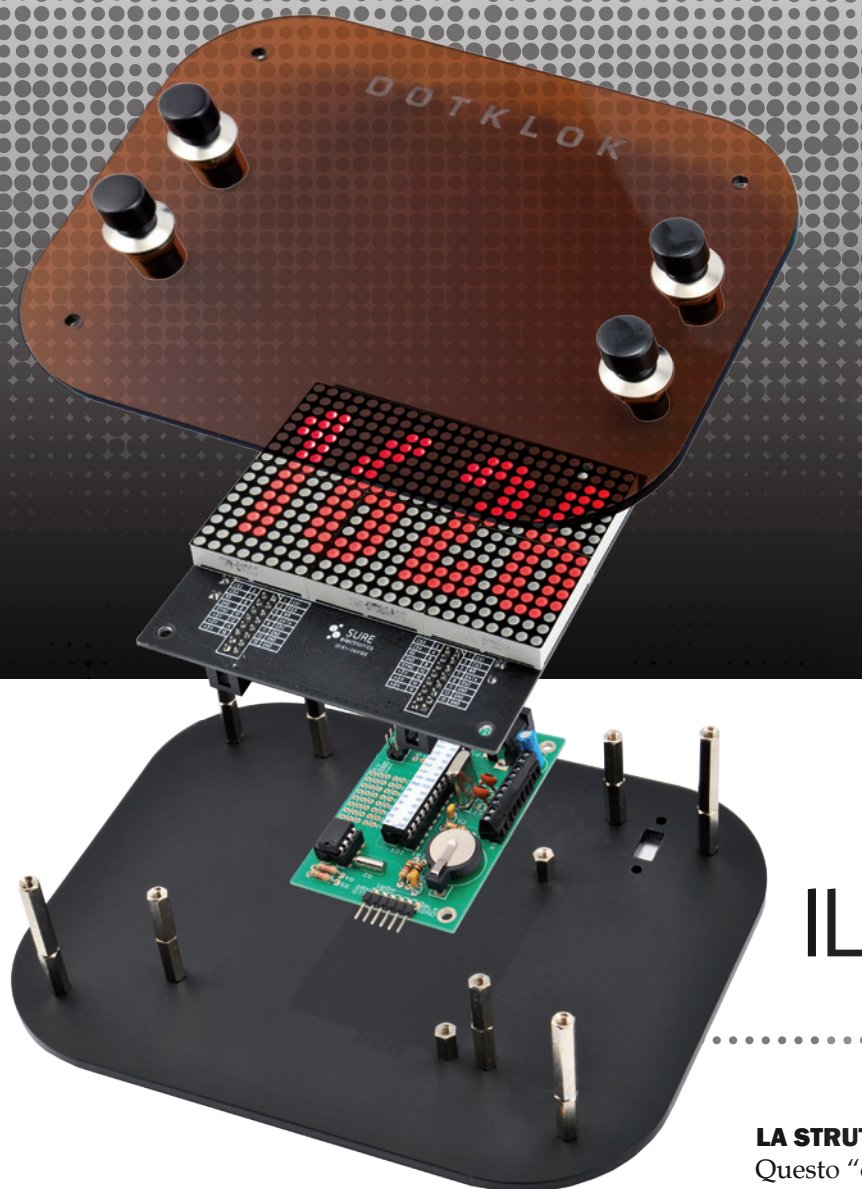
Via Val Sillaro 38 • 00141 Roma • Tel: 06-8104753

Questo mese ci dedichiamo all'assemblaggio di questo originale orologio Open Source, ideato e sviluppato da Andrew O'Malley, basato su Arduino.



DOTKLOK, IL MONTAGGIO

..... di ANDREW O'MALLEY E SIMONE MAJOCCHI



Il progetto di questo particolare orologio nasce, come abbiamo visto lo scorso mese, dalla fantasia di un canadese - Andrew O'Malley - che ha pensato di creare una vera e propria piattaforma di visualizzazione del tempo, facendo ruotare tutto attorno a tre componenti fondamentali. Con un orologio in tempo reale, un display a matrice di LED e un Arduino, Andrew ha creato il suo Dotklok che ha poi messo in open source per dare a tutti la possibilità di realizzarne uno e di aggiungere il proprio contributo alle animazioni già esistenti. Nel numero scorso abbiamo analizzato lo schema elettrico e la struttura del software, quindi adesso è finalmente arrivato il momento di metter mano a cacciavite, stagno e saldatore per realizzare questo interessante dispositivo.

LA STRUTTURA DI DOTKLOK

Questo "orologio" ha fra le sue peculiarità una forma particolare, con due pannelli in acrilico nero e bronzo che incorporano a sandwich i due stampati: quello con il microcontrollore e l'RTC nella parte posteriore e quello con la matrice di LED nella parte anteriore. Leggendo l'intervista all'autore, riportata in queste pagine, scoprirete che ogni elemento di questo dispositivo ha un suo "perché", così come lo hanno i 4 pulsanti frontali e il deviatore posteriore. Prima di passare al montaggio vero e proprio, vale la pena ripassare velocemente lo schema elettrico, dove il microcontrollore ATmega328 con clock a 16 MHz è collegato al display a matrice di LED attraverso i pin per il collegamento con il protocollo seriale su due fili HDR2, mentre l'interrogazione e la scrittura dei dati sul chip orologio RTC Dallas DS1307 è gestita tramite I2C sui pin SCL ed SDA presenti sul microcontrollore.

na la modernità della tecnologia Arduino con il gusto retrò dei primi dispositivi elettronici.

IL MONTAGGIO DELLO STAMPATO

Questo dispositivo è disponibile in kit da Andrew O'Malley e da Futura Elettronica, ma può essere realizzato anche partendo da un Arduino UNO a cui aggiungere il display a matrice e il DS1307 di Dallas Semiconductors assieme a pochi altri componenti su bread-board.

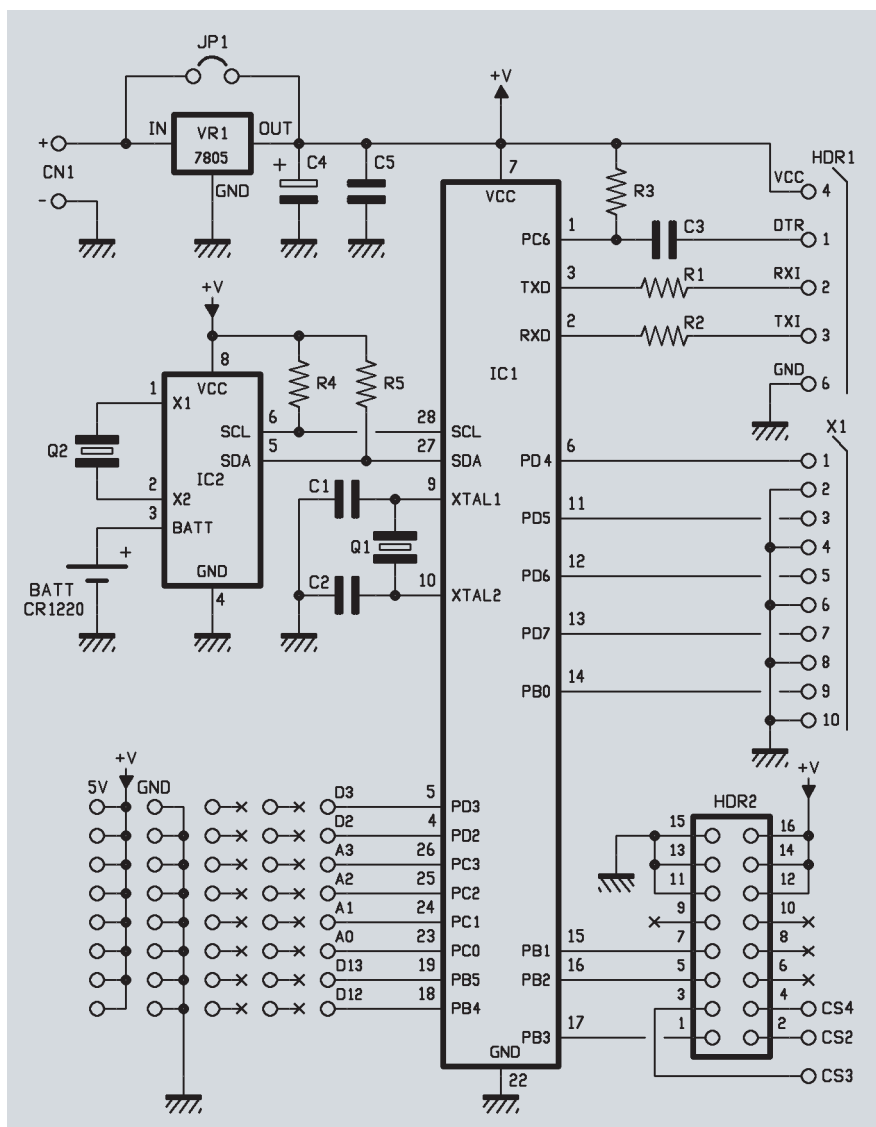
Il nostro kit include anche i due pannelli in acrilico con incisione laser delle varie scritte e diciture. A voi la scelta. In questo articolo vi guidiamo comunque nell'assemblaggio del dispositivo presupponendo l'acquisto del kit, mentre chi sceglie il fai da te completo potrà fare riferimento allo schema elettrico per individuare i vari collegamenti.

Lo stampato del kit è a doppia faccia, con fori metallizzati e ha un lato saldature e uno componenti, che sono tutti di tipo tradizionale. La prima cosa da fare riguarda il jumper JP1 che – come visibile dallo schema elettrico – crea un ponticello fra ingresso dell'alimentazione e piedino di uscita del regolatore di tensione.

Se pensate di usare un alimentatore non eroga 5V stabilizzati bensì una tensione superiore, allora dovete lasciare il ponticello aperto e montare il 7805.

Nel nostro kit è invece presente un alimentatore stabilizzato che eroga 5V con la corrente richiesta e quindi il ponticello va chiuso e il regolatore omesso (**Fig. 1**).

Il secondo passaggio riguarda il montaggio del pinstrip con i contatti piegati a 90 gradi, necessari per il collegamento di un modulo USB/Seriale con chip FTDI, come quello disponibile da Futura Elettronica con il codice FTDI5V e



Cinque pin sono dedicati al collegamento dei quattro pulsanti e del deviatore, mentre i tradizionali TX ed RX della seriale sono resi disponibili tramite il connettore HDR1. Su questo connettore può essere montato un modulo FTDI tramite il quale si può accedere al microcontrollore in modalità di programmazione e comunicazione "Arduino style". Molte porte del microcontrollore restano libere e infatti sul circuito stampato l'autore ha previsto un'area in stile breadboard dove questi pin sono resi disponibili assieme a una serie di pin 5V e GND. Sfruttando quest'area è possibile espandere il circuit-

to - come ha fatto ad esempio l'utente Solarmax di Instructables - aggiungendo anche funzionalità hardware, mentre il codice sorgente è commentato - come abbiamo visto il mese scorso - in modo adeguato ed è quindi facilmente modificabile. Potete trovare la modifica di Solarmax all'indirizzo <http://www.instructables.com/id/Adding-a-dimmer-to-your-DOTKLOK/>. Il display a matrice è disponibile sia in verde, sia in rosso, con luminosità equivalente. I pulsanti, infine, sono intenzionalmente grandi e un po' datati proprio per conferire all'intero dispositivo un aspetto che combi-

Fig. 1

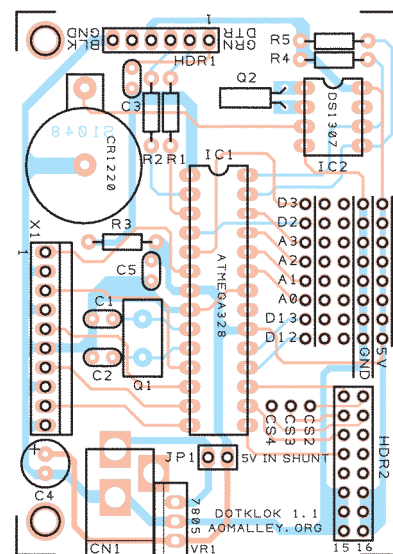
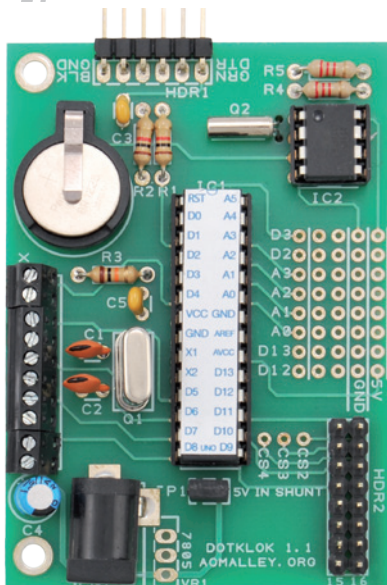


prodotto da Sparkfun. Il modulo vi permetterà di riprogrammare il microcontrollore una volta che lo avrete montato sul circuito. Se non avete intenzione di intervenire sul firmware, potete omettere il connettore, mentre la programmazione è anche possibile tramite una scheda Arduino UNO con le modalità già più volte documentate sulla nostra rivista e comunque reperibili sul sito di Arduino (<http://arduino.cc/en/Tutorial/ArduinoISP>). Dopo questi due passaggi, il montaggio dei componenti sullo stampato può procedere con le abituali modalità, facendo caso al verso di montaggio dei due zoccoli per gli integrati e alla diversità fra i due quarzi. Il quarzo a 16 MHz è di tipo a basso profilo, mentre quello per l'RTC, da 32,768 KHz è di tipo cilindrico. Dopo aver montato anche i componenti passivi, prima di inserire gli integrati negli zoccoli, vanno montati il porta batteria CR1220 e i due connettori X1 e HDR2.

Il primo è composto da una serie di 5 connettori a vite da due poli e questi vanno montati in modo tale che i fori in cui andranno inseriti i fili siano rivolti verso l'esterno. HDR2, invece, è realizzato affiancando due file di pinstrip singoli da 8 posizioni; la parte più lunga va verso il lato dei componenti (verso l'alto), badando di appoggiare bene il supporto di plastica alla basetta dal lato componenti.

Il connettore per l'alimentazione è di tipo standard, a tre contatti, ed è delle dimensioni adatte al

[piano di MONTAGGIO]



Elenco Componenti:

R1; R2: 1 kohm
R3: 10 kohm
R4; R5: 2,2 kohm
C1; C2: 22 pF ceramico
C3: 100 nF ceramico
C4: 10 μ F 63 VL elettrolitico
C5: 100 nF ceramico
IC1: ATMEGA328
IC2: DS1307
VR1: 7805
Q1: Quarzo 16 MHz
Q2: Quarzo 32.768 kHz

Varie:

- Morsetto 2 poli passo 2,54mm (5 pz.)
- Zoccolo 4+4
- Zoccolo 14+14
- Strip maschio 2 vie
- Strip maschio 6 vie 90°
- Strip maschio 8 vie (2 pz.)
- Jumper
- Porta batteria CR1220
- Batteria CR1220
- Plug alimentazione
- Display a matrice di LED
- Circuito stampato

collegamento con l'alimentatore fornito nel kit; se intendete usare un altro alimentatore, assicuratevi che abbia lo spinotto delle dimensioni corrette e con il positivo in centro.

IL MICROCONTROLLORE

Nel kit il microcontrollore è già programmato e ha il bootloader Arduino assieme al firmware di Dotklok; se state realizzando autonomamente il circuito partendo da un Arduino, potete anche tenere la board come parte del progetto e utilizzare una basetta millefori o una breadboard per i collegamenti: il firmware potrà essere caricato sull'Arduino con le normali procedure partendo

dai sorgenti disponibili sul sito dell'autore, come indicato il mese scorso. Se invece avete un Arduino, ma volete conservarlo per altre applicazioni, potete procurarvi un nuovo ATmega328, programmarlo con il bootloader attraverso la metodica ArduinoISP e infine sostituirlo sulla vostra scheda Arduino per caricare il firmware. Al termine potrete rimettere al suo posto il microcontrollore originale per riportare il vostro Arduino in piena efficienza, rimanendo con l'ATmega correttamente caricato con Firmware e bootloader. Va da sé che potete anche programmare il bootloader sull'ATMega328 nuovo, caricare il firmware sulla scheda Arduino

Come usare Dotklock

Questo orologio dispone di un numero variabile di rappresentazioni tradizionali, animate e astratte di data e ora. A seconda del firmware installato, le animazioni e alcune funzionalità potrebbero differire da quanto descritto qui.

Sul frontale si trovano quattro pulsanti: a sinistra dall'alto B1 e B2, a destra dall'alto B3 e B4.

Sul pannello posteriore si trova un deviatore per la modalità Random: quando il deviatore è chiuso a mezzanotte Dotklok sceglie una nuova animazione a caso fra quelle disponibili.

Manualmente è possibile cambiare l'animazione in ogni momento con i pulsanti B3 e B4.

Per impostare l'ora dovete essere nella prima animazione – quella che appare all'accensione – e tenendo premuto B1 potete usare B3 e B4 per cambiare l'ora, poi tenendo premuto B2 potete usare B3 e B4 per cambiare i minuti. Il passaggio da 12 a 24 ore si ottiene premendo contemporaneamente B1 e B2.

Alcune animazioni non dispongono della modalità 12 ore mentre altre non dispongono di quella a 24.

Per modificare la data, dovete essere nella seconda animazione che – appunto – mostra su una riga la data e sotto ora, minuti e secondi (**Fig A**).

Da questa schermata si può regolare solo la data con la seguente procedura: premendo B1, con B3 e B4 si regola il mese, poi premendo B2, sempre con B3 e B4 si regola

il giorno. Per l'anno si devono tener premuti contemporaneamente B1 e B2, agendo su B3 e B4 per aumentare o diminuire l'anno.

La luminosità è regolabile quando vi trovate nell'animazione con i secondi trascorsi e quelli che mancano a fine giornata (**Fig. B**)

Ricordando che B3 e B4 cambiano l'animazione, per cambiare la luminosità è necessario tenere premuto B1 e poi agire su B3 e B4.

Fig. A



Fig. B



e prendere per Dotklok il micro dalla scheda e sostituirlo con quello su cui avete messo il solo bootloader (saltate un passaggio). Per cambiare il firmware, in mancanza dell'adattatore USB/seriale, potete ancora usare un Arduino collegandolo opportunamente, oppure rimuovere il chip da Dotklok e usare la scheda Arduino come fosse il programmatore. In generale, vi consigliamo di fare la sperimentazione sulle funzionalità, le animazioni e i diversi firmware disponibili prima di montare la scheda e il display fra i due pannelli a meno che non

disponiate dell'adattatore seriale/usb che permette il collegamento diretto al PC con l'IDE Arduino.

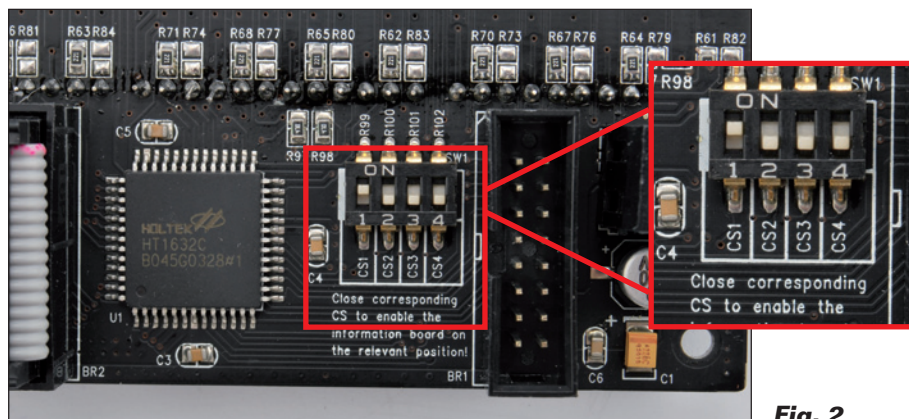


Fig. 2

Fate riferimento all'articolo pubblicato sul numero scorso, da pagina 58, per la descrizione dettagliata di tutta la struttura del firmware di Dotklok.

IL DISPLAY A MATRICE

Questo componente è prodotto da Sure Electronics e si basa su un processore dedicato che dialoga in SPI con il microcontrollore. Per praticità è stato mantenuto il connettore a 16 poli anche se sei di questi sono riservati a V+ e GND e solo due sono destinati ai dati e quattro alla selezione della scheda. Il cavo flat, già attestato con i due connettori femmina, fa parte della confezione e quindi è più conveniente aggiungere HDR2 che rifare un connettore più piccolo. Per utilizzare la matrice, va impostato su ON il micro deviatore che corrisponde a CS1 (**Fig. 2**). Questo meccanismo permette di collegare fino a quattro schede in cascata, inviando i dati in modo selettivo semplicemente selezionando la destinazione tramite l'attivazione della linea del relativo pin CS.

Il connettore a sinistra è quello d'ingresso – da collegare al PCB del Dotklok o ad Arduino nel caso di versione "fai da te", mentre quello accanto al dip switch è il connettore d'uscita verso la scheda successiva.

PRIMO COLLAUDO

Quando il chip DS1307 non è alimentato e la batteria al litio non è presente nel porta batteria, viene perso l'orario, che va regolato in qualche modo. Il firmware prevede il test del funzionamento dell'RTC e della presenza di un orario valido, in mancanza del quale viene presa la data e l'ora di compilazione del firmware stesso. Se state montando la versione in kit, l'ora verrà impostata sull'ora e data di compilazione e caricamento sul chip fornito, mentre se avete appena programmato il firmware, l'ora sarà approssimativamente giusta. Inserite la batteria al litio nel porta batteria, collegate il flat cable da HDR2 alla matrice LED (nel connettore a sinistra, con i led sul lato opposto del PCB e il controller in basso) come in Fig. 3.

Fate attenzione a rispettare l'abbinamento da Pin1 del PCB a Pin1 del display.

Ora potete girare il display in modo che sia visibile e dare alimentazione al circuito; se è tutto a posto, sullo schermo appare prima la versione del Firmware e poi la prima modalità di visualizzazione definita nel firmware con l'apposita costante (Fig. 4).

Se l'orologio funziona mostrando l'ora (probabilmente sbagliata per i motivi prima illustrati), potete passare alla fase successiva di preparazione dei quattro pulsanti e del deviatore. Se non appare nulla, scollegate l'alimentazione, controllate se il CS della scheda è 1 e se il verso del connettore è corretto, quindi controllate la presenza di JP1 sullo stampato e riprovate. Se ancora non avete alcun risultato, provate a collegare il dispositivo tramite modulo USB/seriale e lanciate l'IDE di Arduino: tramite console dovreste ricevere dei messaggi. Come altra soluzione



Fig. 3

potete riprogrammare il micro con il firmware (consigliamo la versione 1.14 se volete molte animazioni) e avere già con la conferma della programmazione una prima testimonianza della funzionalità del circuito. Riprogrammato il chip o verificata l'esistenza di messaggi sulla seriale, rifate i collegamenti facendo attenzione al connettore flat, magari inserito in modo errato.

I PULSANTI

Con quattro pulsanti e un deviatore potete controllare le varie funzioni di Dotklok.

Dato che questi andranno fissati a pannello, la lunghezza dei fili da saldare va gestita per successive approssimazioni. Inizialmente potete usare una trentina di centimetri di filo bipolare per ciascun pulsante, poi potrete accorciarli ad una lunghezza adeguata in base anche a come deciderete di farli passare attorno alle torrette esagonali di supporto. I quattro tasti vanno montati con un ordine preciso nei collegamenti, come da disegno del pannello (Fig. 5).

Collegate al connettore sul PCB, partendo dal lato della batteria al litio, prima il deviatore B5, poi B1, B2, B3 e infine B4, con quest'ultimo che è collegato alla coppia di contatti dal lato del connettore di alimentazione.

Prima di collegare i pulsanti al connettore, ricordate di fissarli

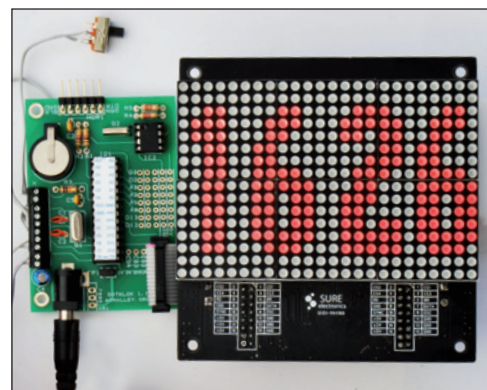


Fig. 4

al pannello, altrimenti avrete problemi con i fili.

IL MONTAGGIO MECCANICO

Con la parte elettronica collaudata, possiamo procedere alla sistemazione degli stampati fra i due pannelli.

Il pannello posteriore ospita due colonnine corte (C), quattro colonnine medie (D) e quattro colonnine lunghe (A), come indicato nel disegno (Fig. 6).

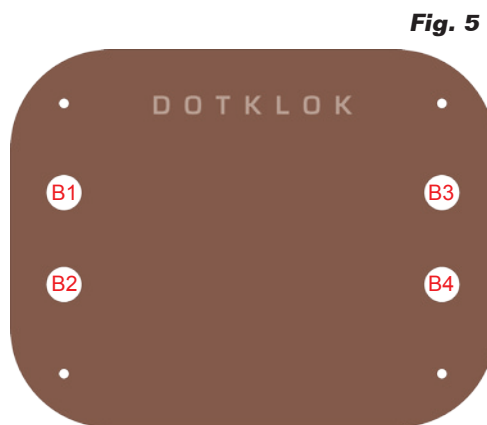


Fig. 5

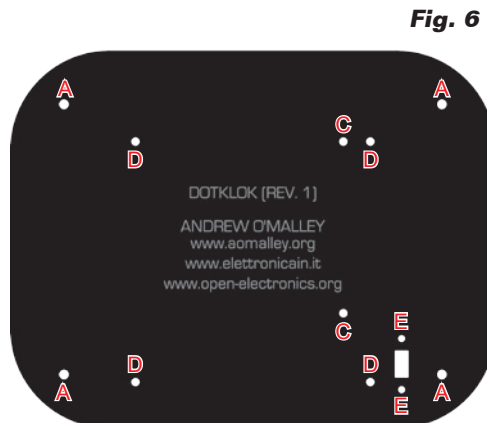


Fig. 6

Due chiacchiere con **Andrew O'Malley**

Per realizzare questo articolo, abbiamo preso contatti con il suo autore, Andrew O'Malley, con il quale abbiamo avuto una fitta corrispondenza. Abbiamo pensato di chiedergli anche un po' di retroscena del progetto in una mini intervista che riportiamo qui di seguito.

D: Cosa ti ha spinto a progettare il Dotklok?

R: Sono sempre stato affascinato dagli orologi, ma non mi sono mai cimentato nella realizzazione di un semplice orologio elettronico in quanto non volevo riprodurre qualcosa che era già in circolazione. Ero al corrente degli orologi Pong e Word e intanto stavo realizzando delle strutture astratte basate sulla luce, al cui interno c'era il medesimo display usato in Dotklok. Da lì mi è sorta l'idea di utilizzare quella soluzione come piattaforma su cui sviluppare sia quegli orologi di cui ero al corrente, sia le mie animazioni originali, come quelle che includono gli Space Invaders, sia Pacman, sia le animazioni più astratte che avevo in mente. È stato molto interessante e avvincente lavorare con animazioni che mostrano il passare del tempo in modi più generali e astratti, in contrasto con le tradizionali semplici indicazioni numeriche. Come ho iniziato a programmare queste varie animazioni mi è anche venuto in mente che l'orologio dovesse cambiare da solo le animazioni ed è così che è nata l'idea del cambio automatico e casuale della modalità di visualizzazione a mezzanotte:



in questo modo ci si svegli ogni mattina con un'animazione diversa. Questa combinazione di animazioni che mostrano l'ora scelta in modo casuale mi ha colpito come qualcosa di unico rispetto a tutto il resto, e per questo aveva qualcosa di peculiare che valeva la pena condividere con gli altri.

D: Quante versioni hai realizzato prima che venisse finalizzata quella corrente?

R: Ci sono state alcune versioni beta dell'orologio in cui ho sperimentato con il posizionamento dei pulsanti e la forma del contenitore, ma tutto l'hardware è stato progettato abbastanza rapidamente. È stato progettato durante l'estate 2011 al NYC Resistor hackerspace di New York City. È stata la programmazione delle varie animazioni che ha portato via la maggior parte del tempo, restando anche l'attività che continuo a seguire per realizzare nuove animazioni, dato che ancora spesso me ne vengono in mente.

D: Per quale motivo lo hai reso Open Source?

R: Ho reso Dotklok Open Source perché ho tratto tanti benefici da altri progetti Open Source e ho sentito che era un progetto approfondito e completo che potevo rilasciare, inclusi il software, il PCB e il contenitore. Ho anche immaginato Dotklok come una piattaforma per il tempo che altri avrebbero potuto modificare: per aggiungere allarmi, timer o anche semplici videogiochi; quest'ultimo è anche il motivo per cui i pulsanti sono sui lati, così che possano essere riassegnati per una piccola piattaforma per giochi portatile.

D: Quali sono le aspettative della community degli utenti e ti saresti aspettato più o meno feedback dagli utenti?

R: La parte di Dotklok relativa alla community si è rivelata difficile da gestire. Questo deriva dal fatto che non sono stato molto fortunato con il software di gestione del forum, appesantito dalla quantità di lavoro da fare per mantenere il forum libero dallo spam, perdendo quindi risorse sul fronte dello sviluppo e della manutenzione dei contenuti veri e propri.

Dotklok ha anche una pagina di Facebook (<http://www.facebook.com/dotklok>) che ho inizialmente usato per dare supporto agli utenti e per condividere contenuti, ma Facebook ha rimosso la funzionalità delle Discussioni dalle pagine, funzione che permetteva di gestire e organizzare i contenuti ed ora è tutto solo in ordine cronologico. La gestione di un forum o di un portale per gli utenti di Dotklok è qualcosa per cui potrei anche cercare dell'aiuto. Per quanto riguarda il feedback dagli utenti, in effetti sono soddisfatto della risposta avuta dalla community. Ciascuno sembra contento del proprio Dotklok e qualcuno ha aggiunto delle caratteristiche come una foto resistenza per gestire automaticamente l'illuminazione, oppure si è cimentato in un contenitore diverso dall'originale. Mi fa piacere quando qualcuno manda immagini del proprio orologio completato. Sono anche stato fortunato nell'aver alcune recensioni pubblicate su Youtube, realizzate da qualche utente soddisfatto da Dotklok. Non ero certo di cosa aspettarmi in termini di quantità, quindi sono soddisfatto di quanto è successo finora.

D: Qualche aneddoto divertente legato a Dotklok?

R: Mi vengono in mente due storie... La prima riguarda il fatto che Dotklok ha alcune animazioni astratte che rendono difficile la lettura dell'ora esatta; a casa mia ho altri dispositivi che indicano l'ora in modi inconsueti. Un mio amico era seduto nel mio salotto quando mi ha detto: "Ci sono tre orologi qui e ancora non ho idea di che ora sia!"

La seconda mi è stata riferita da un amico che realizza dei Dotklok progettando autonomamente i contenitori; stava viaggiando nel periodo delle feste con un Dotklok personalizzato quando in un controllo di sicurezza aeroportuale gli agenti hanno individuato il circuito nel suo bagaglio a mano durante l'ispezione ai raggi X. Data la sua peculiarità il bagaglio è stato immediatamente messo da parte e l'amico ha dovuto spiegare che quell'oggetto misterioso era semplicemente un orologio fatto in casa.

Si comincia con le due colonnine da inserire nei fori C: su queste due colonnine va fissato lo stampato con il microcontrollore, badando di tenere il connettore a

10 vie per i pulsanti verso l'esterno (verso il foro rettangolare del deviatore). I componenti devono essere rivolti verso l'interno di Dotklok e quindi non vanno

rivolti verso il pannello. Poi dovete fissare le quattro colonnine medie nei fori (D) e procedere al cablaggio dei fili di deviatore e pulsanti sul connetto-

Guarda le animazioni del nostro DotKlok



Vuoi vedere come funziona DOTKLOK? In questo video vengono mostrate tutte le animazioni predefinite selezionabili mediante i pulsanti. Scansiona il QR Code con il tuo smartphone!



re X1. Se non lo fate ora, poi sarà ben difficile farlo con il display che impedisce l'accesso alle viti di serraggio dei fili.

Anche il flat cable verso il display va collegato in questa fase.

Posizionate il display senza far ruotare il cavo flat, ma solo ripiegandolo. In questo modo la matrice di LED sarà allineata al centro del pannello anche se la parte inferiore è vuota. Se sbagliate di 180 gradi avrete il cavo flat che fa una torsione e la matrice risulta decentrata visibilmente.

I fori filettati delle colonnine distanziali potrebbero non corrispondere immediatamente con i quattro fori dello stampato del display, ma tenendo le viti di fissaggio un po' allentate potrete avere il gioco necessario a fare gli allineamenti senza sforzare nulla. Ricordate che il pannello posteriore è opaco e risente meno di eventuali graffi, mentre quello frontale è trasparente e mette in evidenza ogni graffio: se potete maneggiatelo solo quando necessario e tenetelo appoggiato su una superficie morbida come un panno di cotone o un pezzo

di pluriball da imballaggio.

Quando la struttura del PCB, del display e del deviatore è definitiva, ovvero con le viti serrate e i fili collegati, potete procedere al fissaggio delle ultime quattro colonnine distanziali. La manovra va fatta con attenzione in quanto avete anche il pannello frontale vincolato dai fili dei pulsanti e ad ogni movimento vi "tirate dietro" anche il pannello con il rischio di rigarlo. Nella figura 7, l'ultimo passaggio prima del fissaggio del pannello frontale (Fig. 7).

Quando il pannello posteriore è pronto con i distanziali, fissateci quello frontale, con delicatezza e senza stringere i bulloni più del necessario per tenere fermo il pannello: l'acrilico si crepa se serrate anche solo un bullone con forza eccessiva.

In questa fase non ci devono essere intralci al passaggio dei fili dei pulsanti e questi non devono creare tensioni o passare di fronte alla matrice di LED. Meglio dei fili abbondanti che tesi, anche per consentire di riaprire Dotklok per eventuali modifiche e interventi. Con tutto montato, fate un giro di controllo sui bulloni accessibili sul pannello posteriore: devono essere stretti quanto basta per essere saldi, ma non vanno sforzati. Potrebbe essere necessario tenere ferma la colonnina distanziale con una pinza mentre serrate i bulloni, quindi tenetela a portata di mano.

MESSA IN OPERA

Ora non vi resta che collegare nuovamente l'alimentazione allo spinotto che ora si trova nella parte bassa del contenitore, dietro alla basetta del display, sul

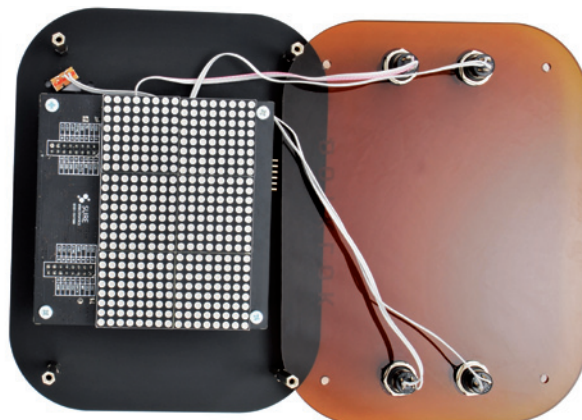
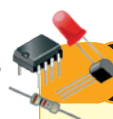


Fig. 7

lato sinistro.

Prima di posizionare il vostro Dotklok sulla scrivania o su un ripiano della libreria – o dove vorrete – vi ricordiamo che va regolato l'orologio con la procedura che descriviamo nel box dedicato all'uso di Dotklok.

Vi invitiamo anche a prender parte alla community dei possessori di Dotklok visitando la pagina di Andrew e iscrivendovi al forum. Anche Elettronica In è aperta a proposte di modifiche e nuove animazioni, quindi se ne realizzerete una, saremo lieti di dare spazio alla vostra creazione sul nostro blog (blog.elettronica.in).



per il MATERIALE

Questo originale orologio è disponibile in scatola di montaggio, cod. FT1048-RK (display rosso) o FT1048-GK (display verde) al prezzo di 78,00 Euro IVA compresa. Il kit comprende tutti i componenti, le minuterie, il microcontrollore già programmato, il CS forato e serigrafato, il display a matrice, l'alimentatore da rete e i due pannelli in plexiglass forati e incisi.

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



CONTROLLIAMO UNA 3DRAG A DISTANZA TRAMITE RASPBERRY PI

Una soluzione Embedded Linux per stampare in remoto e controllare la vostra 3Drag senza tenere impegnato un personal computer.

di FRANCESCO DE FABIANI

Stufi di dover lasciare il personal computer acceso per ore ed ore in garage, senza contare l'inutile spreco di energia, per stampare con la vostra 3D printer?

La soluzione è utilizzare come sostituto del notebook o del pc una scheda RaspberryPi. Dapprima abbiamo ipotizzato d'installare la distribuzione del

software Repetier-Host per linux direttamente su RaspberryPi, poi ci siamo imbattuti in una versione specifica per la piccola scheda marchiat col lampone. Il suo nome è Repetier-Server e fa al caso nostro perché creando una pagina web all'indirizzo IP di Raspberry è possibile sia controllare a distanza la stampante, sia stampare dei modelli

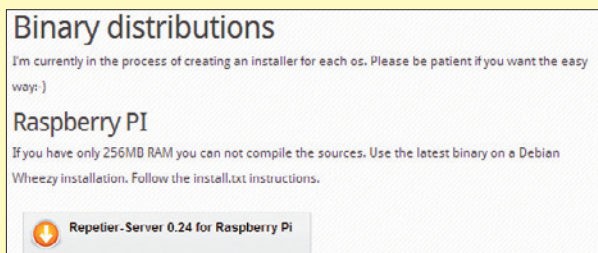
semplicemente caricando il G-Code sul server, senza la necessità di tenere il pc acceso. Per installare Repetier-Server è necessario recarsi all'indirizzo <http://www.repetier.com/repetier-server-download/> scaricare il file 'Repetier-Server 0.24 for Raspberry Pi' e seguire le istruzioni che riportiamo nel box dedicato. Al termine dell'installazio-

ne la pagina web sarà reperibile all'indirizzo IP di Raspberry alla porta 8080.

Dopo aver reso il tutto controllabile a distanza abbiamo riscontrato la necessità di visualizzare la stampa in corso per verificare che tutto funzioni, così abbiamo collegato una webcam alla seconda porta USB della scheda RaspberryPi.

Installazione di Repetier-Server su Raspberry Pi

1. Recarsi all'indirizzo <http://www.repetier.com/repetier-server-download/>
2. Scaricare il file 'Repetier-Server 0.24 for Raspberry Pi'.



3. Spostarlo all'interno della directory di Raspberry `/home/pi` avvalendosi di WinSCP
4. Decomprimere il file con WinSCP:
 - 1) Cliccare con il tasto destro del mouse sul file appena spostato
 - 2) Spostarsi su *Comandi personalizzati*
 - 3) Selezionare *UnTar/GZip...*
 - 4) Confermare cliccando su *OK*
5. A questo punto aprire la cartella appena creata *Repetier-Server-Pi*
6. Nel file *install.txt* troverai le istruzioni per l'installazione di

Repetier-Server da riga di comando, è anche possibile effettuare i passaggi con WinSCP:

- 1) Modificare uno degli esempi di configurazione contenuti in *configs*:
 - *Version=1.0*
 - *Active=false*
 - *Name=3Drag*
 - *slugName=3Drag*
 - *device=/dev/serial/by-id/<troverete il nome del vostro dispositivo collegando la stampante e andando a vedere all'interno del percorso appena citato>*
 - *baudrate=250000*
 - *pingPong=false*
 - *protocol=0*
 - *okAfterResend=true*
 - Completare le altre opzioni con le proprie impostazioni e preferenze
 - *tempUpdateEvery=1*
 - Salvare il nuovo file con nome *3Drag.conf*
- 2) Copiare il file eseguibile *RepetierServer* nella cartella `/usr/bin`
- 3) Copiare il file *repetier-server.conf* in `/etc`
- 4) Creare la cartella *Repetier-Server* in `/var/lib`
- 5) Creare le cartelle *www*, *configs*, *storage* e *languages* all'interno della directory appena creata `/var/lib/Repetier-Server`

TUTTO SOTT'OCCHIO CON UNA WEBCAM

Dopo aver realizzato tutto il sistema, appare evidente la necessità di poter verificare che tutto funzioni. Ancora una volta RaspberryPi ci viene incontro divenendo con poche modifiche un webcam server che permette di visualizzare l'immagine della stampa in tempo reale, direttamente dal pannello di controllo della pagina web di Repetier (**Fig. 1**).

Per prima cosa bisogna installare un'applicazione che acquisisca l'immagine dalla webcam collegata alla porta USB e la carichi su Repetier. Abbiamo scelto Motion perché, essendo nato come software per la videosorveglianza, è semplice da usare e possiede una vasta gamma di impostazioni personalizzabili. Per installare Motion il primo passo è come sempre quello di assicurarsi di avere installati sul proprio RaspberryPi le versioni più recenti dei software e

dei driver digitando nel terminale di RaspberryPi i comandi

```
sudo apt-get update
e
sudo apt-get upgrade
```

A questo punto è possibile installare Motion con il comando

```
apt-get install motion
```

Perché quest'ultimo svolga correttamente il suo compito di webcam server è necessario modificare alcune impostazioni. Digitando da terminale

```
sudo nano /etc/motion/motion.conf
```

sarà possibile configurarlo: trovare la voce

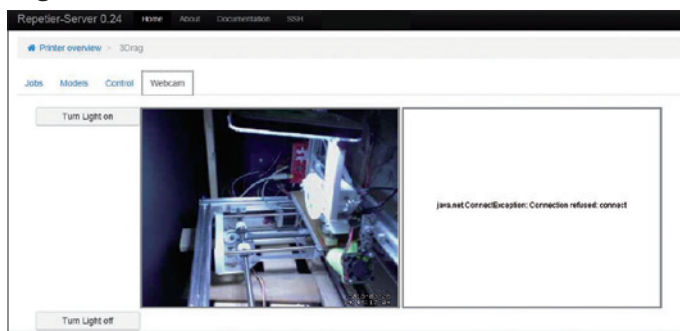
Daemon = OFF

e sostituire **OFF** con **ON**, questo consentirà a Motion di essere eseguito in modalità "demone", cioè in background; in seguito trovare la voce

webcam_localhost = ON

e modificare **ON** con **OFF**, ciò permetterà anche agli altri computer connessi alla rete domestica di visuali-

Fig. 1



- 6) Copiare il nostro file *3Drag.conf* all'interno di */var/lib/Repetier-Server/configs*
- 7) Copiare il contenuto di *www* e *languages* all'interno delle rispettive cartelle in */var/lib/Repetier-Server*
- 8) Per fare in modo che il Server si avvii in automatico all'accensione di Raspberry:
 - Copiare il file *Repetier-Server.init* all'interno di */etc/init.d/Repetier-Server*
 - Dargli i permessi di esecuzione e lettura attraverso la digitazione del comando `sudo chmod 755 /etc/init.d/Repetier-Server` dal terminale
 - Attivare lo start script con il comando `sudo update-rc.d Repetier-Server defaults`
 - Per attivare\disattivare\riavviare il Server manualmente digitare i seguenti comandi
`sudo /etc/init.d/Repetier-Server start`
`sudo /etc/init.d/Repetier-Server stop`
`sudo /etc/init.d/Repetier-Server restart`

7. L'installazione è terminata, ora è possibile accedere al Server con il proprio browser preferito all'indirizzo
 <IP del Raspberry>:8080

È consigliabile impostare la scheda con indirizzo IP statico in modo che non cambi ad ogni reboot.

lizzare l'immagine dalla pagina web creata da Motion. Sempre con nano sarà possibile modificare molti altri parametri, come la frequenza di frame al secondo o la dimensione in pixel dell'immagine, ma che è consigliabile lasciare all'impostazione di default in modo da non appesantire troppo la CPU non molto potente di RaspberryPi. Per assicurarsi che all'avvio il motion service sia lanciato effettivamente come daemon si deve ancora modificare una configurazione in questo modo:

```
sudo nano /etc/default/motion
```

sostituire in
 start_motion_daemon= **no** con **yes**.

Finalmente possiamo lanciare il programma con l'istruzione

```
sudo service motion start
```

e verificare che funzioni andando all'indirizzo IP di RaspberryPi, alla porta 8081. Per visualizzare l'immagine direttamente dalla pagina di controllo di Repetier bisogna apportare ancora alcune modifiche. Per prima cosa dovete copiare l'appli-



cazione Java Cambozola.jar, scaricabile dall'indirizzo <http://www.charliemouse.com/code/cambozola/>, in */var/lib/Repetier-Server/www*, in seguito aprire il file *printer.php* copiato al momento dell'installazione del server Repetier nella directory appena citata. Adesso basterà aggiungere all'interno del "Control Panel" all'interno del <body> del file php queste due righe di codice:

```
<applet code=com.charliemouse.cambozola.Viewer
      archive=cambozola.jar width="320"
height="240" style="border-width:1; border-
color:gray; border-style:solid;"> <param name=url
value="http://localhost:8081"> </applet>
```

e il gioco è fatto.

Così facendo apparirà nella pagina dei comandi manuali della stampante un riquadro con l'immagine aggiornata ogni secondo della webcam.

CONTROLLARE VENTOLA E ILLUMINAZIONE DEL PIATTO DI STAMPA A DISTANZA

Proseguendo nella personalizzazione della postazione di stampa, sfruttiamo il nostro controllo remoto tramite pagina web per gestire la ventola di raffreddamento e l'illuminazione dell'area di stampa.

Per poter controllare ventola e illuminazione a distanza saranno sufficienti alcune semplici modifiche.

Iniziamo con il software residente sulla scheda di controllo. Per chi non l'avesse mai fatto, il firmware Marlin è reperibile alla sezione Software del sito ufficiale della stampante 3Drag, vale a dire

<http://3dprint.elettronica.in.it/>. Una

volta scaricato, per modificarlo, compilarlo e caricarlo sulla scheda Sanguinololu avremo bisogno della versione 0023 dell'IDE di Arduino, scaricabile dall'indirizzo seguente <http://www.arduino.cc/en/Main/Software>.

Cominciamo col modificare il file *pins.h*, spostandoci nella sezione relativa alla scheda Sanguinololu cerchiamo la voce

```
#define FAN_PIN
```

e sostituiamo -1 (che disabilita questa funzione) con 4 (il pin n°4 del microprocessore Atmel è l'unico tra quelli inutilizzati a poter generare un segnale PWM). In seguito aggiungiamo la dichiarazione del pin che

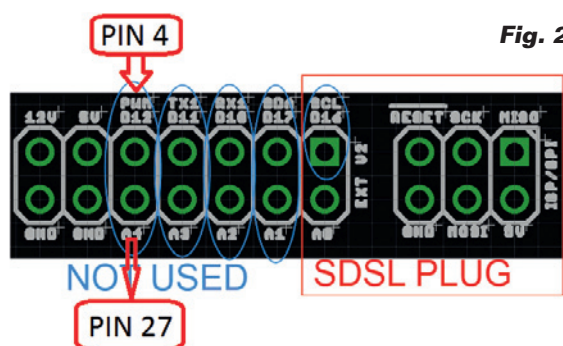


Fig. 2

farà accendere e spegnere la luce in questo modo :

```
#define LIGHT_PIN 27
```

(il pin n°27 corrisponde al 4° pin di INPUT analogico, ma se configurato correttamente può benissimo fungere da OUTPUT digitale). Ora possiamo dedicarci al file Marlin.pde. Qui aggiungendo nel 'void setup()' l'istruzione:

```
pinMode (27, OUTPUT);
```

abbiamo configurato il pin 27 come OUTPUT digitale. Giunti a questo punto dobbiamo fare in modo che alla ricezione di un determinato comando la luce si accenda o si spenga. Per fare questo si devono aggiungere nello "switch" relativo ai comandi "M" collocato nel 'void loop()' due "case" con numero pari a quello del comando (non ancora utilizzato dal firmware) che vogliamo associare ad accensione o spegnimento della luce, come ad esempio:

```
case 800:
    WRITE (LIGHT_PIN, HIGH);
    break;
case 801:
    WRITE (LIGHT_PIN, LOW);
    break;
```

Adesso si può salvare il nuovo firmware. Per caricarlo su Sanguinololu è necessario scaricare i sanguino file da questo indirizzo <https://github.com/jmgialcone/sanguino1284p>, estrarli e poi copiarli all'interno della cartella hardware di Arduino. Adesso potete riaprire l'IDE 0023 e selezionare dal menù a tendina Tools>Board>Sanguino, disconnettere l'alimentazione della stampante, individuate il pulsante di reset sulla scheda Sanguinololu e tenetevi pronti a premerlo appena pochi secondi dopo aver dato il via all'upload dall'IDE. Per quanto riguarda l'hardware, basta pilotare con i segnali prodotti dai pin della scheda due Mosfet collegati a ventola ed illuminazione (Fig. 2).

N.B. Per un errore di fabbricazione il pin 4 del mi-

croprocessore Atmega644p, utilizzato per il controllo della ventola, non corrisponde con l'omonimo pin della scheda Sanguinololu 1.3, ma bensì con il 12.

Infine, per poter accendere e spegnere la luce direttamente dal pannello di controllo è necessario aggiungere due pulsanti sempre nel file printer.php ricopiando queste due righe di codice all'interno del già citato Control Panel :

```
<button class="btn span2 inputonline notprinting"
onclick="sendCmd('M800')"><?php _("Turn Light on")?></button>
<button class="btn span2 inputonline notprinting"
onclick="sendCmd('M801')"><?php _("Turn Light off")?></button>
```

AUTOMATIZZARE L'ACCENSIONE

Sfruttando la flessibilità del programma Slic3r, possiamo fare in modo che la ventola sia opportunamente gestita tramite comandi impartiti dal Gcode. Tramite le opzioni che indichiamo qui di seguito, permetteremo a Slic3r di inserire nei punti opportuni i comandi per l'attivazione e lo spegnimento della ventola. Ecco dove intervenire.

Aprirete la console di Slic3r, nella pagina di configurazione dei parametri di stampa, spostatevi dapprima nel menu *Filament Settings* nella sezione *Cooling*, qui abilitate il raffreddamento automatico e salvate cliccando sull'icona a forma di disco accanto al nome del profilo; in seguito andate nel menu *Printer Settings* nella sezione *Custom G-Code* e quindi aggiungete i comandi per l'accensione e lo spegnimento dell'illuminazione del piatto di stampa all'elenco di comandi che vengono inviati prima e dopo la stampa (M800 per accensione; M801 per spegnimento), quindi salvate.

STAMPIAMO IL NOSTRO PRIMO OGGETTO DA REPETIER-SERVER

Dopo aver aperto la pagina web generata dal nostro Repetier-Server, nella **Home** cliccate sul pulsante *Activate* per connettere la stampante a RaspberryPi. Dopo che lo stato sarà mutato da *Offline* a *Online* cliccate su *Use*. Ora è possibile spostarsi alla sezione *Control* ed effettuare la calibrazione degli assi riportandoli alla posizione di home, cliccando sull'icona a forma di casa. Adesso spostatevi nel tab *Job* e caricate il file G-Code contenente le istruzioni da inviare alla stampante per stampare l'oggetto. Al termine dell'upload, cliccando su *Start* e poi su *yes* inizierà la stampa dell'oggetto una volta che l'estrusore avrà raggiunto la temperatura predefinita. Sempre dal pannello *Job* si può visualizzare lo stato di avanzamento della stampa. Nel pannello di controllo è invece possibile tenere sotto controllo la stampa attraverso l'immagine aggiornata ogni secondo della webcam. ■

GONFIATUTTO PROGRAMMABILE

di GABRIELE DAGHETTA



Quante volte, prima di partire per le vacanze, da provetti automobilisti ci ricordiamo di gonfiare i pneumatici dell'auto? E quante, nel farlo, non rammentiamo qual è la giusta pressione di gonfiaggio? E quante volte abbiamo cercato di gonfiare dei palloncini per una festa, facendone scoppiare la metà perché abbiamo esagerato con l'aria? Sicuramente situazioni del genere

sono capitate alla maggior parte di voi e in certe occasioni vi siete resi conto di quanto sarebbe stato comodo avere un apparecchio in grado di regolare la pressione dell'aria in modo automatico, sollevandovi dall'onere di dover stare attenti ad ogni tacca del manometro e pronti a mollare il dito dal pulsante della pistola ad aria. Ebbene, questo apparecchio ve

lo proponiamo qui, tra le pagine di Elettronica In; si tratta di un regolatore elettronico della pressione dell'aria che può funzionare in tre modalità: in quella manuale indica la pressione raggiunta e può essere usato o come una comune pistola di gonfiaggio, i cui controlli (gonfia/sgonfia) sono costituiti dai tasti cursore freccia su e freccia giù o come sistema

semiautomatico che permette di impostare la pressione di gonfiaggio desiderata, in quella automatica permette di scegliere tra 8 programmi di gonfiaggio definiti dall'utente (identificati con il nome del veicolo od oggetto da gonfiare) e porta automaticamente l'elemento da gonfiare alla pressione predefinita (sgonfiandolo se è più gonfio di quel che serve o gonfiandolo se ha una pressione più



Collegato in serie al tubo di un compressore d'aria, permette di gonfiare pneumatici, palloni, materassini e gonfiabili, in modo manuale o automatico alla pressione definita mediante un apposito menu.

bassa di quella imposta-
ta) mentre nella moda-
lità programmazione
possiamo definire i
parametri di gonfiaggio
e i nomi degli 8 pro-
grammi. Un sistema del
genere è simile a quello
che si trova in alcune
stazioni di servizio
all'estero (per esempio
in Germania nei distri-
butori ARAL) dove per
gonfiare i pneumatici
basta impostare la pres-
sione voluta, collegare
la pistola e premere un

tasto: la macchina fa
tutto automaticamente
e segnala su un display,
oltre che con un avviso
acustico, il raggiungi-
mento della pressione
ottimale e quindi la fine
del ciclo.

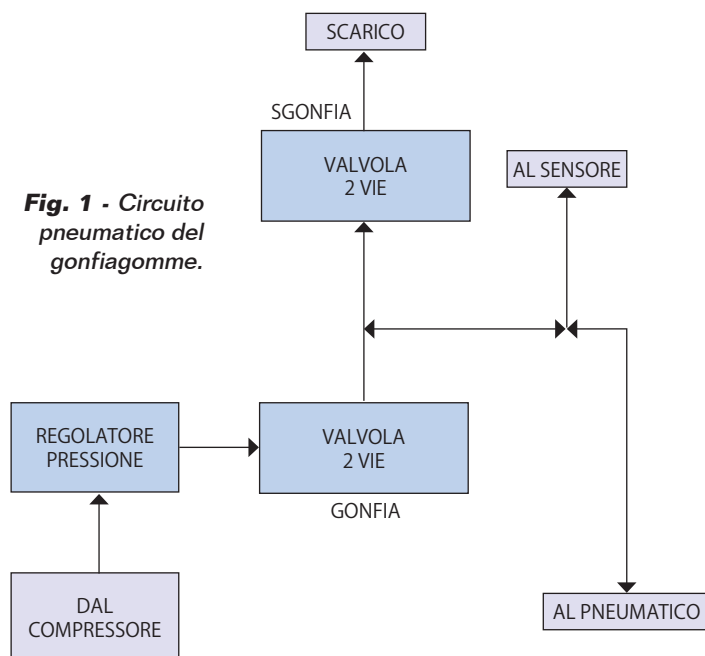
IL NOSTRO SISTEMA
Vediamo meglio di
cosa si tratta: il nostro
apparato è sostanzial-
mente un controllore
della pressione nel
condotto che porta
l'aria dall'uscita del

compressore al rac-
cordo per le valvole,
capace di intervenire su
due elettrovalvole, una
delle quali fa passare
l'aria, mentre l'altra
sfia l'aria contenuta
nel pneumatico. Pertan-
to dev'essere collegato
ad un compressore
d'aria, con l'ingresso
all'uscita del manicotto
di quest'ultimo; per
agevolare il collegamen-
to, il nostro apparato è
dotato di bocchettoni
standard ad innesto per
compressori.
Il nostro gonfiapneu-
matici non richiede la
pistola, ma un semplice
tubo provvisto di attac-
co con aprivalvole per
le gomme dei veicoli o
spillo per i palloni; la
funzione della pisto-
la, ovvero la leva per

gonfiare o il pulsanti-
no per sgonfiare che
troviamo nelle pistole
montate sui tubi delle
colonnine di gonfiaggio
delle stazioni di servi-
zio, qui viene svolta dai
due pulsanti freccia su e
freccia giù.

Con riferimento alla **Fig. 1**, le due valvole sono inserite una tra l'arrivo dell'aria dal compressore e il raccordo di gonfiaggio e l'altra fra questo ed uno sfiato dotato di silenziatore (un cappuccio poroso...) per ridurre il rumore dell'aria quando viene fatta uscire dalla macchina.

In questa tubazione a "T" abbiamo inserito un trasduttore di pressione, che ci serve a rilevare la pressione e comunicarla





al microcontrollore che gestisce l'unità; tale informazione viene poi utilizzata per visualizzare il valore dell'attuale pressione del pneumatico o pallone nella modalità manuale, ovvero per visualizzare e stabilizzare la pressione e compiere in automatico il ciclo di gonfiaggio/sgonfiaggio, nella modalità automatica. Inoltre si trova un regolatore di pressione che serve a limitare a 6,7 bar la pressione nel circuito pneumatico; tale regolatore andrà registrato mediante un'apposita procedura che descriveremo più avanti in questo articolo.

SCHEMA ELETTRICO

Prima di descrivere dettagliatamente il funzionamento, diamo uno sguardo al circuito elettrico del gonfiagomme, il cui cuore è il microcontrollore PIC18F2550, il quale gestisce il display a 4 righe per 20 caratteri, comanda le elettrovalvole tramite i MOSFET T1 e T2, legge la tastiera a matrice 4x4 (4 righe per 4 colonne) e lo stato del sensore di pressione U1; inoltre il micro, essendo dotato di interfaccia USB, può dialogare con un Personal Computer in modo da essere configurato attraverso un'utility software che potete scaricare dal nostro sito web www.elettronica.in.it. Oltre a fare ciò, il PIC comanda il cicalino BZ1, attraverso il quale fornisce le segnalazioni quale quella di avvenuto gonfiaggio.

L'intero circuito viene alimentato da un alimentatore stabilizzato in grado di fornire 9÷12 Vcc e una corrente di circa 1 ampere; la tensione d'ingresso viene inviata al regolatore integrato lineare U3, che ne ricava 5 volt per alimentare il microcontrollore, il display e il trasduttore di pressione, quindi al DC/DC switching basato sull'integrato U4, che ricava 11,7 volt con i quali vengono fatte

funzionare le elettrovalvole. La stessa tensione d'ingresso giunge ad un ulteriore regolatore, che è il caricabatteria Microchip MCP73844-8.4 usato per mantenere in carica il pacco batterie Li-Ion da 7,2 V, indispensabile per consentire l'utilizzo del sistema all'esterno e in generale dove non c'è la rete elettrica.

Analizziamo le singole parti iniziando con lo stabilizzatore a 5 volt, che fornisce la tensione occorrente al microcontrollore: impiega l'integrato TC1262 (U5) della Texas Instruments, che è un regolatore lineare a basso dropout (LDO) utilizzato nella canonica configurazione e preceduto dai condensatori C1 e C6 per filtrarne la tensione di ingresso da ripple e disturbi provenienti dall'alimentatore e seguito da altri due condensatori che filtrano la componente di uscita, così da alimentare il PIC con una tensione non solo stabilizzata, ma anche priva di disturbi che potrebbero destabilizzarne il funzionamento. Il componente è incapsulato in contenitore SMD. Il secondo regolatore è basato sull'integrato MC34063AD, anch'esso di produzione TI, che con pochissimi componenti esterni realizza un completo regolatore buck /boost in grado di accettare in ingresso da 3 a 40 Vcc ed erogare in uscita fino ad 1,5 ampere; nel circuito si occupa di fornire gli 11,7 volt stabilizzati al circuito di potenza, ossia quello che tramite i MOSFET comanda i solenoidi delle elettrovalvole. Nel nostro caso l'integrato opera in configurazione boost ad una frequenza di oscillazione determinata dal condensatore collegato tra il piedino 3 e massa; la tensione d'uscita (Vout) dipende dalla retroazione che il partitore R12/R13 esercita verso il piedino 5 e si determina con la formula:

$$V_{out} = 1,25 (1 + R12/R13)$$

Funzionando in modo boost, l'integrato ha la bobina L1 sul collettore del transistor di uscita, il cui emettitore, connesso internamente al piedino 2, finisce a massa insieme al pin 4 (GND della circuizione interna del regolatore). Il funzionamento è il seguente: l'oscillatore interno sviluppa un'onda triangolare che in un comparatore di tensione viene confrontata con il potenziale di retroazione (piedino 5) in modo da ottenere impulsi la cui larghezza dipende dalla tensione erogata dal converter. Gli impulsi pilotano la base del transistor di potenza integrato nel componente, il quale quando è in piena conduzione fa fluire corrente nell'induttore L1 caricandolo; nelle pause, l'induttore tende a reagire alla brusca interruzione della corrente generando una tensione inversa che per qualche istante cerca di mantenere inalterato il flusso ed il verso della corrente che, ora, può sfuggire attraverso il diodo Schottky D4, attraverso il quale C3 riceve impulsi che lo caricano. Dunque, nei periodi di conduzione il transistor carica l'induttanza, la quale nelle pause restituisce al C3 l'energia che ha immagazzinato.

L'effetto di stabilizzazione della tensione di uscita si manifesta riguardo sia alle fluttuazioni della tensione d'ingresso, sia rispetto al variare del carico, perché se la differenza di potenziale erogata varia, la retroazione permette al regolatore PWM realizzato dal comparatore e dall'oscillatore di adeguare la larghezza degli impulsi, aumentandola se serve più energia perché la tensione ai capi di C3 sta scendendo, ovvero riducendola se detta tensione sta salendo e serve perciò meno energia.

Passiamo adesso a vedere come funziona il caricabatteria, che si affida ad un chip completamente automatico in grado di compiere la corretta curva di ricarica delle batterie agli ioni di litio, curva che prevede una prima carica a corrente costante fino a circa 4 V per cella e poi la carica a tensione costante, puntando al valore di 4,2 V, sempre per cella.

La particolarità dell'MCP73844 è che prevede una fase ausiliaria di preconditionamento (a bassa corrente) quando le celle sono praticamente scariche, prima di avviare la carica standard; inoltre questo integrato permette di definire la corrente di carica costante mediante l'applicazione del resistore R17 applicato in serie al MOSFET deputato alla carica, usando il piedino SENSE. La carica a corrente costante avviene a 330 mA. L'interruttore SW permette di sconnettere il circuito quando la batteria è in carica e alimentarlo una volta completata la carica e sconnessione l'alimentatore-caricabatteria dal plug di alimentazione PWR.

Diamo adesso uno sguardo alla sezione di visualizzazione, che consta di un display PCM2004D, gestito nella maniera tipica di questi display: infatti, analogamente ai classici 2 righe per 16 caratteri o 20 caratteri basati sul controller Hitachi HD44780 o compatibile, prevede un bus dati di 8 bit (D1÷D7), una linea di RW, una RS ed una E; l'RW indica al display se deve acquisire i dati in arrivo (Write) o produrre sul bus dati diretti al microcontrollore (Read) e normalmente è posto a massa, perché come nel nostro caso tipicamente il display si usa solo in modo passivo. Il contatto E è l'enable e permette di accendere e spegnere la logica del modulo, ovvero attivare e disattivare la visualizzazione (nel

nostro caso è governato dalla linea RC0 del micro); l'RS serve invece per dire al display, nella modalità Write, se i dati in arrivo sul bus si riferiscono a comandi oppure a caratteri da visualizzare. Per comando si intende ad esempio lo spostamento del cursore o l'inizio della scrittura da una determinata posizione. Andiamo ora a vedere la sezione del microcontrollore che si occupa di leggere la tastiera a matrice: le linee RB0, RB1, RB2 ed RB3 corrispondono alle righe e le RB4÷RB7 alle colonne; per rilevare la pressione dei tasti il firmware che gira nel microcontrollore usa la modalità multiplex, ponendo a livello logico alto una per volta e in rapida sequenza le linee RB4, RB5, RB6, RB7, quindi andando a leggere la condizione logica di RB0÷RB3. Ogni volta che viene premuto un tasto, il microcontrollore verifica qual è la linea tra RB4÷RB7 attiva al momento e la correla con la RB0÷RB3 sulla quale è stato letto l'1 logico.

Notate che a causa della scarsità di linee a disposizione nel microcontrollore, per le colonne 3 e 4 abbiamo utilizzato RB6 ed RB7 in comune con il connettore ICSP per la programmazione in-circuit (RB7 fa anche da PGD ed RB6 da PGC); inoltre le righe 1 e 2 condividono RB0 ed RB1 con SDA ed SCL dell'PC-Bus utilizzato per leggere la pressione rilevata dal sensore di pressione SENS. I diodi D1 e D2 servono proprio per evitare che l'attività sul bus disturbi la lettura della tastiera. Il sensore è un componente della SensorTechnics capace di misurare pressioni dell'aria fino a 6,89 bar (è per questo che il nostro apparato è tarato per regolare fino a 6,7 bar) corrispondenti a 100 psi del sistema metrico inglese e di esprimerne il valore

mediante stringhe di dati che invia su richiesta all'PC-Bus che ne costituisce l'interfaccia. Il componente presenta un offset, ossia un valore base di 410 che fornisce alla pressione atmosferica, ossia senza rilevare alcuna pressione sulla sua membrana.

In vero, il nostro sensore potrebbe anche fornire un'indicazione analogica della pressione misurata, grazie al piedino 2, sul quale si rileva una tensione dell'ordine di 0,04 V/psi (circa 0,59 V/bar).

COME FUNZIONA

Bene, ora che abbiamo analizzato il circuito elettronico possiamo spiegare come funziona il sistema di gonfiaggio, riprendendo quanto detto, cioè che può operare in tre modalità: manuale, automatica e programmazione. All'accensione e dopo la calibrazione automatica, il display propone un menu contenente tre voci tra le quali si può scorrere.

Selezionare modalità	
Manuale	(1)
Automatica	(2)
Programmazione	(3)

Nella modalità **Manuale (1)**, sfruttando i tasti freccia su e giù si apre l'aria o si sgonfia, come si farebbe con la leva e il pulsante della pistola tradizionale del gommista.

Quando il valore scende a zero, freccia giù non sortisce più alcun effetto. Quando si preme la freccia su, il microcontrollore comanda l'apertura dell'elettrovalvola che porta l'aria dal compressore all'uscita; premendo freccia giù, quest'elettrovalvola resta chiusa e si apre quella di scarico, che fa defluire l'aria dal condotto di uscita al silenziatore. Durante il gonfiaggio/sgonfiaggio, il display mostra la pressione rilevata

sia numericamente che mediante una barra.

MODALITA' MANUALE
3,20 bar (46,40 psi)
- ■■■■■■ +
Zero(3) Imposta(4)

Notate che, analogamente a quanto accade con la pistola del gommista o della stazione di servizio, quando si apre l'aria per gonfiare viene spesso visualizzata una pressione più alta di quella reale.

Selezionando la voce "Imposta" col tasto 4, il microcontrollore richiede di impostare, con i tasti numerici e freccia sinistra/destra, la pressione desiderata, da confermare premendo il tasto OK, allorché il display chiede conferma (**Confermi?**).

MODALITA' MANUALE
0,00 bar (0,00 psi)
Imp. press: 1,00 bar
Confermi?(OK)

Premendo nuovamente OK si avvia il gonfiaggio/sgonfiaggio.

MODALITA' MANUALE
0,65 bar (9,42 psi)
Press. Imp: 1,00 bar
Operaz. in corso

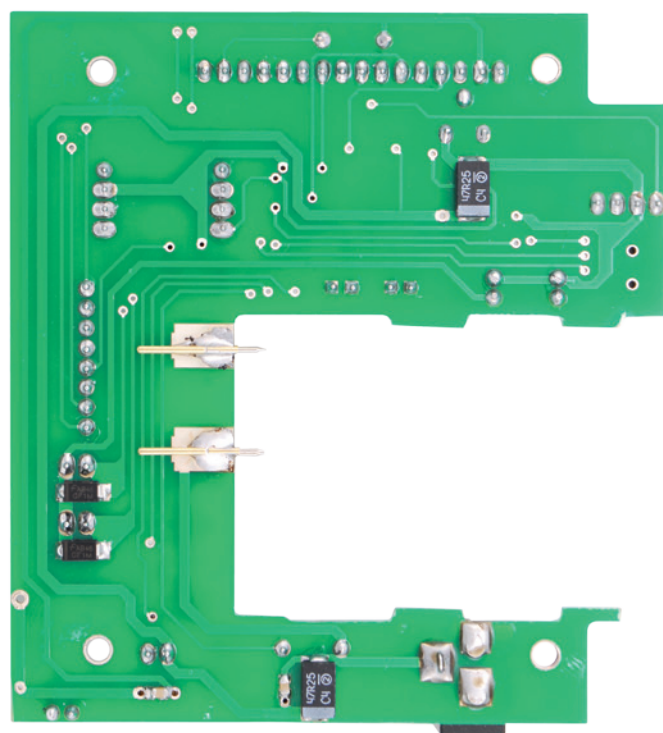
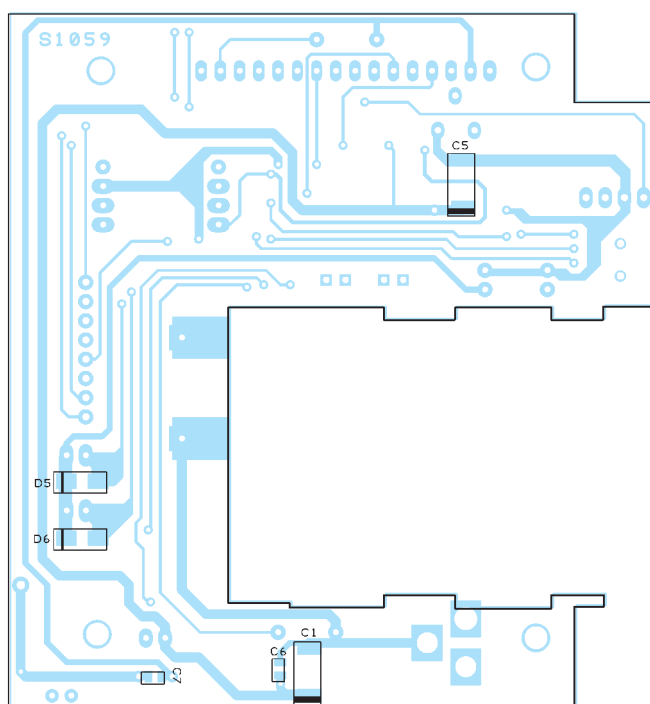
Notate che il gonfiaggio automatico non avviene aprendo continuamente la valvola, in quanto ciò causerebbe malfunzionamenti. Per capire ciò basti immaginare cosa accade quando si preme la leva della pistola della colonnina dell'aria dal benzinai: nel momento in cui si apre la valvola, il manometro segna una pressione decisamente maggiore di quella effettiva e lasciando la leva vediamo che in realtà

I sensori nei pneumatici

Dal novembre 2012 la Comunità Europea richiede ai costruttori di vetture – per l'omologazione dei nuovi modelli – l'adozione del TPMS, ovvero del Tyre Pressure Monitoring System. Ogni sistema pneumatico/cerchione viene quindi dotato di un sensore che, tramite un segnale a radiofrequenza di bassa intensità, comunica con una centralina la pressione rilevata. Questa soluzione permetterà a chi guida di essere avvisato quando lo pneumatico non sta operando entro i parametri previsti, peggiorando consumi, aderenza e usura. Si tratta di un'idea molto valida e per questo sono già in arrivo dei kit di retrofit per dotare della medesima funzionalità TPMS anche i veicoli che per la loro "età" ne sono privi, attraverso sensori interni o addirittura esterni da installare al posto dei tappi delle valvole. Oltre a questo, per ovvi motivi di discriminazione fra i segnali generati dai sensori, questi vanno abbinati alla centralina ogni volta che si effettua qualche intervento su di essi. Le centraline dei modelli più prestigiosi possono funzionare solo con sensori originali, abbinati



con procedure effettuabili con attrezzature in dotazione alle officine autorizzate, creando qualche problema di costo in caso di cambi gomme fra estate e inverno che prevedano la sostituzione della valvola. Purtroppo se si deve sostituire la valvola, c'è agganciato il sensore e quindi se si cambia la valvola, anche il sensore andrebbe sostituito pur essendo ancora perfettamente operativo. Da qui l'idea di qualche azienda di produrre sensori sostitutivi che possono "clonare" il codice di quelli montati originariamente, evitando quindi le procedure di abbinamento sulla centralina. Il mercato del TPMS è comunque appena ai primi passi nell'aftermarket e come i sensori di parcheggio sono diventati accessori a basso costo e facilmente reperibili, molto probabilmente potremo assistere alla nascita di interessanti dispositivi dedicati a questa applicazione.



Elenco Componenti:

R1: 10 ohm
 R2÷R8: 4,7 kohm (0805)
 R9, R10: 680 ohm (0805)
 R11a, R11b: 0,1 ohm 1W (2010)
 R12: 47 kohm (0805)
 R13: 5,6 kohm (0805)
 R14: 180 ohm (0805)
 R15: -
 R16: -

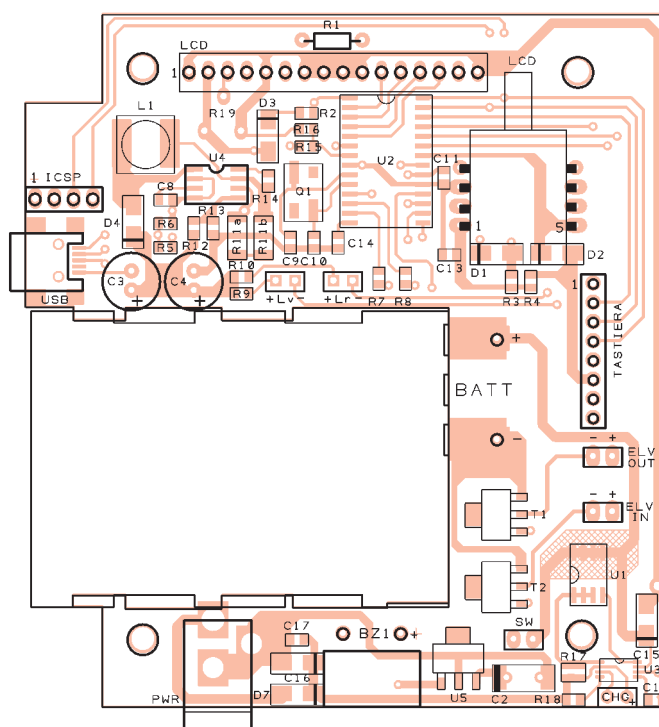
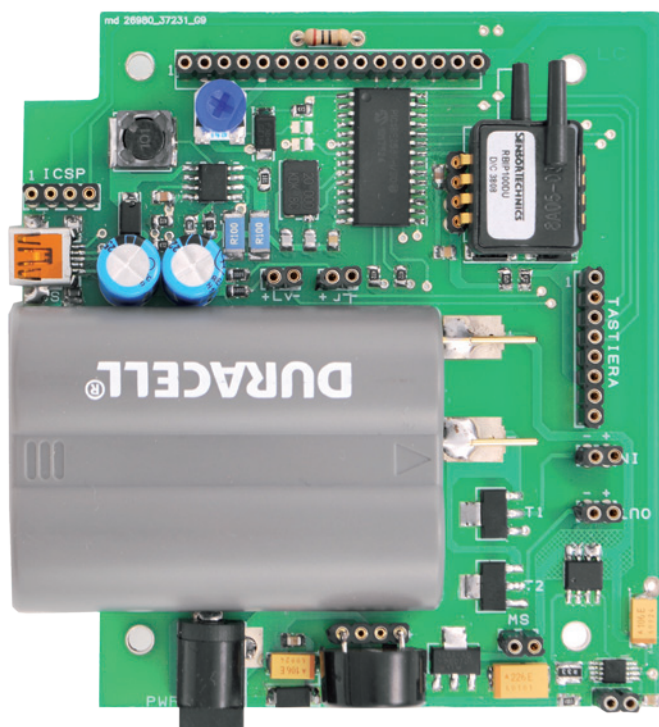
R17: 330 mohm (1206)
 R18: 100 kohm (0805)
 R19: Trimmer 10 kohm
 C1: 47 μ F 25 VL tantalio (D)
 C2: 22 μ F 16 VL tantalio (D)
 C3, C4: 220 μ F 25 VL elettrolitico
 C5: 47 μ F 25 VL tantalio (D)
 C6, C7: 100 nF ceramico (0805)
 C8: 1,5 nF ceramico (0805)

C9, C10: 15 pF ceramico (0805)
 C11: 220 nF ceramico (0805)
 C12, C14: 100 nF ceramico (0805)
 C15, C16: 10 μ F 16 VL tantalio (C)
 C17: 100 nF ceramico (0805)
 D1, D3: GF1M-E3
 D4: MBRS140TRPBF
 D5÷D7: GF1M-E3
 Q1: Quarzo 20 MHz (C7S)

la pressione è molto più bassa. Questo accade perché aprendo la valvola del compressore, l'aria, dovendo passare in un condotto che si assottiglia quale quello della valvola di gonfiaggio di un pneumatico, deve aumentare di velocità per mantenere la pressione costante; succede, però, che oltre una certa sezione si determina una resistenza che impedisce l'aumento della velocità e dell'aria, quindi il condotto si oppone in una certa misura al passaggio, determinando una pressione diversa da quella dell'interno del pneumatico.

Se il sistema andasse ad aprire la valvola dell'aria in arrivo dal compressore e si arrestasse quando il trasduttore di pressione rileva una pressione uguale a quella impostata, in realtà il gonfiaggio non avverrebbe correttamente, ma si fermerebbe a una pressione decisamente minore. Per evitare questo problema, il nostro sistema in gonfiaggio apre e chiude periodicamente la valvola che fa affluire l'aria dal compressore all'uscita (cioè al pneumatico), quindi durante la chiusura va a leggere il dato fornito dal sensore di pressio-

ne. Facendo la misura in questo modo, siamo certi di leggere la pressione nel pneumatico e non quella fornita dal compressore. Analogamente, in fase di sgonfiaggio viene aperta e chiusa la valvola di scarico e la pressione si misura durante la chiusura. Ma c'è di più: per adeguare il rapporto tra apertura dell'elettrovalvola e chiusura, prima di effettuare il gonfiaggio o lo sgonfiaggio il microcontrollore calibra il sistema per ottenere un gradiente di pressione ottimale e funzionale all'oggetto da gonfiare. La calibrazione si ottiene aprendo e



U1: NDS8434A/NDS8435A
 U2: PIC18F2550-I/SO
 U3: MCP73844-840I/MS
 U4: MC34063AD
 U5: TC1262-5.0VDB
 LCD: Display 20x4 (LCD20X4)
 T1, T2: IRFL110PBF
 BZ1: Buzzer 12V con elettronica
 L1: Induttanza 100 μ H 790 mA

USB: Connettore Mini-USB
 SENS: Sensore pressione RBIP100DU

Varie:
 - Plug alimentazione
 - Batteria DRNEL3
 - Tastiera a matrice 4x4
 - Strip Femmina 2 poli (6 pz.)
 - Strip Femmina 4 poli torniti (3 pz.)

- Strip Femmina 8 poli (2 pz.)
 - Strip Femmina 16 poli (1 pz.)
 - Strip Maschio 2 poli (6 pz.)
 - Strip Maschio 4 poli (1 pz.)
 - Strip Maschio 8 poli 90° (2 pz.)
 - Strip Maschio 16 poli lungo (1 pz.)
 - Interruttore da pannello
 - Contenitore plastico G421
 - Circuito stampato

chiudendo per un tempo molto breve la valvola di entrata o quella di scarico e misurando il gradiente di pressione; in altre parole, la macchina apre la valvola dal compressore o dello scarico per un tempo fisso (10 ms) e poi la chiude, calcolando la variazione della pressione per unità di tempo (bar/ms) in aumento o in diminuzione. I
 Il firmware del microcontrollore contempla sette range del coefficiente di variazione, ai quali corrispondono altrettanti periodi di apertura della relativa valvola in gonfiaggio o sgonfiag-

gio. Per fare un esempio, se nella fase di test il circuito rileva in gonfiaggio un gradiente di salita della pressione di 0,01 bar per 10 millisecondi, adotterà intervalli di gonfiaggio minori di quelli che imposterebbe se rilevasse un gradiente di 0,03 bar/10ms. Tutto questo serve perché se, si gonfia un palloncino, un tempo fisso di gonfiaggio/pausa e misura può essere eccessivo ed invece corto per un grande pneumatico. Stabilito qual è il coefficiente di crescita, inizia il gonfiaggio adeguando i periodi di gonfiaggio

a quale delle fasce corrisponde l'elemento da gonfiare; quindi ripete questo ciclo fin quando la lettura a valvola chiusa non dà il valore voluto, allorché la procedura termina e la cosa viene segnalata da un avviso acustico, nonché visivo tramite il display.

MODALITA' MANUALE
1,02 bar (14,79 psi)
Press. Imp: 1,00 bar
Op. eseguita _

Vediamo adesso la modalità **Automatica (2)** nella quale c'è

un database di 8 modelli i cui parametri di gonfiaggio sono personalizzabili a piacimento dall'utente. Entrando in questa modalità (con il tasto 2)

MODALITA' AUTOMATICA
Selezionare veicolo
(1) AUDI A3
(2) FORD FIESTA

si sceglie il numero corrispondente al programma che si vuole eseguire poi il tipo di gomma e di gonfiaggio, poi si conferma con OK per iniziare il ciclo.

AUDI A3 Ut.norm. _
Ruota Ant. 2,20 bar
0,00 bar (0,00 psi)
Esegui(OK) Canc(ANN)

Per definire i dati relativi agli 8 programmi bisogna entrare in modalità **Programmazione (3)** quindi scegliere la voce di menu "Imposta veicolo e pressioni" col tasto 4.

Imposta veicolo (4)
e pressioni
Programmaz. da PC(5)

Dato che il sistema nasce principalmente per gonfiare i pneumatici degli autoveicoli, per ciascun programma si possono definire le pressioni di: ruota anteriore, posteriore e di scorta, quindi per le prime due si definisce la pressione normale e di pieno carico (per la ruota di scorta non ha senso specificare una pressione di pieno carico). Vediamo dunque come effettuare la programmazione: entrati in una posizione di memoria, se questa è libera si introducono i dati da zero, mentre se è occupata appaiono i vecchi dati, che si possono modificare.

MODALITA' PROGRAMM.
Selezionare veicolo
(1) AUDI A3
(2) FORD FIESTA

Ciascun programma può prendere il nome di un tipo di veicolo, da introdurre nella prima voce, che, per la posizione 1 è:

Veicolo 1: _ _ _ _

Definito il tipo di veicolo (sono a disposizione 8 caratteri per ciascun nome) si passa a definire la pressione per la ruota anteriore, posteriore o di scorta; se dovete impostare il programma di gonfiaggio automatico di palloni e palloncini, definite uno dei valori a scelta. Sotto la voce **Veicolo:** _ _ _ _ trovate la scritta:

Ruota anteriore

Veicolo 1: AUDI A3
Ruota anteriore
Utilizzo normale
Pressione: 2,20 bar

Potete cambiarla in **Ruota posteriore** o **scorta** semplicemente spostandovi con i tasti freccia laterali. Definito qual è la ruota, sotto trovate l'indicazione:

Utilizzo normale/Pieno carico

Anche in questo caso, con i tasti freccia laterali definite la pressione normale o di pieno carico, cui potete poi far gonfiare automaticamente le gomme. Per la posizione **Ruota di scorta** non è prevista la scelta dell'utilizzo: accanto alla voce Utilizzo si trovano dei trattini.

Alla fine potete impostare la pressione corrispondente, mediante i tasti numerici:

Gli elementi occorrenti a realizzare il circuito pneumatico del sistema di gonfiaggio si possono acquistare presso un rivenditore di articoli tecnici o di materiale per impianti pneumatici. Gli elementi occorrenti sono i seguenti:

- 1) tappo Ø 6 mm;
- 2) raccordo automatico in ottone Ø 6 mm femmina - 1/8" maschio;
- 3) raccordo automatico in ottone Ø 4 mm femmina - 1/8" maschio;
- 4) tubo in gomma Ø 4x2 mm (l=200 mm circa);
- 5) raccordo a "T" in ottone 1/8" F/F/M;
- 6) raccordo a "T" in ottone 1/8" M/F/M;
- 7) elettrovalvola pilota 2 vie 1/8" F - F (2 pezzi - 7a) con bobina 12Vdc 5W (444mA) (2 pezzi - 7b);
- 8) silenziatore 1/8";
- 9) raccordo automatico girevole 90° in tecnopolimero Ø 6 mm femmina - 1/8" maschio;
- 10) tubo in poliuretano Ø 6x4 mm (l=3,5 cm);
- 11) molla l=15 mm, Øint 6,6 mm, Ø filo 0,7 mm;
- 12) riduttore di pressione miniaturizzato 90°;
- 13) riduzione cilindrica 1/4" maschio 1/8" femmina;

Pressione: _ _ _ _ bar

Con OK si salva e si passa a definire una nuova ruota o nuova condizione di carico. Ogni volta che si preme il pulsante OK, il display risponde con il messaggio **Operazione eseguita!**

Operazione eseguita!

Fatto tutto, si esce dal menu premendo ANN/CNCL un passo alla volta. La semplice pressione di ANN/CNCL permette di tornare indietro di un passo per rivedere la configurazione del programma. Notate che in tutte le modalità si possono impostare valori di pressione a passi di 1/100 bar, quindi con molta precisione; il display mostra sempre i valori in bar e in PSI, oltre (nella sola modalità manuale) ad una barra

Realizzare il circuito pneumatico

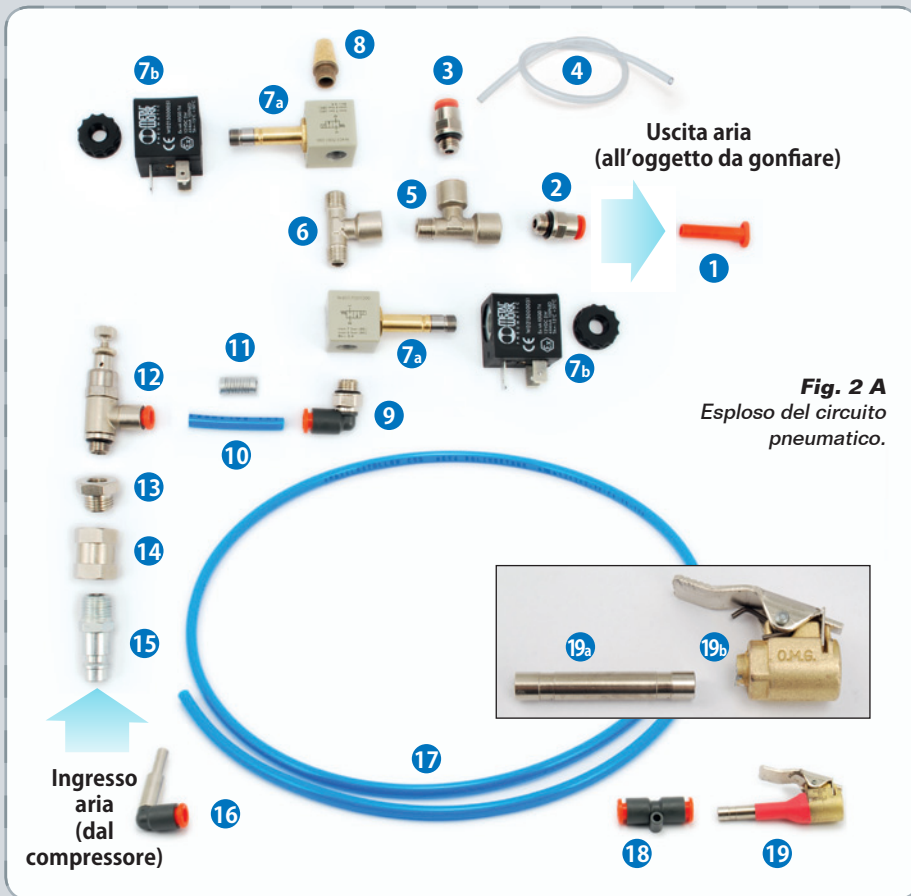


Fig. 2 A
Esploso del circuito
pneumatico.

- 14) raccordo intermedio 1/4" femmina - femmina;
- 15) innesto rapido maschio 1/4" in tecnopolimero Ø 6 mm femmina - maschio;
- 16) raccordo automatico 90° in tecnopolimero Ø 6 mm femmina - maschio;
- 17) tubo in poliuretano Ø 6x4 mm (l=1,2 m);
- 18) raccordo automatico diritto in tecnopolimero Ø 6 mm femmina - femmina;
- 19) prolunga Ø 6 mm maschio - maschio (19a) con attacco per valvola pneumatici (19b);

Le singole parti vanno assemblate come visibile nella **Fig. 2 B** usando, per le giunture a filetto, del nastro bianco di teflon per impianti idraulici, che assicurerà la necessaria tenuta.

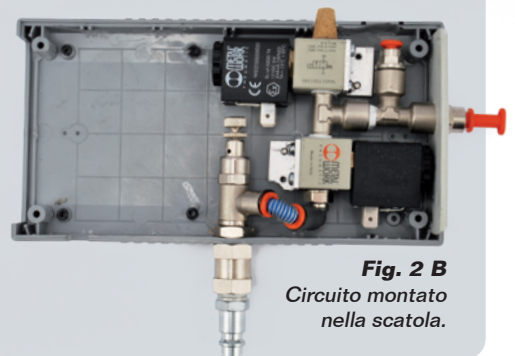


Fig. 2 B
Circuito montato
nella scatola.

che indica la pressione di gonfiaggio.

IMPOSTAZIONE DA COMPUTER

Il nostro gonfiagomme può anche essere configurato da PC collegandolo, tramite l'apposita presa mini USB di cui dispone, ad un Personal Computer nel quale sia stato installato l'apposito software "Gonfia Pneumatici Automatico" scaricabile dal nostro sito Internet www.elettronica.in.it. Il software consente di definire il database degli otto programmi che normalmente, in stand-alone, si imposterebbero mediante la tastiera del sistema; per l'esattezza, con il programma su PC è possibile leggere i dati dei veicoli impostati in programmazione, impostarne di nuovi, cancellare tutti i dati in memoria, esportare il database (in un file ad estensione .ini) ed ottenere i dettagli della periferica.

Per consentire il collegamento al PC, il microcontrollore è stato programmato in modo da essere visto come periferica HID (Human Interface Device) quindi come un mouse o una tastiera; ciò consente di far identificare la nostra unità senza bisogno di driver aggiuntivi. Per utilizzare l'unità è necessario collegarla al PC tramite cavo USB quindi stabilire la connessione scegliendo l'ultima voce del sottomenu **Programmazione**, ossia **Programmazione da PC (5)**.

Connessione a PC....

A connessione avvenuta, il display dell'unità mostra il messaggio seguente:

**Connessione a PC
attiva**

Una volta stabilito il collegamento con il computer e lanciato il software, la connessione col PC è attiva. Avviate il software (**Fig. 3**), quindi aprite il menu a tendina che riepiloga le periferiche HID e selezionare quella corrispondente al gonfiagomme; a quel punto premendo il tasto "Connetti a Periferica" il collegamento viene correttamente instaurato e i dati letti dal gonfiapneumatici appaiono nella schermata (**Fig. 4**). Va notato che per poter utilizzare il sistema in simulazione HID, il microcontrollore PIC18F2550 deve comportarsi come una tastiera o un mouse, ovvero deve ciclicamente trasmettere dati verso l'USB del computer; ciò perché le periferiche HID vengono gestite verificando che ciclicamente producano dei dati indicanti la posizione del puntatore, i tasti premuti ecc. Se questo refresh periodico non

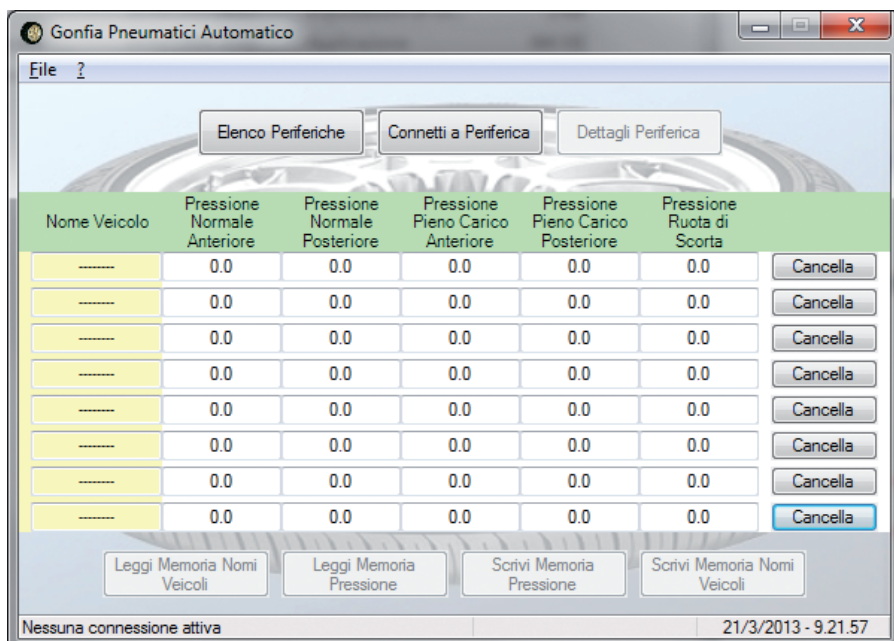


Fig. 3 - Schermata di avvio del programma da PC.

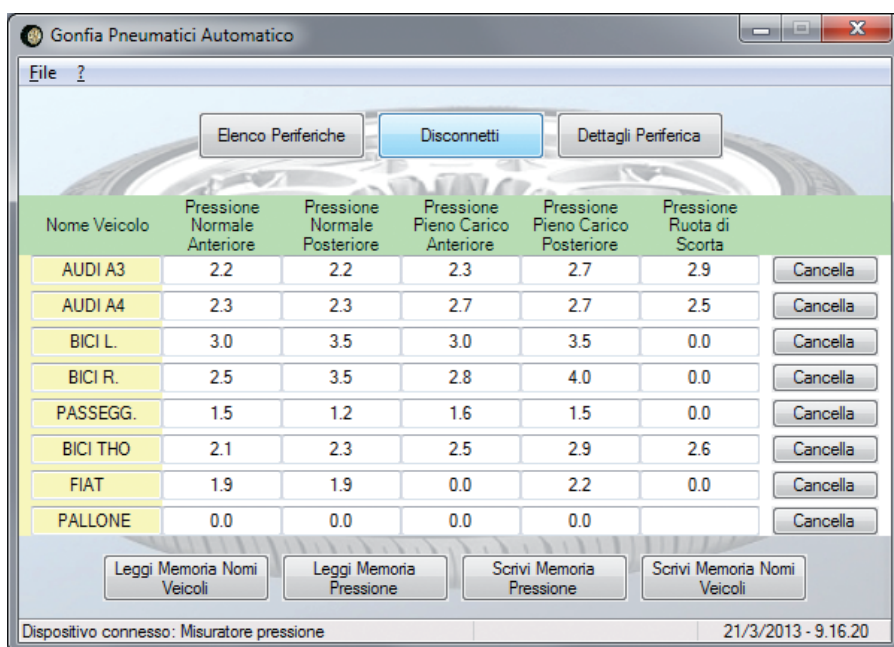


Fig. 4 - Dati nella schermata del programma acquisiti dal gonfiapneumatici.

avviene, il computer considera disconnessa la periferica HID, il che nel nostro caso si traduce nell'interruzione della comunicazione con l'unità.

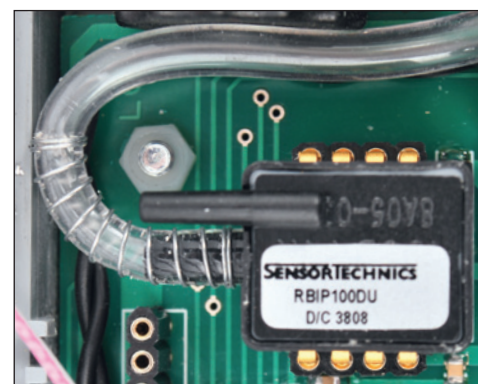
REALIZZAZIONE E COLLAUDO

Passiamo adesso a descrivere la costruzione del gonfiagomme, che consta di una parte meccanica ed una elettronica; quest'ulti-

ma è la scheda di controllo ed è realizzata su un circuito stampato a doppia faccia di cui trovate le tracce rame nel nostro sito www.elettronica.in sotto forma di file immagine, pronte per essere stampate in modo da ottenere le pellicole per la fotoincisione. Per il montaggio dei componenti, che sono per la maggior parte in SMD, seguite i disegni e le

illustrazioni in queste pagine, ricordando di iniziare con gli elementi a più basso profilo e rispettare la polarità e il verso indicati per i componenti a semiconduttore, il cicalino e i condensatori elettrolitici. La tastiera a matrice 4x4 e il display LCD è previsto che vengano montati all'esterno e collegati mediante due connettori single-in-line realizzati da pin-strip femmina a passo 2,54 mm. Il sensore di pressione va montato su uno zoccolo costruito usando due file di 4 contatti a tulipano e, una volta assemblata la parte pneumatica, raccordato mediante un tubicino con la porzione di tubo che si trova tra la valvola dell'arrivo dal compressore e quella di scarico dell'aria. Per assicurare la tenuta del collegamento tra tubo e sensore, sull'estremità del tubo stesso conviene applicare una molla, ad esempio quella di una penna, come mostrato in Fig. 5. Approntato il circuito elettronico, programmate il microcontrollore connettendo al connettore SIL siglato ICSP il programmatore e avviando il trasferimento del firmware. A questo punto dovete realizzare il circuito pneumatico, che si compone delle parti riepilogate nell'apposito riquadro (Fig. 2 A), da interconnettere secondo le foto Fig. 2 B e Fig. 8; per le giunture servirà del nastro

Fig. 5



in teflon. Realizzare la parte pneumatica dell'apparato non è difficile: bisogna solo ricordare che ogni raccordo filettato deve essere avvolto con tre/quattro giri di nastro in teflon, che fa da guarnizione; per avvitare bene le parti può non bastare la forza delle mani, almeno per i raccordi in metallo. Per aiutarvi, potete stringerli con delle pinze o chiavi, ma senza forzare troppo (basta un po' più della forza applicabile a mani nude) e senza danneggiare le superfici. Il circuito pneumatico va realizzato all'interno del contenitore plastico del

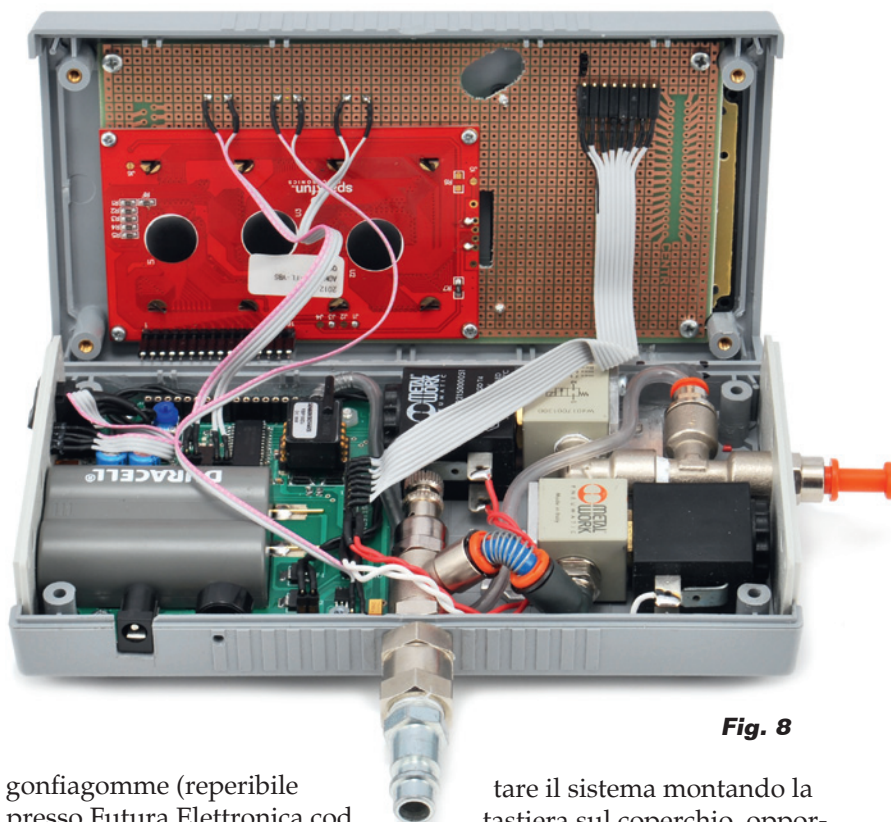


Fig. 8



Fig. 6

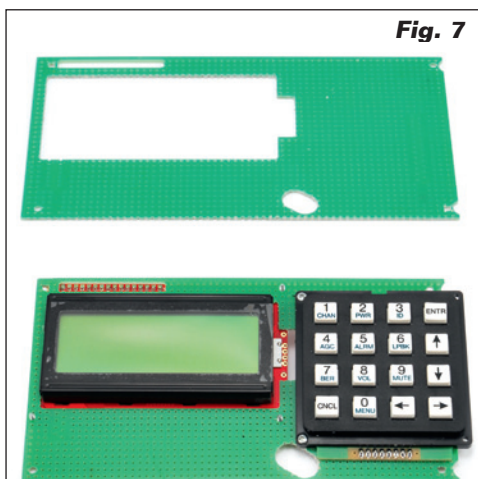
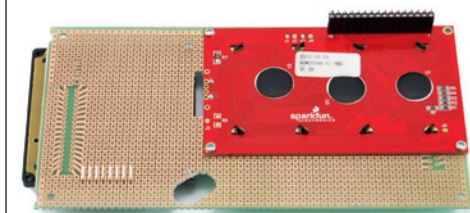


Fig. 7

Vista frontale del display e tastiera montati sulla basetta millefori.



Vista posteriore.

gonfiagomme (reperibile presso Futura Elettronica cod. G421), ripiegandolo mediante gli appositi raccordi angolari; sul tubo in plastica che raccorda l'arrivo dal compressore con la prima elettrovalvola è opportuno applicare la molla di contenimento. Notate che ciascuna delle elettrovalvole è composta da un solenoide attuatore e da una valvola vera e propria, che viene azionata dall'attuatore: questi due componenti si acquistano separatamente e si uniscono durante l'assemblaggio. Realizzato il circuito pneumatico e montate sul contenitore le flange di attacco (raccordi automatici) del compressore e del tubo di gonfiaggio (anche lo scarico con silenziatore va su uno dei lati del contenitore, in modo da sfogare all'esterno l'aria in eccesso). Ora bisogna fissare la basetta all'interno del contenitore quindi collegare a questa i fili delle due elettrovalvole, ricordando che la valvola di apertura dell'aria in arrivo dal compressore va connessa ai contatti ELV IN e quella di scarico a ELV OUT. Comple-

tare il sistema montando la tastiera sul coperchio, opportunamente forato e lavorato sia per far uscire i tasti, sia per lasciar affacciare all'esterno il display, che monterete anch'esso sul coperchio. Nel nostro prototipo abbiamo montato display, LED di segnalazione e tastiera su un pezzo di basetta millefori (Fig. 7) e fissato il tutto al coperchio del contenitore opportunamente forato (Fig. 6). Per collegare al circuito stampato tastiera e display, usate dei flat-cable con intestati dei pin-strip idonei ad entrare nei connettori SIL previsti nel circuito.

A questo punto, se avete completato il cablaggio potete passare alla taratura del regolatore di pressione; allo scopo, alimentate ed accendete l'unità, quindi premete e tenete premuto OK sulla tastiera mentre compare la scritta iniziale "calibrazione in corso...".

Calibrazione
in corso
Attendere prego
.....

Dopo i tre beep (1 lungo + 2 brevi) rilasciando OK si entra nella funzione di taratura della valvola.

TARATURA VALVOLA

- 1 Chiudere l'uscita
- 2 Premere 'ENTR'
- 3 Regolare valvola

Chiudete il raccordo di uscita con l'apposito tappo e collegate il compressore al raccordo di ingresso.

Adesso premete ENTER: il circuito fa uscire l'aria in eccesso e segnala la pressione esistente; a questo punto, usando un cacciaviti a lama di dimensioni adeguate dovete regolare la vite del regolatore di pressione fino ad ottenere sul display il valore di circa 6,7 bar, raggiunto il quale il display indica "pressione corretta". Allora si può premere il tasto **Annulla** e poi **freccia dx**: il sistema esce dalla calibrazione ed entra nel normale utilizzo.

Notate che all'accensione sul display appare offset 410 (l'indicazione può essere 410÷415, in base all'offset del sensore da voi usato) che è il valore corrispondente alla pressione atmosferica. Per l'uso, vi conviene rinominare i tasti della tastiera applicando sulla loro superficie delle etichet-

te autoadesive: i pulsanti numerici restano gli stessi, mentre * diventa ANN/CNCL, # la freccia a sinistra, D la freccia destra (SEL), B la freccia su (PAG-) e C la freccia giù (PAG+); il tasto ENTR/OK dovrà corrispondere all'A. Siccome i tasti numerici servono anche per introdurre i nomi dei programmi nella modalità **programmazione (3) - imposta veicolo e pressioni (4)**, vi può essere utile scrivervi anche la loro corrispondenza, che è:

1 = simboli . , ' / + -

2 = ABC

3 = DEF

4 = GHI

5 = JKL

6 = MNO

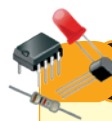
7 = PQRS

8 = TUV

9 = WXYZ

0 = _

Naturalmente per i tasti numerici vale la regola che una pressione nelle procedure che richiedono l'introduzione di lettere attiva il primo simbolo alfabetico o di punteggiatura, due rapide pressioni valgono per il secondo, tre rapide pressioni valgono per il terzo e per ultimo viene introdotto il numero associato al tasto. Nelle operazioni che richiedono la definizione dei soli numeri (valori di pressione), ogni singolo



per il MATERIALE

Tutti i componenti utilizzati in questo circuito sono facilmente reperibili. Il master del circuito stampato, il firmware per la programmazione del micro e il software sono scaricabili gratuitamente dal sito della rivista (www.elettronica.in.it). I componenti relativi alla sezione pneumatica sono reperibili presso i rivenditori della società Metal Work (www.metalwork.it). Il contenitore plastico utilizzato (cod. G421) è reperibile presso Futura Elettronica.

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11,
21013 Gallarate (VA)

Tel: 0331-799775 • Fax: 0331-792287
<http://www.futurashop.it>

intervento sul tasto introduce il numero corrispondente.

In linea generale ricordate che:

- per spostarsi da un carattere all'altro bisogna usare le frecce sinistra e destra;
- per cancellare un carattere scritto bisogna premere 0;
- per tornare rapidamente al menu principale quando ci si trova in un sottomenu, occorre premere e mantenere premuto il tasto 0.

Elettronica In

Sei un appassionato di elettronica? Ti piace progettare, costruire, modificare qualsiasi cosa purché abbia a che fare con transistor, display o microcontrollori? Vorresti condividere le tue conoscenze con gli altri lettori?

Cosa aspetti? ... Entra subito a far parte dei

COLLABORATORI HARDWARE della nostra Rivista

Realizzare un progetto per ElettronicaIn è molto semplice!

Dopo aver concordato con la redazione le linee guida del progetto, ti invieremo il materiale necessario con quale farai le prove e realizzerai un primo prototipo, anche su breadboard. Successivamente ci manderai il disegno dello stampato; alla sua realizzazione pratica provvederemo noi che ti rimanderemo la basetta ed i componenti per realizzare il prototipo finale, quello che poi andremo a fotografare e che presenteremo sulle pagine della rivista. Agli schemi elettrici definitivi provvederemo sempre noi mentre a te non rimarrà altro che descrivere il progetto nello stile della rivista.

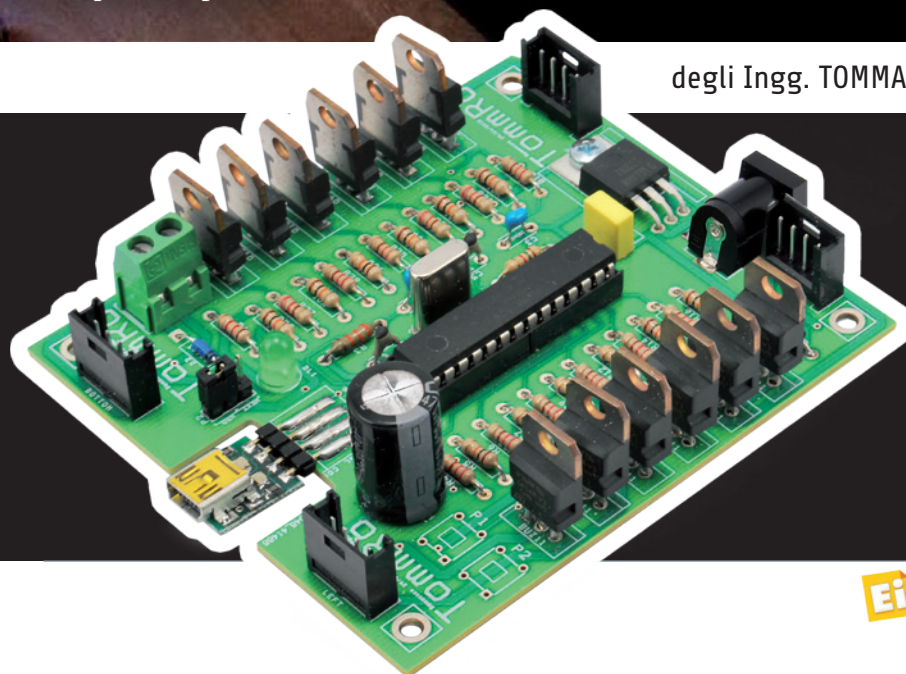
Manda una mail a: redazione.hardware@elettronica.in.it

Non perdere
questa opportunità!
**Contattaci
oggi stesso!**

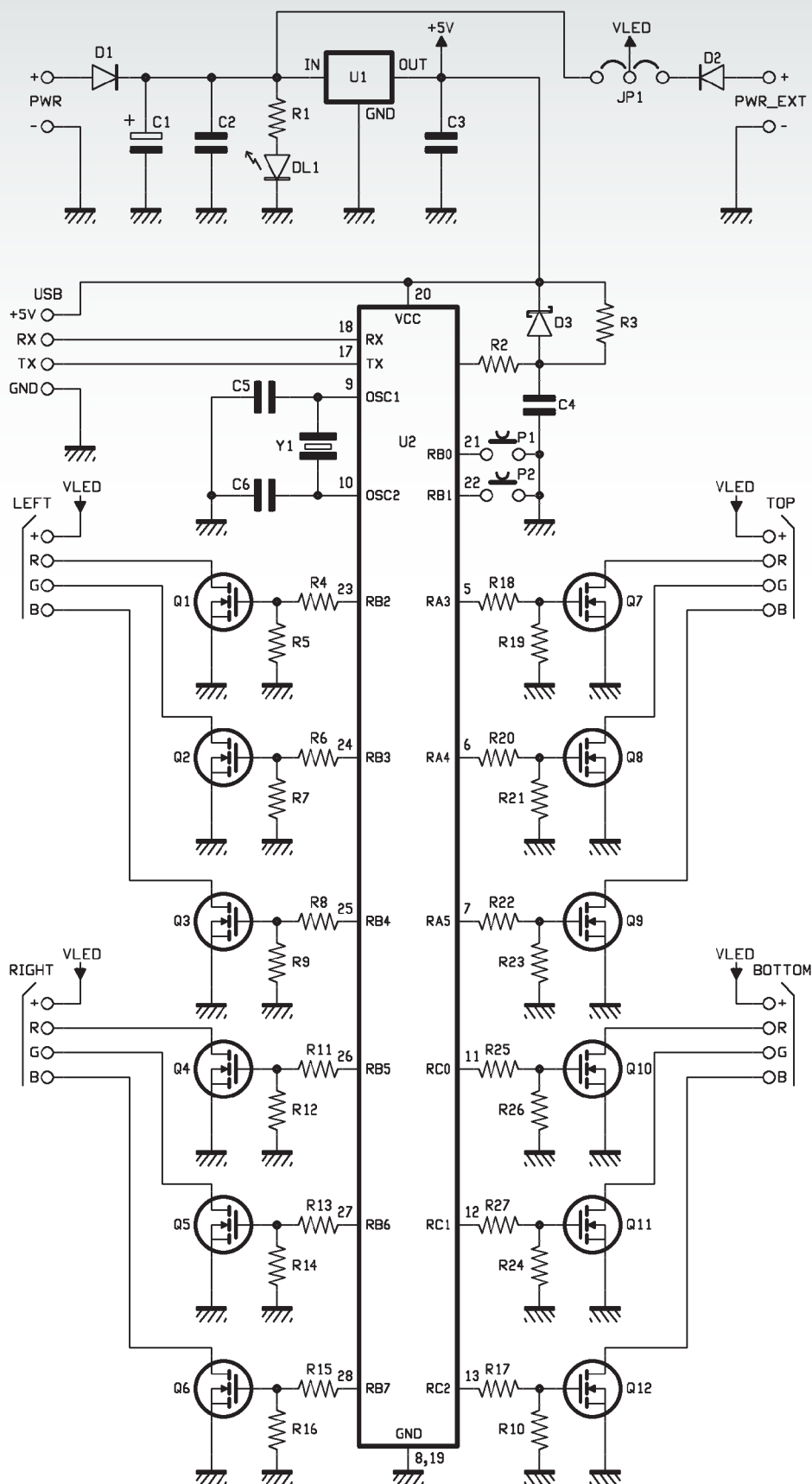
L'avvento dell'alta definizione e la disponibilità di schermi di dimensioni sempre maggiori ha permesso a molti di realizzare in casa propria una piccola sala cinematografica - o home theater - dove gustare con elevata qualità acustica e visiva sia le trasmissioni televisive, sia i contenuti su supporti ottici (DVD e BluRay), sia la musica. L'home theater è anche l'insieme delle apparecchiature audio e video necessarie per tale riproduzione domestica. Non essendoci specifiche tecniche da soddisfare, si possono realizzare sistemi economici da poche centinaia di euro e dalle scarse prestazioni, ma anche complessi impianti

SISTEMA HOME THEATER VIDEO LIGHT

degli Ingg. TOMMASO E ALESSANDRO GIUSTO



Miglioriamo l'effetto scenico di un sistema Home Theater domestico aggiungendo quattro strisce led RGB in modo da ottenere sensazioni visive il più coinvolgenti possibili.



da decine di migliaia di euro, dalle prestazioni elevatissime.

Una caratteristica molto importante per l'appassionato di home theater è che il sistema risulti il più "coinvolgente" possibile; dare una definizione del termine coinvolgente è abbastanza complicato in quanto si presta a diverse interpretazioni che dipendono anche dall'utilizzatore finale e dall'ambiente esterno. È innegabile che vedere un film la sera, a luci spente, senza rumori ambientali e con l'audio ad un volume leggermente più elevato rispetto al solito è sicuramente più intrigante che non la mattina, a basso volume e in condizioni di rumorosità esterna.

Altro aspetto che potrebbe migliorare il coinvolgimento emotivo è la presenza di eventuali luci esterne comandate per rappresentare al meglio quello che viene visualizzato a video: per esempio se la scena è ambientata al buio le luci potrebbero essere a bassa luminosità se non addirittura spente; nelle scene ambientate in acqua oppure in aria le luci dovrebbero essere di tonalità blu o azzurre; nelle scene nei boschi o con abbondante vegetazione di colore verde; in ambienti caldi rossi, ecc...

Infine più sono le luci che possono essere comandate indipendentemente, più attraente può essere il risultato finale. È chiaro che, al pari dei canali audio, più sono i "canali delle luci" gestiti, più coinvolgente sarà il risultato finale. Da questa breve descrizione è abbastanza evidente che se si desidera realizzare un sistema di questo tipo sono necessari come minimo due blocchi funzionali: il primo può essere rappresentato come un blocco a cui in ingresso vengono fornite le immagini rappresentate a video; in questo blocco le informazioni vengono

elaborate e come uscita vengono forniti degli opportuni comandi per pilotare le luci. Il secondo blocco riceve come input i comandi forniti dal primo blocco funzionale e, in accordo con tali comandi, genera gli opportuni colori da una palette completa. Il primo blocco è tipicamente un sistema software che gira sul computer utilizzato per la riproduzione video e che con tecniche specifiche "cattura" le immagini visualizzate, le elabora opportunamente e comunica in uscita i colori da rappresentare; la soluzione più comune passa dall'impiego di una porta usb o seriale per la comunicazione e il formato per rappresentare il colore desiderato è RGB, Red Green Blue.

Il blocco funzionale è invece una scheda elettronica collegata tramite porta usb o seriale al PC, dotata di una o più uscite capaci di pilotare, a seconda dei livelli ricevuti, le luci RGB.

In questo articolo realizzeremo un sistema hardware e software per un impianto di luci ambiente composto da 4 distinte strisce LED RGB; ogni striscia sarà comandata indipendentemente con un diverso canale legato al contenuto del video nelle posizioni left, right, top e bottom, ovvero lato sinistro, destro, alto e basso dell'immagine video. Come vedremo più avanti, ogni LED che compone le terne di luci RGB, verrà pilotato in PWM in modo da realizzare ben 16.777.216 sfumature di colore (256x256x256), con una buona approssimazione di tutti i colori dell'arcobaleno.

SOFTWARE BOBLIGHT

Relativamente al primo blocco funzionale, su Internet si trovano diverse applicazioni, la maggior parte gratuita. Uno dei software più conosciuti, semplice ed affidabile, è BobLight, disponibile

liberamente in Rete. È possibile scaricare la versione originale dal sito <http://blogger.xs4all.nl/loosen/articles/408184.aspx>, mentre una copia già configurata ad hoc per il nostro sistema è disponibile sul sito internet di Futura Elettronica.

Il software è disponibile per sistemi operativi Windows e Linux e l'ultima versione attualmente disponibile è la 1.3; non è richiesta nessuna procedura di installazione, mentre per poter funzionare è necessario che, a seconda dell'hardware utilizzato, venga configurato correttamente (fate riferimento al box Configurazione BobLight per i dettagli).

Il programma viene fornito in un pacchetto compresso di cui fanno

parte diversi file; a noi interessano i 2 eseguibili (boblighd.exe e boblight-getpixel.exe) e il file di configurazione boblight.conf.

Il software supporta 4 diversi tipi di periferiche (momo, sound, popen e LTBL – Let There Be Light) ciascuna con le proprie caratteristiche hardware e software. La nostra attenzione si è concentrata sul protocollo Momo (il più semplice tra quelli disponibili) basato sulla comunicazione seriale tramite porta COM (virtuale nel nostro caso in quanto utilizzeremo un convertitore usb/seriale).

Nel nostro progetto sono supportati 4 diversi canali indipendenti e per la comunicazione vengono utilizzati diversi comandi composti ciascuno da 3 byte per i valori

Protocollo di comunicazione

La scheda elettronica si aspetta di ricevere il valore dei canali RGB secondo il protocollo predisposto per la gestione di 4 canali RGB.

La configurazione della porta seriale è a 9600 di baud rate, nessun controllo di flusso, nessuna parità, 8 bit di dati, e 1 bit di stop.

Essendo predisposti 4 canali RGB la scheda si aspetta di ricevere "pacchetti" di 12 byte ripetuti ciclicamente ed in continuazione. Il formato dei pacchetti è il seguente:

```
Byte 01: Duty Cycle PWM colore rosso canale left.
Byte 02: Duty Cycle PWM colore verde canale left.
Byte 03: Duty Cycle PWM colore blu canale left.
Byte 04: Duty Cycle PWM colore rosso canale right.
Byte 05: Duty Cycle PWM colore verde canale right.
Byte 06: Duty Cycle PWM colore blu canale right.
Byte 07: Duty Cycle PWM colore rosso canale top.
Byte 08: Duty Cycle PWM colore verde canale top.
Byte 09: Duty Cycle PWM colore blu canale top.
Byte 10: Duty Cycle PWM colore rosso canale bottom.
Byte 11: Duty Cycle PWM colore verde canale bottom.
Byte 12: Duty Cycle PWM colore blu canale bottom.
```

Per convenzione il formato di ciascun duty cycle è il seguente: 0x00 indica duty cycle al 0% (luce spenta); 0xFF duty cycle al 100% (luce accesa alla massima luminosità) con tutti i restanti valori (0x01,...,0xFE) che indicano rispettivamente le percentuali tra 0% e 100% (avendo 256 valori la risoluzione della percentuale sarà di circa 0.39%2).

Il protocollo non prevede nessun carattere di sincronismo pertanto molto importante è che prima che vengano inviati i byte l'hardware sia già pronto a riceverli. La perdita della sincronizzazione comporta che i byte dei canali vengano confusi e quindi comandata un'uscita RGB errata al posto della corretta.

Configurazione Boblight

Il software boblight supporta diverse tipologie hardware di periferiche, con relativo protocollo di comunicazione. La configurazione viene gestita tramite il file di testo boblight.conf; vediamo una breve descrizione mentre per quella completa rimandiamo al relativo sito Internet <http://blogger.xs4all.nl/loosen/articles/408184.aspx>.

Il file di configurazione è diviso in 4 sezioni: "global", "device", "color" e "light".

Sezione GLOBAL

La sezione global è composta da 11 parametri:

- **Interface:** indirizzo ip (utilizzato nel caso di comunicazione tcp/ip).
- **Port:** porta tcp/ip (utilizzato nel caso di comunicazione tcp/ip).
- **Timeout:** timeout per i clients (utilizzato nel caso di comunicazione tcp/ip).
- **Interpolation:** attivazione interpolazione. Impostare a 1 per attivare; 0 per disattivare.
- **Proportional:** configurazione proporzionale (da 0.0 a 100.0).
- **Saturation:** configurazione saturazione.
- **Value:** configurazione valore.
- **Valuerange:** configurazione valuerange.
- **Method:** configurazione method (valori validi sono average e median).
- **Use:** configurazione use (valori validi sono 0 o 1).
- **Threshold:** configurazione threshold (range tra 0 e 255).

Sezione DEVICE

La sezione device è composta da 9 parametri:

- **Name:** nome periferica.
- **Type:** tipo periferica (configurazione protocollo; valori validi sono LTBL, momo, atmo, popen e sound. Nel nostro caso utilizzeremo Momo).
- **Output:** se la periferica ha il proprio output, questo parametro indica il tipo.
- **Rate:** baudrate della porta seriale.
- **Channels:** numero di canali

di ciascun canale RGB completo (1 byte per il singolo canale Red; 1 byte per il canale Green e 1 byte per il Blue). Come convenzione è stato definito che un byte di valore 0x00h indica "luce canale completamente spenta" (luminosità azzerata); il valore 0xFF indica "luce completamente accesa" (luminosità massima) mentre tutti gli altri indicano i relativi valori intermedi. La configurazione della porta seriale virtuale utilizzata è quella standard a 9600 come baud rate, nessun controllo di flusso, 8 bit di dati, 1 bit di stop e nessuna parità.

Per la corretta esecuzione è necessario lanciare entrambi gli eseguibili; questi verranno ridotti a icona nella Traybar di Windows e si metteranno in esecuzione in background. Facciamo notare che le nostre prove sono state eseguite utilizzando Windows XP con cui abbiamo verificato il corretto funzionamento.

HARDWARE SCHEDA VIDEOLIGHT

Dopo aver predisposto il PC all'analisi delle immagini visualizzate e all'invio delle impostazioni dei valori RGB, passiamo a vedere come è composta e come funziona la scheda hardware di gestione delle strisce LED RGB. Come già detto, il PC è stato configurato per trasmettere le informazioni tramite protocollo

seriale, pertanto anche la scheda elettronica dovrà essere munita di una porta compatibile (nel nostro schema abbiamo integrato un convertitore usb/seriale). Inoltre, avendo deciso di gestire 4 distinti canali, l'hardware dovrà essere munito di 4 uscite predisposte per il pilotaggio di 4 strisce led RGB. Dal punto di vista elettronico, le strisce LED RGB in commercio compatibili con il nostro hardware possono essere viste come singoli led di colore rosso, verde e blu collegati tra loro con anodo comune (tramite resistenza di limitazione) al positivo della tensione di alimentazione e con il catodo portato in uscita su un connettore. Per realizzare tutti i colori dell'arcobaleno è necessario pilotare ciascun gruppo di LED in PWM, con una frequenza sufficientemente alta per evitare che l'occhio percepisca lo sfarfallio e duty cycle variabile. Se viene acceso un solo LED alla volta, il colore risultante sarà quello del singolo LED; se vengono accesi tutti e 3 il colore risultante sarà il bianco. Infine, per realizzare i diversi colori, si utilizzerà la possibilità che offre il PWM di variare digitalmente l'intensità percepita di ciascun colore primario (R, G e B) in base alla durata dell'impulso generato. Passiamo ora ad analizzare lo schema elettrico del circuito: le strisce LED RGB che

utilizzeremo per questo progetto sono del tipo ad anodo comune; per pilotarle avremo bisogno di un semiconduttore che mandi il catodo di ciascun gruppo di LED a massa in base al segnale PWM. Le strisce, via via che aumentano di lunghezza, possono arrivare a consumare molta corrente e, comunque, non possono essere pilotate direttamente con le sole porte di un microcontrollore. Per questo abbiamo utilizzato dei Mosfet a bassa resistenza per gestire senza problemi strisce anche di una certa lunghezza senza che il componente si scaldi in modo avvertibile. Nella scelta ci siamo concentrati su Mosfet a canale N ed in particolare su dei BUZ11 o equivalenti che, come evidenziato dal datasheet, sono in grado di supportare carichi fino a 30 amper (più che sufficienti dato che 5 metri di striscia RGB consumano al massimo 4,5 ampere).

Nello schema sono presenti ben 12 Mosfet (uno per ciascuno dei 3 colori RGB, per i possibili 4 canali) identificati da Q1 a Q12; ognuno è munito di resistenza di pull-down in modo da evitare false attivazioni all'accensione della scheda ed è collegato ad un'uscita del PIC. Il microcontrollore PIC18F2520 (U2) è il cervello del sistema, colui che riceve i comandi dal PC e pilota opportunamente ciascun Mosfet. Come si vede è presente

della periferica (ogni canale RGB completa conta per 3; singolo canale R, G e B).

- **Interval:** intervallo (espresso in microsecondi) tra una trasmissione e la successiva.
- **Prefix:** caratteri di prefisso del protocollo.
- **Period:** parametro valido per periferiche sound.
- **Buffer:** parametro valido per periferiche sound.

Sezione COLOR

La sezione color è composta da 5 parametri:

- **Name:** nome colore.

- **Rgb:** valore rgb del colore (espresso in numero esadecimale a 6 cifre).
- **Gamma:** livello gamma (valido tra 0.0 e infinito).
- **Adjust:** valore aggiustamento finale (validi valori tra 0.0 e 1.0).
- **Blacklevel:** valore minimo del colore.

Sezione LIGHT

La sezione light è composta da 12 parametri:

- **Name:** nome luce.
- **Color:** colore abbinato alla luce.

- **Interpolation:** specifica se la luce usa l'interpolazione.
- **Hscan:** scanrange orizzontale (espresso in percentuale).
- **Vscan:** scanrange verticale (espresso in percentuale).
- **Proportional:** aggiustamento parametro proporzionale.
- **Saturation:** moltiplicatore parametro saturazione.
- **Value:** moltiplicatore parametro valore.
- **Valuerange:** aggiustamento parametro valuerange.
- **Use:** attivazione/disattivazione luce.
- **Method:** metodo scanning

clients.

- **Threshold:** impostazione threshold (valido da 0 a 255).

Nel pacchetto zippato scaricabile dal sito di Futura Elettronica è già compreso un file di configurazione realizzato e testato ad hoc per la scheda hardware presentata in questo articolo. L'unica modifica necessaria al file è l'impostazione del numero di porta COM che viene creata sul vostro PC al primo collegamento.

un quarzo a 20 MHz con i rispettivi condensatori di stabilizzazione. Come vedremo più avanti, analizzando il software, al PIC non è richiesta una grossa mole di lavoro ed un'elevata complessità di calcolo; è necessario però che l'onda PWM sia la più precisa possibile e con frequenza elevata; è questo il motivo per cui abbiamo scelto un quarzo a 20 MHz. Analizzando lo schema si nota che i canali sinistro e destro vengono gestiti dai pin 2,3, 4, 5, 6 e 7 della porta B; il canale top dai pin A3, A4 e A5 mentre il bottom dai pin C0, C1 e C2. Per l'interfacciamento seriale verso il computer è stato scelto il conosciuto ed affidabile modulo convertitore usb/seriale (disponibile in Futura Elettronica con il Cod. 7100-FT782M) configurato con tensione di funzionamento a 5V.

Infine è presente la sezione di alimentazione sviluppata attorno al regolatore LM7805 (componente U1, utilizzato per regolare i 12V di ingresso in 5V stabilizzati utilizzati come alimentazione dal PIC e componenti TTL). Questa sezione è composta dal plug di alimentazione (PWR), da un diodo per la protezione (D1) da eventuali inversioni di polarità, da alcuni condensatori (C1, C2 e C3) per stabilizzare le tensioni di alimentazioni in caso di picchi elevati di assorbimento ed infine

da un led verde (DL1) di segnalazione di scheda alimentata. Come ultima nota evidenziamo la presenza di una morsettiera a 2 poli (PWR_EXT) che può essere utilizzata se si desidera far funzionare le strisce RGB ad una tensione diversa da quella fornita per l'alimentazione della scheda. Infatti sulle uscite delle strisce RGB viene fornita una tensione di alimentazione VLED che, tramite il ponticello a 3 vie P2, può essere selezionata o tra il +12V fornito come alimentazione o dal polo positivo di PWR_EXT.

SOFTWARE SCHEDA VIDEOLIGHT

Il software della scheda elettronica risulta abbastanza semplice. Le operazioni necessarie sono solo due: ricezione dalla porta seriale dei 12 byte dei colori delle 3 strisce RGB e, in funzione di questi, impostazione dei duty cycle dei PWM delle uscite.

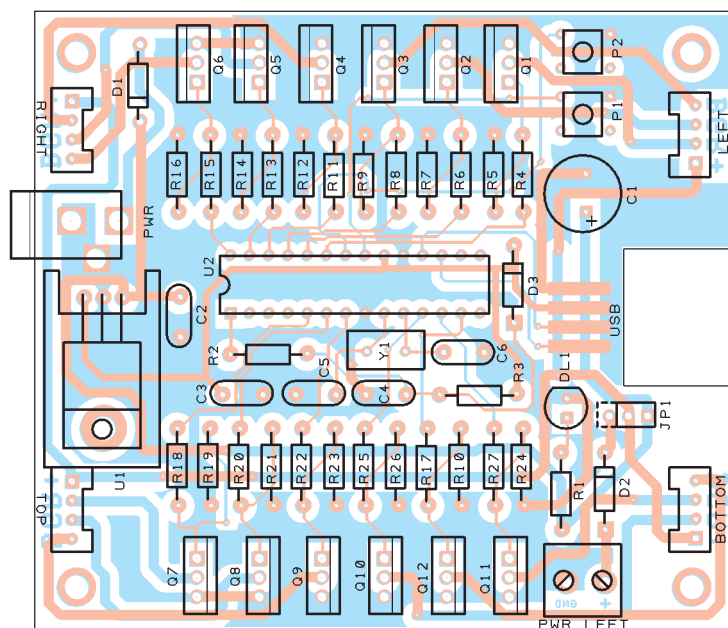
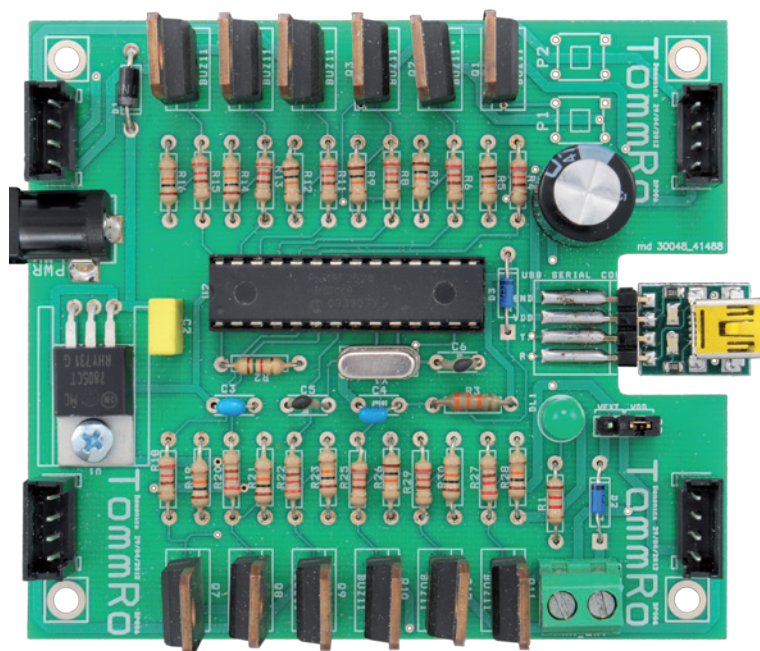
Entrambe le operazioni sono gestite con routine basate sugli interrupt software della MCU. La prima routine è collegata all'interrupt di ricezione di un byte sulla porta seriale e, quando attivata, semplicemente memorizza i byte ricevuti all'interno di un buffer circolare che, nel ciclo principale del programma, viene letto per estrarre, analizzare e memorizzare i byte come duty cycle da impostare alle uscite. La seconda

routine invece viene gestita tramite l'interrupt collegato al timer 2 del PIC; tra le varie esecuzioni degli interrupt le uscite hardware vengono attivate o disattivate a seconda dei byte ricevuti dalla porta seriale in modo da generare i duty cycle richiesti.

REALIZZAZIONE PRATICA

La scheda hardware è stata progettata utilizzando solo componenti di tipo tradizionale e pertanto può essere realizzata anche in proprio disponendo di un saldatore purché abbia la punta ragionevolmente fine. Le linee guide da seguire sono le solite: partire saldando per primi i componenti più piccoli e bassi e in seguito procedere fino ai componenti più alti. Unica nota riguarda il montaggio del convertitore USB/seriale che, come in altri progetti, andrà saldato direttamente sulle piazzole disponibili sullo stampato (è presente anche un taglio dello stampato stesso). Montata la scheda è il momento di passare alle strisce LED RGB. Queste sono solitamente predisposte e marcate per essere tagliate nella lunghezza desiderata; segnaliamo però di prestare la massima attenzione in quanto in commercio ne esistono di diversi tipi, ciascuno con le proprie indicazioni per quanto riguarda il "taglio". Tipicamente viene indicato

[piano di MONTAGGIO]



Elenco Componenti:

C1: 470 μ F 35 VL elettrolitico	R5, R7, R9, R10, R12, R14, R16, R19, R21, R23, R24, R26: 10 kohm	Varie:
C2: 220 nF 100 VL poliestere	P1, P2: Microswitch	- Plug alimentazione
C3, C4: 100 nF multistrato	DL1: LED 5 mm verde	- Morsetto 2 poli
C5, C6: 15 pF ceramico	D1, D2: 1N4007	- Zoccolo 14+14
Y1: Quarzo 20 MHz	D3: SD103	- Strip maschio 3 vie
R1: 1,2 kohm	U1: 7805	- Jumper
R2: 1,5 kohm	U2: PIC18F2520 (MF1067)	- Connettore ad innesto maschio 4 vie da CS (4 pz.)
R3: 33 kohm	Q1÷Q12: P36NF06	- Vite 10 mm 3 MA
R4, R6, R8, R11, R13, R15, R17, R18, R20, R22, R25, R27: 220 ohm	USB: Interfaccia USB/TTL (FT782M)	- Dado 3 MA

un numero che indica ogni quanti led è possibile eseguire il sezionamento (anche sulla serigrafia è aggiunta un'indicazione). Per le nostre prove abbiamo utilizzato strisce che potevano essere sezionate ogni 3 chip carrier, ma raccomandiamo di verificare il vostro caso e prestare la massima attenzione, tagliando in mezzo alle coppie di piazzole verticali e lasciandone una serie in ciascuno spezzone.

Tagliate le strisce, è il momento di predisporle al collegamento verso la scheda hardware; nello specifico andranno collegati i 4 fili che portano i segnali.

Anche in questo caso ogni strip ha le proprie caratteristiche ma solitamente, nei punti in cui è possibile eseguire il taglio, sono presenti delle predisposizioni (piazzole) per la saldatura delle terminazioni dei fili. Nel caso sia presente l'isolamento in silicone trasparente, questo va sollevato e poi tagliato quel tanto che basta a scoprire le piazzole, da ripulire prima della saldatura e comunque da trattare con attenzione in quanto possono danneggiarsi facilmente se si indugia troppo con la punta del saldatore. Per maggior chiarezza suggeriamo di utilizzare fili di colore nero per i positivi di alimentazione (anodo strisce a led) e 3 fili di colore rosso, verde e blu per i catodi dei led RGB.

Altra accortezza che noi stessi abbiamo utilizzato è, dopo la saldatura, di isolare e proteggere il capo della striscia utilizzando un pezzetto di guaina termorestringente (oppure, nel caso non si disponga di tale attrezzatura, è possibile farlo tramite nastro isolante da elettricista). Dopo aver saldato i fili, dall'altro lato del cavetto andranno montati i connettori equivalenti a quelli montati sulla scheda. Anche

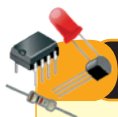
per questa operazione esistono attrezzature specifiche; se non disponibili suggeriamo di saldare i fili direttamente alla scheda.

COLLAUDO VIDEOLIGHT

Terminato il montaggio dell'hardware possiamo passare al test del sistema e realizzare il collegamento con il PC.

La scheda elettronica dispone di: un connettore plug di alimentazione a cui andrà collegato l'alimentatore; 4 connettori da utilizzare per il collegamento alle strisce LED RGB; il connettore USB del relativo convertitore (questo andrà utilizzato per la connessione al PC tramite cavo USB). Sotto ai connettori delle strisce viene indicata la relativa piedinatura e il simbolo + indica la tensione positiva di alimentazione e in sequenza i simboli R, G e B per le uscite dei Mosfet dei canali red, green e blue.

Come tutti i convertitori USB, anche questo richiede l'installazione su computer del relativo driver software che, però, è lo stesso utilizzato per la scheda Arduino



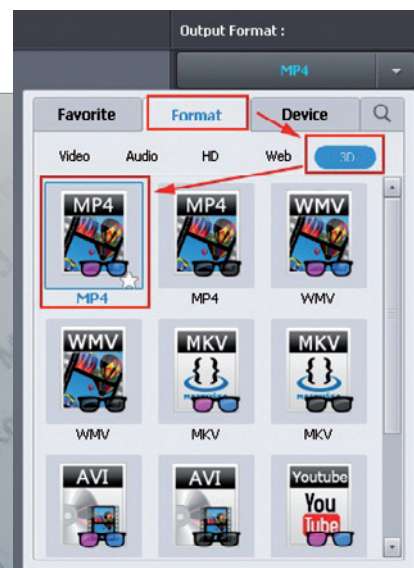
per il MATERIALE

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 software ed il firmware utilizzato per programmare il microcontrollore PIC18F2520. L'interfaccia USB/TTL FT782M può essere acquistata già montata presso Futura Elettronica al prezzo di 15,00 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>

Il 3D anche sul computer

Il nostro progetto prevede che il video sia riprodotto su PC per poter usufruire delle routine di analisi dell'immagine del software Boblight. Se siete un fortunato possessore di schermo compatibile con i formati 3D, sappiate che il passaggio da PC non vi costringe a rinunciare a questa caratteristica dato che tutti i player più recenti sono stati aggiornati per supportare gli attuali formati 3D. Oltre ai due formati più semplici, con ogni frame suddivisa nelle due immagini affiancate o sovrapposte, ci sono anche i formati in cui il contenuto è codificato a risoluzione intera e intercalato alternando i frame sinistro e destro. I BluRay 3D offrono al momento il massimo della qualità - con la modalità FramePacking - in quanto il frame è full HD (1920 x 1080p) e il frame rate è raddoppiato; un particolare pacchetto di linee nere fra i due frame permette al televisore di



riconoscere questo formato e attivare la propria modalità di visualizzazione 3D. La banda richiesta per questo tipo di formato è il doppio rispetto al normale HD e per questo chi trasmette via satellite o in Digitale Terrestre non può ancora offrire questo livello qualitativo, mentre attraverso una soluzione che comprime ciascun frame attraverso una particolare soluzione di sub campionamento si riesce ad avere una qualità ragionevole senza incidere troppo sulla quantità di dati da inviare per avere il 3D in alta definizione.

(pertanto se avete già lavorato con Arduino di questo aspetto potete non preoccuparvi).

Terminato il collegamento, alimentate la scheda, quindi avviate in sequenza il software BobLight (prima l'eseguibile boblightd e in seguito boblight-getpixel) e infine la riproduzione video sul PC. L'operazione va eseguita in questa sequenza (prima accendere la scheda e dopo avviare i software per PC) in quanto il protocollo non prevede campi di sincronizzazione e pertanto, prima che vengano inviati dati sulla porta seriale, è necessario che l'hardware sia già pronto a riceverli.

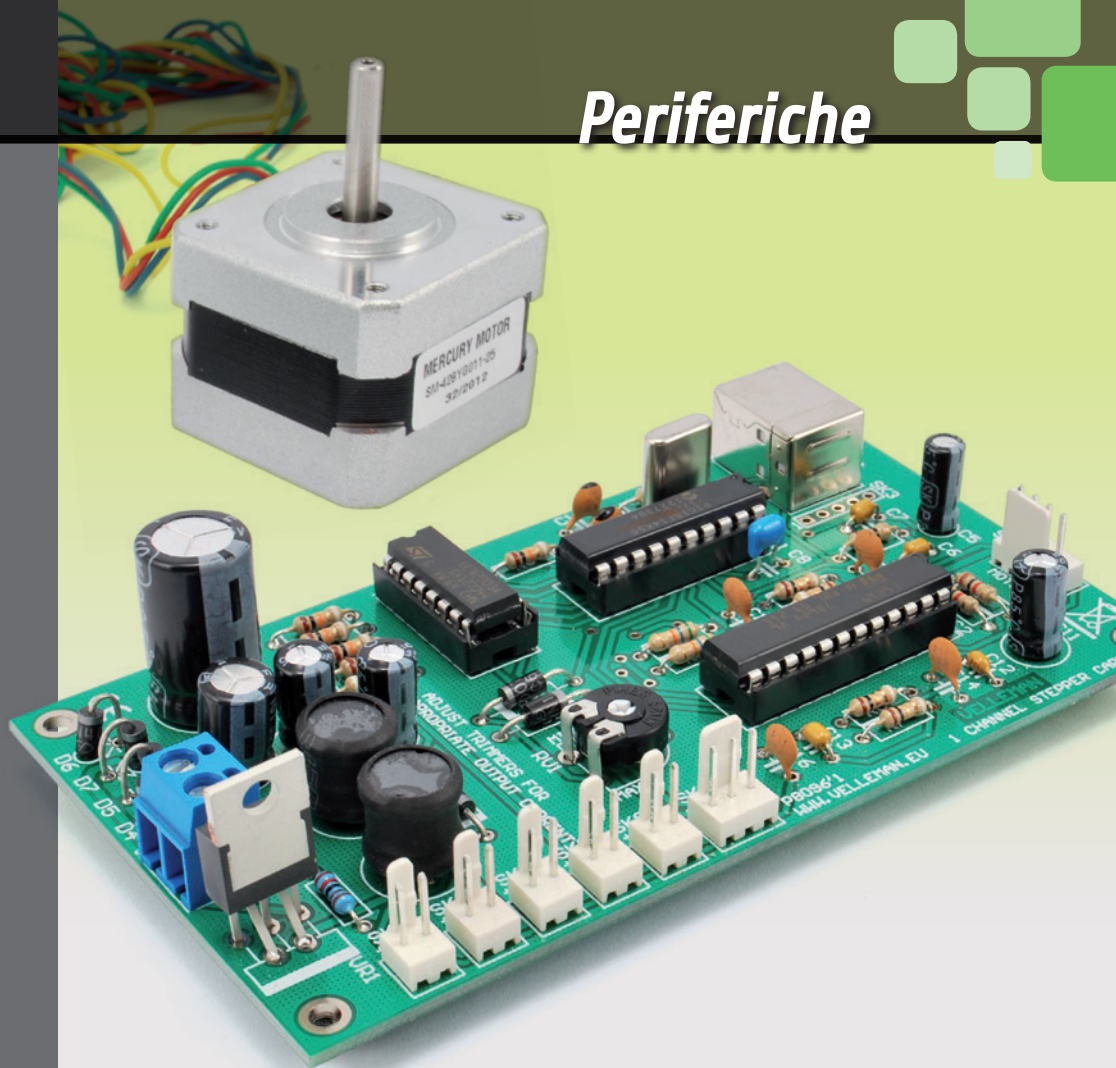
Se qualcosa non dovesse funzionare, è possibile eseguire due ulteriori verifiche più semplici dell'hardware montato. La scheda si aspetta di ricevere le informazioni in sequenza ciclica e secondo un protocollo definito (riferirsi al box Protocollo di comunicazione).

Come primo collaudo è possibile

aprire sul PC un software emulatore seriale (per esempio HyperTerminal), impostare la corretta configurazione della porta seriale virtuale e provare ad inviare in sequenza dei byte. In corrispondenza dei byte inviati le luci esterne dovrebbero cambiare secondo il relativo protocollo (attenzione: i byte vanno inviati come numeri puri e non come caratteri ASCII).

In seguito come secondo collaudo, dopo aver avviato i 2 software di boblight, è possibile aprire Paint di Windows e disegnare diversi rettangoli dei colori base nelle 4 posizioni dei canali video e verificare che i led RGB cambino correttamente colore. In chiusura di articolo facciamo notare che, nel normale utilizzo del PC, il software Boblight è perfettamente funzionante e tarato per l'analisi video; se si utilizza il PC per vedere dei film, in alcune scene può invece succedere che l'analisi dei colori tenda troppo verso le varie intensità del bianco e non dei colori.

Controlliamo da qualsiasi computer dotato di presa USB un motore passo-passo bipolare, oppure attiviamo un utilizzatore generico in modo on/off e leggiamo lo stato di 5 ingressi normalmente aperti.

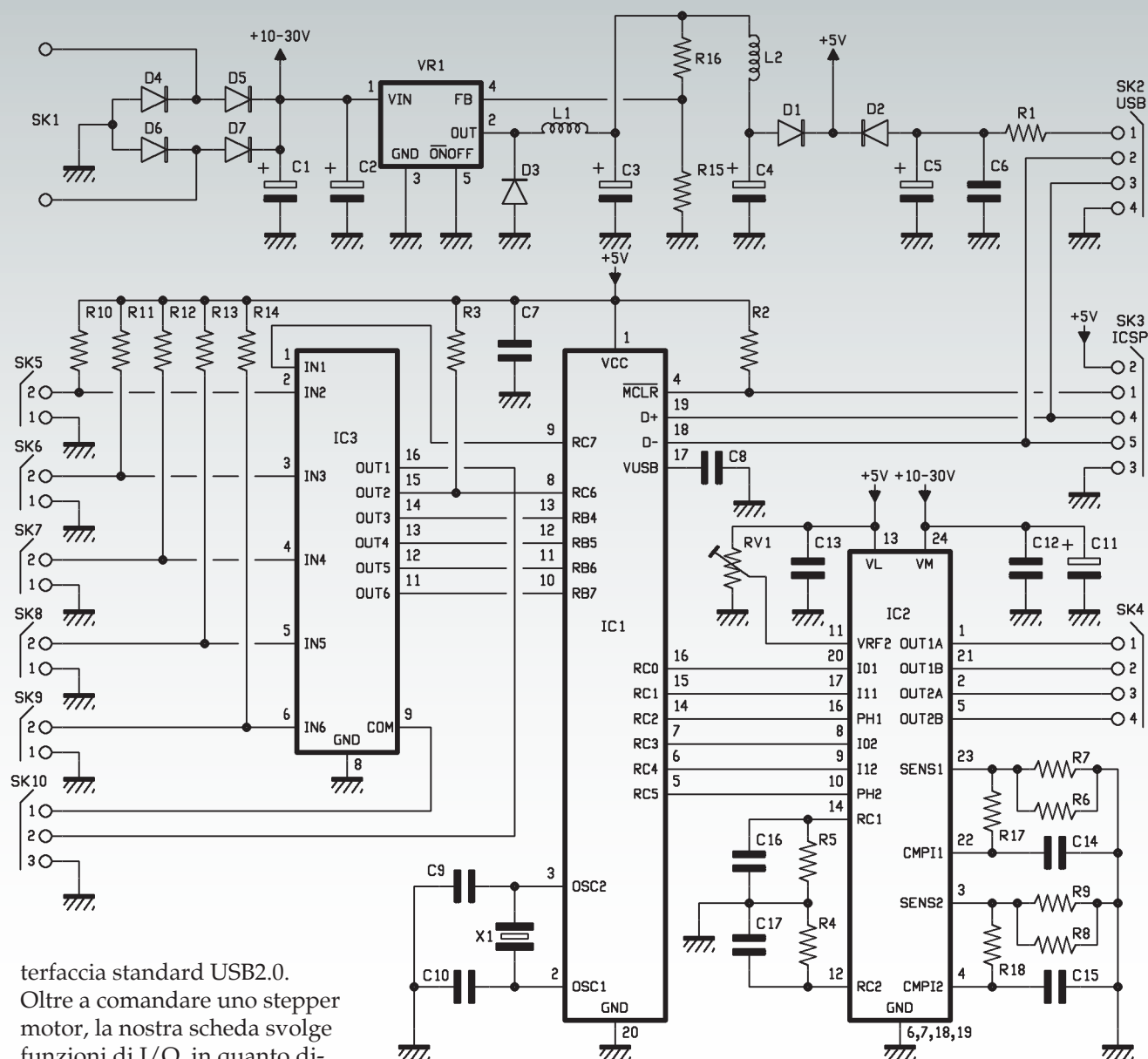


SCHEDA USB PER STEPPER MOTOR

di DAVIDE SCULLINO

I motori passo-passo (anche detti stepper motor) sono particolari motori elettrici rotativi che, diversamente da quelli tradizionali, presentano la caratteristica di ruotare a passi di ampiezza (espressa in gradi sessagesimali) ben precisa e di poter fermare il proprio albero in posizioni determinabili con esattezza, grazie al fatto che dispongono di coppie di avvolgimenti alimentando i quali si può determinare la rotazione in un senso o nell'altro, ovvero fermare l'albero. Per queste loro qualità, sono molto utilizzati negli azionamenti, nelle stampanti, in svariati macchinari (ad esempio macchine CNC) e - ormai lo sapete se avete seguito il nostro

progetto della 3Drag - nelle stampanti 3D. Controllare gli stepper motor è quindi molto utile ed è una delle cose che ci viene richiesta sovente, soprattutto da Personal Computer; per questa ragione abbiamo pensato di proporvi il progetto di una scheda controller interfacciata via USB, in grado di gestire da PC, tramite un apposito software scaricabile dal nostro sito www.elettronica.in.it, un motore passo-passo bipolare funzionante da 12 a 24 V, chiaramente in continua. Si tratta di un circuito molto compatto collocabile all'interno dell'apparato che contiene il motore passo-passo, interfacciabile con un comune cavo USB perché dispone di un'in-



terfaccia standard USB2.0. Oltre a comandare uno stepper motor, la nostra scheda svolge funzioni di I/O, in quanto dispone di un'uscita open-collector capace di erogare fino a 500 mA e di cinque input digitali di cui può acquisire lo stato logico; in realtà queste linee sono progettate per leggere l'apertura e la chiusura di un contatto o di un transistor, ma è possibile anche comandarle in tensione purché con livelli 0/5 volt. L'unità è stata dunque pensata per controllare da computer una macchina in maniera completa, ovvero gestirne un motore passo-passo e un attuatore (che può essere anche un semplice LED) oltre che leggere lo stato di finecorsa, switch di vario genere, sensori

con uscita TTL, pulsanti per l'arresto di emergenza ecc. Diamo subito uno sguardo al circuito studiandone lo schema elettrico, che ce ne mostra la struttura e la composizione.

SCHEMA ELETTRICO

La scheda di controllo per lo stepper-motor è formata essenzialmente da un microcontrollore dotato di interfaccia USB e da un driver con due ponti ad H per il controllo di motori tradizionali o passo-passo. Completano l'insieme un line-driver ULN2003 che serve a far leggere al microcon-

trollare i cinque ingressi (oltre che a comandare l'uscita on/off) ed uno stadio di alimentazione che permette di ricavare la tensione stabilizzata occorrente al microcontrollore e all'ULN2003 e quella non stabilizzata che fa funzionare il driver per il motore. Il microcontrollore è un PIC18F14K50 Microchip capace di elaborare a 12 MIPS, con architettura ad 8 bit, dotato di memoria di programma di tipo Flash da 16 kB, RAM da 768 byte, EEPROM dati da 256 byte, quattro timer (uno da 8 bit e 3 da 16 bit), un ADC 10 bit a 9 canali

ed un modulo PWM associabile a 5 I/O; ma la caratteristica più importante di questo micro è che integra varie periferiche di comunicazione: oltre ad un USART e ad una SPI/I2C-Bus, dispone di una periferica USB Full Speed compatibile USB 2.0.

Il PIC18F14K50 può lavorare con un clock interno fino a 48 MHz (la frequenza base può essere impostata tra 31 kHz e 16MHz) ovvero con uno esterno, moltiplicato attraverso un PLL 4x; nel nostro circuito il clock è definito dal quarzo da 12 MHz il quale, con l'aiuto dei condensatori C9 e C10, scandisce il funzionamento dell'oscillatore interno la cui frequenza base è 12 MHz e viene moltiplicata per quattro.

Il microcontrollore è programmato per svolgere essenzialmente due compiti: il primo è dialogare con il computer mediante la porta USB e interpretare i comandi corrispondenti; il secondo è controllare il driver per stepper motor.

Quest'ultimo è un L6219 della ST ed è un driver monolitico (incapsulato in contenitore DIP plastico) che contiene due ponti ad H, adatto perciò ai motori passo-passo bipolari, ma anche utilizzabile per il controllo bidirezionale di motori in continua tradizionali. Il componente è progettato per essere gestito tramite linee TTL-compatibili, ossia con tensioni 0/5 V, il che ci permette di controllarlo direttamente con gli I/O del microcontrollore. Ciascuno dei ponti può sopportare tensioni continue fino a 46V e integra quattro diodi di protezione dei transistor di commutazione, utili a sopprimere le correnti inverse che si generano negli avvolgimenti all'interruzione dell'alimentazione. Un'ulteriore protezione di cui l'integrato è provvisto è quella che previene

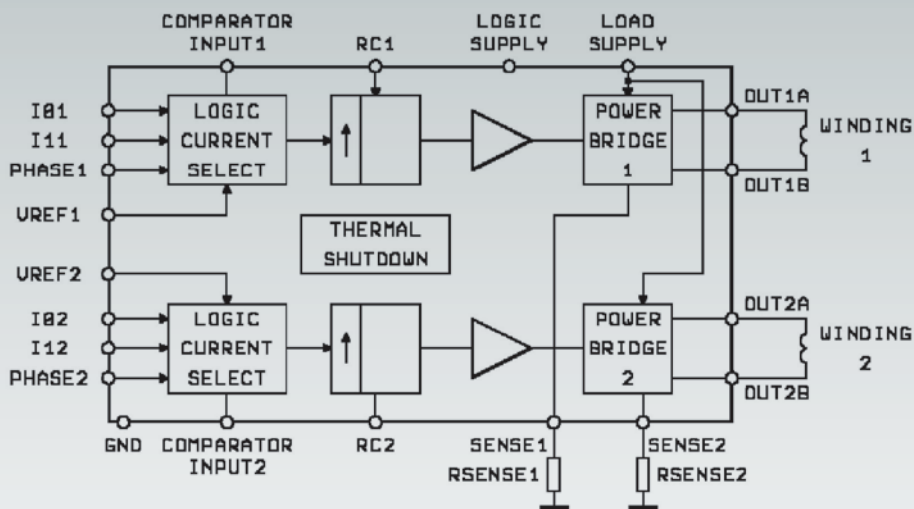
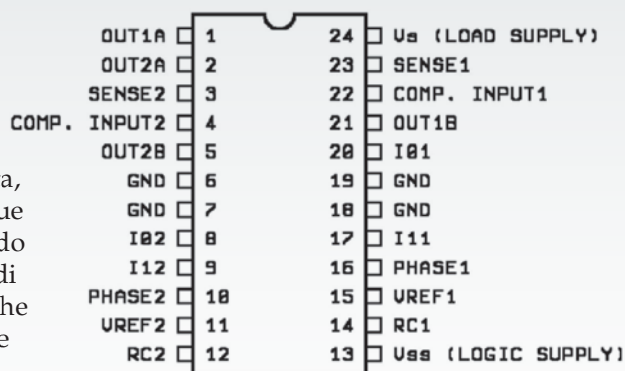


Fig. 1 - Schema a blocchi e piedinatura dell'integrato L6219.

la conduzione incrociata, ossia simultanea, dei due lati di ogni ponte quando si verifica l'inversione di direzione del motore (che comporta un'inversione della corrente erogata); ciò evita che un ponte si chiuda sull'altro, con ovvie conseguenze. A completare il set di protezioni c'è quella termica, che interdice gli stadi di potenza (più esattamente, interdice i transistor superiori dei ponti) nel caso la temperatura del chip ecceda il margine operativo (Fig. 1). Ciascun ponte è formato da transistor BJT e può erogare impulsi di corrente da 750 mA, con picco all'avvio fino ad 1 A. I ponti dell'integrato vengono pilotati mediante gli impulsi prodotti da un modulatore PWM (Pulse Width Modulation) interno che opera a frequenza fissa e che a sua volta pilota i comparatori utilizzati per impostare la corrente di uscita e la protezione in corrente che il componente prevede. Attraverso il piedino PHASE corrispondente (PHASE1 vale per il ponte 1 e PHASE2 per il ponte 2) si imposta il verso di rotazione del motore, ovvero la



polarità degli impulsi alle metà di ciascun ponte ad H. I piedini I01 e I11, nonché I02 e I12, permettono di definire la soglia di corrente erogabile dai transistor del ponte, secondo la **Tabella 1**, dalla quale si vede che quando entrambi gli ingressi di controllo sono a livello alto il ponte si disattiva; ciò viene ottenuto interdicendo i transistor superiori del ponte corrispondente. Con I0 a livello basso e I1 allo stato logico alto, il ponte eroga 1/3 della corrente massima, mentre con l'impostazione contraria (1 0) la corrente erogata raggiunge i 2/3 del massimo; con entrambi i piedini a livello basso,

Tabella 1

I0	I1	Current level
H	H	No current
L	H	Low current 1/3 IO max
H	L	Medium current 2/3 IO max
L	L	Maximum current IO max

CARATTERISTICHE TECNICHE

- Tensione di alimentazione c.a.: 9÷24Veff
- Tensione di alimentazione c.c.: 10÷30V
- Corrente assorbita: max 2A
- Corrente continuativamente erogata al motore: 2 x 750mA
- Corrente di picco nel motore: 1A
- Protezione da sovracorrente
- Protezione termica
- 5 ingressi per contatti
- 1 uscita open collector 500mA

si abilita il ponte ad erogare il massimo della corrente permessa. La limitazione della corrente viene ottenuta intervenendo sulla soglia di commutazione del comparatore del modulo PWM, ovvero facendo interdire i transistor prima o dopo a seconda del valore di corrente desiderato. Sullo stesso comparatore interviene la protezione in corrente, realizzata rilevando la corrente erogata da ciascun ponte (il rilevamento si effettua sull'emettitore) mediante le resistenze collegate tra i piedini SENSE1 (per il ponte 1) e SENSE2 (per il ponte 2) e retrocedendone parte agli ingressi COMPARATOR INPUT 1 e COMPARATOR INPUT 2: quando la tensione determinata dalla corrente di uscita ai capi delle predette resistenze supera la soglia impostata, il comparatore si blocca e spegne i transistor. L'intervento è sui soliti transistor superiori del ponte, che vanno in interdizione in un tempo determinato dalle reti RC collegate ai piedini RC1 (ponte 1) ed RC2 (ponte 2). La soglia di spegnimento corrisponde a circa 0,5 V sui piedini SENSE. La regolazione del valore massimo di corrente ammessa all'uscita si fa con il trimmer il

cui cursore è collegato ai piedini VREF dei due ponti; la tensione fornita da esso viene confrontata nel comparatore e stabilisce la massima soglia, ovvero il valore massimo corrispondente a quando i piedini I01-I11 e I02-I12 si trovano a livello basso.

Ruotando il cursore del trimmer verso il positivo di alimentazione si aumenta la Vref e di conseguenza si abbassa la massima corrente erogata dai ponti, mentre portando il cursore verso massa la corrente cresce.

In linea generale, all'aumentare della tensione Vref e di quella sui SENSE si abbassa la corrente erogata e viceversa.

Quanto all'ULN2003, serve al microcontrollore da interfaccia per acquisire lo stato di cinque linee d'uscita a transistor open collector o interruttori a contatto pulito (ma anche di linee digitali TTL 0/5V) ed anche per comandare un utilizzatore che assorba fino a 500 milliampere e funzioni con una tensione continua di valore compreso tra 5 e 15 volt, quindi anche un semplice relé per alimentare la macchina in cui è inserito lo stepper motor o un LED utilizzato per segnalare localmente un avvenimento. L'ULN2003 è un line-driver a 7 canali composti ciascuno da un Darlington NPN configurato ad open collector: gli ingressi, TTL-compatibili, facenti capo alle basi (in serie a ciascuna si trova un resistore di limitazione della corrente) sono localizzati ai piedini 1 (IN1) 2 (IN2) 3 (IN3) 4 (IN4) 5 (IN5) 6 (IN6) e 7 (IN7) mentre le corrispondenti uscite sono sui pin 16 (OUT1) 15 (OUT2) 14 (OUT3) 13 (OUT4) 12 (OUT5) 11 (OUT6) e 10 (OUT7). Ogni uscita fa capo al collettore di un Darlington ed è protetta da un diodo (connesso con l'anodo al collettore) integrato nel chip che serve

nel caso si debbano controllare carichi induttivi (come ad esempio i relé) mentre gli emettitori sono tutti in comune, collegati al piedino GND.

Quanto ai diodi di protezione, i loro catodi sono uniti e collegati internamente al piedino COM D. Nel nostro circuito usiamo i line-driver dal 2 al 6 per leggere lo stato di altrettanti ingressi; ogni input dell'ULN2003 è provvisto di resistore di pull-up in modo da mantenere, a riposo (cioè ad ingresso aperto) il relativo Darlington in saturazione e quindi la corrispondente uscita a zero logico. Le uscite dei line-driver 2÷6 vengono lette dalle linee RB4, RB5, RB6, RB7 ed RC6, le prime delle quali dispongono del pull-up interno, mentre RC6 richiede il resistore di pull-up esterno, che nello schema elettrico è siglato R3. Il primo driver è invece comandato dal microcontrollore mediante la linea RC7 e la sua uscita è disponibile su un'apposita morsettiera, alla quale si deve collegare sul morsetto 1 lo stesso positivo che alimenta il carico alimentato dall'uscita stessa (OUT1); notate che in questo caso il potenziale applicato a COM D è irrilevante per le uscite dei restanti Darlington, dato che fornendo almeno 5 volt i diodi di protezione interni all'ULN2003 vengono polarizzati inversamente e comunque non conducono, né influenzano il funzionamento degli OUT2, OUT3 ecc.

Quanto al settimo ed ultimo driver dell'ULN2003, in questa applicazione non viene utilizzato. Stabilito ciò, possiamo analizzare il funzionamento dell'intero circuito, ossia spiegare come lavora il microcontrollore: dopo il power-on-reset, il firmware che gira in esso inizializza gli I/O impostando RC0÷RC5 come uscite dedicate al controllo dell'L6219,

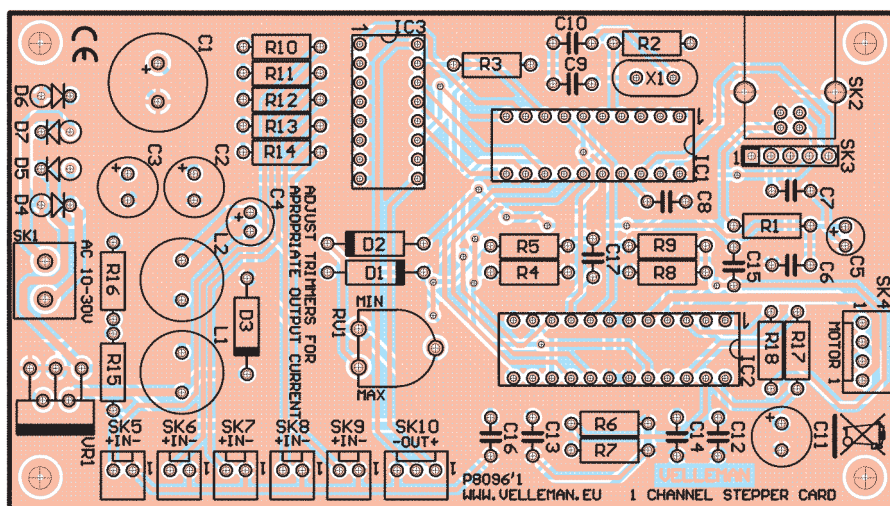
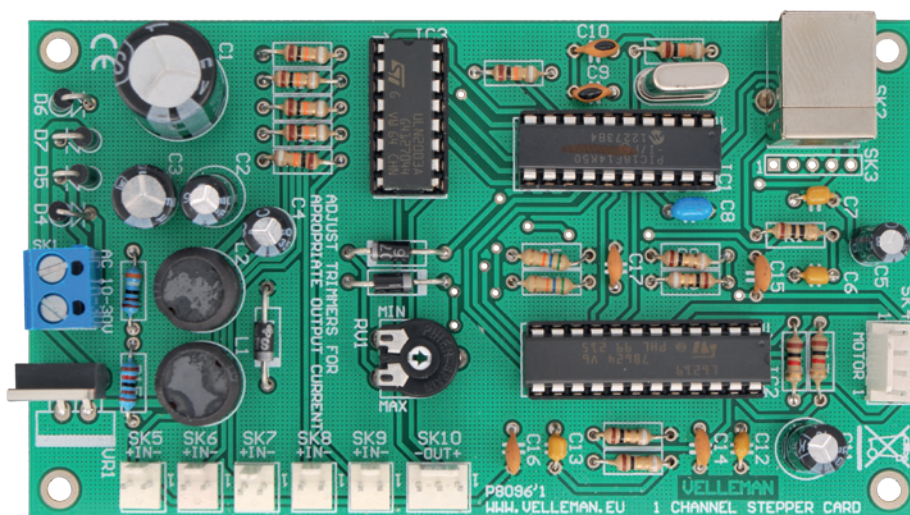
[piano di MONTAGGIO]

Elenco Componenti:

R1 ÷ R3: 10 ohm
 R4; R5: 56 kohm
 R6 ÷ R9: 1 ohm
 R10 ÷ R14: 10 kohm
 R15: 1,2 kohm
 R16: 3,9 kohm
 R17 ; R18: 1 kohm
 RV1: Trimmer 10 kohm MO
 C1: 1000 μ F 25 VL elettrolitico
 C2: 100 μ F 25 VL elettrolitico
 C3: 330 μ F 25 VL elettrolitico
 C4: 100 μ F 25 VL elettrolitico
 C5: 4,7 μ F 63 VL elettrolitico
 C6; C7: 100 nF ceramico
 C8: 470 nF ceramico
 C9; C10: 22 pF ceramico
 C11; C13: 100 μ F 25 VL elettrolitico
 C14; C17: 820 pF ceramico
 D1; D2: 1N4007
 D3: SB130
 D4; D7: 1N4007
 X1: Quarzo 12 MHz
 VR1: LM2575TADJ
 L1; L2: Bobina 330 μ H
 IC1: PIC18F14K50-I/P (VK8096)
 IC2: L6219
 IC3: ULN2003

Varie:

- Morsetto 2 poli
- Zoccolo 8+8
- Zoccolo 10+10
- Zoccolo 12+12
- Connettore USB-B da CS
- Connettore ad innesto 2 vie (5 pz.)
- Connettore ad innesto 3 vie (1 pz.)
- Connettore ad innesto 4 vie (1 pz.)
- Piedini adesivi (5 pz.)
- Circuito stampato



RC6 come ingresso senza pull-up (su questo I/O non è previsto il pull-up interno) per la lettura dell'OUT2 dell'ULN2003, RC7 come uscita per comandare l'IN1 dell'ULN2003; le linee RB4÷RB7 vengono anch'esse inizializzate come ingressi, ma provvisti di pull-up interno e servono a leggere lo stato dei line-driver 2, 3, 4, 5, 6 dell'ULN2003.

Quanto alle linee dedicate al controllo del motore passo-

passo, RC0 ed RC1 impostano la corrente limite nel motore o il blocco del driver del primo ponte, mentre RCRC3 ed RC4 fanno altrettanto con il secondo ponte; invece RC2 fornisce il segnale che imposta la polarità degli impulsi di corrente forniti dal ponte 1 ed RC5 l'analogo segnale che definisce il verso della corrente fornita dal ponte 2.

Le altre linee del microcontrollore sono dedicate all'interfaccia

USB (RA1/D-/PGC è il canale dati D-, mentre RA2/D+/PGD è il canale D+) ed alla programmazione in-circuit (RA3/MCLR/VPP); notate che RA1 ed RA2, che nel normale utilizzo gestiscono il traffico dei dati sull'USB, in programmazione servono a programmare il micro (mediante il connettore ICSP).

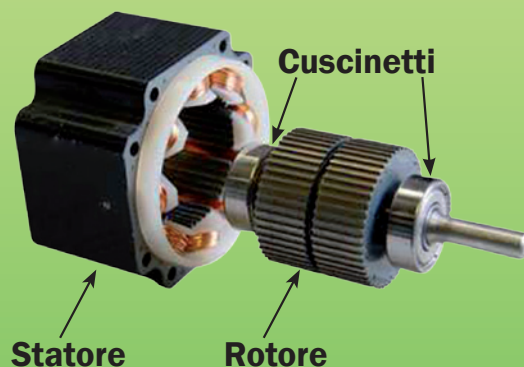
Una volta completata l'inizializzazione degli I/O, il firmware entra nel loop principale atten-

Il motore passo-passo

Questa macchina elettrica è un particolare motore in continua nel quale il rotore, in presenza di alimentazione, non gira linearmente ma può essere comandato a scatti e fermato in una determinata posizione angolare. Lo stepper motor ha sullo statore quattro avvolgimenti realizzati su due nuclei con le relative espansioni polari (queste sono tipicamente 4 oppure 8), ciascuno dei quali ha una coppia di terminali o fili prelevabili dall'esterno della carcassa; il rotore gira generalmente su cuscinetti ed è costituito da un magnete permanente dotato di tante espansioni polari quante sono quelle dello statore, posto in movimento dal variare del verso della corrente negli avvolgimenti dello statore. All'interno dello statore sono presenti piccoli denti che si affacciano su quelli del rotore; in realtà sono esattamente affacciati al rotore solo i denti appartenenti ad una espansione polare e a quella opposta; le altre coppie sono sfalsate rispettivamente di $1/4$, $1/2$ e $3/4$ del passo dei denti. Il rotore viene magnetizzato quando il motore è già assemblato, cioè con il rotore alloggiato all'interno dello statore; a riposo, il flusso

magnetico generato dal rotore si richiude attraverso il materiale ferromagnetico da cui è composto lo statore. Se si estrae il rotore dalla carcassa, viene a mancare il circuito magnetico creato con lo statore e perciò il rotore si auto-smagnetizza.

Gli elementi caratteristici di un motore passo-passo sono il numero di passi compiuti in un giro, ovvero i gradi per passo, la configurazione degli avvolgimenti, e naturalmente la tensione e la corrente assorbite da ciascun solenoide o da tutti nella modalità indicata. Riguardo alla configurazione, la distinzione si fa in base al collegamento dei capi ed in relazione al flusso della corrente negli avvolgimenti: il motore funziona in modo bipolare quando essi sono connessi in serie a due a due in modo che vi scorra corrente elettrica nei due versi, mentre si ottiene la modalità unipolare quando le bobine sono percorse in un solo verso. Nel primo caso il pilotaggio si effettua tramite un circuito a ponte, capace di invertire la polarità tra i capi delle serie, mentre nel secondo è sufficiente uno stadio push-pull, composto ad esempio da transistor NPN con gli emettitori a massa ed i collettori



liberi, ponendo il centrale di ogni solenoide del motore al positivo d'alimentazione.

Il circuito che deve controllare lo stepper motor dipende dalla configurazione voluta e dal numero di avvolgimenti: se ciascuna bobina ha due capi, escono 8 fili, che devono ridursi a 4 nella modalità bipolare e a 6 in quella unipolare: nel primo modo basta unire i capi interni e lasciarli scollegati dal resto della circuiteria, portando al driver solamente gli estremi del primo avvolgimento, del secondo, del terzo e del quarto, mentre nella seconda configurazione occorre sempre unire i fili centrali, quindi portarli oltre ai 4 estremi verso il driver push-pull, e si ottengono perciò sei cavetti. Il motore passo-passo a 6 o 8 fili può funzionare in configurazione sia unipolare che bipolare.

Il rotore del motore passo-passo si presenta come una coppia di ruote dentate affiancate e solidali

dendo l'arrivo di dati dal computer via USB e quando li riceve, li elabora per eseguire le richieste, ovvero acquisire lo stato degli input, comandare l'uscita open collector e, naturalmente, far ruotare o fermare il motore passo-passo. Ciò che viene fatto dipende dal software che gira nel PC e che descriveremo tra breve, dopo aver spiegato il funzionamento dell'ultimo stadio del circuito: quello di alimentazione. L'intera scheda richiede una tensione d'alimentazione continua o alternata: nel caso della continua,

servono da 11 a 30 volt, mentre applicando l'alternata occorrono da 9 a 24 Veff. L'alimentatore della scheda è composto da due stadi: un ponte di Graetz, che raddrizza la tensione d'ingresso, ottenendo impulsi che, filtrati dal C37 e dai condensatori elettrolitici posti a valle, sviluppano una componente continua che fa funzionare il driver per gli stepper motor; un regolatore DC/DC switching il cui ingresso è alimentato dalla tensione presente a valle del solito ponte a diodi e che serve a fornire i 5 Vcc

stabilizzati alla logica. Notate che alimentando la scheda in continua, il ponte raddrizzatore consente di applicare la tensione senza riguardo per la polarità, dato che il ponte raddrizzatore presenta la stessa polarità tra i punti di uscita + e - indipendentemente da qual è la polarità all'ingresso (punti ~). Lo stadio DC/DC impiega il regolatore switching LM2575-ADJ di National Semiconductors per ricavare circa 5,3 volt stabilizzati, erogando una corrente che può raggiungere 4 ampere; il pregio

all'albero costituite da un nucleo magnetico (le due ruote sono permanentemente magnetizzate, una con polarità Nord, l'altra con Sud) mentre i denti, che costituiscono le espansioni polari, sono di materiale ferromagnetico. Il numero di denti dipende dal numero di passi/giro che il motore deve compiere. La dentatura delle due ruote non è allineata ma sfasata di 1/2 del passo dei denti: il dente di una delle due ruote corrisponde quindi alla gola dell'altra.

Per far girare il motore occorre pilotare ad impulsi i vari capi in modo da far spostare la polarità del campo elettromagnetico dello statore in senso orario o antiorario, mentre per bloccarlo è sufficiente mantenere la polarità nel momento in cui va fermato. Nei motori unipolari, che hanno quattro bobine, la sequenza di alimentazione è 1, 2, 3, 4 per un verso di rotazione e 4, 3, 2, 1 per quello opposto; la corrente nelle singole bobine deve circolare sempre nello stesso senso. Invece nei motori bipolari, che hanno due bobine, la sequenza di comando è 1-2, 1-2 (invertendo la polarità) per un verso e 2-1, 2-1 (invertendo la polarità, per quello opposto).

di questo circuito integrato è che funziona richiedendo solo pochi componenti passivi tra cui una piccola induttanza e il tutto occupando una superficie grande come quella di un normale 78xx dotato di un modesto dissipatore. Un'occhiata allo schema elettrico conferma la semplicità dell'insieme: troviamo l'M2575-ADJ montato nella classica configurazione, che riceve dal ponte a diodi la tensione non regolata, filtrata dall'elettrolitico C2; i piedini 3 e 5 sono collegati a massa, come prescritto nelle note

applicative date dal costruttore, mentre il 4 riceve la tensione riportata dall'uscita attraverso la rete di retroazione serve a fissare il valore della tensione erogata dal componente. Infine, il piedino 2 fa capo al transistor di uscita dell'integrato, che fornisce impulsi di corrente all'induttanza L1 intervallati da pause nelle quali si interdice; durante tali pause, L1 restituisce l'energia immagazzinata forzando un flusso di corrente nel condensatore C3, che pertanto viene caricato e assume una tensione continua di 5,3 volt. La funzione del diodo veloce D3 è impedire che, durante le pause, l'extratensione inversa dovuta alla scarica dell'induttore L1 danneggi lo stadio finale dell'M2575-ADJ; ma non solo, perché l'SB130, proprio andando in conduzione in tali fasi, consente di chiudere il circuito della bobina su C3, consentendone la carica che altrimenti sarebbe impossibile.

L'integrato LM2575-ADJ può assumere varie tensioni d'uscita, in funzione sia della differenza di potenziale applicata all'ingresso (che può spaziare tra 5 e 40 volt) sia della componente retrocessa dalla rete di retroazione. Nel nostro circuito, la tensione fornita all'uscita (V_{out}) si ricava dalla formula:

$$V_{out} = V_{ref} (1 + R_{16}/R_{15})$$

dove V_{ref} è la tensione del riferimento interno e vale 1,23 V. L'induttore si calcola in due fasi; innanzitutto ricavando il parametro ET ($V \times ms.$) dalla formula:

$$ET = (V_{in} - V_{out}) \frac{1.000 \times V_{out}}{V_{in} \times f}$$

dove f è la frequenza di lavoro, che nell'integrato è fissa a 52 kHz.

Affinché la costante ET risulti in volt per ms, la frequenza f deve essere espressa in kHz. Ottenuta ET, dal grafico fornito dal costruttore si trova il valore di induttanza più adatto (Fig. 2). Una volta scelto il valore dell'induttanza, con la seguente formula si trova quello del condensatore d'uscita (C_{out} , che in questo caso è C3):

$$C3 \geq 13.300 \times \frac{V_{in}}{V_{out} \times L}$$

In essa L è l'induttanza, espressa in mH per avere il valore del condensatore in microfarad. La componente continua presente ai capi dell'elettrolitico C3 viene filtrata dalla cella LC formata dall'induttore L2 e dal condensatore C4, il cui scopo è sopprimere eventuali residui di commutazione in modo da fornire al PIC una tensione pulita e priva di disturbi che potrebbero alterarne il funzionamento. Il diodo D1 porta la tensione ricavata dall'M2575-ADJ alla linea di alimentazione a 5 V per il microcontrollore e la logica ed

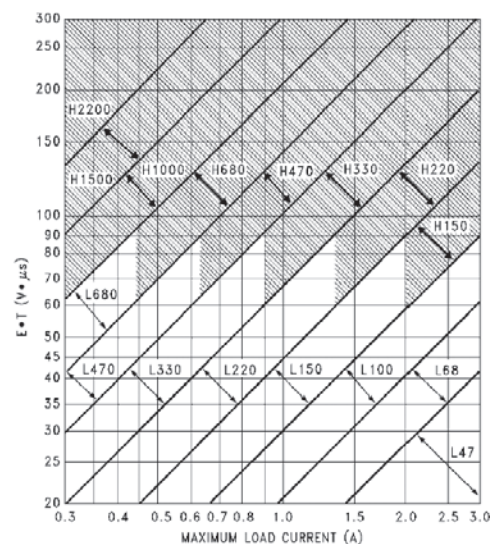


Fig. 2 – Grafico per la determinazione del valore dell'induttanza dell'M2575.

è stato inserito perché il circuito prevede la possibilità di essere alimentato, almeno per le sole funzioni di dialogo con il PC, acquisizione dei cinque input e controllo dell'uscita on/off, tramite la connessione USB.

A tal proposito la linea 5V VBUS in arrivo dal connettore USB, filtrata da R1, C5 e C6, viene portata al catodo del D1 mediante il diodo D2. I due diodi formano una porta OR e permettono di fornire al microcontrollore poco meno di 5 volt (nel caso dell'alimentazione tramite un alimentatore, si tratta di 5,3-0,7 V della caduta sul diodo, mentre con l'USB la tensione è 5-0,7 V) indipendentemente da quale sia la fonte di alimentazione.

REALIZZAZIONE PRATICA

Vediamo ora come si costruisce la scheda USB descritta in queste pagine, per la quale è stato previsto un circuito stampato a doppia

faccia ottenibile per fotoincisione e del quale trovate nel nostro sito le tracce lato rame in formato stampabile.

Una volta in possesso della basetta, non resta che saldarvi i componenti a partire da quelli a più basso profilo, ossia resistenze, diodi (quelli del ponte di Graetz, ossia D4, D5, D6 e D7, vanno montati in verticale) trimmer e zoccoli per microcontrollore, ULN2003 ed L6219, quindi i condensatori, partendo da quelli non polarizzati. Proseguite con il quarzo e la presa USB. L'integrato LM2575ADJ deve essere montato in verticale e non richiede dissipatore di calore.

Per le connessioni con le bobine dei motori passo-passo potete utilizzare dei connettori maschio quadripolari tipo JST o delle morsettiere miniatura passo 2,54 mm; quanto ai cinque ingressi e all'uscita on/off, vale lo stesso discorso: potete usare morsettie-

re miniatura o connettori a passo 2,54 mm rispettivamente bipolari e tripolare. Per l'alimentazione è richiesta una morsettieria bipolare da stampato a passo 5 mm.

Non dimenticate di inserire e saldare negli appositi fori un pin-strip a 5 poli che vi servirà per la programmazione in-circuit del microcontrollore o per l'eventuale debug. Per tutte le connessioni tra il motore, gli switch degli ingressi, l'uscita ecc., fate riferimento alla Fig. 3, che illustra il cablaggio delle varie sezioni usando i connettori preposti. Completato il circuito e inseriti gli integrati nei rispettivi zoccoli, dopo la programmazione del PIC (che va fatta senza fornire alimentazione in quanto provvede la connessione USB) potete alimentare il tutto con un alimentatore da rete capace di fornire la tensione e la corrente richieste dal motore passo-passo che intendete controllare. Se usate

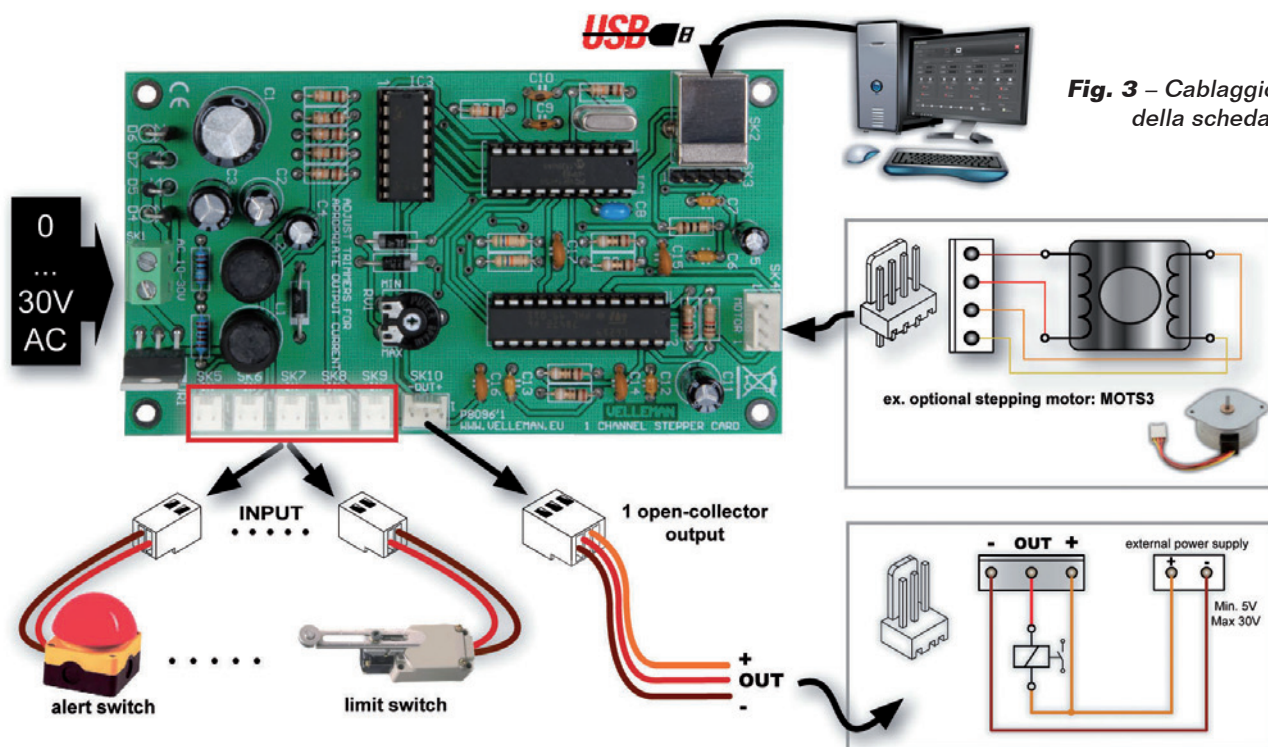


Fig. 3 – Cablaggio della scheda.

Fig. 4 – Schermata del software di gestione della scheda.

l'uscita on-off, l'alimentazione le verrà fornita dallo stesso circuito di alimentazione dell'utilizzatore che andrete a controllare con essa. Diciamo che per alimentare la scheda basta una corrente di 2 ampere. Ora accendete il computer e installate in esso il file eseguibile contenente il software di controllo, ottenuto scompattando l'archivio motorsetup.zip scaricabile dal nostro sito www.elettronica.in.it; vi sarà altresì necessario installare i driver della scheda, che poi si identificano con i driver Microchip del microcontrollore contenente la porta USB, opportunamente customizzati. I driver verranno richiesti al primo collegamento della scheda al computer, quindi suggeriamo di installarli prima del software di controllo, così da fare in modo che una volta installato, questo sia in grado di rilevare l'unità.

IL SOFTWARE DI CONTROLLO

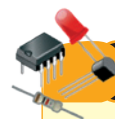
Si tratta di un'utility che gira in ambiente Windows (compatibile con Windows 2000/XP, Vista, Windows 7) e che una volta installata ed avviata mostra una schermata con il pannello di controllo visibile in Fig. 4. Nel menu a tendina Card Type dovete scegliere K8096 e, nella casella sottostante (Port), definire la porta di comunicazione (COM virtuale) corrispondente all'USB utilizzato dal computer per connettersi alla scheda di controllo.

Fatto ciò potete iniziare a verificare come funziona l'insieme, anche aiutandovi con la funzione DEMO, che permette di emulare le funzionalità anche senza aver collegato la scheda. Se quest'ultima è effettivamente collegata al computer, facendo clic sul pulsante Connect avviate il dialogo con essa e potete controllare il motore passo-passo e l'uscita on/off open collector, ovvero acqui-



sire lo stato degli ingressi. Notate che il software di controllo è universale e vale sia per la scheda qui descritta, sia per l'unità di cui parleremo in un articolo a parte, capace di governare quattro stepper motor; per questa ragione, se scegliete K8096, che poi è la scheda qui descritta, potete impartire comandi solo nel riquadro Motor 1, oltre che, naturalmente, in Input (lettura ingressi) e Output (controllo uscita on/off). Per l'esattezza, in Motor 1 disponete dei pulsanti Left e Right facendo clic sui quali potete impartire i comandi di rotazione rispettivamente levogira e destrogira, oltre che di Torque e Stop, che rispettivamente avviano la rotazione e la bloccano. Nelle caselle Steps e Speed potete definire il numero di passi/giro che il motore dovrà compiere (in accordo con le caratteristiche dello stepper motor) e di giri/minuto (anche questo valore dovrà essere compatibile con i dati del motore passo-passo che userete). Nell'archivio zip contenente il software di controllo e i driver, si trovano anche utili contenuti come la dll che potete incorporare in software personalizzati

da voi realizzati, per integrare la gestione della scheda in altri programmi; il materiale fornito a supporto e installato insieme al software (completata l'installazione, nel menu Avvio/Start>Tutti i programmi, si trova la voce Velleman, dalla quale si accede ai vari contenuti) permette anche di associare agli ingressi la gestione manuale del motore passo-passo. ■

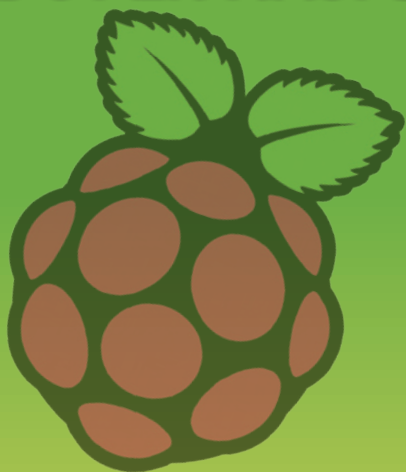


per il MATERIALE

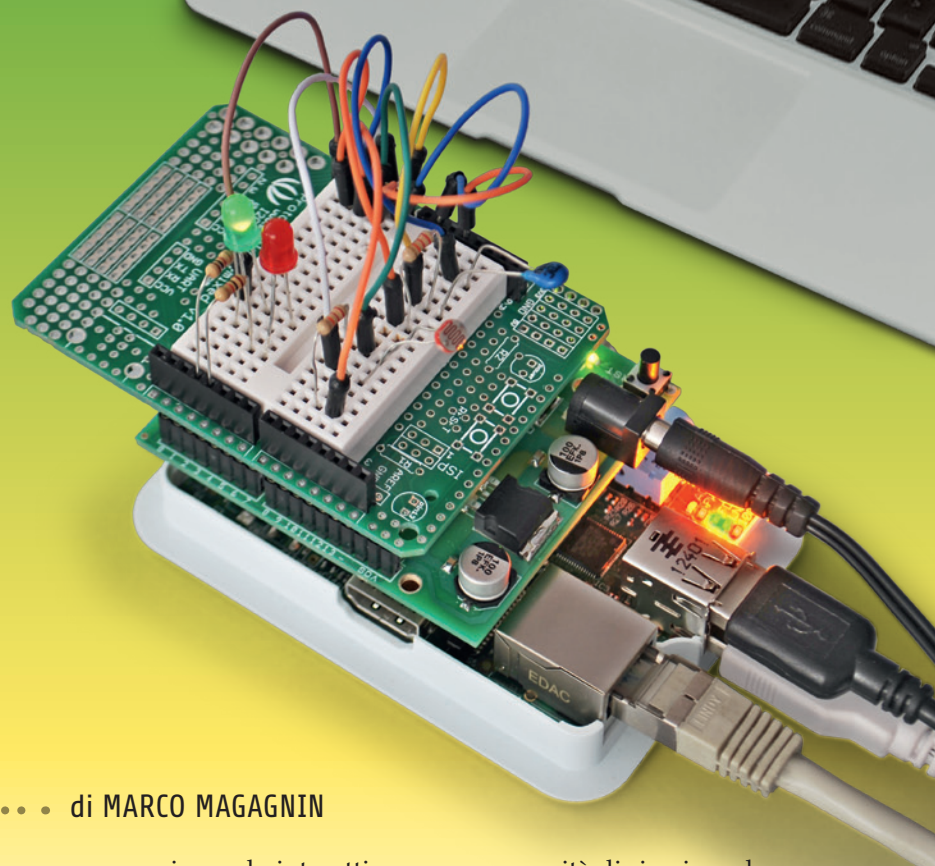
Questo controller USB per stepper motor è disponibile in scatola di montaggio. Il kit - prodotto da Velleman e distribuito da Futura Elettronica - comprende tutti i componenti, la basetta forata e serigrafata, il micro programmato e le minuterie. Non è compreso l'alimentatore né il motore passo-passo. Il kit (cod. K8096) costa 43,00 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>

TERMOSTATO WEB CON LO SHIELD DI ESPANSIONE ADC PER RASPBERRYPI



Completiamo la realizzazione del termostato web con la realizzazione della pagina HTML che include la dashboard emoncms e il pulsante per modificare le soglie di temperatura alle quali intervengono i sistemi che simulano il riscaldamento e il condizionamento.



..... di MARCO MAGAGNIN

Dopo aver configurato il framework emoncms che gestisce gli indicatori grafici alimentati dalle misure dei sensori di temperatura e luminosità e lo stato della simulazione dei sistemi di riscaldamento e condizionamento, completiamo l'applicazione con i programmi di presentazione della pagina web.

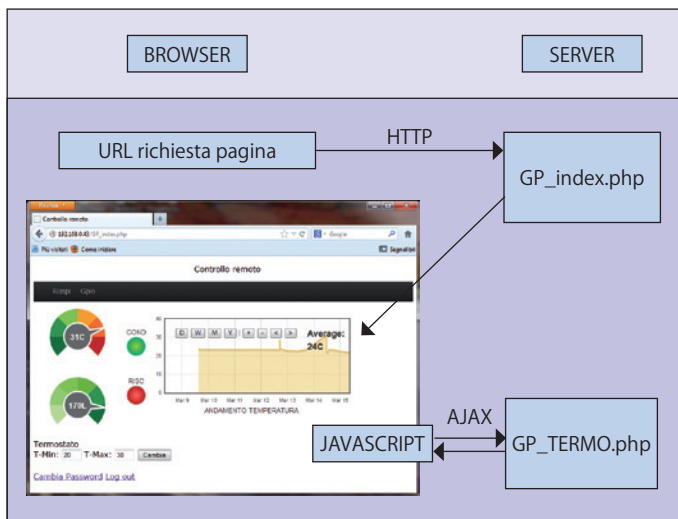
La particolarità principale è l'utilizzo della tecnica AJAX (Asynchronous JavaScript and XML) per gestire la configurazione delle soglie di temperatura

in modo interattivo, senza necessità di ricaricare la pagina ogni volta.

PROGRAMMA DI GESTIONE DELLA PAGINA WEB

Alcune considerazioni sulla struttura dell'interfaccia web nel nostro progetto.

Vogliamo una pagina dove possano essere eseguiti comandi e impostazioni, la risposta ai quali non richieda di ricaricare ogni volta la pagina stessa. Il nostro obiettivo è di realizzare una pagina che, una



volta aperta se ne stia al suo posto accettando i nostri comandi e ritornando i risultati dopo la modifica interattiva dei soli campi interessati. Prendiamo come riferimento lo schema di Fig. 1. Per ottenere questo risultato dobbiamo fare in modo che la pagina HTML possa eseguire delle azioni proprie, indipendenti dal server. Utilizzeremo il linguaggio Javascript che ci permette di inserire delle istruzioni (script) all'interno del codice HTML della pagina web, che verranno eseguite sul client a fronte di specifiche richieste. L'utilizzo di Javascript nelle pagine HTML è una delle tante tecniche che vanno sotto il nome di HTML dinamico (DHTML Dynamic HTML). Javascript permette di eseguire delle azioni all'interno della pagina web senza necessità di interessare il server. Questo significa che il browser che utilizziamo contiene al suo interno l'interprete Javascript per l'esecuzione delle istruzioni. Per aggiornare i contenuti utilizziamo la tecnica AJAX che ci permette di gestire lo scambio di dati in sottofondo tra la pagina web ed il server, in modo asincrono. In sintesi, per gestire la nostra pagina web dobbiamo adottare tecniche di pagine web dinamiche (PHP) coadiuvate da tecniche Javascript e di gestione interattiva dei contenuti (AJAX). Crediamo che basti. Per restare il più semplici possibili utilizzeremo queste tecniche nel modo più elementare possibile, giusto per dare nota della loro esistenza. Comunque qui il terreno si fa un po' in salita. Nello schema di Fig. 1 abbiamo visto la pagina GP_index.html che viene richiamata per prima e che si colloca come coordinatore di tutte le altre funzionalità. L'azione "Cambia" viene eseguita dal programma GP_TERMO.php separato rispetto al programma che gestisce la pagina principale, che viene richiamato ad ogni pressione del pulsante.

Fig. 1

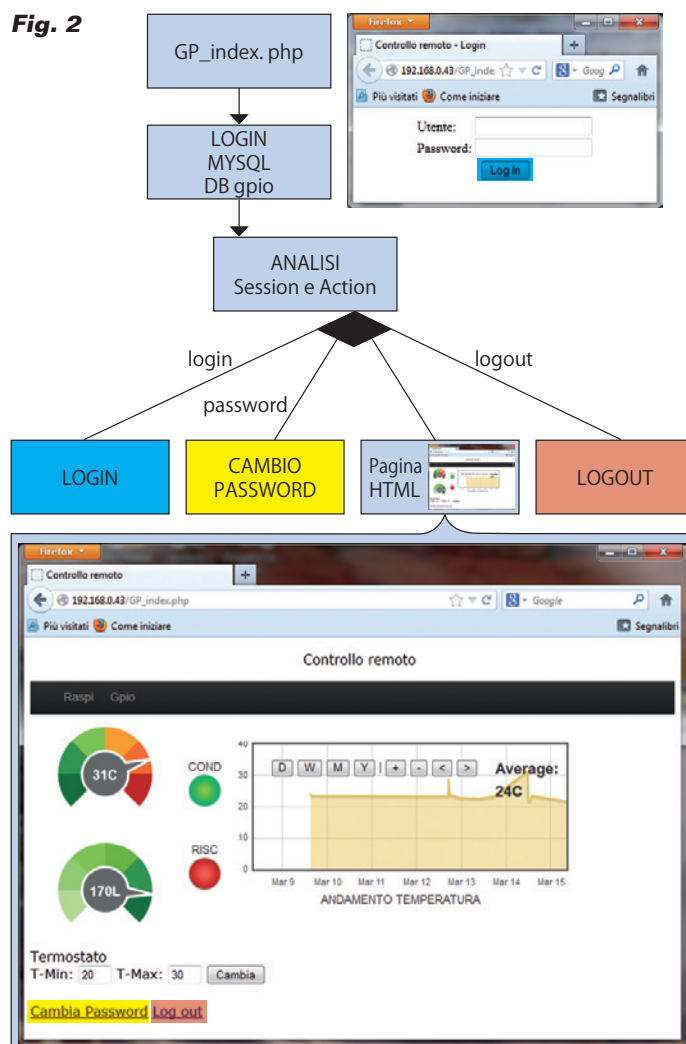
Vediamo di descrivere il funzionamento del programma GP_index.php, almeno nelle sue linee essenziali.

Nella Fig. 2 i rettangoli colorati che evidenziano le singole funzionalità nelle immagini delle pagine web corrispondono ai colori dei blocchi nel diagramma di struttura, che elaborano le funzionalità stesse, e che trovano riscontro nelle corrispondenti sezioni del listato.

Quando un utilizzatore esterno richiede la pagina, come prima operazione il programma si connette al database "gpio" ospitato dal server MySQL. Poi verifica se l'utente si collega per la prima volta, in tal caso restituisce la pagina di login con la richiesta di Userid e Password. Ricordiamo che l'utente è "admin" e la password "gpio".

Dopo il buon esito del login viene presentata la pagina principale del progetto. Questa pagina è divisa in due aree, un'area di sola visualizzazio-

Fig. 2



ne con la rappresentazione grafica dello stato del sistema e un'area di "comando" con i campi e i pulsanti necessari ad attivare le diverse funzionalità. Sono presenti due collegamenti per cambiare la password e per eseguire il logout. Abbiamo lasciato come ultima riga un campo dove vengono riportati i messaggi di ritorno del programma attivato con la tecnica Ajax.

Tralasciamo la spiegazione dei rami di Primo collegamento, Cambio password e Logout per concentrarci sulle sezioni di programma che gestiscono il pulsante "Cambia" e i campi associati.

Una sola osservazione riguarda l'inclusione nella nostra pagina della pagina grafica di emoncms, che abbiamo risolto con l'inclusione della pagina mediante un tag `<iframe>`.

"Iframe" significa **"inline frame"**: in una pagina che non utilizza una struttura a frame è possibile creare un frame "al volo" in qualsiasi punto del codice HTML, grazie a questo tag.

Possiamo specificare la larghezza e l'altezza del tag, mentre gli attributi di visualizzazione sono gli stessi del tag `<frame>`: si tratta di una vera e propria finestra verso l'esterno contenuta in un documento ordinario. Come riferimento abbiamo utilizzato il link alla pagina di menu del nostro server emoncms, utilizzando l'APIKey di sola lettura per l'autorizzazione all'accesso, come abbiamo già visto alla fine della personalizzazione della pagina emoncms.

Ora analizziamo più approfonditamente il meccanismo AJAX associato ai comandi operativi.

Vediamo che nella sezione di programma che gestisce la pagina principale è presente un tag `<script type="text/javascript">` al quale segue un lungo elenco di istruzioni decisamente non PHP, terminato dal tag `</script>`. Le istruzioni contenute

nel blocco sono istruzioni Javascript, che si occupano della gestione della parte interattiva della pagina HTML.

Queste istruzioni vengono passate all'interno della pagina HTML per essere eseguite, a richiesta, direttamente all'interno del browser. Javascript è un linguaggio di scripting e quindi richiede un interprete per poter essere eseguito. Ovviamente l'interprete fa parte del codice dei browser più moderni.

Come potete notare i moduli javascript sono organizzati in funzioni. Ciascuna delle quali può prevedere o meno dei parametri.

Di seguito vediamo il frammento di codice HTML che realizza la funzionalità di modifica delle soglie di temperatura. Possiamo modificare i valori nei campi T-Min e T-Max e poi premere il pulsante "Cambia" per inviare i valori al server. Rispetto al codice di una normale pagina HTML notiamo i tag id all'interno delle sezioni di descrizione degli elementi, caselle di testo e pulsante, in più, nel tag dell'elemento "button" è presente la gestione dell'evento onclick, che innesca la funzione javascript "spedisci Termo", passandogli come parametri i valori dei campi tmin e tmax (**Listato 1**).

Alla pressione del pulsante "Cambia" viene attivata la funzione Javascript "spedisci Termo" che, ricordiamo, viene eseguita all'interno del browser, recepisce i due valori di temperatura e inserisce il messaggio di attesa nella casella "descrizione", ultima in fondo alla pagina, che riporta i messaggi di comunicazione tra gli "attori" AJAX e poi, in squisito stile programmazione ad oggetti istanzia una richiesta al server web di nome "richiesta", per la quale prenota l'esecuzione della funzione `updateasincronoTermo` per quando verranno ricevute delle risposte da parte del server, imposta una chiamata con metodo

Listato 1

```
Termostato
<br>
T-Min: <input type="text" name="Tmin" id="tmin" value="" . $xMin . \'\' size="3"/>
T-Max: <input type="text" name="Tmax" id="tmax" value="" . $xMax . \'\' size="3"/>
      <input type="button" value="Cambia" id="buttonTermo"
      onclick="spedisciTermo(document.getElementById(\'tmin\').value,
      document.getElementById(\'tmax\').value);"/>
```

Listato 2

```
function spedisciTermo(Tmin,Tmax)
{
    document.getElementById("descrizione").innerHTML = "--- Attendi ... ---";
    server = "AJ_TERMOPHP?action=Termo" + "&Tmin=" + Tmin + "&Tmax=" + Tmax;
    richiesta = new XMLHttpRequest();
    richiesta.onreadystatechange = updateasincronoTermo;
    richiesta.open("GET", server, true);
    richiesta.send(null);
}
```

Listato 3

```
function updateasincronoTermo(){
    if ((richiesta.readyState == 4) && (richiesta.status == 200))
    {
        esito = richiesta.responseText;
        iparr = esito.split(",");
        Stato = iparr[0];
        Tmin = iparr[1];
        Tmax = iparr[2];
        document.getElementById("tmin").value = Tmin;
        document.getElementById("tmax").value = Tmax;
        document.getElementById("descrizione").innerHTML = esito;
    }
}
```

Listato 4

```
$MySQLUsername = "gpio";
$MySQLPassword = "pi";
$MySQLHost = "localhost";
$MySQLDB = "gpio";

$dbConnection = mysql_connect($MySQLHost, $MySQLUsername, $MySQLPassword);
mysql_select_db($MySQLDB, $dbConnection);

$query3 = mysql_query("SELECT xMin, xMax FROM xTemp WHERE xLocale = 1;");
$tempRow = mysql_fetch_assoc($query3);
$xMin = $tempRow['xMin'];
$xMax = $tempRow['xMax'];

mysql_close();
```

Listato 5

```
<iframe src="http://192.168.0.43/emoncms/admin?id=1&apikey=5a79c4482e414309e9b6b98dc8651f2a" width="800" height="300" frameborder="0" ></iframe>
```

GET e la invia. La stringa null nel metodo send è associata al fatto che abbiamo eseguito la chiamata con il metodo GET. Il metodo GET passa i parametri al server direttamente nell'URL, dopo il punto di domanda, mentre il metodo POST passa i parametri in una stringa separata, che può contenere molti più dati di quelli gestibili con il metodo GET. Nel caso avessimo usato il metodo POST, i dati associati alla chiamata andrebbero inseriti al posto del "null" (Listato 2).

A quest'ora il server avrà già risposto da tempo, infatti vedrete molto raramente la scritta ---Attendi---, nel qual caso viene attivata la funzione `updateasincronoTermo()` che, se non vi sono errori - cioè lo stato della risposta è uguale a 4 e lo stato della richiesta è uguale a 200 - intercetta il messaggio di ritorno, costituito da campi divisi da virgole (lo ab-

biamo costruito noi nel programma lato server, come vedremo presto), lo "spacchetta" in campi singoli e aggiorna i valori nei campi `tmin` e `tmax` con i valori ricevuti di ritorno. Poi riporta l'intero messaggio di ritorno nel campo "descrizione". Potete anche eliminare questa informazione scrivendo "" nel campo o quello che preferite. Con questo abbiamo finito e come avrete notato, durante tutto questo lavoro la pagina non viene assolutamente ricaricata, ma rimane fissa al suo posto (Listato 3). Vediamo nell'estratto di codice successivo (Listato 4) come reperire dei dati da un database MySQL in PHP. Nel programma `GP_index.php` ci serve accedere ai valori di soglia minimo e massimo impostati per la funzione termostato, per evidenziarli nella pagina web.

Iniziamo con la connessione al database, per la quale è ne-

cessario specificare i parametri dell'host dove risiede il database, il nome del database, l'utente e la password abilitati alla sua gestione.

Si esegue la connessione con il metodo `mysql_select_db` e poi possiamo eseguire tutti i comandi SQL necessari; nel nostro caso sono la SELECT nella tabella `xTemp` dei valori `xMin` e `xMax` per il locale 1. Per l'esecuzione dei comandi utilizziamo il metodo `mysql_query` che restituisce il risultato della query nella collezione `$query3`. Nel nostro caso viene restituita una sola riga, in caso di restituzione di più righe è necessario eseguire un ciclo for che esaurisca le righe di risposta. Per ciascuna riga si indirizza il "cursore" della query alla riga in esame con il metodo `fetch_assoc` che restituisce i campi della riga nell'array dizionario di nome `$tempRow` e da questo si leggono i valori dei campi indicizzandoli con il loro nome, `xMin` e `xMax`. Prima di terminare il programma si chiude la connessione al database.

Visto che stiamo parlando del programma `GP_index.php`, prima di passare ad un altro

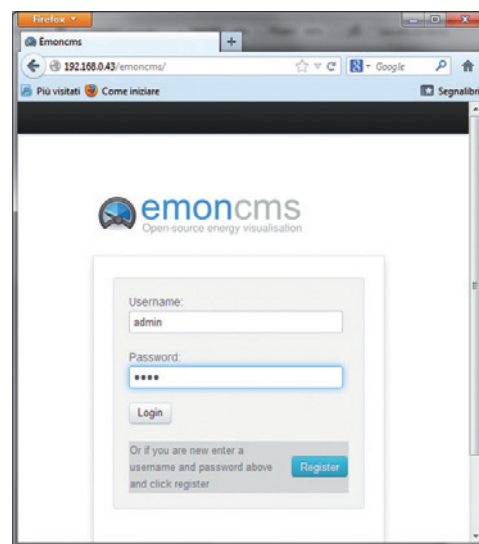


Fig. 3

argomento sistemiamo la configurazione dell'URL e dell'API-Key necessari a richiamare la dashboard emoncms nell'iframe del programma stesso.

Per fare questa operazione dobbiamo riaprire l'applicazione di configurazione di emoncms. Nel browser inseriamo l'URL <indirizzo-IP-RaspberryPi>/emoncms.

Nella finestra di **Fig. 3** inserite utente e password, noi vi avevamo suggerito utente "admin" e password "gpio". Selezionate il pannello "Dashboard" e poi cliccate sull'icona share (**Fig. 4**), copiate dal campo URL del browser il link alla pagina appena creata (**Fig. 5**) e inseritelo nel reference dell'iframe nel programma GP_index.php, insieme all'APIkey di lettura (**Listato 5**).

GESTIONE TERMOSTATO

Per la gestione del termostato abbiamo già descritto il programma, che scheduleremo a tempo e che legge i sensori collegati al convertitore analogico digitale, calcola i valori di temperatura e luminosità, li confronta con i valori di soglia sulla tabella xTemp del database e aggiorna il server emoncms con i valori letti e lo stato dei LED che rappresentano l'impianto di riscaldamento e di condizionamento.

Vedremo più avanti come utilizzare lo scheduler di sistema "cron" per eseguire il programma a intervalli determinati, risparmiando molte risorse rispetto all'esecuzione di cicli temporizzati con il programma sempre in esecuzione.

Dobbiamo ancora analizzare il programma PHP che realizza la funzione AJAX nella pagina web interattiva.

Il programma GP_TERMO.php viene richiamato dalla richiesta in formato URL che proviene dalla

funzione javascript della pagina web, innescata dalla pressione del pulsante "Cambia" (**Listato 6**).

A questo punto sotto il controllo del server Apache2 viene eseguito il programma GP_TERMO.php, visibile nel Listato 2, che esegue la connessione al database con le stesse modalità descritte nel paragrafo dedicato alla pagina GP_index.php e poi esegue l'aggiornamento (UPDATE) del record nella tabella xTemp con i valori ricevuti nell'URL.

Alla fine viene restituito con l'istruzione print il semplice messaggio di ritorno

OK, 20, 28, Termo

con i campi separati da virgole che verranno intercettati dalla funzio-

ne javascript updateasincronoTermo() nella pagina web.

SCHEDULATORE CRON

Possiamo lanciare a mano il programma ProvaADC2.py per fare qualche prova, poi conviene adottare una struttura più solida per eseguire il programma in modo automatico.

Introduciamo ora il servizio di schedulazione cron che ci permetterà di inviare una nuova serie di dati al nostro sito emoncms ad intervalli prestabiliti, per esempio ogni minuto.

Il servizio cron è il servizio predefinito in GNU/Linux per la gestione dell'esecuzione di programmi e comandi a tempo.

Per funzionare, cron si appoggia a una tabella che contiene le

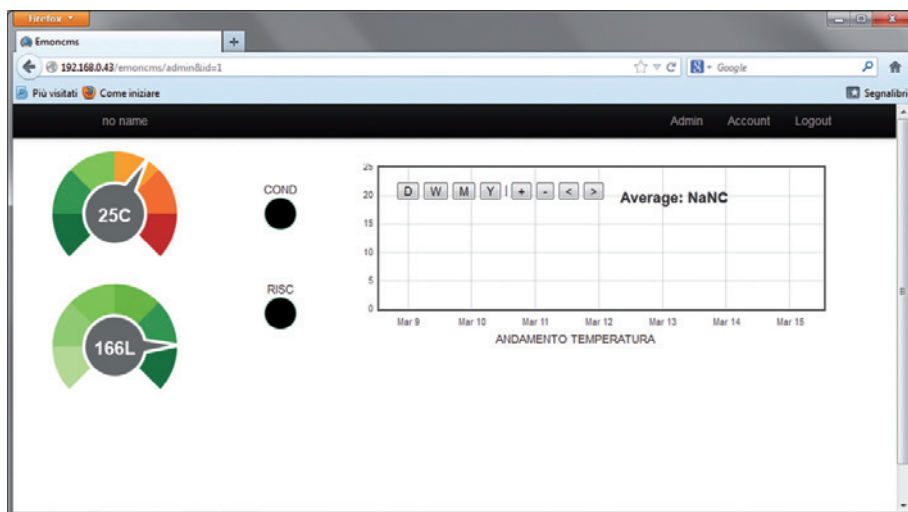


Fig. 4

```
GNU nano 2.2.6      File: /tmp/crontab.pBXJ9s/crontab      Modified
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow  command
*/1 * * * * python /home/ProvaADC2.py > /dev/null
```

Fig. 5

Listato 6

```
<?php
// GP_TERMO.php

$MySQLUsername = "gpio";
$MySQLPassword = "pi";

$MySQLHost = "localhost";
$MySQLDB = "gpio";

If (($MySQLUsername == "USERNAME HERE") || ($MySQLPassword == "PASSWORD HERE")){
    print 'ERROR - Please set up the script first';
    exit();
}

$dbConnection = mysql_connect($MySQLHost, $MySQLUsername, $MySQLPassword);
mysql_select_db($MySQLDB, $dbConnection);

$action = $_GET['action'];
$Tmin = mysql_real_escape_string($_GET['Tmin']);
$Tmax = mysql_real_escape_string($_GET['Tmax']);
if ($action == "Termo"){
    mysql_query("UPDATE xTemp SET xMin='$Tmin', xMax='$Tmax' WHERE xLocale=1;");
    mysql_close();
}
print 'OK,' . $Tmin . ',' . $Tmax . ',' . $action;
?>
```

“istruzioni” circa i programmi da eseguire e in quali istanti lanciarli. La tabella utilizzata da cron si gestisce con la funzionalità “crontab” che ha la sintassi:

crontab <opzione> dove opzione può assumere i valori -e, -l e -r.

Il comando

crontab -e

ci permette di aprire il file di configurazione predefinito di crontab come in **Fig. 5**: Ogni riga del file di configurazione ha la struttura riportata nel **Comando 1**. Ciascun parametro può essere configurato come serie di valori es: 0,3,7,9 come un intervallo es: 3-7 oppure un misto dei due es: 0,3,4-6,9 oppure ancora come frequenza */5 (ogni 5 minuti, ore, etc.). Nel nostro caso vogliamo che il programma ProvaADC2.py venga eseguito ogni minuto, quindi digitiamo la linea di configura-

zione:

```
*1 *** python /home/ProvaADC2.py > /dev/null
```

che significa: esegui il programma che misura temperatura e luminosità una volta al minuto di ogni ora, giorno e mese, perdendo tutti gli eventuali messaggi di log che possano venire prodotti. Usciamo dal file con <CTRL>X, rispondiamo “Y” alla richiesta di salvataggio e confermiamo il nome del file con “Invio”. D’ora in avanti ogni minuto verrà eseguito il programma ProvaADC2.py con gli aggiornamenti conseguenti.

Per modificare le impostazioni di cron richiamiamo la funzionalità

crontab -e

Per vedere le impostazioni utilizziamo l’opzione *crontab -l*

mentre per interrompere l’esecu-

zione automatica digitiamo

crontab -r

LANCIAMO GPIOSERVER3.PY IN SOTTOFONDO

Avrete ovviamente notato che quando viene lanciato il programma GPIOServer3.py, questo tiene impegnata la finestra di terminale, impedendovi di eseguire altri comandi. A questo punto potremmo aprire più finestre per lanciare più programmi, che può essere una buona soluzione per il debug, ma ci costringe in ogni caso a tenere aperta la connessione SSH per far funzionare il programma. Una classica architettura “embedded” deve essere in grado di funzionare completamente abbandonata e senza connessioni di controllo. È ora di affrontare la gestione dei processi in background, ovvero senza interazione e senza necessità di una finestra di terminale aperta.

Descriviamo due modalità che agiscono, per così dire, a diverse profondità nel sistema operativo. Il primo metodo consiste nel consegnare l’esecuzione del programma ad un processo che, una volta lanciato, libera la linea di comando ma resta comunque collegato alla vita della finestra. Quando chiudiamo la finestra il processo viene interrotto. Per eseguire il programma in questa modalità si aggiunge una “&” alla fine del comando come, per esempio:

```
python /home/GPIOServer3.py &
```

Si dà invio e poi ancora invio: il programma resta in esecuzione in sottofondo e la linea di comando resta utilizzabile. Vediamo come interrompere il processo senza necessariamente chiudere la finestra “madre”. Quando abbiamo lanciato il

```
* * * * * comando da eseguire
- - - - -
| | | | | +---- giorno della settimana (0-6) (0=Domenica)
| | | | | +----- mese (1-12)
| | | +----- giorno del mese (1-31)
| | +----- ora (0-23)
| +----- min (0-59)
+-----
```

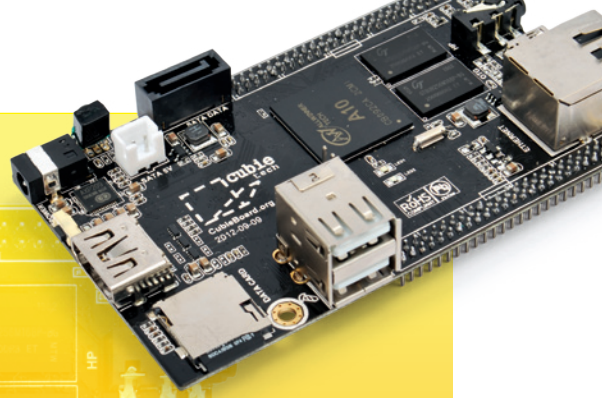
Comando 1

CubieBoard, nuova scheda ARM Android/Linux

Continua l'uscita sul mercato di veri e propri "computer" con a bordo processori ARM RISC. La scheda Cubieboard, delle dimensioni di poco superiori a quelle di RaspberryPi (circa 10 x 6 cm), è dotata di un SoC Allwinner A10 che contiene un processore Cortex A8 - limitato ad 1 GHz per contenere i consumi - e una GPU Mali-400, 1 GB di RAM e 4GB di memoria Flash a bordo. Nasce con Android 4.0.4 già installato, ma è in grado di far girare diverse distribuzioni di GNU/Linux come Ubuntu 10.04 e 12.04, Arch Linux, Fedora ARM e Raspbian "wheezy" (sì, la stessa distribuzione di RaspberryPi) e quindi può essere utilizzato in diversi ambiti, dalla



smart TV, per la quale è previsto un sensore infrarosso in modo da poterlo comandare da un telecomando, fino a diventare una vera e propria pattaforma di sviluppo, coadiuvata dalla presenza di un connettore SATA per il collegamento di un disco fisso "vero". Completano la dotazione il connettore HDMI, lo slot SDCard, 2 USB, la porta Ethernet, ingresso e uscita audio e ben 96 pin di I/O che includono tutti i bus di utilizzo "elettronico" I2C, SPI, seriali, VGA e video composito. Non mancheremo di parlarne.



processo con il metodo indicato sopra abbiamo ricevuto in risposta un numero di processo. Tutti i processi sono identificati da un numero. Possiamo vedere l'elenco dei processi attivi con i relativi identificativi utilizzando il comando ps (process status):

```
ps -e
```

Per elencare solo i processi che ci interessano possiamo filtrarli concatenando il comando ps con il comando grep (general regular expression print) che ricerca in una lista di file le stringhe di testo che soddisfano l'espressione di filtro

```
ps -e | grep python
```

Troveremo elencati solo i processi eseguiti dall'interprete python, cioè i nostri.

Per controllare le caratteristiche di un singolo processo, ad esempio per verificare qual è il programma in esecuzione utilizziamo il comando

```
ps numero-processo
```

Per terminare intenzionalmente un processo che gira in sottofon-

do, dobbiamo conoscere il numero di processo che lo identifica ed utilizzare il comando

```
kill numero-processo
```

La seconda modalità ci permette di eseguire i programmi in processi ancora più profondi, totalmente svincolati dalla finestra di terminale. Lanciamo il programma con il comando nohup e il carattere "&" alla fine:

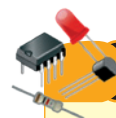
```
nohup python /home/GPIOServer3.py &
```

Con il programma lanciato in questo modo, possiamo tranquillamente chiudere la finestra di terminale e anche disconnetterci dalla sessione SSH con il programma che continuerà a funzionare. Nohup significa No Hang Up ed è nato ai tempi delle connessioni su rete commutata via modem, quando ci si connetteva in remoto utilizzando le normali linee telefoniche per amministrare i sistemi e lanciare processi che dovevano continuare a funzionare anche dopo avere chiuso la connessione (telefonata).

Anche in questo caso, per terminare intenzionalmente il proces-

so si utilizza il comando *kill numero-processo*

Se i processi che lanciamo utilizzano lo stdout, cioè la destinazione - per esempio - delle istruzioni print, che normalmente viene indirizzata sulla finestra terminale, quando li lanciamo in sottofondo possiamo oscurare lo stdout oppure reindirizzarlo verso un file, per esempio per poter leggere



per il MATERIALE

Tutti i componenti utilizzati per realizzare lo shield di espansione per RaspberryPi utilizzato in questo progetto e descritto nel numero 173 sono di facile reperibilità. Il master del circuito stampato può essere scaricato dal sito della rivista così come i programmi di esempio. 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>

Fig. 6

```

192.168.0.43 - KITTY
root@raspberrypi:~# nohup python /home/GPIOServer3.py > /dev/null &
[1] 4221
root@raspberrypi:~# nohup: ignoring input and redirecting stderr to stdout

root@raspberrypi:~#

```

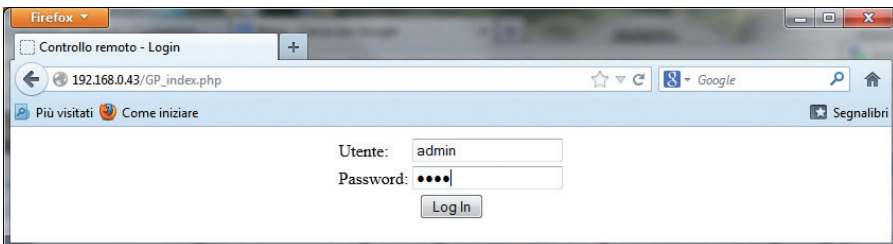
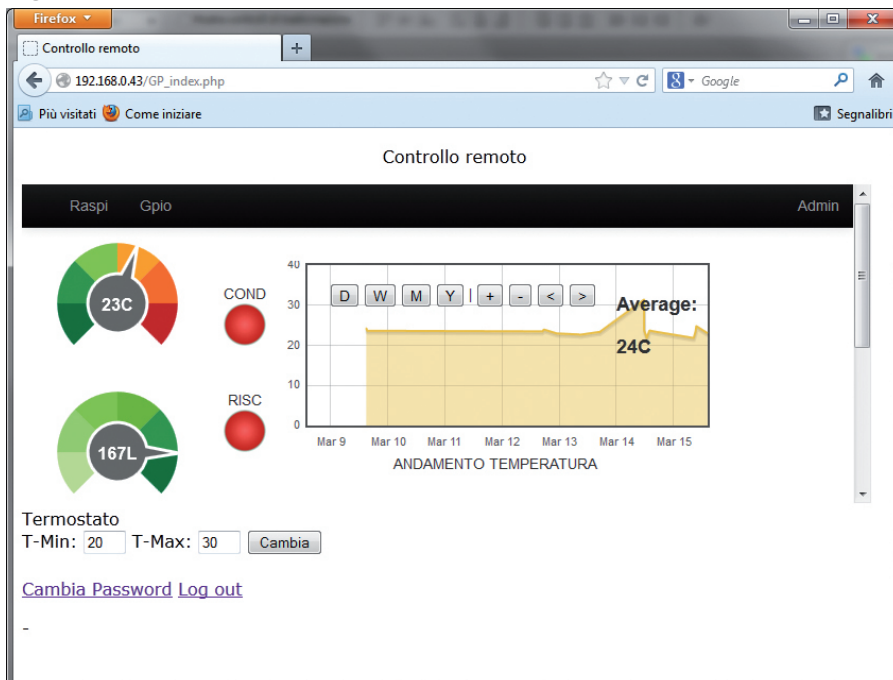
Fig. 7

```

192.168.0.43 - KITTY
root@raspberrypi:~# nohup python /home/GPIOServer3.py > /dev/null &
[1] 4221
root@raspberrypi:~# nohup: ignoring input and redirecting stderr to stdout

root@raspberrypi:~# ps -e | grep python
4221 pts/0    00:00:04 python
root@raspberrypi:~# ps 4221
  PID TTY          STAT       TIME COMMAND
 4221 pts/0    S          0:04      python /home/GPIOServer3.py
root@raspberrypi:~# kill 4221
  PID TTY          STAT       TIME COMMAND
[1]+  Terminated                  nohup python /home/GPIOServer3.py > /dev/null
root@raspberrypi:~#

```

Fig. 8**Fig. 9**

i messaggi successivamente, a scopo di debug (Fig. 6 e Fig. 7). Nel primo caso usiamo il comando

```
nohup python /home/GPIOServer3.py > /dev/null &
```

nel secondo caso

```
nohup python /home/GPIOServer3.py > /home/GPIOServer.log &
```

Per provare l'applicazione usciamo da emoncms con "Logout", richiamiamo la pagina principale con l'URL

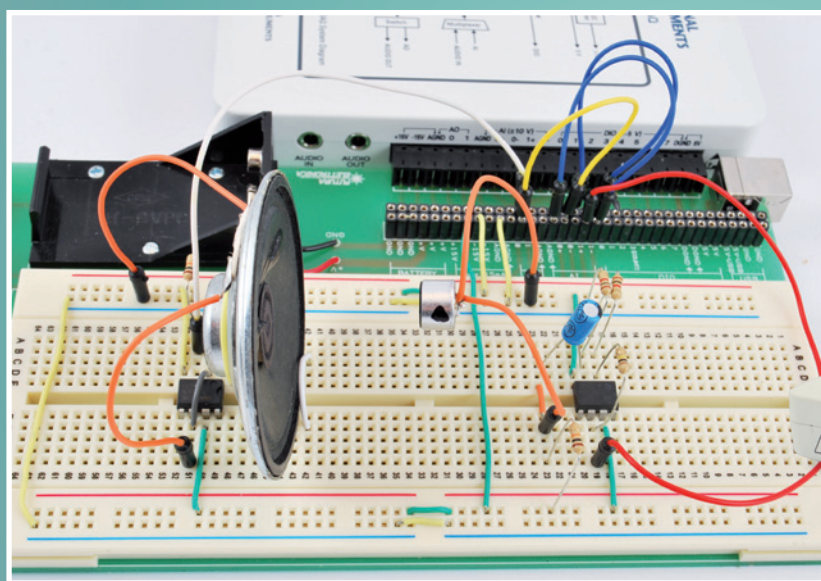
```
http://<IP-RaspberryPi/GP_index.php
```

alla pagina di login rispondiamo con "admin" e "gpio" (Fig. 8). Cambiando i valori di soglia della temperatura e scaldando la resistenza NTC tenendola tra le dita possiamo vedere attivarsi i LED veri e simulati a seconda delle condizioni di temperatura. Nel caso in cui l'aspetto grafico della pagina non vi soddisfi, riaprite emoncms in amministrazione, modificate il layout e riprovate (Fig. 9). Eccoci alla fine del percorso che ci permette di progettare e realizzare applicazioni embedded in ambiente GNU/Linux in generale, non soltanto utilizzando RaspberryPi. Chiaramente si è trattato solo di un assaggio, ma speriamo di essere riusciti a trasferire le informazioni necessarie e capire dove andare a guardare quando avremo necessità particolari. Ai tempi dei libri, quello che abbiamo cercato di descrivere stava nascosto all'interno di una "mezza libreria" di manuali. Oggi con internet a disposizione è molto più semplice approfondire architetture e tecniche, a patto di sapere dove cercare. Buona sperimentazione. ■

myKIT, AMPLIFICAZIONE AUDIO E LABVIEW

di ANDREA LETTIERI

Un operazionale TL082 per imparare le basi di amplificazione, microfoni e altoparlanti in questa nuova esercitazione con myDAQ e LabVIEW.



In questa esercitazione viene presentata la teoria e la pratica degli amplificatori audio non invertenti, con un'applicazione pratica finalizzata ad amplificare il segnale proveniente da un microfono. Dopo aver realizzato un circuito in grado di amplificare il segnale di un microfono, questo verrà utilizzato per misurare le prestazioni di un altoparlante.

IL MICROFONO PREAMPLIFICATO

Le capsule microfoniche electret sono molto diffuse in virtù delle loro prestazioni e del costo ridotto. Il

modo più semplice per descrivere il funzionamento di un microfono è di immaginarlo come un resistore il cui valore di resistenza varia quando viene colpito da un'onda sonora. Quando una corrente di polarizzazione scorre attraverso questa ipotetica resistenza variabile, la tensione ai capi varia di conseguenza; separando mediante un condensatore la parte variabile dalla polarizzazione (fissa) si ottiene un segnale che descrive l'onda sonora. Realizzate il circuito visibile nello schema di Fig. 1, dove la resistenza R_4

fornisce la corrente di polarizzazione per il microfono. Il condensatore C1 impedisce di amplificare anche la tensione continua che scorre attraverso il microfono, permettendo solo al segnale variabile di giungere allo stadio amplificatore. La resistenza R2 provvede alla polarizzazione dell'ingresso non invertente dell'operazionale TL081 e forma, insieme a C1, un filtro passa alto la cui frequenza di taglio si calcola con la seguente formula:

$$f_c = \frac{1}{2\pi R_2 C_1} = \frac{1}{2\pi (1k\Omega)(10\mu F)} = 15,92Hz$$

L'integrato TL081 e le tre resistenze R1 ed R3 formano il circuito amplificatore non invertente, il guadagno del quale può essere calcolato con la formula:

$$A_V = \frac{V_{OUT}}{V_{IN}} = 1 + \frac{R_3}{R_1} = 1 + \frac{100k\Omega}{1k\Omega} = 101$$

Se si utilizza il simulatore Multisim (seguite le istruzioni fornite nell'esercitazione 4) per riprodurre il circuito di Fig. 1, si ottiene lo schema mostrato in Fig. 2; allora si possono analizzare le prestazioni del circuito mediante la funzionalità AC Analysis, i cui risultati sono riportati in Fig. 3.

Nel modello Multisim, anche in questo caso si è utilizzata una sezione dell'operazionale TL082 al posto del TL081, in quanto quest'ultimo non è disponibile nelle librerie Multisim.

L'analisi della simulazione mostra che il guadagno è 40 dB, come previsto dai calcoli, e che la frequenza

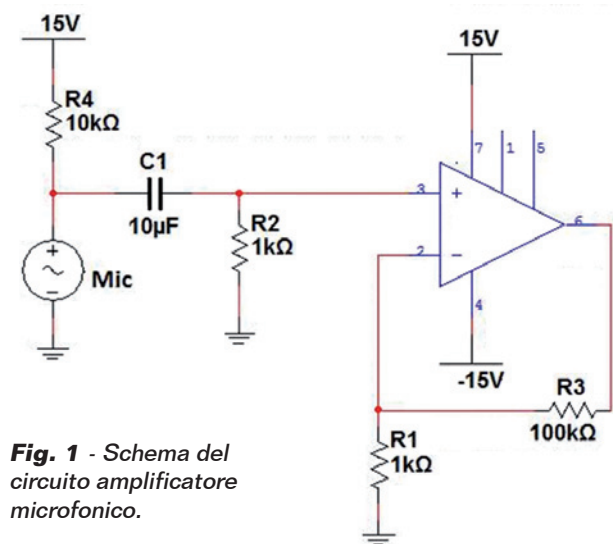


Fig. 1 - Schema del circuito amplificatore microfonico.

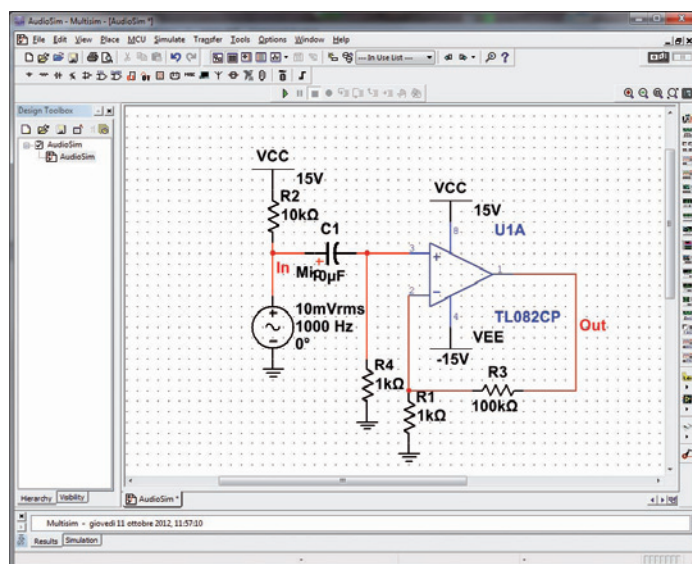


Fig. 2 - Schema Multisim dell'amplificatore microfonico.

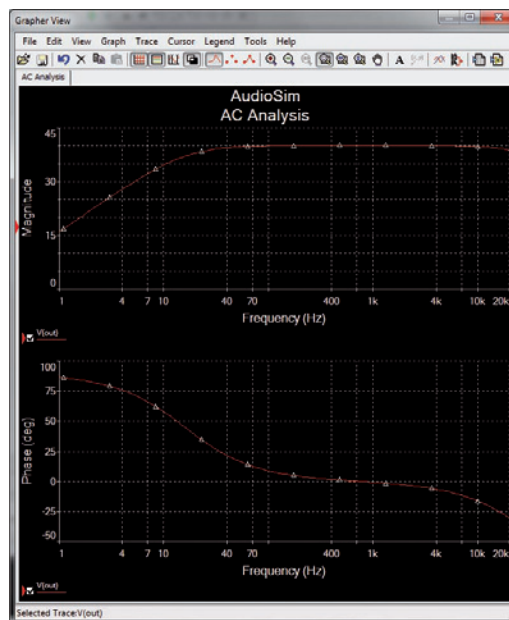


Fig. 3 - AC Analysis delle prestazioni del circuito.



Fig. 4

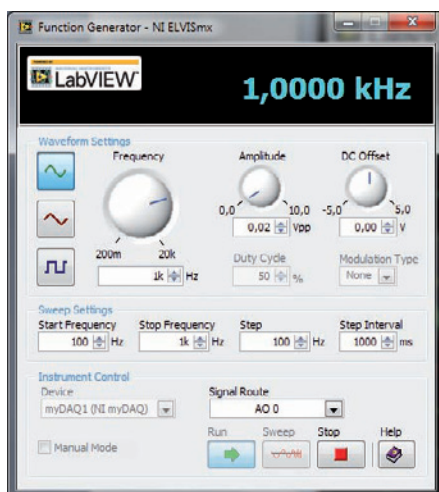
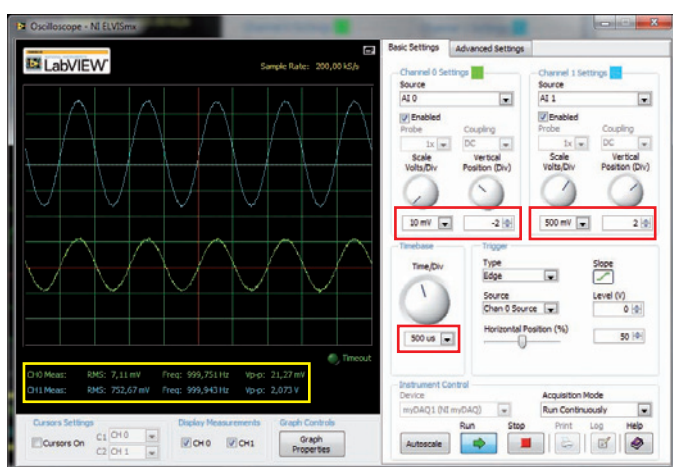


Fig. 5

banda passante è utilizzare il prodotto guadagno/ banda passante (GBW) specificato dal produttore. L'amplificatore operazionale TL081 ha un GBW di 3 MHz. Dividendo questo numero per il nostro guadagno di 101, possiamo prevedere una banda passante per il circuito amplificatore di 30 kHz, in accordo con i risultati della simulazione. Lo strumento Bode Analyzer, incluso nel set di strumenti myDAQ, può essere utilizzato per verificare il comportamento del circuito reale, una volta realizzato sulla breadboard. Nell'eseguire il cablaggio fate attenzione a rispettare le corrette polarità del condensatore elettrolitico e del microfono. Per verificare il circuito con il Bode analyzer, si deve collegare il connettore ingresso analogico 1 del myDAQ (AI 1+) al pin 6 dell'amplificatore operazionale. L'ingresso analogico 0 (AI 0+) deve essere collegato al terminale positivo del condensatore elettrolitico da 10 μ F, insieme all'uscita analogica AO 0, mentre i connettori (AI 0-) e (AI 1-) devono essere collegati a massa. Il diagramma di Bode della Fig. 4 conferma i risultati ottenuti dai calcoli ed in sede di simulazione. La Fig. 5 si riferisce alle prestazioni del circuito ottenute con la stessa configurazione di collegamenti a myDAQ, con un segnale emesso dal generatore di segnali configurato per un'uscita in onda sinusoidale a 1 kHz e 0,02 Vpp.

Una delle possibilità offerte dall'amplificatore microfonico appena realizzato è utilizzarlo per misurazioni e registrazioni di grandezze del mondo reale, quali, per esempio, le prestazioni di un altoparlante. Per pilotare un altoparlante con il myDAQ è necessario interporre un circuito buffer, che può essere realizzato a partire dal circuito di Fig. 1 rimuovendo la resistenza di retroazione e collegando l'uscita



di taglio del filtro passa basso è circa 15,9 Hz. La frequenza di taglio superiore è determinata dalla banda passante dell'amplificatore operazionale. Per piccoli segnali, il metodo più semplice per determinare la

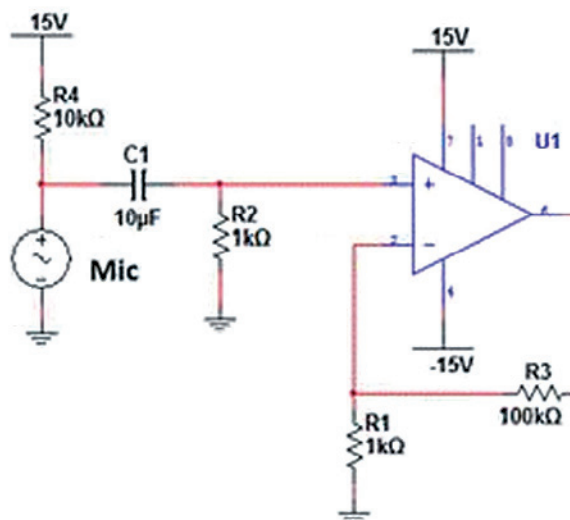
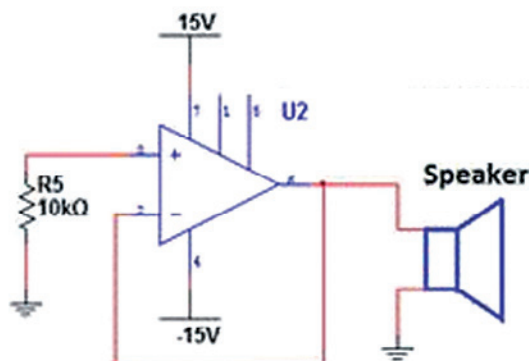
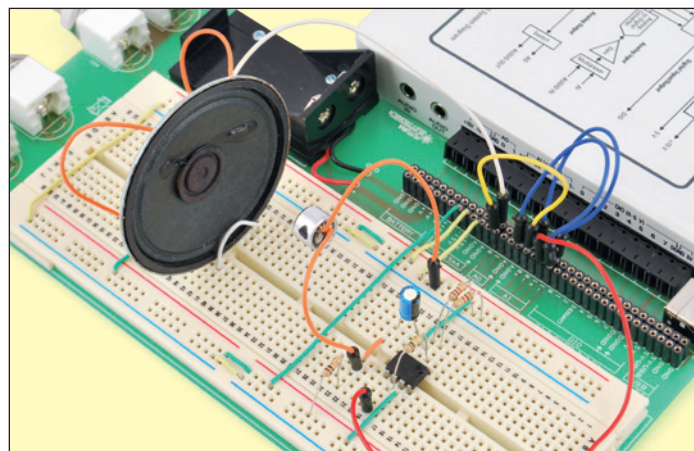


Fig. 6



Fig. 7



dell'altoparlante visibile in Fig. 6.

La misura della risposta in frequenza del circuito altoparlante/microfono può essere eseguita connettendo il terminale di uscita analogica AO 0 e l'ingresso analogico AI 0+ di myDAQ all'ingresso del circuito buffer dell'altoparlante, mentre il connettore di ingresso analogico AI 1+ deve essere collegato al pin 6 dell'amplificatore operazionale del circuito del microfono. I connettori AI 0- e AI 1- degli ingressi analogici devono essere collegati a massa.

Dopo aver aperto lo strumento Bode Analyzer, si può eseguire l'analisi con le impostazioni evidenziate in Fig. 7. Può essere utilizzato lo strumento Dynamic Signal Analyzer (DSA) per visualizzare il contenuto in armoniche del suono prodotto dall'altoparlante. Lasciate in funzione il generatore di segnali con le impostazioni precedenti.

Aprire lo strumento DSA e modificate le impostazioni come evidenziate in Fig. 8, poi selezionate il pulsante Run.

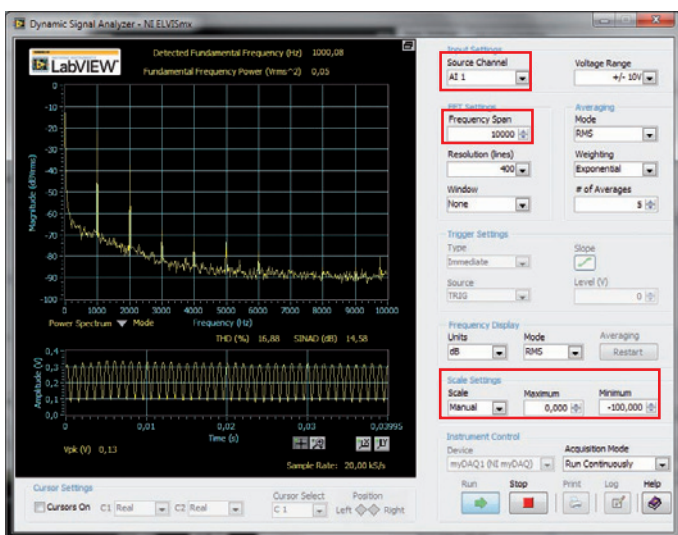
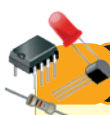


Fig. 8

dell'amplificatore operazionale direttamente all'ingresso invertente dello stesso. In questo modo il guadagno del circuito risulta essere uguale a 1. Questo significa che la tensione in uscita dall'amplificatore operazionale coincide con la tensione in ingresso. A questo punto ci si può chiedere quale sia il vantaggio di adottare un circuito che non presenta alcun guadagno; la risposta è che il buffer consente di trasferire il segnale senza caricare la fonte che lo fornisce, in quanto esso provvede ad erogare al carico la corrente necessaria. Dunque, il buffer fa da adattatore di impedenza.

Come si vede nello schema di Fig. 6, la corrente all'altoparlante è fornita dall'amplificatore operazionale e non dal generatore di segnali di myDAQ. L'impedenza vista dal generatore di segnali, nel caso del circuito di Fig. 6 è data dal valore della resistenza R5: nella fattispecie è 10 kohm. È ora tempo di realizzare sulla breadboard il circuito di pilotaggio



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

In questa sesta puntata ci occupiamo del firmware, di come permetta di intervenire sui vari parametri funzionali e dei messaggi che vengono forniti sul display a cristalli liquidi.



OpenWheels

IL FIRMWARE

di MIRCO SEGATELLO

Durante i mesi impiegati a sviluppare OpenWheels, abbiamo lavorato molto per realizzare un firmware che fosse al contempo completo di funzioni ed efficiente in termini di velocità. Abbiamo utilizzato variabili di tipo float solo dove strettamente necessario e implementato un sistema ad interrupt per scandire, con i giusti tempi, le operazioni di stabilizzazione. Non abbiamo lesinato sui sistemi di protezione e sulla messaggistica di diagnostica in quanto, come progetto open source, deve essere facile da gestire. L'hardware già descritto nei precedenti articoli è in versione definitiva, anche se è rimasto in sospeso un piccolo ma fondamentale dettaglio: la scala del giroscopio. Nei primi collaudi abbiamo usato il giroscopio con la scala dei 30°/sec - sicuramente il valore ottimale durante la marcia di OpenWheels - che però in alcune situazioni ha portato ad un overflow della lettura. È importante in queste applicazioni sfruttare i sensori con la loro massima portata, affinché il convertitore ADC utilizzato per la lettura sfrutti l'intera gamma di valori; proprio per questo motivo la scheda di controllo prevedeva i

due ponticelli JP3 e JP4, da cortocircuitare a seconda del fondo scala più idoneo. Per evitare dei fuori scala nella lettura del giroscopio abbiamo preferito, successivamente, utilizzare la scala dei 120°/sec, funzione che si ottiene cortocircuitando il jumper JP4 e non più il jumper JP3, come invece visibile nella foto a pag. 46 del numero 174 della nostra rivista. Solo i firmware dalla versione 1.1 in poi supportano questa modalità.

IMPOSTAZIONI TRAMITE LCD DI OPENWHEELS

La gestione di tutti i parametri modificabili dall'utente avviene tramite il display LCD, il quale svolge anche la funzione di monitoraggio dei principali valori di diagnostica. Sono previste tre schermate che possono essere mostrate in sequenza premendo il pulsante "Enter"; vediamo in dettaglio quali informazioni sono disponibili. La schermata numero 1 (default) riporta il livello della batteria, lo stato di funzionamento e, se in stato di errore, riporta anche il relativo codice. Siccome vi possono essere diverse situazioni di errore, che si verificano anche contemporaneamente, questa informazione è visualizzata in chiaro solo nel software della telemetria, mentre sul display LCD compare una sequenza di bit, ciascuno corrispondente a uno specifico errore secondo quanto riportato in **Tabella 1**.

La schermata in **Fig. 2a** - la prima della sequenza normale - in cui simuliamo il display LCD, indica che tutto funziona regolarmente, OpenWheels si trova nella modalità di StandBy e la batteria è al 56%.

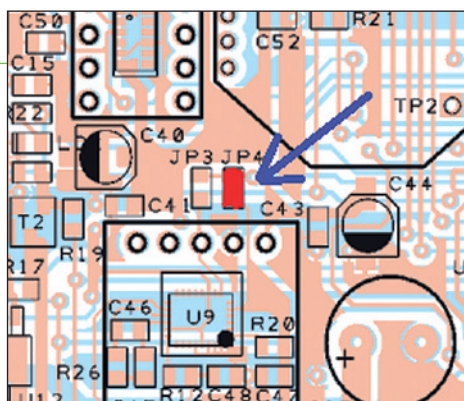


Fig. 1 - Corretta impostazione dei jumper JP4

Fig. 2a

56%
STANDBY

La schermata Fig. 2b riporta invece una situazione di errore con codice 10000000 che corrisponde (Tabella 1) ad un errore nella valutazione dell'angolo. Questa situazione si verifica, ad esempio, se si inclina troppo OpenWheels.

Fig. 2b

56%
ERROR 10000000

Tutti gli errori, escluso il bit 0, comportano l'immediata modalità di StandBy ed il conseguente Shut-Down del driver di potenza. È probabile che alla prima accensione, essendo tutti i valori della EEPROM nulli, si presenti l'errore al bit 8; vedremo più avanti come impostare i parametri di funzionamento. Da notare che non vengono visualizzati gli zeri non significativi a sinistra, quindi se l'errore è al bit 5 avremo "ERROR 100000" e così via. Mentre è visualizzata la schermata con l'errore, premendo contemporaneamente P1 e P2 si ottiene l'azzeramento degli errori ed il conseguente ripristino di tutte le funzioni. La seconda schermata - Fig. 3 - riporta, nella prima riga, le indicazioni relative alla tensione della batteria e della relativa corrente erogata, mentre nella seconda riga, i valori dell'accelerometro e del giroscopio così come letti dal convertitore

Tabella 1 - Codifica errori display LCD.

bit	Segnalazione
0	Se "1" indica driver motori attivo.
1	Se "1" indica che la tensione della batteria è sotto la soglia minima.
2	Se "1" indica che la tensione della batteria è sopra la soglia massima (non implementato).
3	Se "1" indica che la corrente assorbita dalla batteria ha superato la soglia massima.
4	Se "1" indica che si è verificato un errore nella lettura del potenziometro dello sterzo.
5	Se "1" indica che si è verificato un errore nella lettura dell'accelerometro.
6	Se "1" indica che si è verificato un errore nella lettura del giroscopio.
7	Se "1" indica che si è verificato un errore nel calcolo dell'angolo
8	Se "1" indica che si è verificato un errore nei parametri contenuti nella EEPROM

ADC del microcontrollore. Si tratta dei valori grezzi (RAW) non ancora elaborati in cui per la sola lettura dell'accelerometro è applicata una piccola media per eliminare parte del rumore intrinseco.

Fig. 3

24,67V 12,45A
ACC=502 GYR=385

Da questa schermata premendo P1 e P2 contemporaneamente è possibile entrare nella sezione riguardante l'impostazione dei parametri PID. L'impostazione dei parametri P, I, D avviene tramite i pulsanti P1 e P2 dopo aver selezionato il parametro da modificare con il pulsante Enter. La terza schermata - Fig. 4a - riporta, nella prima riga, il valore dell'angolo di inclinazione di OpenWheels e il valore dello sterzo; nella riga sottostante sono riportati i valori relativi alla tensione inviata ai motori, destro e sinistro; ad un valore negativo corrisponde la rotazione in senso inverso delle ruote.

Fig. 4a

-5,78° STR=4
L 56% R 48%

Per il corretto funzionamento di OpenWheels verificate che il valore dell'angolo aumenti inclinando in avanti il manubrio e che il valore dello sterzo aumenti inclinando a destra il manubrio.

Da questa schermata premendo contemporaneamente P1 e P2 si

avvia la procedura per lo ZERO dell'accelerometro e del giroscopio (Fig 4b); è fondamentale che durante questa procedura OpenWheels rimanga perfettamente immobile in posizione verticale. Questi valori saranno memorizzati nella EEPROM e quindi ad ogni accensione potrete usare immediatamente OpenWheels senza la necessità di eseguire la calibratura. È fondamentale eseguire questa procedura al primo avvio e ogni qual volta OpenWheels tende a muoversi pur mantenendolo in posizione perfettamente verticale.

Fig. 4b

SET ZERO WAIT...

Quando alimentate la scheda di OpenWheels, tramite la porta seriale, sono inviati una serie di parametri relativi all'inizializzazione che possono essere visibili tramite la funzione Serial Monitor dell'IDE di Arduino. Durante la fase di accensione viene inizializzato il display LCD, i registri che saranno usati per il controllo dei moduli PWM e dell'interrupt, il valore di neutro del manubrio e il valore dello zero del sensore di corrente. Via seriale sono inviati i dati dei principali parametri che saranno utilizzati dal firmware, per primi i valori letti dalla memoria EEPROM e usati dal controllo PID, a seguire la scala utilizzata per lo sterzo (SteerScale), i valori dello zero dello sterzo (Steer_Zero), dell'accelerometro (Acc_Zero), del giroscopio (Gyro_Zero) e del sensore di corrente (IBat_Zero).

LA REGOLAZIONE DEL PID

Vediamo ora l'impostazione dei parametri del controllo PID e come essi influiscono sul funzionamento. È opportuno, inizialmente, impostare a zero i tre parametri PID e modificarli con cautela seguendo le indicazioni seguenti. **Non azionate il pulsante di START per nessun motivo se prima non avete azzerato tutti i parametri PID ed impostato lo Zero dei sensori.** Con tutti i parametri a zero anche premendo il pulsante di START non verrà fornita energia ai motori, sul display comparirà comunque la scritta RUN ad indicare che i driver sono stati attivati (Fig. 6).

56%
RUN

Per assicurare la massima sicurezza, abbiamo inserito la funzione di Soft-Start che consente di azionare i motori gradualmente quando si preme il pulsante di START e quando lo si rilascia, evitando bruschi movimenti del mezzo. Infatti, potrebbe capitare di avere OpenWheels leggermente inclinato, azionando il pulsante di START si avrebbe una brusca partenza, rendendo il mezzo incontrollabile. Con la funzione di Soft-Start, quando si preme il pulsante di START i motori sono azionati gradualmente in un lasso di tempo di circa due secondi, sufficienti per evitare situazioni pericolose; sul display la scritta STANDBY è sostituita dalla scritta RUN solo dopo che sia terminata la procedura di Soft-Start, momento in cui è possibile salire ed utilizzare il mezzo.

Il primo parametro da aggiornare è il termine proporzionale (P) che indica essenzialmente quale velocità raggiunge OpenWheels in base all'inclinazione del manubrio (Fig. 7); un valore elevato significa

che OpenWheels sarà molto sensibile all'inclinazione del manubrio, quindi molto reattivo. Per prendere confidenza con il mezzo consigliamo di partire da un valore basso, ad esempio 50, ed eventualmente alzarlo successivamente, a passi di cinque, dopo aver preso confidenza con il mezzo. Se provate ora a premere il pulsante di START (ma senza necessariamente salirci sopra) vedrete che inclinando il manubrio OpenWheels prende velocità; se provate a salirci noterete che difficilmente rimarrete in equilibrio. Ricordiamo che OpenWheels necessita delle giuste regolazioni e in questa prima fase non è garantito il funzionamento ottimale; occorre anche mettere in conto l'inevitabile inesperienza con questi tipo di mezzi, pertanto consigliamo di utilizzare sempre la massima cautela adottando sistemi protettivi come caschetto, ginocchiere e polsiere.

KP=110

Il passo successivo (Fig. 8) consiste nel regolare il parametro derivativo (D): esso agisce sulla stabilità di OpenWheels perché influisce sui motori in conseguenza di repentine variazioni di inclinazione. È bene iniziare dal valore zero ed alzarlo progressivamente piano piano, man mano che si fanno delle prove; valori troppo bassi impediscono la stabilizzazione e quindi durante la marcia OpenWheels tende a cadere in avanti o indietro, mentre valori troppo alti tendono a portare una sovra compensazione nella quale i motori cominceranno a vibrare.

KD=30

Procedete per passi successivi, alzando gradatamente l'uno (P) e

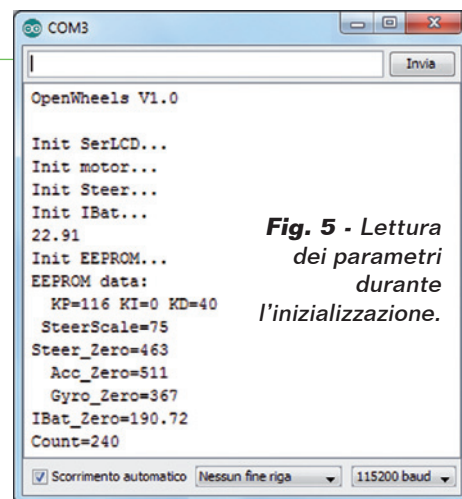


Fig. 5 - Lettura dei parametri durante l'inizializzazione.

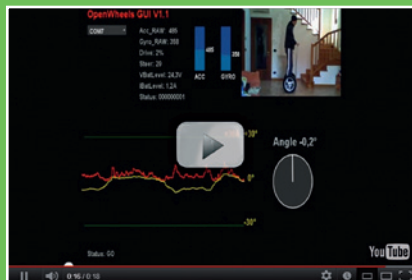
l'altro (I) parametro, sino ad ottenere la completa stabilità del mezzo; la regolazione di questi soli due parametri permette il funzionamento di OpenWheels.

Il parametro integrativo (I) in Fig. 9, pur rientrando di fatto nel ciclo della stabilizzazione, in pratica influisce sulla modalità di conduzione del mezzo nel lungo termine. Infatti quando si raggiunge una certa velocità è necessario mantenere inclinato in avanti il manubrio e conseguentemente si poggeranno i piedi su di una pedana a sua volta inclinata; questo a lungo andare è scomodo. Il parametro integrativo, essenzialmente, svolge una funzione di incremento progressivo della velocità del mezzo. Essendo noi stessi, tramite l'inclinazione del manubrio, a regolare la velocità, questo si traduce in un naturale e progressivo arretramento del manubrio, consentendo di raggiungere la velocità di crociera desiderata ma con la pedana in posizione orizzontale. Occorre essere molto cauti con questo parametro poiché l'incremento automatico della velocità può essere pericoloso se non si riesce a gestirlo al meglio; consigliamo di partire dal valore zero e operare per incrementi di una unità, con la massima cautela.

KI=5

Sul sito dedicato a OpenWheels potrete visionare alcuni video relati-

Guarda i test del firmware di OpenWheels



Vuoi vedere come si comporta Open Wheels in casa e all'esterno con il firmware descritto in questo articolo?

Scansiona il QR Code con il tuo smartphone!



vi alla sua conduzione; alcuni video fanno riferimento alla telemetria che, tramite la tecnica del picture-in-picture, permette di vedere l'andamento dei dati acquisiti in relazione alla conduzione del mezzo. L'analisi di questi dati consente di apportare migliorie al firmware e di valutare l'effettiva funzionalità della parte meccanica, soprattutto per quanto riguarda il rapporto di riduzione tra i motori e le ruote. Esiste un quarto parametro (**Fig. 10**) denominato *Steer* che consente di impostare la sensibilità dello sterzo; dalle nostre prove abbiamo visto che il valore 75 è risultato ottimale. Più alto è questo parametro e maggiore sarà la sterzata in proporzione all'inclinazione (destra e sinistra) del manubrio. OpenWheels dispone di un'alta manovrabilità ed è in grado di ruotare su se stesso da fermo, questo però potrebbe essere un problema ad alta velocità perché sarebbero possibili virate così strette da diventare pericolose oltre che inutili. Per tale motivo abbiamo dotato il nostro mezzo di una sorta di sterzo progressivo in cui la sensibilità di sterzata vari in base alla velocità del mezzo; in questo modo per basse velocità si potranno eseguire curve molto strette (ed

anche ruotare su se stessi) mentre, man mano si aumenta la velocità, saranno possibili curve sempre meno accentuate. Anche questo è un meccanismo di sicurezza per impedire che involontariamente si commettano delle manovre pericolose che mettano a rischio l'incolumità del passeggero.

Fig. 10

Steer=75

I principali parametri, di uso comune, sono modificabili tramite il display LCD, mentre altri sono impostabili solo modificando le opportune righe di programma; si tratta di parametri più specifici che consigliamo di modificare solo se in possesso delle dovute conoscenze.

I PARAMETRI NEL FIRMWARE

Questi valori sono riportati nelle prime righe del modulo principale del firmware, tra essi troviamo il parametro *Cycle_Freq* che si riferisce alla frequenza (Hertz) del ciclo della stabilizzazione. Verrebbe voglia di aumentare questo valore nel tentativo di incrementare le prestazioni di OpenWheels. In realtà, viste le masse in gioco, non sono necessarie frequenze troppo alte, inoltre occorre fare i conti con i limitati tempi a disposizione per i calcoli matematici. Alcuni parametri sono necessari alla definizione dello stato dell'alimentazione, primo fra essi il valore *VBatScale* che definisce la scala nella misura della tensione della batteria; il valore è legato al partitore di tensione formato dalle resistenze R10 e R11. Il parametro *VBatLmax* indica il valore di tensione per il quale la batteria si considera pienamente carica, mentre il valore *VBatLmin* è il valore al di sotto del quale la batteria si considera scarica. Questo comporta anche uno stato di errore ed il conseguente stato di STANDBY di OpenWheels. Tutti questi parametri

sono già stati fissati tenendo conto delle prove che abbiamo effettuato, ma è probabile che alcuni di essi - come ad esempio il limite massimo della corrente oppure la minima tensione della batteria - possano essere modificati in base a nuove prove. Un serio problema riscontrato nelle nostre prove riguardava la stabilizzazione del mezzo al variare della tensione di alimentazione, infatti quando la tensione della batteria scende, proporzionalmente diminuisce la tensione sui motori facendo mancare di fatto la necessaria potenza al controllo. Per evitare questo inconveniente utilizziamo la lettura della tensione della batteria per compensare la potenza sui motori così da risultare indipendente dallo stato di carica della batteria; le prestazioni, compresa la velocità massima, risultano così costanti sino alla completa scarica della batteria. Nella lettura della corrente sulla batteria, il parametro *IBatScale* definisce la scala con la quale viene valutata la corrente sulla batteria e il valore è ricavato in base al valore misurato dal sensore di corrente U1. Il valore *IBatMAX* è di massima importanza in quanto definisce il limite massimo di corrente erogata dalla batteria oltre il quale OpenWheels si porta immediatamente nello stato di STANDBY. Nel firmware sono presenti le variabili *IbatLevel* e *IbatLevelSmooth* che si riferiscono rispettivamente alla misura di corrente istantanea (ovvero quella misurata ad ogni ciclo) e quella media. Il valore istantaneo viene usato per limitare la massima corrente assorbita, mentre il valore medio è quello usato nella telemetria. Nel modulo denominato IMU trovate i valori *Acc_Scale* e *Gyro_Scale* che fanno riferimento alla scala adottata per i due sensori presenti nella scheda elettronica. Abbiamo già spiegato come ricavare questi valori, la loro modifica è necessaria solo se userete un modello diverso di sensori. ■

mOway: il piccolo robot superaccessoriato

di SIMONE MAJOCCHI

Robot e scuola hanno stabilito da tempo anche in Italia un lungo sodalizio: con la robotica i ragazzi di scuole medie e superiori possono cimentarsi in attività che li mettono di fronte alla risoluzione dei problemi, all'analisi, alla logica e alla programmazione, pure offrendo una forte componente di sfida e l'opportunità di lavorare in gruppo. Negli anni le piattaforme utilizzate si sono avvicendate, con iniziative tecnologiche specifiche di singoli istituti contrapposte talvolta a scelte concordate a livello nazionale.

Ora c'è un nuovo arrivato che – almeno sulla carta – sembra avere tutti i requisiti per erodere quello che fino ad oggi era terreno quasi esclusivo del mattoncino più famoso del mondo.

Il progetto mOway è spagnolo e si basa su un attento studio delle esigenze pratiche degli studenti e degli insegnanti. Rispetto ad altre soluzioni, si tratta di un robot semovente a due ruote e di dimensioni sorprendentemente contenute, ma ciò nonostante, ha delle caratteristiche di rilievo.

Chiuso ma non troppo

Ad un primo sguardo, mOway si presenta come una specie di mini veicolo con due

Pensato per la scuola e per gli appassionati di robotica, questo piccolo robot è un concentrato di tecnologia, immaginata per soddisfare esigenze precise.

grandi ruote posteriori e un ball caster nascosto nella parte frontale. La scocca superiore blu metallizzata si abbina con un certo stile al silver delle altre parti, mentre un ampio slot nella parte superiore lascia immaginare la possibilità di aggiungere qualcosa. L'architettura di mOway si basa su un PIC 18F86J50 a 4MHz che pilota due motori in continua dotati di riduttore ed encoder per il feedback digitale del loro funzionamento che, in pratica, diventa simile a quello di un servomotore.

Risolto l'aspetto del movimento, mOway dedica le risorse del processore e di altri circuiti di supporto a una schiera di sensori che permettono di avere un controllo non indifferente sull'ambiente circostante. In Fig. 1 trovate lo



Fig. 1

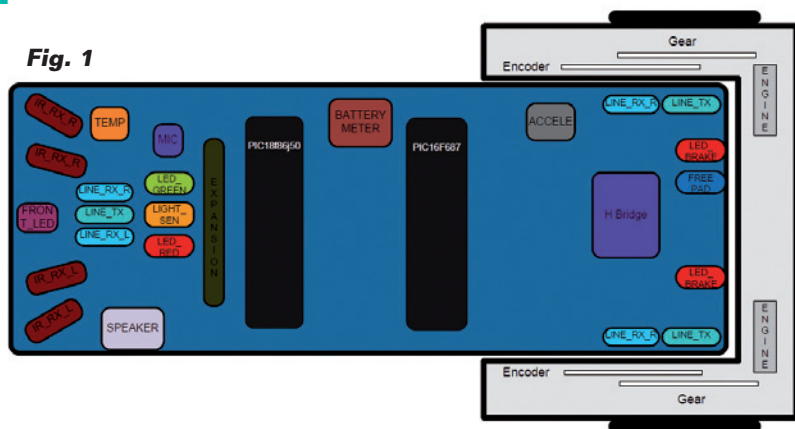


Fig. 2

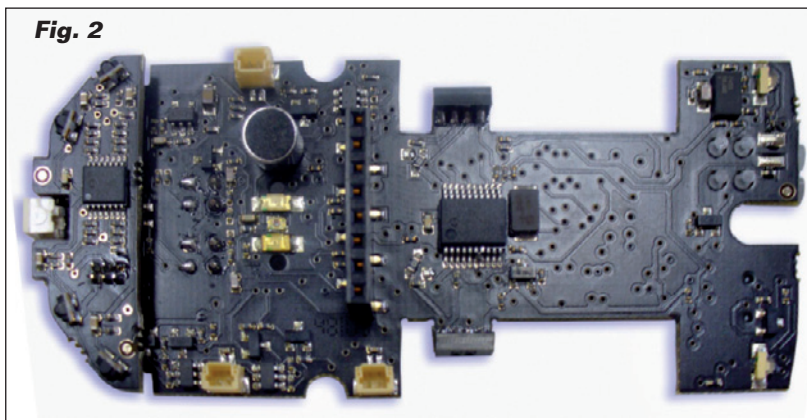
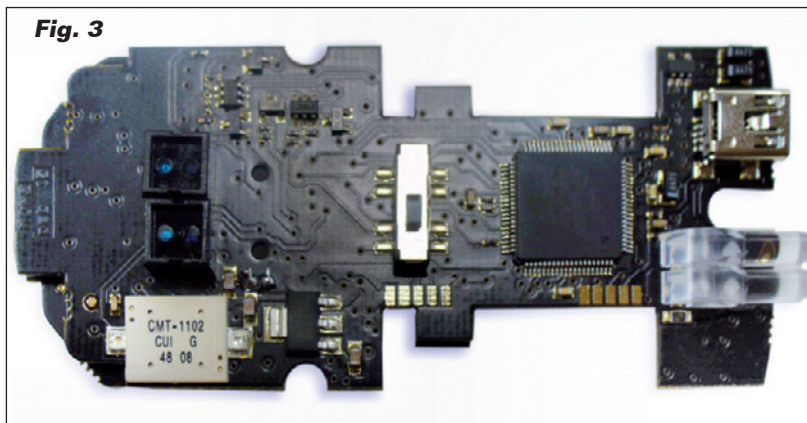


Fig. 3



schema dei vari componenti presenti sulla basetta, mentre in **Fig. 2** e **Fig. 3** potete vedere la basetta vera e propria su entrambi i lati.

Nella parte inferiore si trovano due sensori di linea, quelli che emettono un raggio infrarosso e lo rilevano con un fototransistor e che solitamente permettono di realizzare con facilità un programma inseguitore di linea. Essendo due a distanza ravvicinata, rendono ancora più semplice restare sulla linea scura se questa ha una larghezza adeguata (sono visibili a sinistra in **Fig. 3**, sopra

l'altoparlante che è chiaro). Il gruppo successivo di sensori è sempre di tipo ottico e sfrutta due emettitori a infrarossi posti a sinistra e a destra del musetto per illuminare gli eventuali ostacoli che quattro sensori rileveranno per riflessione. Questi sensori sono orientati per "sentire" oggetti circa a 45 gradi rispetto all'asse centrale e sono gestibili singolarmente.

Il sensore successivo è per la luce e sfrutta un ingegnoso sistema a guida ottica per convogliare la luce raccolta da una finestrella a mezza luna posta frontalmente sopra al mOway (**Fig. 4**). La luce rilevata è quindi quella di fronte al musetto e questo permette al robot di capire da che parte questa proviene e agire di conseguenza. Un sensore di temperatura posizionato sulla basetta vicino ai quattro sensori frontali permette di leggere la temperatura ambiente; si tratta di una NTC collegata ad uno degli ingressi analogici e fa parte dell'arsenale di sensori con i quali immaginarsi delle applicazioni e dei progetti.

Il microfono, posizionato internamente e quindi soggetto al rumore generato dal motore e dagli ingranaggi, viene letto sia a livello digitale, sia analogico per consentire al robot di rilevare presenza di rumore e l'intensità del rumore stesso; ovviamente la lettura per essere significativa dovrebbe essere effettuata a robot fermo, così da non avere rumori interni.

Dietro a uno dei LED posteriori si trova l'accelerometro a tre assi di Freescale (MMA7455L di tipo MEMS) che il PIC gestisce attraverso quattro pin di ingresso e due di uscita, mentre un pin analogico è dedicato alla lettura della carica della batteria che alimenta l'intero circuito.

Proseguendo nell'analisi dei collegamenti, troviamo quattro LED indirizzabili singolarmente; si inizia con quelli posteriori che sono visti come un unico LED, poi il led frontale, che è di tipo bianco ad elevata luminosità, mentre dalla stessa guida ottica del sensore di luce viene convogliata la luce degli ultimi due led di colori diversi.

Un altoparlante permette infine di emettere suoni di varie frequenze.

In **Tabella 1** trovate il riassunto dei collegamenti che fanno capo al microcontrollore. Da notare come sia previsto anche un pin

Fig. 4



Tabella 1

Pin PIC	I/O	Sensor
PORT A		
RA0	I	Light
RA1	I	Central left infrared receiver
RA2	I	Right line sensor receiver
RA3	I	Side left infrared receiver
RA5	I	Left line sensor receiver
PORT B		
RB1	I	First interruption of the accelerometer
RB2	I	Second interruption of the accelerometer
RB3	O	Speaker
RB5	O	Top red LED
RB6	O	Top green LED
PORT C		
RC7	O	Front LED
PORT D		
RD1	O	Line sensor transmitter
RD4	I	SD0 signal for the SPI communication (accelerometer)
RD5	O	SDI signal for the SPI communication (accelerometer)
RD6	O	Clock signal for the SPI communication (accelerometer)
RD7	I	Chip Select for the SPI communication (accelerometer)
PORT E		
RE5	O	Brake LED
PORT F		
RF5	I	Side right infrared receiver
RF6	I	Central right infrared receiver
PORT H		
RH5	I	Temperature sensor
RH6	I	Battery measurer
RH7	I	Microphone
PORT J		
RJ6	O	Infrared transmitter
RJ7	I/O	Free pad

libero (RJ7) per aggiungere ulteriore elettronica al robot mOway.

La vera peculiarità di questo prodotto sta però nel suo connettore, accessibile dallo slot che si trova nella parte superiore, grazie al quale è possibile inserire una serie di schede. I collegamenti al microcontrollore accessibili tramite questo connettore sono riportati in **Tabella 2**.

Lunga durata

Un altro elemento da non sottovalutare è l'uso di una batteria LiPoly che garantisce in condizioni normali quasi due ore di funzionamento, mentre la presenza di schede nello slot di espansione può ridurre l'autonomia che, comunque, è sempre molto interessante per chi deve portare il proprio mOway in gara e non può permettersi di rimanere senza energia. La ricarica avviene tramite un collegamento miniUSB a 5V, an-



Fig. 5

Tabella 2

Pin Expa	I/O	PIC
Pin1	O	Vcc 3.3v
Pin2	O	GND
Pin3	I/O / PMD3 / AN12 / P3C / C2INC	RH4
Pin4	I/O / PMA5 / AN7 / C2INB	RF2
Pin5	I/O / SCK1 / SCL1	RC3
Pin6	I/O / SD01 / C2OUT	RC5
Pin7	I/O / SDI1 / SDA1	RC4
Pin8	I/O / INT	RB0

che sfruttando un collegamento alla porta USB di un computer.

Le schede accessorie

La possibilità di potenziare le funzionalità di questo robot attraverso l'inserimento nell'apposito slot di schede aggiuntive è sicuramente una buona idea e già ora sono disponibili quattro diverse schede testate e funzionanti: RFusb, WiFi, videocamera (**Fig. 5**) e Millefori (**Fig. 6**).

In particolare, la scheda RFusb (**Fig. 7**) permette di comunicare facilmente fra robot e computer, ma anche fra più robot, senza doversi avventurare nelle complicazioni della comunicazione senza fili in WiFi. La modalità con cui il microcontrollore dialoga con il chip di ricezione – un nRF24L01 prodotto da Nordic Semiconductors – è tramite il protocollo SPI. Altri dispositivi compatibili con questo protocollo sono quindi ideali per il collegamento a mOway tramite il connettore di espansione.

Fig. 6



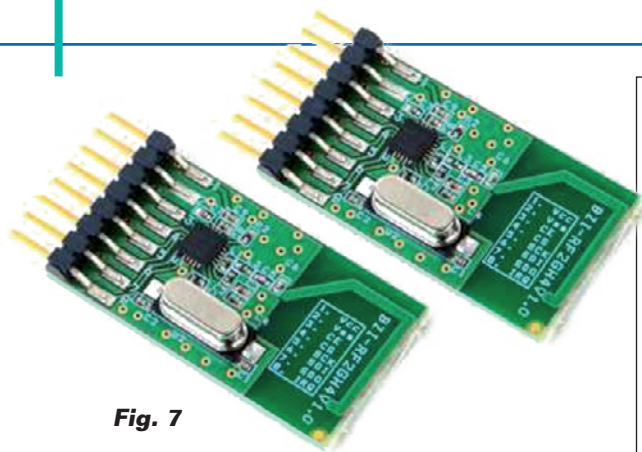


Fig. 7

Programmare mOway

Trattandosi di un robot basato su un microcontrollore della famiglia PIC 18F di Microchip, ed essendo fornito con ampia documentazione su tutti i vari aspetti della componentistica e dei sensori collegati, si ha la possibilità di programmarlo partendo dal livello più basso, ovvero attraverso l'assembler o il C18.

Con mOway viene fornito un CD di installazione in cui si trova una serie di programmi, driver e tool, inclusi i link agli strumenti Microchip necessari alla programmazione e al test a basso livello. Chi vuole può quindi programmare con l'MPLab IDE potendo anche contare sulle librerie di supporto necessarie a pilotare nel modo corretto tutte le varie funzionalità del robot; le librerie fanno parte del pacchetto d'installazione mOway World Pack. Al proposito, la documentazione è ben dettagliata e illustra passo passo come accedere agli esempi in assembler che permettono di capire e gestire tutti i vari aspetti dell'hardware; in **Fig. 8** abbiamo riprodotto un esempio di come va programmato il mOway tramite l'IDE MPLab.

Data l'importanza della scheda RFusb per l'interazione e il controllo del robot, è fornita anche la libreria specifica da utilizzare in MPLab per accedere alle funzionalità di comunicazione senza fili.

Ovviamente l'approccio con assembler e C18 è adatto agli istituti superiori e a quegli appassionati che vogliono il pieno controllo del robot, dei suoi sensori e dei tempi di reazione del codice. Chi invece vuole sfruttare un approccio più orientato al problem solving, al ragionamento e all'apprendimento, può utilizzare l'ambiente di programmazione mOwayWorld che sfrutta il paradigma del flowchart per consentire anche ai meno esperti di creare del codice eseguibile con pochi clic.

```
list p=18F86j50
;* PIC register definition
#include "P18F86j50.INC"
;* Reset vector
org 0x1000
goto INIT
;* Program memory
org 0x102A

;*****[MOWAY LIBRARY]*****
#include "lib_sen_moway.inc"
;*****

INIT
;Microcontroller configuration to use the sensors
call SEN_CONFIG
;Green LED blink
call LED_TOP_GREEN_ON_OFF
;*****[MAIN PROGRAM]*****

MAIN

;Check central left obstacle sensor
movlw OBS_CENTER_L
movwf SEN_CHECK_OBS
call SEN_OBS_DIG

;If a obstacle is detected robot avoids it
btfsc SEN_OBS,0
call LED_FRONT_ON
btfss SEN_OBS,0
call LED_FRONT_OFF
goto MAIN

goto MAIN
;*****
```

Fig. 8

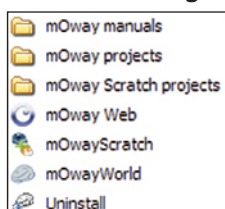
mOway World Pack

Il cuore del pacchetto software di mOway è mOwayWorld, l'evoluzione della precedente interfaccia grafica mOwayGUI. Per installarlo si parte dal kit di distribuzione che contiene tutto il necessario all'interfacciamento e alla programmazione, dai driver per Windows fino agli esempi in assembler. Quando viene lanciato, controlla anche la presenza di eventuali versioni più aggiornate e propone il download in caso affermativo. In **Fig. 9** abbiamo il risultato dell'installazione completa del pacchetto.

Nella nostra installazione si nota anche la presenza di mOwayScratch, una particolare soluzione che utilizza il collegamento RF fra mOway e computer per passare alla programmazione delle interazioni tramite Scratch (una soluzione che ricorda la modalità Firmata + Processing del mondo Arduino) che sfrutta la potenza del computer per pilotare secondo un programma realizzato tramite flowchart e condizioni il robot, mentre quest'ultimo si limita a mandare i dati dei sensori e a ricevere i comandi per i propri motori e LED.

Tornando all'interfaccia di mOwayWorld, vi-

Fig. 9



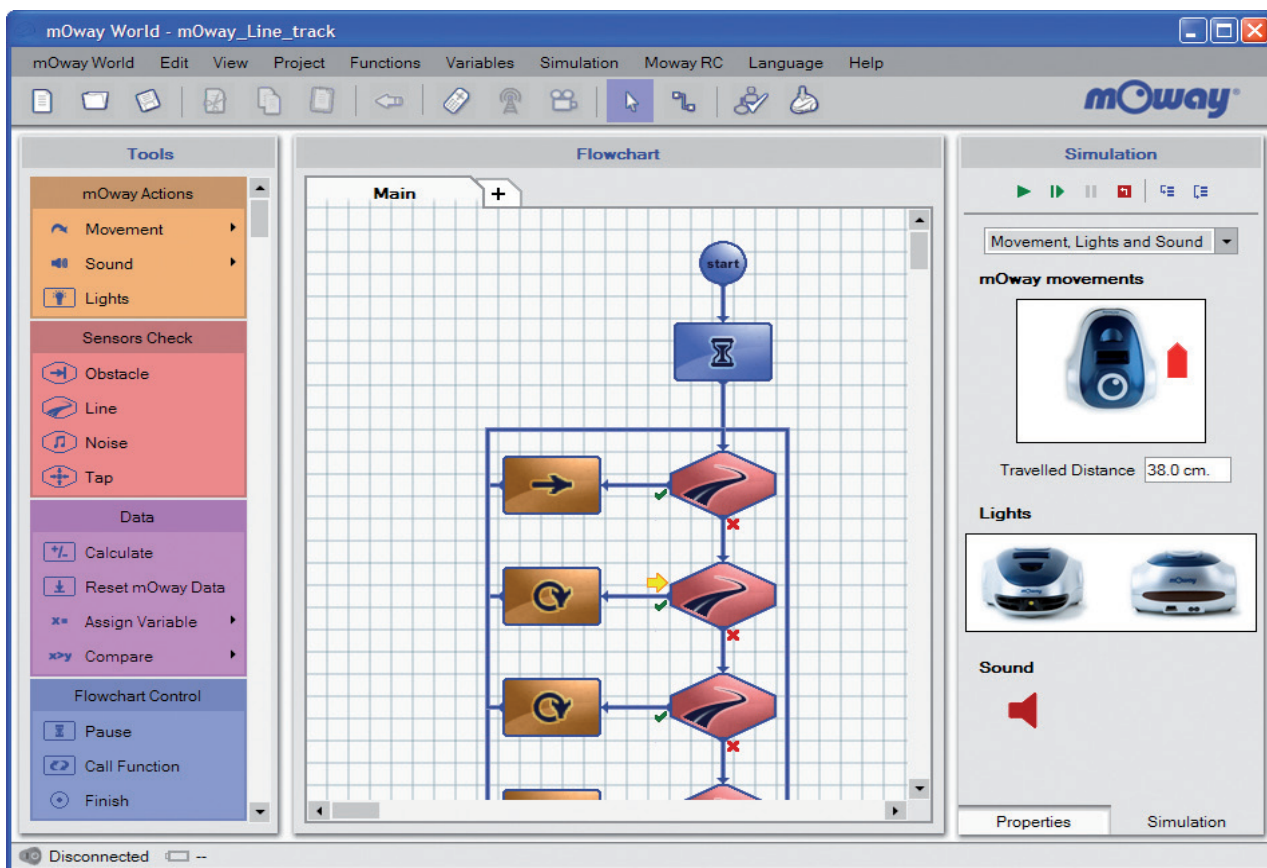


Fig. 10

Fig. 11

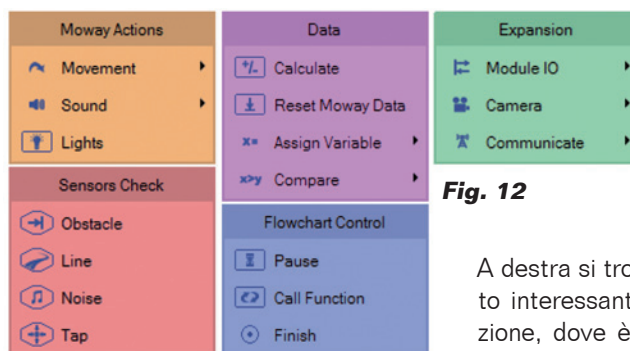
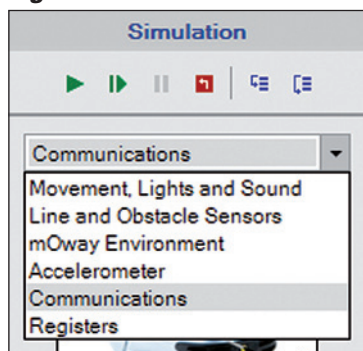


Fig. 12

sibile nella Fig. 10, si notano quattro aree strutturate in blocchi, finestre e barra degli strumenti.

Nella parte superiore troviamo i menu e le icone: da qui è possibile caricare e salvare i progetti, avviare la simulazione, attivare la comunicazione RF o trasferire il codice sul robot. A sinistra si trova la palette con tutte le funzioni e i blocchi che possono essere usati per costruire il diagramma di flusso (o flowchart) che ha il proprio spazio nell'area centrale, quadrettata, dell'interfaccia. Oltre al flusso principale è anche possibile creare delle funzioni (o subroutine) in sottopagina.

A destra si trova invece un'area molto interessante, quella della simulazione, dove è possibile avere varie rappresentazioni della situazione dei sensori e del robot, così da poter interagire e simulare virtualmente qualsiasi situazione reale (Fig. 11).

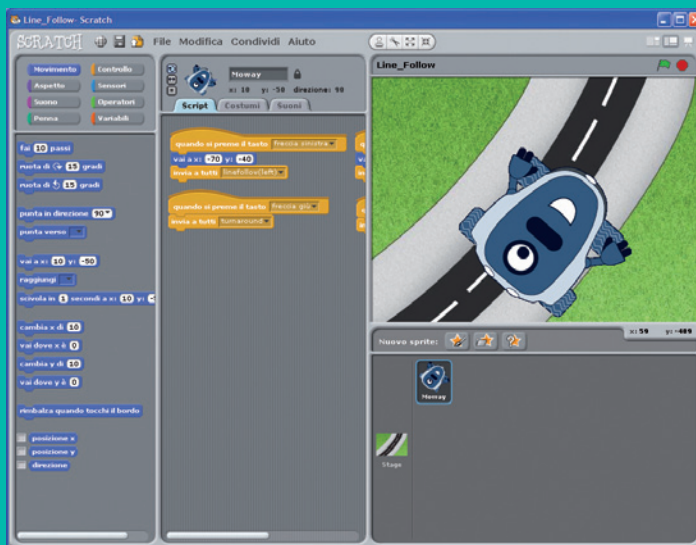
Da questa interfaccia è anche possibile telecomandare tramite scheda RF il mOway, caricando su di esso l'opportuno programma RadioControl.

In Fig. 12 abbiamo tutti i blocchi disponibili per creare i programmi e si può notare come l'arsenale a disposizione sia completo. Per ogni blocco trascinato sul diagramma vanno poi definiti i dettagli.

Con un doppio clic si apre il pannello con le varie opzioni disponibili e questo permette di intervenire con grande precisione su ciascun

Scratch

Scratch è sviluppato dal Lifelong Kindergarten Group dei Media Lab del MIT, con il supporto finanziario della National Science Foundation, di Microsoft, Intel Foundation, Nokia, Iomega, e dei consorzi di ricerca dei Media Lab del MIT. È un linguaggio di programmazione che rende semplice creare storie interattive, animazioni, giochi, musica e arte e condividere le creazioni sul web. Quando i ragazzi creano e condividono i loro progetti Scratch, acquisiscono importanti idee matematiche e computazionali e allo stesso tempo imparano a pensare creativamente, a ragionare con sistematicità e a lavorare in collaborazione. Con mOway, Scratch acquista una dimensione fisica che consente ai ragazzi di creare da un lato in un ambiente grafico e dall'altro di interagire con un ambiente reale; in questa situazione i sensi sono di mOway, il ragionamento è dell'applicazione scritta in Scratch, e le azioni



sono nuovamente di mOway. L'interfaccia di programmazione Scratch è colorata e interattiva, proprio per attirare l'attenzione dei più piccoli.

aspetto funzionale del programma in fase di costruzione.

In Fig. 13, ad esempio, vediamo tutte le opzioni possibili per il blocco "Movimento", dove si parametrizzano tutte le possibili opzioni di rotazione del robot fino ad ottenere esattamente il comportamento desiderato. Il livello di dettaglio offerto da questo sistema di sviluppo del codice è, a nostro avviso, molto più elevato di quanto ci si potrebbe aspet-

tare, e per questo il livello e la complessità delle applicazioni può crescere ben oltre le aspettative iniziali. La funzione di controllo sulla coerenza e funzionalità del diagramma di flusso permette di verificare almeno la correttezza formale del diagramma realizzato, mentre la funzione di visualizzazione del codice permette di vedere l'assembler strutturato in blocchi commentati prima che questo sia trasformato in file esadecimale scaric-

Fig. 13

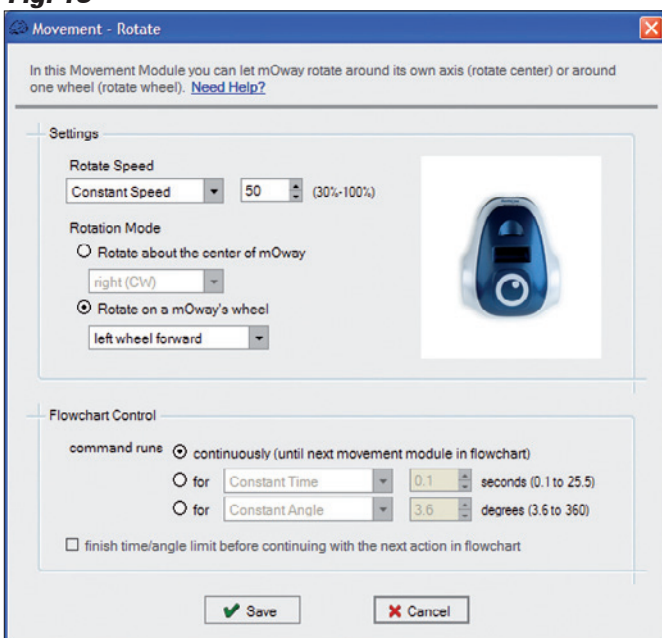
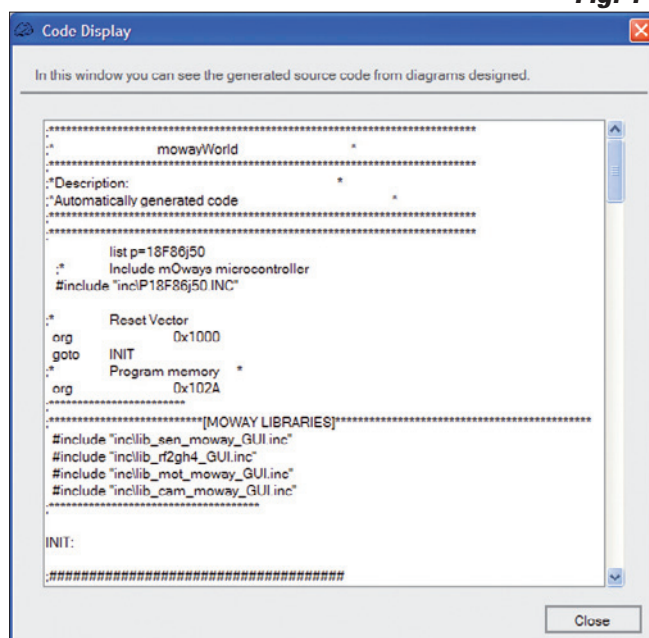


Fig. 14



cabile sul microcontrollore. Questa modalità di visualizzazione del codice assembler non solo offre un modo per avvicinarsi a questa modalità di programmazione, ma consente anche interventi di rifinitura sul codice generato, oppure l'uso del diagramma di flusso per creare un'ossatura assembler su cui intervenire in un secondo momento.

In Fig. 14 l'inizio del codice per la modalità "line follower", parte degli esempi disponibili.

Un supporto continuo

La ricchezza della soluzione mOway per la scuola, ma anche per chi vuole sperimentare seriamente con questo tipo di robotica, non si ferma a quello che è incluso nella scatola: sul web il produttore permette di avere a disposizione lezioni, esempi e altro materiale didattico e sta facendo nascere una community. Purtroppo la lingua è ancora un ostacolo in quanto manca l'italiano, ma questa lacuna verrà sicuramente colmata a breve, visto l'interesse anche nel nostro paese per questo robot.

Il supporto è composto da una serie di video tutorial, da documenti che guidano passo passo nella realizzazione di esercitazioni specifiche e da download che raggruppano in pratici archivi compressi la documentazione e il codice necessario all'esercitazione.

A livello di video, è possibile vedere praticamente cosa questi robot riescano a fare sia singolarmente, sia in gruppo, evidenziando le reali capacità di reazione e di rilevazione del territorio circostante. Interessanti anche i video che mostrano come il collegamento RF sia utilizzabile fra robot per coordinare operazioni e comportamenti "da sciame".

Conclusioni

Molte piattaforme robotiche, recentemente presentate sul mercato e proposte ai corpi docenti in varie nazioni, nel nome della flessibilità e dell'espandibilità sono state realizzate con strutture di una certa dimensione e complessità. Il progetto mOway va sicuramente controcorrente e apparentemente il robot sembra più un giocattolo che una piattaforma robotica, mentre internamente contiene il set necessario di sensori già ben posizionato e ingegnerizzato; il tutto conta sul supporto di un ambiente di programmazione di tipo



flowchart molto flessibile e di spessore. Meno elettronica "allo scoperto" e maggiore attenzione alle applicazioni quali la programmazione e lo sviluppo di processi e ragionamenti per la risoluzione di problemi; meno saldatore sempre collegato e più opportunità educative – quella del modulo RF per dialogare fra robot ci pare una bella innovazione – che spaziano dalle scuole medie per i classici problemi di labirinti e line follower fino ad arrivare agli istituti tecnici che possono interagire con una smart city ricca di stimoli e problematiche d'interazione. □

I prodotti Minirobots sono distribuiti in Italia da Futura Elettronica srl. mOway è disponibile nella confezione basic con 1 robot al prezzo di 180,000 Euro e nella confezione De Luxe Kit (Euro 362,00) che comprende 2 mOway robot, 2 moduli RF a 2,4 GHz da inserire nel bus di espansione, 1 chiavetta USB RF, CD con software "mOway World" e manuale. I prezzi si intendono IVA compresa.

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



per il MATERIALE

L'ITALIA ALLA SHELL ECO-MARATHON

L'agguerrito gruppo di partecipanti italiani a questa competizione può contare su team di comprovato valore; da mesi lavorano al perfezionamento dei propri veicoli con l'intenzione di primeggiare in varie categorie.



a cura di SIMONE MAJOCCHI

Mentre state leggendo queste pagine, a Rotterdam 13 team italiani stanno gareggiando nella Shell Eco-marathon con altri 200 team provenienti da tutto il mondo per superare ogni record in materia di efficienza energetica; è la più importante e innovativa competizione per studenti che si svolge ogni anno in Europa, America e Asia, con Rotterdam che è il luogo scelto per

l'Europa. Le gare europee si sviluppano su un percorso cittadino avendo a disposizione solo un litro di carburante che per alcuni team è sufficiente a coprire una distanza pari a quella fra Rotterdam e Mosca. I veicoli, suddivisi nelle due categorie Prototipi e Urban Concept utilizzano sia carburanti tradizionali come benzina, diesel, etanolo e GTL, sia carburanti alternativi come idrogeno e batterie elettriche.

Le giornate dal 15 al 19 maggio vedranno gareggiare per il nostro paese anche "H2politO - molecole da corsa" di Torino e "ZeroC" di Carpi, da cui abbiamo raccolto una testimonianza diretta del loro lavoro.

H2politO

Il Team H2politO – molecole da corsa è composto da studenti di Ingegneria del Politecnico di Torino



**L'attuale Team Leader di H2polito
Alessandro Messana.**



Il Team H2polito 2013.



**IDRApegasus - 2012 - EUROPEAN SHELL
ECO-MARATHON, Rotterdam (Olanda): Percorsi
3023 km/L di benzina equivalente, 337 km/kW.**

**XAM - 2012 - EUROPEAN SHELL ECO-MARATHON,
Rotterdam (Olanda): Percorsi 107 km/L, Primo
premio "Communication Award".**



**XAM 2.0 - 2012 - FUTURE CAR CHALLENGE Londra:
percorsi i 100 km che separano Brighton da Londra.
Premio Best Extended Range Vehicle Prototype.**

con un obiettivo comune: quello di partecipare ed eccellere alla **Shell Eco-marathon**. Ogni studente contribuisce a diversificare ed ampliare le conoscenze del Team, spaziando tra molteplici discipline legate ai corsi di laurea di appartenenza: Autoveicolo, Meccanica, Informatica, Elettronica, Aerospaziale, Energetica, Matematica, Materiali, Telecomunicazioni, Meccatronica, Gestionale, Cinema e Mezzi di Comunicazione e Design Industriale. È proprio questa eterogeneità di conoscenze che permette di affrontare un problema da più parti, sviluppando capacità di comunicazione e comprensione reciproca, per poi concordare una soluzione comune.

La missione del Team è quella di affiancare alla fortissima preparazione teorica che il Politecnico offre, il "mettere in pratica" le conoscenze acquisite per essere in grado di portare a compimento un progetto, dalla carta alla pista.

Il risultato della passione e del lavoro del Team sono diversi veicoli realizzati e sviluppati dal 2008 ad oggi, già divenuti punti di riferimento della loro categoria alla Shell Eco-marathon Europe. In particolare:

IDRA08, IDRA 09 (che implementata è diventata IDRA10 e poi IDRA11) e IDRApegasus prototipi a tre ruote due anteriori e una posteriore mossi da una fuel cell a idrogeno che alimenta un motore

elettrico ad alta efficienza (200 W). XAM (eXtreme Automotive Mobility) un urban concept ibrido parallelo alimentato a etanolo.

Poi nel 2012 il Team ha realizzato XAM 2.0 un urban concept ibrido serie di tipo ERev per la partecipazione alla Future Car Challenge da Brighthon a Londra, un veicolo targato in grado di correre non solo più su un circuito ma su strade e autostrade ad una velocità massima di 80 km/h.

Ecco dalle parole del Team Leader Alessandro Messana gli obiettivi del Team H2polito per la Shell Eco-marathon di quest'anno a Rotterdam: *"Quest'anno il team sta lavorando molto sul powertrain delle due vetture. L'obiettivo principale con IDRApegasus è quello di arrivare tra i primi due in classifica e migliorare la nostra percorrenza di almeno altri 500 km/L. Con XAM, gareggeremo per entrare finalmente tra i top 3 team nella categoria ibridi alimentanti a etanolo. Il team è molto motivato da questo punto di vista, e le possibilità di raggiungere questi obiettivi potrebbero veramente essere a portata di mano. Basterà impegnarsi costantemente e lavorare duramente fino a maggio."*

ZeroC

Escorpio è il nome del prototipo con il quale il team ZeroC - gruppo di studenti e insegnanti dell'ITIS "Leonardo da Vinci" di Carpi (MO) - gareggia alla Shell Eco-marathon;

il prototipo deriva da una delle numerose invenzioni di Leonardo Da Vinci, genio dal quale anche la scuola prende il nome.

Il team ZeroC è nato grazie ad una campagna di sensibilizzazione riguardo le polveri sottili nel 2006, effettuata perché l'Emilia Romagna risultava (e tutt'ora risulta) essere una regione molto più inquinata rispetto alla media nazionale. Il team ZeroC ha iniziato a partecipare alla S.E.M. dal 2007, attirando fin da subito molti ragazzi i quali hanno potuto applicare ad altissimo livello la teoria studiata sui banchi di scuola, creando con le proprie mani (grazie anche all'aiuto di alcuni docenti come: Stefano Covezzi, Marco Vidoni, Riccardo Arletti, Elisabetta Ferretti Franco Bonaccini e Aureliano Levani) un prototipo innovativo, fino al 2012 alimentato tramite idrogeno.

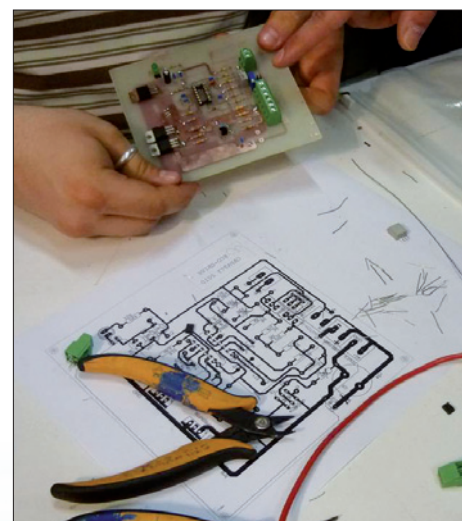
Nelle sei edizioni passate il team ha sempre raggiunto importanti risultati classificandosi a livello europeo sempre nei primissimi posti, ottenendo nel 2011 il primo posto per le scuole europee e conquistando il prestigioso Technical Innovation Award nel 2012.

Per il 2013 la vettura è stata modificata in modo abbastanza "importante". Nel 2012 ZeroC ha partecipato alla sezione ad idrogeno mentre quest'anno partecipa alla sezione electric battery, con le seguenti caratteristiche: la macchina utilizza, per il powertrain, 10 batterie di tipo LIPO (polimero di litio) per un totale di 37V - 6.7Ah; nella parte superiore della carrozzeria è integrato un modulo fotovoltaico costituito da 10 celle, con possibilità di utilizzo dell'energia proveniente dal modulo sia nel powertrain, sia sotto forma di accumulo nella batteria principale; Escorpio usa un BMS (Battery Management System) che controlla la gestione della carica in entrata e in uscita alla batteria, coordinato

da un Energy Manager (fatto dal team) che supervisiona i flussi di energia e sovrintende alla parte riguardante la sicurezza elettrica del funzionamento dell'intero sistema, ed infine viene usato un MPPT (Maximum Power Point Tracking) che ottimizza l'efficienza del pannello fotovoltaico.

Il team ZeroC partecipa alla gara di Rotterdam con la speranza di ottenere buoni risultati, anche se questo è il loro primo anno in questa categoria. Pochissimo tempo fa (Febbraio 2013), grazie alla gentilissima e generosa collaborazione con Fondtech Aerolab, il team ha testato l'aerodinamica del veicolo nella Galleria del vento di Sant'Agata bolognese (BO). Verso Aprile, grazie ad un'altra gentile e generosa collaborazione instaurata con Autodromo di Modena, il team ha eseguito una sessione di prove per testare sul campo il funzionamento generale del prototipo, poco prima della partenza per Rotterdam.

Il prossimo mese potremo sapere quali traguardi sono stati raggiunti dai nostri studenti in questa competizione e potremo presentarvi nuove testimonianze in proposito. □





**ZERO
ROBOTICS**
SUMMER SPHERES PROGRAM

ZERO ROBOTICS: PER DUE ANNI IN FINALE

“Tutto il lavoro di squadra è stato svolto in piena autonomia. I professori avevano solo il compito di supervisionare l'andamento generale, dandoci la possibilità di prendere da soli le decisioni, anche quelle più importanti.”

di CESARE IACOBELLI e ANTONIO SPANO - ITIS PININFARINA - MONCALIERI (TO)

Il concorso Zero Robotics è stato creato nel 2009 dal Laboratorio di Sistemi Spaziali del Massachusetts Institute of Technology (MIT) negli Stati Uniti d'America.

Viene introdotto per la prima volta in Europa nel 2011 in via sperimentale per coinvolgere altre nazioni nel mondo (Italia, Germania, Gran Bretagna, ecc.). Nel 2012 la gara si estende ad altre nazioni europee (Spagna, Portogallo, Olanda) rendendo la competizione un vero campionato Europeo. In questi due anni la competizione, pur avendo molte fasi in comune e facendo scontrare squadre dei due continenti, ha sempre tenuto classifiche separate (USA, Europa).

Gli spheres

La competizione ha come obiettivo programmare un piccolo

satellite a basso costo (50.000 euro) in grado di muoversi in assenza di gravità e che opera normalmente sulla stazione spaziale internazionale (ISS).

La difficoltà della sfida sta nel controllare - tramite un software sviluppato dagli studenti - il satellite che deve effettuare determinate azioni. I satelliti hanno forma quasi sferica e si muovono liberamente in assenza di gravità, grazie alla spinta esercitata dalla fuoriuscita di anidride carbonica da alcuni ugelli, posizionati sulla superficie degli Spheres. La prima parte della gara (qualificazioni e semifinali) si svolge in un ambiente simulato computerizzato tridimensionale, in grado di far percepire ai partecipanti il reale comportamento dei satelliti nello spazio. Nella fase finale i programmi vengono caricati direttamente

sugli Spheres e questi vengono fatti muovere direttamente all'interno della stazione spaziale. Per rendere la competizione più realistica, il MIT inventa ogni anno una storia, creando uno scenario di fantasia in cui i robot devono muoversi.

Nel 2011 la Terra, a corto di un gas chiamato Helium 3, manda in missione nello spazio i piccoli Spheres, con il compito di estrarre da due asteroidi differenti l'Helium 3. Nel 2012 si tratta invece di spostarsi da una base all'altra evitando di incappare in detriti spaziali lasciati dall'astronave concorrente.

Il lavoro degli studenti

Le diverse possibilità offerte



Tabella delle posizioni raggiunte nelle due edizioni

Edizione	2011-12	2012-13
Eliminatorie	48° su 153 a livello mondiale, 12° su 20 a livello europeo	13° su 34 a livello europeo
Semifinali	2° su 6 alleanze europee	5° su 6 alleanze europee
Finali	3° ASSOLUTI	2° ASSOLUTI

dagli scenari portano all'importanza di scegliere la giusta strategia per ottenere il maggior punteggio e battere l'avversario. La gara si svolge infatti tramite scontri diretti tra 2 spheres.

Diventa importante avere quindi studenti che si occupano di programmazione e studenti che si occupano di strategia.

Gli strategi si sono cimentati nello studio di ogni piccola sfaccettatura delle regole di gioco, per ideare delle possibili strategie da implementare nella programmazione degli Spheres. Il requisito minimo per lo studio del manuale è stato quello di una buona conoscenza dell'inglese tecnico per avere una piena comprensione del testo.

Dopo la stesura di un'ipotetica strategia, i programmatori la implementano tramite il codice. Per la compilazione del programma, i programmatori hanno studiato a fondo la parte del manuale contenente le funzioni pre-programmate messe a disposizione dei partecipanti che permettono di effettuare i principali movimenti dei satelliti. Il linguaggio utilizzato è il C e l'ambiente di sviluppo è disponibile all'interno del sito di Zero Robotics.

Nelle due competizioni il lavoro degli studenti si è svolto prevalentemente al di fuori dell'orario scolastico, sia fermandosi dei pomeriggi a scuola, sia tramite incontri in videoconferenza (con Skype). Questo è poi diventato indispensabile quando si è creata l'alleanza con le altre squadre collocate a Roma e in Germania.

Durante il periodo che ha preceduto le prime eliminatorie, si sono messi a disposizione alcuni responsabili del progetto per tenere delle lezioni/chiarimenti e per valutare l'avanzamento dei progressi dei team torinesi.



Organizzazione della gara

Nella prima fase ogni squadra ha gareggiato con tutte le squadre europee e americane; alla fine sono state stilate delle graduatorie separate per Europa e Stati Uniti. Le simulazioni sulla piattaforma sono state effettuate dal MIT ed i risultati comunicati sul sito della manifestazione.

Nella seconda fase, le migliori 18 squadre europee sono state divise in tre gruppi - in ordine decrescente di punteggio - e si sono formate le alleanze in cui dovevano essere presenti una squadra per gruppo e non ci poteva essere più di 2 squadre della stessa nazione; in questo modo sono stati creati 6 team europei.

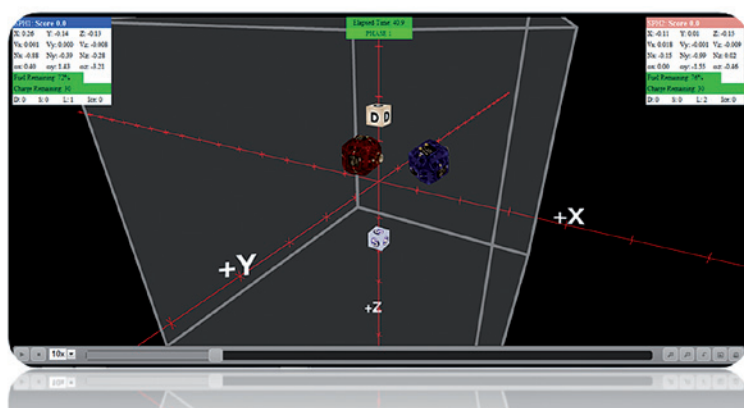
L'appartenenza a nazioni diverse ha spinto gli studenti a dialogare con compagni con conoscenze e metodologie di lavoro spesso diverse. I migliori team europei (3 nel 2011 e 6 nel 2012) avrebbero assistito in diretta, presso la sede dell'ESA a Noordwijk in Olanda alla competizione finale con le sfere reali a bordo della ISS.

Cosa hanno detto i partecipanti

Ecco il commento scritto da alcuni degli studenti che hanno partecipato alla competizione.

Edizione 2011-12: *"Inizialmente, il nostro gruppo contava più degli attuali cinque membri. Successivamente ai primi incontri, soprattutto quelli organizzati dai responsabili del progetto, alcuni componenti della squadra, avendo compreso le difficoltà che avrebbero dovuto affrontare, si sono tirati indietro.*

Tutto il lavoro di squadra è stato svolto in pie-



na autonomia. I professori avevano solo il compito di supervisionare l'andamento generale, dandoci la possibilità di prendere da soli le decisioni, anche quelle più importanti. Questo ci ha permesso di responsabilizzarci, rendendoci più consapevoli del fatto che il successo dipendeva esclusivamente dalle nostre capacità e impegno. Grazie a questi abbiamo ricevuto anche una lettera dal Ministro dell'Istruzione Francesco Profumo, il quale si è complimentato con noi per il successo ottenuto e questo è stato motivo di grande orgoglio. Non sono mancati i momenti di svago e divertimento, soprattutto durante il soggiorno ad Amsterdam. Oltre ad aver visitato la città, abbiamo avuto l'opportunità di far parte di un tour illustrativo all'interno dell'ESA dove c'è stato offerto un ricco banchetto. Non abbiamo riscontrato particolari problemi tranne per la questione comunicativa tra i vari team della stessa alleanza. Abbiamo avuto alcuni problemi sull'accordarsi sugli orari per sviluppare insieme il progetto. La nostra alleanza era formata da due squadre tedesche e la nostra. Le squadre tedesche sono state un po' reticenti a comunicare con noi. Loro erano disponibili solamente la mattina, quando invece noi avevamo lezione. In definitiva l'esperienza è stata molto costruttiva e divertente. Ci ha permesso di sviluppare uno spiccato spirito di squadra nonché approfondire la lingua inglese. Questa è stata un'esperienza unica e irripetibile che ricorderemo per sempre e che auguriamo ai futuri partecipanti di poterla vivere, proprio come l'abbiamo vissuta noi."

Ecco invece la lettera inviata da due dei partecipanti del 2012-2013 ai propri compagni:

"La nostra avventura cominciò ad aprile dell'anno scorso, quando alla Robocup jr (un'altra competizione riguardante la robotica) conoscemmo i reduci della vecchia squadra "WhyNotPinin" che ci raccontarono la loro esperienza proprio come stiamo facendo noi con questa lettera. Ebbene, noi fummo subito colpiti e chiedemmo ai nostri professori alcune

Team che ha partecipato alla competizione del 2011. Da sinistra: Russo Matteo, Fabbro Luca, Patanè Davide, Russotto Carmelo Andrea e accucciato Di Chio Boris.



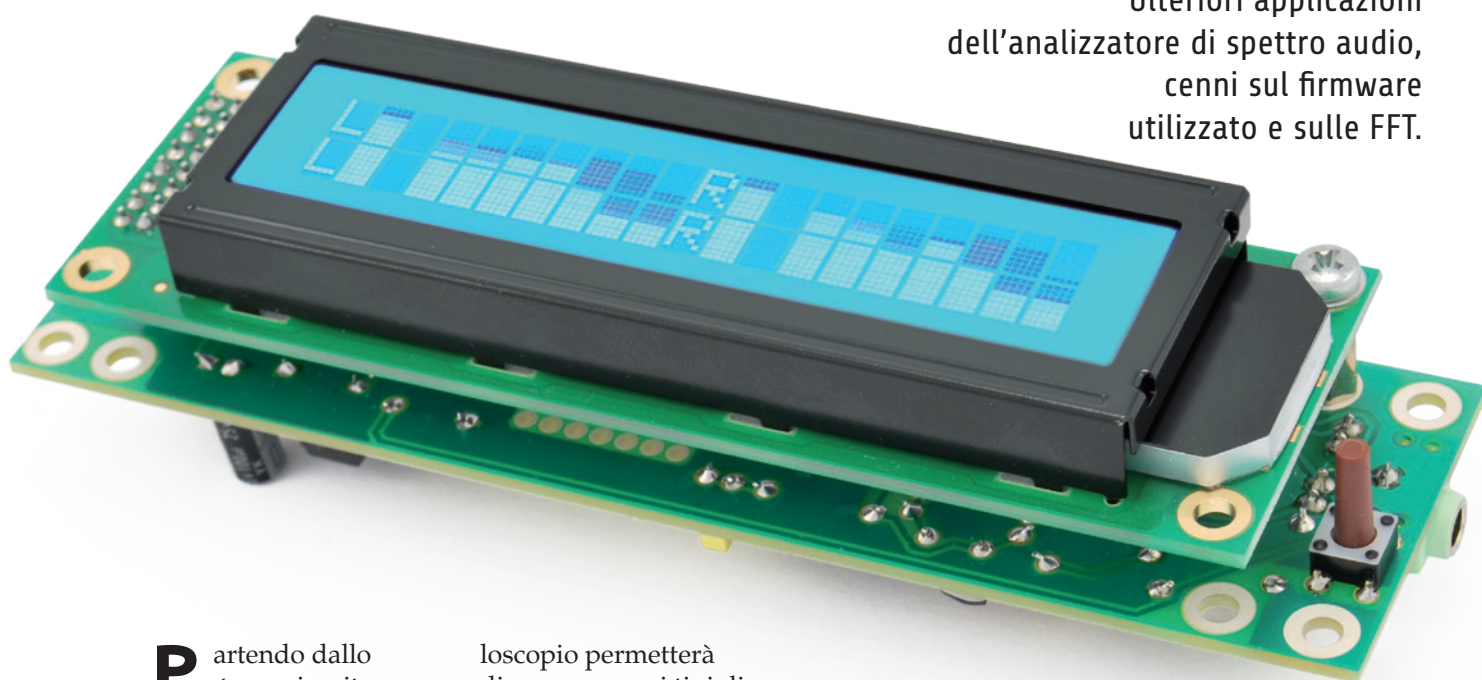
Team che ha partecipato alla competizione del 2012. Da sinistra: Saggese Davide, il Prof. Spano, Feno Dennis, Trezza Alessandro, Rossi Matteo, Marrella Igor, Ponzino Flavio.

dritte per cominciare con la Zero Robotics. Essendo finalisti dell'anno precedente, le selezioni erano già superate, quindi si è cominciato a lavorare "seriamente" da settembre. Assieme alle colonne portanti della nuova squadra "WhyNotPinin", i nostri esptissimi compagni di quinta, e assieme ai promettenti compagni di terza, abbiamo raggiunto senza troppe difficoltà il cuore della competizione, la formazione delle alleanze che ci saremmo portati avanti fino alla finale. Ricordo come se fosse ieri quel sabato di novembre così determinante anche per la formazione, volendo, della classifica finale. Da quel momento siamo diventati un tutt'uno con i nostri Amici del liceo Democrito di Roma e della squadra "LazySpace" di Berlino. Il codice per guidare i piccoli Spheres (i satelliti che si trovano all'ISS) era uno per tutta la squadra alleata e il lavoro di squadra è stato fondamentale. A colpi di scambi di algoritmo tramite Skype e modifiche sull'IDE del Mit, siamo riusciti ad arrivare tra le prime sei alleanze, che avrebbero assistito all'esecuzione del proprio codice direttamente dall'ISS e di conseguenza partecipato alla finale della competizione. Il codice per la finale è stato consegnato "a porte chiuse" rispetto alle altre fasi di gara a circa metà di dicembre e fino alla simulazione finale, non abbiamo saputo niente riguardo la gara. Sarebbe stata comunque una soddisfazione il semplice fatto di poter assistere dal vivo a tutto ciò, ma con un po' di fortuna (è servita anche quella) abbiamo raggiunto il secondo posto in Europa. Adesso, a due mesi circa dalla finale, possiamo dire di aver tratto da questa esperienza i requisiti necessari per provare a fare ancora meglio l'anno prossimo, sperando di trasmettere a nuovi compagni di squadra l'esperienza che è stata trasmessa a noi. Con l'augurio di aver condiviso nel migliore dei modi la nostra avventura, auguriamo a tutti una "spaziale" giornata." □

ANALIZZATORE DI SPETTRO AUDIO FFT SU DISPLAY LCD

dell' Ing. SALVATORE COSENTINO

Ulteriori applicazioni dell'analizzatore di spettro audio, cenni sul firmware utilizzato e sulle FFT.



Partendo dallo stesso circuito descritto nel numero di Marzo 2013, in questa puntata aggiungeremo un semplice circuito di filtro passa-basso con la funzione di integrare un segnale PWM generato su uno dei piedini liberi del microcontrollore (di seguito μ CU) per applicarlo ad un oscilloscopio esterno. Questo segnale replicherà le informazioni mostrate sul display LCD, ma la maggiore risoluzione dello schermo dell'oscil-

loscopio permetterà di generare vari tipi di spettro con diverse risoluzioni, ben più definite rispetto al piccolo LCD. In questo caso saranno immediatamente visibili le limitazioni del nostro circuito che genera fino a 64 singole barre di frequenza con risoluzione di solo 8 bit: ciò si traduce in una dinamica di ingresso di circa 40dB. Un secondo scopo di questo articolo, del tutto slegato da quello primario del circuito, è quello di illustrare l'uso

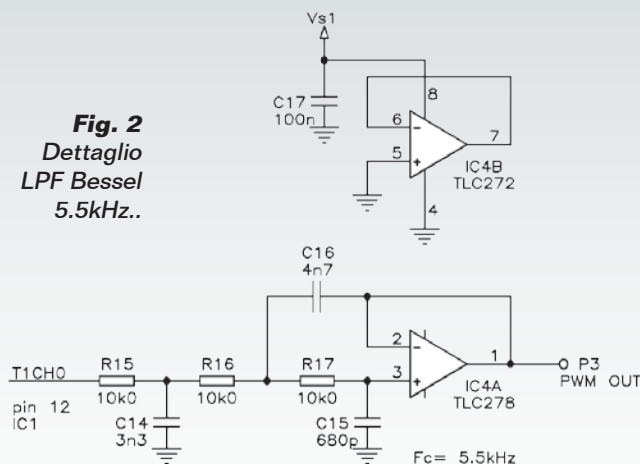
pratico della tecnologia PWM (Pulse Width Modulation, ovvero modulazione dell'ampiezza dell'impulso) per la conversione di un segnale digitale in analogico. In un box dedicato in queste pagine esaminiamo la procedura

FFT mentre il pezzo è incentrato sulla descrizione della generazione e dell'elaborazione del segnale PWM perché, volendo approfondire un po' la teoria, c'è parecchio da dire nonostante la semplicità del circuito proposto.



Fig. 1

Fig. 2
Dettaglio
LPF Bessel
5.5kHz..



DESCRIZIONE DEL CIRCUITO

Utilizziamo uno dei piedini liberi del nostro μ CU per generare un segnale PWM che rappresenta l'analisi di spettro in maniera analogica sullo schermo di un oscilloscopio; si tratta del pin 12 collegato all'uscita del canale 0 del Timer 1. Il circuito esterno è costituito semplicemente da un filtro passa-basso attivo, realizzato usando un singolo amplificatore operazionale dei due contenuti in un circuito integrato ad 8 piedini. Il circuito, data la sua semplicità, viene montato all'interno della piccola area milliferi riservata agli "esperimenti" ricavata all'interno del circuito stampato proposto, con la tecnica del montaggio punto-punto e utilizzando i terminali dei componenti per la connessione diretta tra uno e l'altro; vengono aggiunti un paio di fili isolati per i collegamenti più lunghi. Lo schema del circuito è riportato in Fig. 2.

L'Op-amp suggerito, il TLC272 è un economico integrato in tecnologia CMOS il cui range di modo comune (tensione ammessa) comprende la tensione negativa, in questo caso la massa, sia per l'ingresso, sia per l'uscita, permettendo l'alimentazione a tensione singola. Poiché la tensione massima di questo integrato è di poco superiore ai 12V, ma l'uscita non riesce ad arrivare vicino alla tensione di alimentazione (ci si approssima fino a circa 1,6V in meno), affinché l'uscita del circuito riesca a raggiungere la piena tensione generata in PWM dal μ CU (+5V), occorre che la tensione di alimentazione di IC4 sia maggiore di 5V. Per questo motivo ricaveremo questa tensione, Vs1, collegandoci al terminale positivo di C1 ove sono presenti circa 11V. L'integrato proposto come Op-Amp è una delle scelte migliori come rapporto costo / prestazioni / assorbimento e disponibilità, alimentato a 10-12V ed è quello che suggeriamo in caso di acquisto, ma non è obbligatoriamente l'unica soluzione. Un'alternativa per questo esperimento è il vetusto LM348, un circuito bipolare a basso assorbimento e di prestazioni poco

Non crediamo che sia il caso di approfondire troppo l'argomento FFT in questo contesto. Le FFT rappresentano una tecnica matematica piuttosto complicata che coinvolge la teoria dei numeri complessi e necessita di basi adeguate per essere pienamente descritta e capita. Non è il caso generico dell'hobbista tipico e risulta arduo pure a chi, come l'autore, queste cose le ha studiate molto tempo fa. E' però interessante capire cosa sono di base, a cosa servono e come sono implementate nel programma della nostra μ CU.

FFT è l'acronimo di Fast Fourier Transform ovvero, in italiano, Trasformata di Fourier Veloce. La Trasformata di Fourier è una tecnica matematica che estrae il contenuto armonico, ovvero quello nel dominio della frequenza, di un segnale variabile nel tempo. Per questo motivo è spesso utilizzata, come nel nostro caso, per realizzare analizzatori di spettro o - più genericamente - per compiere analisi di contenuto spettrale per particolari elaborazioni dei segnale quali quelle di compressione audio-video. La Trasformata di Fourier è descritta da formule complesse che possono essere (più o meno) facilmente sviluppate su computer e microprocessori di adeguata potenza. Il calcolo eseguito dai computer viene svolto su un numero finito di campioni temporali del segnale e non rappresenta compiutamente l'analisi spettrale generica di un segnale temporale, ma solo di un suo frammento che può essere ripetitivo o meno come l'onda generata da un generatore di segnali o una conversazione umana. Per questo motivo, nel caso dei computer, si parla più precisamente di Trasformata di Fourier Discreta (ovvero per un numero finito di punti): in inglese il termine corrispondente è DFT (Discrete Fourier Transform). La FFT è un caso particolare di DFT che si può applicare quando il numero dei campioni è pari a una generica potenza N di 2, ovvero 4, 8, 16, 32, 64, 128, 256, ecc.. Questo riduce notevolmente il numero dei calcoli da effettuare per calcolare la Trasformata e, pur restando comunque un compito arduo, lo rende più facile e quindi "veloce" da eseguire su un computer o un microprocessore. Da qui il nome.

L'algoritmo della FFT è stato riscoperto relativamente di recente da Cooley e Tukey, due scienziati dei laboratori IBM, nel 1965 in modo indipendente, ma era già noto a Gauss che lo aveva descritto nel 1805 in un suo diario. L'algoritmo che svilupparono Cooley e Tukey aprì la strada qualche anno dopo, con l'aumentare progressivo della potenza di calcolo disponibile, all'implementazione delle FFT sui computer in tempo reale. L'algoritmo in questione è di tipo ricorsivo e consiste nel dividere e trasformare in 2 pezzi di $N/2$ gli N campioni di ingresso che costituiscono un "vettore". Questo spiega perché sia particolarmente efficiente quando i campioni sono in numero pari a potenza di 2. Rispetto al caso generico delle DFT sono inoltre eliminati una gran quantità di calcoli intermedi rendendo parecchio più veloce il processo. Un altro beneficio non trascurabile nel caso dei piccoli processori è il fatto che ogni calcolo fatto su una coppia di elementi del vettore di ingresso o dei vettori intermedi rimpiazza gli elementi di partenza che non devono più essere utilizzati. Questo permette di minimizzare l'uso della memoria RAM in cui questi calcoli sono effettuati: come sappiamo la RAM è un elemento costoso e quindi limitatamente presente sui microprocessori. Durante il calcolo vengono generati N vettori composti da coppie di numeri complessi, ove il primo vettore è il campione di dati temporali in ingresso (vettore puramente reale, la cui parte immaginaria è pari a zero), l'ultimo vettore elaborato rappresenta invece i corrispondenti campioni in uscita della frequenza. In questo caso, come in tutti quelli intermedi, il vettore è in realtà composto da 2 vettori separati: uno per la parte reale e l'altra per la parte immaginaria. La somma geometrica dei due vettori genera il "modulo" del campione, ovvero l'energia corrispondente, il diverso contributo dei due componenti è

invece in relazione alla "fase" che, nel nostro caso, è irrilevante. Alla fine del processo di FFT i due vettori reale/immaginario devono essere post-processati per ricavare i 2N campioni di bande di frequenza. Si noti che si parla proprio di "bande di frequenza" e non di frequenze singole, infatti il risultato comprende altrettante bande di frequenza quanti sono i campioni iniziali, distribuite progressivamente tra 0 ed $F_c/2$ e quindi nuovamente tra $F_c/2$ e 0. In pratica il risultato è simmetrico e la seconda metà del vettore del risultato può essere scartata in quanto identica specularmente alla prima! Ne deriva facilmente che le bande di frequenza in cui il risultato è scomposto sono ampie ciascuna quanto $F_c/2N$. Per fare un esempio concreto se F_c , la nostra frequenza di campionamento, è pari a 32000Hz, $N=7$ ovvero il numero di campioni uguale a 2N è pari a 128, ciascuna banda di frequenza del risultato è ampia $32000/128 = 250$ Hz. Per diminuire l'ampiezza di questa banda elementare ed avere maggiore risoluzione ci sono due possibilità:

1. Aumentare proporzionalmente il numero dei campioni. Se ad esempio aumentiamo i campioni da 128 a 1024 ($N=10$), la banda singola passa da 250 a 31,25Hz. Aumenta però in modo sensibile il tempo necessario all'elaborazione della FFT in modo più che proporzionale alla differenza del numero dei valori (circa 11-12 volte invece che 8). Aumenta anche la quantità di RAM necessaria all'elaborazione. Di converso aumenta la risoluzione del risultato in qualsiasi punto della banda elaborata, ovvero i dettagli.
2. Diminuire la frequenza di campionamento in proporzione. Se diminuiamo la frequenza precedente di 8 volte, ovvero la portiamo a 4000Hz, la banda minima sarà ancora pari a 31,25Hz con 128 campioni in ingresso. La massima frequenza rappresentata in uscita sarà però adesso solo 2kHz invece che 16kHz! Il tempo di elaborazione e la quantità di RAM necessaria sarà in compenso il medesimo di prima.

Dettagli sull'algoritmo FFT di Cooley-Tukey

Come accennato, l'algoritmo consolidato delle FFT prevede la trasformazione progressiva di singole coppie dei vettori complessi generici in altrettante coppie tramite l'applicazione di un procedimento di prodotto e somma reciproca e con un coefficiente pari al seno e al coseno

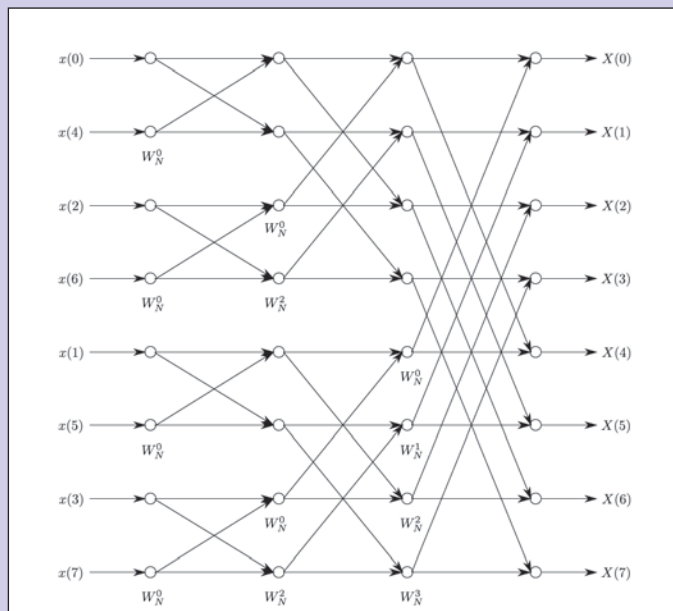


Fig. A - Diagramma a farfalla.

dell'angolo corrispondente alla posizione della coppia nel vettore. Data la coppia complessa FM e FN, si ha:

$$\begin{aligned} FM &= RM + jIM & \rightarrow & FM' = (RM + TR) + j(IM + TI) \\ FN &= RN + jIN & \rightarrow & FN' = (RM - TR) + j(IM - TI) \end{aligned} \quad \begin{aligned} & \text{ove} \\ TR &= RN \cos(\omega) + IN \sin(\omega) \\ TI &= IN \cos(\omega) + RN \sin(\omega) \end{aligned}$$

RM e IM, così come RN e IN, rappresentano le componenti reali e immaginarie di FM e FN. In pratica la seconda coppia è ricavata dal primo valore complesso (FM) cui viene aggiunto o sottratto un secondo termine derivato dal secondo valore della prima coppia (FN) variamente moltiplicato nelle sue componenti reale ed immaginaria da termini pari al seno e al coseno di ω . Questo angolo (ω) varia nelle successive iterazioni del procedimento: è inizialmente pari a 0° e 180° e applicato a metà e metà dei componenti del primo vettore, viene dimezzato a 0° , 90° , 180° e 270° e applicato a quarti delle coppie del secondo vettore, successivamente scomposto in 8 multipli di $22,5^\circ$ e applicato a ottavi delle coppie del 3° vettore, ancora scomposto a multipli di $11,25^\circ$ e applicato a sedicesimi di coppie del 4° vettore e così via fino al vettore N. Come si intuisce il procedimento è piuttosto complesso: ogni trasformazione richiede una decina di operazioni (4 prodotti, 6 somme) che impegnano pesantemente il processore specie se l'aritmetica non è con lo stesso numero di bit suo proprio (calcoli a 16 bit con processori a 8 bit o addirittura calcoli in virgola mobile). Per semplificare e velocizzare lo stesso è indispensabile che tutti i valori utilizzati di seno e coseno siano precalcolati e tabulati in una tabella presente in memoria ROM con la precisione necessaria. Si noti che è necessaria una sola tabella per seno e coseno in quanto è facile applicare $\sin(\alpha) = \cos(\alpha + 90^\circ)$ e ridurne ancora le dimensioni tra 0° e 180° applicando facili trasformazioni trigonometriche tipo $\sin(\alpha) = \sin(180 - \alpha)$. Il procedimento sopra esposto viene applicato a tutte le coppie dei vari vettori secondo un algoritmo detto "a farfalla" (butterfly in inglese) che qui rappresentiamo per 8 campioni (**Fig. A**).

Per complicare un po' le cose (ne abbiamo proprio bisogno!) si può notare che la numerazione dei campioni in entrata non è uguale a quella in uscita. Infatti i campioni in uscita sono in ordine crescente con il loro indice di posizione, quelli in entrata no. Per mettere le cose in ordine occorre spostare i campioni in entrata o quelli in uscita secondo una legge ben precisa, chiamata "bit-reversal". Sebbene spostare i campioni in uscita potrebbe sembrare più logico, è preferibile farlo con quelli in entrata perché in questo caso si agisce su un vettore composto soltanto da termini puramente reali, mentre nel primo caso occorrerebbe spostare i componenti reali e immaginari in uscita, ovvero un lavoro doppio! Prima di procedere al processo vero e proprio dell'algoritmo di FFT, i campioni in entrata vengono quindi riposizionati secondo un nuovo indice di posizione che deriva da quello iniziale convertito in valore binario e invertito a specchio. Per fare un esempio il campione n.21 (0010101 in binario) diventa il n.84 (1010100). Facile no? Una volta finito con l'applicazione dell'algoritmo "a farfalla" il vettore complesso in uscita viene ricomposto in un unico vettore reale calcolando il modulo di ciascun termine in uscita secondo la somma dei quadrati, ovvero il termine N è pari a $X(n) = \sqrt{Xr(n)^2 + Xi(n)^2}$. Come detto all'inizio, la seconda metà del vettore in uscita viene scartata perché simmetricamente uguale alla prima metà.

Lavorando in virgola fissa come facciamo noi sul nostro piccolo processore, tra l'applicazione degli algoritmi su un vettore e il successivo è indispensabile procedere volta per volta a una verifica del valore massimo presente sul vettore e scalare (dividere per 2) tutto il vettore in caso di superamento del massimo consentito per evitare errori di "overflow" (superamento, tracimazione in italiano) che non si tradurrebbero in una semplice tosatura, ma in una forte distorsione sull'intero risultato finale.

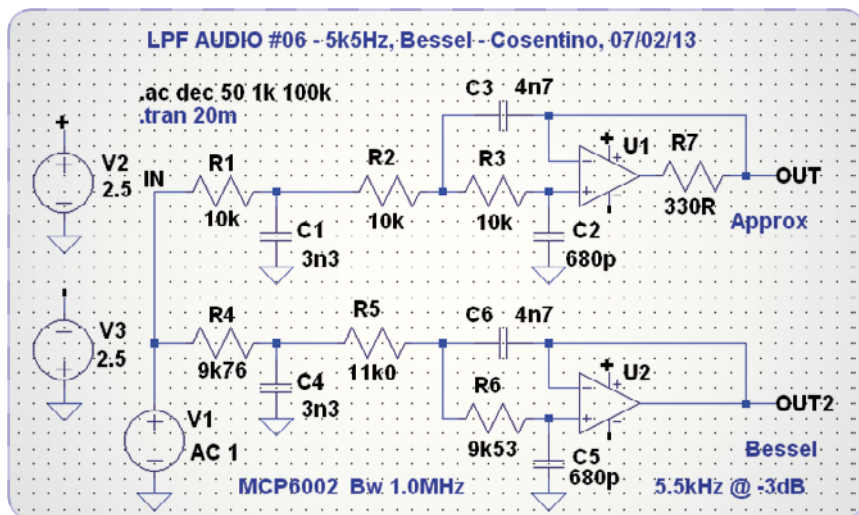


Fig. 3

più che mediocri che è disponibile oramai da oltre 40 anni (!) e per questo è facilmente disponibile nella raccolta di componenti di uso generico nel cassetto dell'hobbista: la differenza di prestazioni misurata sul nostro circuito è nella pratica quasi impercettibile, data la bassa velocità necessaria. Analizzando la risposta all'impulso si nota una leggera esitazione/instabilità quando il segnale sale dopo essersi posizionato in uscita a livello di massa; ciò è dovuto a vari motivi: principalmente alla minore velocità, ma soprattutto al tempo di "svuotamento" delle capacità di giunzione dei transistor del circuito bipolare di uscita. Questa lieve instabilità è in pratica invisibile con il tempo impostato per il display sull'oscilloscopio, pari a 20ms. Un'altra possibilità, inizialmente favorita dall'autore, ma poi scartata nella pratica per l'indisponibilità del circuito adatto in formato "DIP" nel cassetto dell'hobbista citato precedentemente, è quella di adoperare un moderno Op-amp in tecnologia CMOS di tipo "rail-to-rail", capace di approssimare la tensione di alimentazione sia in ingresso, sia in uscita, con alimentazione massima a 5,5-6V. Un esempio eccellente di questo tipo è l'MCP6002 di Microchip, un economico doppio

amplificatore da 1MHz di banda passante, a basso consumo. In questo caso l'alimentazione deve essere ricavata dall'uscita stabilizzata a 5V da IC2 (Vdd) anziché 11V. Come si vede dallo schema, in ogni caso il secondo amplificatore contenuto nell'involucro, inutilizzato, è riferito a massa per non lasciare ingressi aperti. Niente da aggiungere sul resto dei componenti, tutti elementi al 5% con valori standard e di tipo tradizionale.

Vale invece la pena di analizzare un po' meglio il circuito in sé: un filtro passa-basso (LPF l'acronimo inglese) di tipo "Bessel" o a fase lineare. Questo tipo di filtro si differenzia da altri molto usati, come il classico "Butterworth" o a "massima piattezza", o il "Tchebichev" usato in ingresso agli ADC perché la sua curva di risposta, sebbene meno piatta in banda passante e con attenuazione meno ripida in banda attenuata ha la caratteristica di massima linearità di fase. Questa proprietà è molto utile nel nostro caso perché non distorce un impulso a onda quadra in uscita come quelli che genereremo noi: altri filtri avrebbero potuto generare delle fastidiose oscillazioni in salita e in discesa in prossimità dei fronti degli impulsi generati. Anche in questo caso,

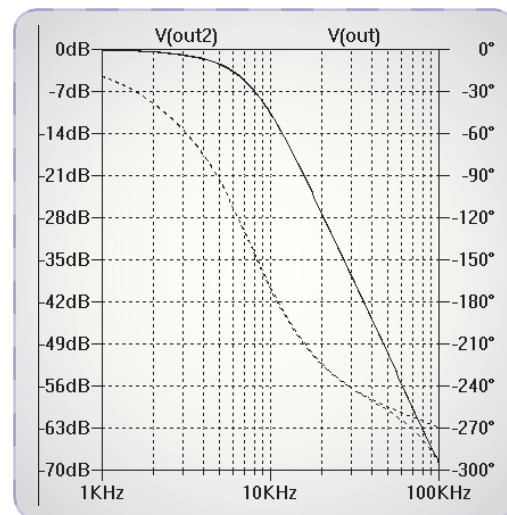


Fig. 4

a scopo di studio, alleghiamo il circuito simulato con LTSpice, con cui è facile compiere esperimenti simulati con diversi componenti ed eventualmente vederne l'azione, sempre su un impulso simulato in entrata. Nello schema proposto abbiamo raggruppato due filtri analoghi, il primo (quello in basso) è realizzato con i componenti teorici che meglio approssimavano la funzione di Bessel, il secondo (quello in alto) con i componenti più vicini scelti nella scala E12, realmente disponibili. Come si può vedere confrontando le

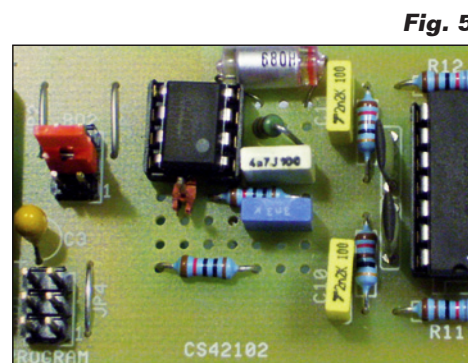


Fig. 5

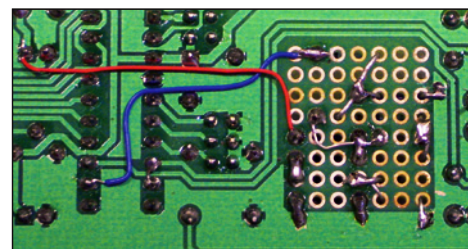


Fig. 6

due uscite sovrapposte nel grafico in uscita alla simulazione, queste sono praticamente coincidenti, confermando la bassa criticità dei filtri di Bessel. Il file LTSpice (LPF audio #06 - 5k5Hz Bessel 3.asc) è disponibile per il download dal sito della rivista assieme agli altri file collegati all'articolo.

REALIZZAZIONE

Data la semplicità del circuito, due fotografie chiariscono meglio come si può montare il circuito all'interno dell'area millefori predisposta dietro al display LCD.

FUNZIONAMENTO

Per il funzionamento occorre caricare il software in versione 1.10 o superiore: questo aggiunge la funzione PWM oltre al normale pilotaggio del display LCD. Con la versione 1.10 sono disponibili due quadri visualizzabili su oscilloscopio contemporaneamente al display LCD: un quadro con tutte le 64 barre di frequenza tra 0 e 16kHz e ciascuna barra ampia 250Hz, e lo stesso quadro con le 7+7 ottave relative ai canali destro e sinistro più i relativi

livelli di picco visualizzati anche su LCD. In ambedue i quadri su oscilloscopio, i segnali sono visualizzati in una finestra, pari a 20ms, che si può facilmente sincronizzare sul fronte in discesa di inizio del quadro, ponendo il trigger attorno ai 4V. La sensibilità è impostata per 4V fondo scala (0dB), 10dB/V: ovvero su schermo sono rappresentati 40dB di dinamica, un po' più che sul display LCD (32dB). A causa di questo è maggiormente visibile il rumore di fondo del processo che, come abbiamo detto nella prima puntata, è all'incirca compreso tra -35 e -40dB dal livello massimo, poco visibile su LCD per la dinamica più limitata, ma già apprezzabile su oscilloscopi con soli 40dB di dinamica. Teniamo anche presente che, per una migliore rappresentazione, la dinamica delle barre a ottave di frequenza sono amplificate di 6dB rispetto alla barra che indica il livello di picco assoluto del segnale. Il segnale PWM è ricavato dalla frequenza di BusClock diviso 256, ovvero è pari a poco più di 78kHz con 20MHz di BusClock.

Il filtro passa-basso di Bessel con $F_t=5,5\text{kHz}$ attenua la frequenza di PWM di oltre 60dB (Fig. 4) rimuovendo ogni traccia visibile della portante di campionamento dall'uscita. Nelle figure successive possiamo osservare il tempo di salita/discesa di un impulso digitalizzato in uscita, pari a 66us e l'ottima corrispondenza con un segnale di onda quadra, ove è assente qualsiasi sovra-oscillazione successiva alla salita/discesa ripida del segnale (Fig. 7 e Fig. 8). L'impulso in ciascuna delle due figure è pari a una singola barra di frequenza digitalizzata in uno spettro a 64 barre in 20ms. Si noti anche come non siano visibili residui della frequenza PWM a 78 kHz.

Per l'abilitazione dell'uscita PWM è necessario posizionare un ponticello su BD2 tra i pin centrali 3 e 4 (Fig. 5): questo agghiuverà la funzione PWM alle 4 funzioni base di visualizzazione sul display LCD ed in contemporanea a 2 di esse. Nelle Fig. 9 e Fig. 10 sono mostrati i display relativi a un segnale sinusoidale in ingresso simultaneamente sui

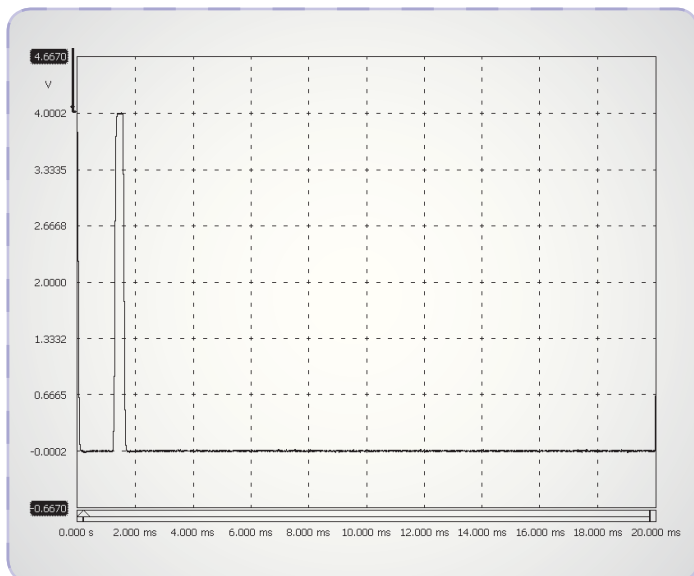


Fig. 7

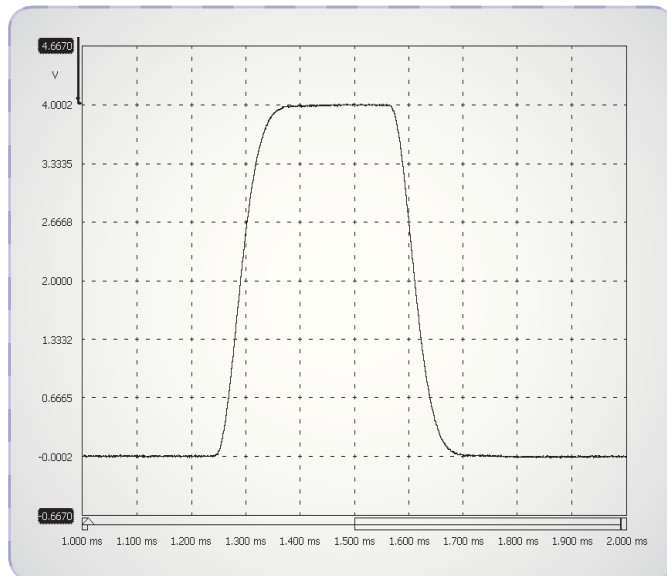


Fig. 8

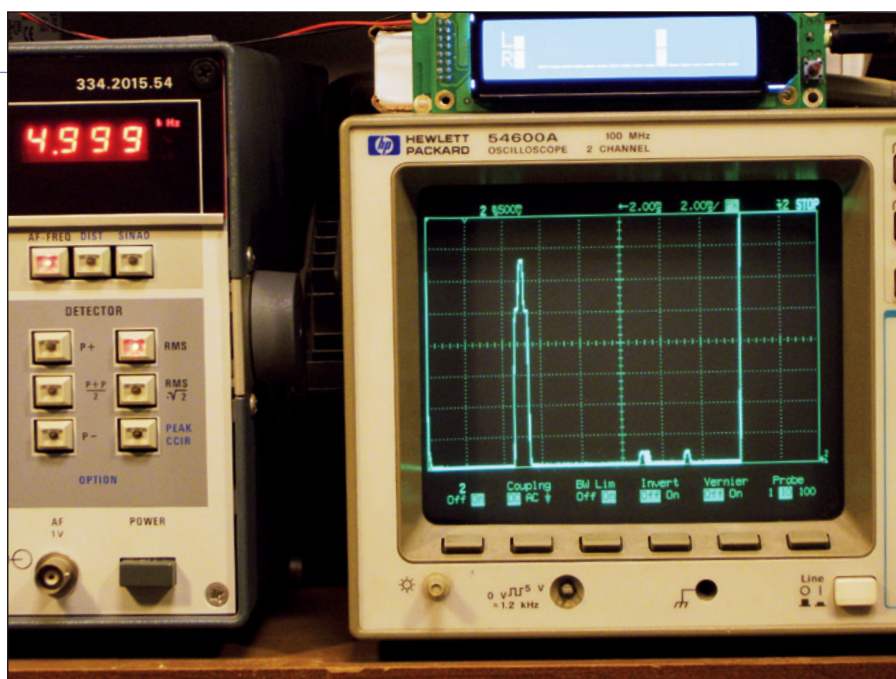


Fig. 9

due canali destro e sinistro, di ampiezza pari a -6dB e frequenza 5kHz.

Si noti il rumore visibile a fondo scala degli oscillogrammi che hanno una dinamica leggermente maggiore dello schermo LCD ovvero 40dB (5dB/div) contro 32 dB (2dB/barretta orizz.). Con segnali di ampiezza maggiore o alcune frequenze particolari (1/4, 1/8, 1/16 di Fc), il rumore è molto più visibile.

Si sono scelti 20ms per un quadro base sull'oscilloscopio perché quasi tutti questi strumenti hanno una scala temporale di 2ms/div, ovvero 20ms per un quadro completo, con rinfresco dello schermo con buona cadenza. Le 64 barre di frequenza tra 0 e 16 kHz vengono invece rappresentate su 16ms (l'oscilloscopio viene mantenuto sempre con la base tempi orizzontale in posizione 2ms/div.) in modo che ad ogni divisione corrispondano esattamente 2 kHz; si veda la **Fig. 9**, dove è rappresentato un segnale a 5,0kHz, proprio a metà della 3a divisione. Nella posizione corrispondente a 16kHz (fine dell'ottava divisione) il segnale va in alto uscendo di schermo e indicando la fine della banda utile. Per ambedue i quadri la

sensibilità verticale dell'oscilloscopio - compresa la sonda di misura - deve essere pari a 0,5V/div., ovvero da 0 a +4,0V sullo schermo.

CENNI SUL PROGRAMMA

Giusto due parole sul programma, semplicemente per dare una idea sulla sua complessità e articolazione. Chi volesse approfondire, può scaricare l'intero programma sorgente, ampiamente commentato e che occupa oltre 20 pagine A4 a doppia colonna da 80 righe l'una e studiarlo comodamente. Intendiamoci, probabilmente metà o più del testo sono proprio i commenti, ma questi sono indispensabili per aiutare a comprendere le funzioni. Il programma è interamente in assembler ed è lungo pochi byte in più di 4kB (4117 Byte per l'esattezza), ovvero occupa solo metà della memoria flash disponibile in un 9S08SH8. La RAM è invece occupata praticamente al 99%: solo 7 byte restano disponibili dei 512 presenti. Il grosso di questo spazio è riservato all'allocazione dei vettori dei campioni iniziali (2x128Byte per i canali destro e sinistro) e altri 128 byte per un terzo vettore dei componenti immaginari generati

durante la conversione FFT dei campioni reali dei due canali. Il rimanente è distribuito tra variabili di calcolo, buffer LCD (40 byte, come i caratteri a display), buffer FFT per i due schermi da 17+1 e da 7+7+1+1 barre ad ottave o frazioni (altri 20byte). Paradossalmente, come spiegato nel box riservato all'algoritmo FFT, volendo aumentare la dimensione dei campioni, volendo ad esempio passare a 256 o 512 o 1024 campioni e/o aumentare la risoluzione da 8 a 16 bit, la dimensione della memoria di programma (flash) aumenterebbe di ben poco mentre la RAM raddoppierebbe parallelamente ad ogni raddoppio della quantità dei campioni o della risoluzione. Un'analisi spettrale di un solo canale di 1024 campioni a 10 /12 o 16 bit con aritmetica a 16 bit richiederebbe 4096 byte di RAM per i dati e circa un centinaio o poco più aggiuntivi per tutto il resto. In pratica, però, occorrerebbe cercare un processore con 6 o 8kB di RAM che è il passo immediatamente superiore a quelli che hanno solo 4kB. Un'impresa non facile per i microcontrollori a 8bit (tra i Freescale a 8 bit sono solo 3 o 4 modelli che hanno più di 4kB di RAM).

Come detto nella prima parte di questo progetto, con 20MHz di BusClock ovvero un ciclo base di istruzione pari a 50ns, un ciclo completo di FFT a 8 bit su 2 campioni di segnale per i canali destro e sinistro da 128 elementi ciascuno, il raggruppamento in ottave sul display e l'aggiornamento dell'LCD richiedono circa 28ms, di cui solo 4 sono quelli in cui vengono campionati i canali audio. Abbiamo perciò un rapporto tra i dati esaminati e i dati disponibili pari a 4/28, ovvero di solo 1 a 7. Con oltre 35 quadri al secondo la cosa non è visibile

ed il display appare funzionante in tempo reale. Aggiungendo 16 o 20ms di processo per la generazione del segnale per il display su Oscilloscopio, il ciclo si allunga a 48ms ma la visualizzazione a display con circa 21 quadri/s è ancora estremamente fluida. Si noti che per buona parte (probabilmente il 99%) del tempo riservato al display su Oscilloscopio, il processore è sostanzialmente inattivo e attende la scadenza della temporizzazione di ogni singola riga. Si sarebbe potuto occupare questo tempo in qualche modo, ma certamente complicandoci la vita, ammesso che potesse servire a qualcosa. Il ciclo di programma inizia campionando contemporaneamente i canali destro e sinistro, prima l'uno e immediatamente dopo l'altro, ad un ritmo di 32kHz. Ovvero 128 + 128 campioni a due a due ogni 31,25us per 4ms in totale. Successivamente, viene eseguito l'algoritmo FFT sul canale sinistro (10,6ms), poi i 64 campioni di frequenza ottenuti vengono raggruppati in ottave o in mezza-ottave e memorizzati sul buffer FFT (20 byte). Quindi è la volta del canale destro, anch'esso processato con FFT, raggruppati e sommati sul buffer (altri 10,8+1ms). Dopo questo processo il segnale memorizzato sul buffer FFT viene moltiplicato per una costante e utilizzato per modulare il registro PWM a 256 passi per ottenere il segnale da inviare all'oscilloscopio se il flag che ne segnala la richiesta è attivo. Questo periodo, come detto prima, è lungo 16 o 20ms e viene passato sostanzialmente ad attendere che scada il tempo per ogni singola barra presentata sull'oscilloscopio. Questo tempo è pari a 250us per ciascuna barra quando si visualizzano 64 barre e a 1ms quando le barre sono

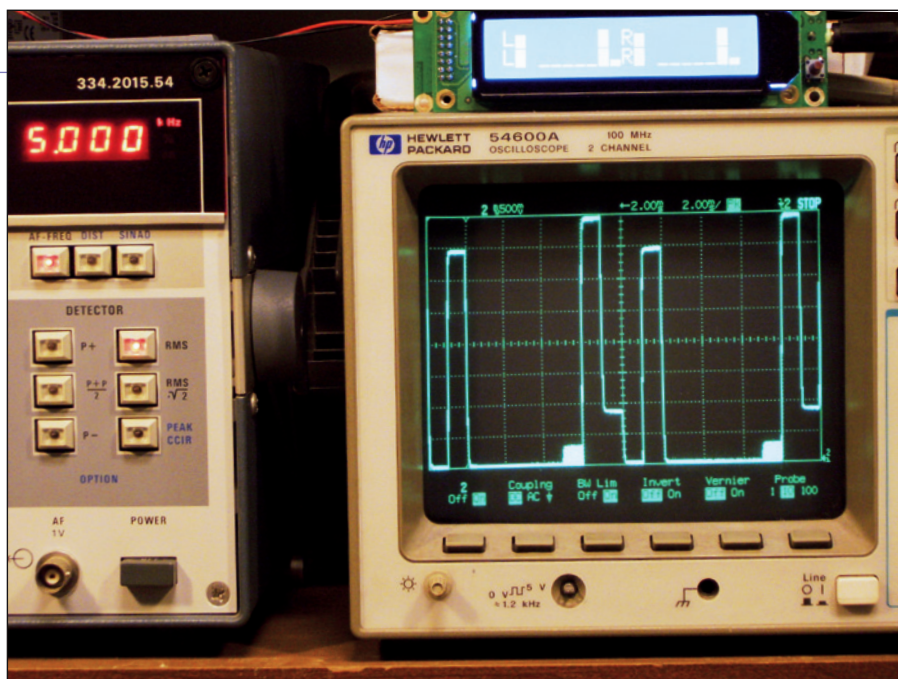
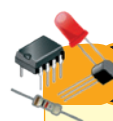


Fig. 10

solo 20 (1+7+1+7 + 4 barre vuote). Se il flag non è attivo, questo processo viene saltato e si passa direttamente a quello successivo che comprende la composizione del segnale sul buffer LCD (all'interno della RAM del μ CU) con 40 caratteri semigrafici di 8 tipi diversi e la successiva scrittura sulla RAM interna al modulo LCD che richiede poco meno di 2ms. Al termine di questo processo vengono esaminati i segnali corrispondenti sulle 3 linee dedicate al commutatore BD2 e al pulsante SW1, predisposti i flag che indicano quale dei 4 o 6 programmi di processo e display eseguire ed il ciclo ricomincia da capo.

Un ultimo dettaglio: durante il campionamento del segnale, nell'attesa tra un campione e l'altro (circa 20us), il segnale viene "finestrato" in tempo reale con una maschera temporale, ovvero moltiplicato con una funzione a forma di collina con fianchi bassi che sale rapidamente al centro con una forma tipica (coseno innalzato) e ben precisa (finestra di Hamming). Questo privilegia nel processo FFT i dati centrali rispetto a quelli iniziali e finali per simulare un segnale di tipo continuo/ripetitivo, ridu-

cendo la grossa discontinuità causata dal brusco inizio e fine del campionamento. Si genera così un display molto migliore per qualsiasi segnale che non sia esattamente centrato all'interno di una precisa banda di frequenza. Di converso si espande la base della barra di frequenza centrale a quelle adiacenti; al proposito si veda la rappresentazione in fotografia del segnale a 5kHz in Fig. 9.



per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono facilmente reperibili in commercio. Il master del circuito stampato ed il firmware col quale è programmato il micro sono scaricabili gratuitamente dal sito della rivista (www.elettronica.in.it). Il microcontrollore Freescale è anche disponibile già programmato (Cod. MF1056) al prezzo di 10,00 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>

Continuiamo a illustrare le principali funzioni del GSM/GPRS/GPS shield e della relativa libreria attraverso semplici esempi da cui potete prendere spunto per i vostri progetti.

GESTIRE I MODULI GSM CON ARDUINO

di MARCO MARTINES

Nella scorsa puntata sono stati presentati una serie di esempi su come utilizzare lo shield per effettuare chiamate, inviare toni DTMF e riconoscere se il mittente di una telefonata entrante fosse o meno salvato in rubrica. Tutto questo è stato possibile utilizzando la classe `call.h`, contenuta all'interno della libreria, relativa - appunto - alla gestione delle chiamate.

In questa seconda puntata verranno presentati degli esempi che tratteranno la gestione degli SMS. Attraverso la classe `sms.h`, vi mostreremo come inviare un SMS quando si verifica una particolare condizione o come estrapolare il contenuto di un SMS ricevuto, interpretandone i comandi contenuti al suo interno.

ESEMPIO 1

Vi mostriamo uno sketch in grado di inviare un messaggio di testo al numero preimpostato quando si verifica una condizione di allarme (**Listato 1**). Tale esempio ricalca il primo tra quelli già presentati la scorsa puntata, relativi alle chiamate, opportunamente modificato per inviare un SMS al posto di effettuare una chiamata. Per questo esempio, è stata scelta come condizione di attivazione il passaggio del valore analogico, letto dal convertitore di Arduino, al di sotto di un valore di soglia prestabilito. In tale caso comunicheremo, attraverso l'invio di un messaggio di testo al numero impostato, il verificarsi della condizione di soglia. Analizziamo il codice per comprenderne al meglio il funzionamento.

Listato 1

```
#include "SIM900.h"
#include <SoftwareSerial.h>
#include "sms.h"

SMSGSM sms;
char number[]="3921234567";
int value=0;
int value_old=0;
char message[40];
char value_str[4];
int pin=A0;

void setup()
{
  Serial.begin(9600);
  if (gsm.begin(2400))
    Serial.println("\nstatus=READY");
  else Serial.println("\nstatus=IDLE");
  value=analogRead(pin);
  value_old=analogRead(pin);
};

void loop()
{
  value=analogRead(pin);
  Serial.println(value);
  if (value<50&&value_old>=50) {
    message[0]='\0';
    strcat(message,"ALLARME LOW, VAL=");
    itoa(value,value_str,10);
    strcat(message,value_str);
    sms.SendSMS(number,message);
  }
  value_old=value;
  delay(1000);
};
```

Come per ogni sketch relativo a questo shield, per prima cosa è necessario includere la libreria relativa (in questo caso sms.h) ed in seguito istanziarne un oggetto. Tale procedimento può essere effettuato con i seguenti comandi:

```
#include "sms.h"
SMSGSM sms;
```

Ogni funzione appartenente a tale classe, per poterla utilizzare, richiede di essere chiamata nel seguente modo.

```
nome_oggetto.nome_funzione(parametri)
```

Nel caso la condizione di allarme sopracitata si verifichi, per poter inviare un messaggio è sufficiente richiamare la seguente funzione.

```
sms.SendSMS(number,message);
```

Il primo parametro della funzione contiene il destinatario del

messaggio, il secondo invece indica la stringa contenente il corpo del messaggio. Ad esempio:

```
char message[]="Questa è una prova";
char number[]="3921234567";
sms.SendSMS(number,message);
```

In questo modo verrà inviato al numero indicato la stringa "Questa è una prova".

In questo esempio, oltre a voler inviare una stringa statica contenente il messaggio di allarme, si è scelto di comunicare anche il valore al momento del rilevamento della condizione di allarme (la verifica potrebbe anche essere effettuata con intervalli molto lunghi).

In questo caso bisognerà convertire il valore analogico letto dal convertitore, sotto forma di intero, in una stringa da concatenare a quella statica.

La funzione che si occupa di tale conversione è la seguente:

```
itoa(int intero, char* stringa, int base)
```

ove la base solitamente è pari a 10 in quanto lavoriamo in base decimale. Se dovessimo lavorare con numeri in basi differenti, sarà necessario modificare tale valore nel modo appropriato. Per concatenare una stringa ad un'altra useremo invece la funzione:

```
strcat(char* stringa1, char* stringa2)
```

Così come consigliato nel precedente articolo, è vivamente consigliato di inserire un'isteresi o un filtro passa-basso per evitare, in caso di oscillazioni intorno al

Listato 1a

```
else if (value>50&&value_old<=50) {
  message[0]='\0';
  strcat(message,"ALLARME HIGH, VAL=");
  itoa(value,value_str,10);
  strcat(message,value_str);
  sms.SendSMS(number,message);
}
```

valore di soglia, l'invio ripetuto di SMS.

Nel caso si volesse aggiungere anche una segnalazione che comunichi il ritorno alla condizione di partenza è sufficiente aggiungere la seguente parte di codice prima dell'aggiornamento della variabile value_old (**Listato 1a**).

ESEMPIO 2

Vediamo ora uno sketch (**Listato 2**) per poter interpretare alcuni comandi ricevuti tramite SMS. Controlleremo lo stato di un'uscita digitale inviando alcuni semplici SMS allo shield.

Sarà possibile, dunque, impartire ad Arduino comandi da remoto, inviandogli semplicemente un messaggio di testo.

Non volendo imporre nessuna limitazione sul mittente del messaggio, per mantenere un livello minimo di sicurezza, è previsto l'utilizzo di una password da inserire ad ogni invio, al fine di evitare che persone non autorizzate possano impartire comandi contro la nostra volontà o a nostra insaputa.

Il primo passo consiste nel controllare la presenza di un nuovo messaggio non letto e, in caso affermativo, memorizzare la posizione in cui questo è salvato sulla SIM.

Il comando che svolge tale azione è il seguente:

```
sms.IsSMSPresent(SMS_UNREAD);
```

il quale restituisce un intero contenente la posizione del primo messaggio non letto. Nel caso in cui non siano presenti nuovi messaggi, restituirà 0.

Per poter analizzare il contenuto è necessario 'aprire' il messaggio e leggerne il contenuto.

```
sms.GetSMS((int)pos,number,message,180);
```

Com'è facilmente intuibile, la funzione GetSMS, si occupa di

Abbiamo parlato di GSM...

Le nostre attività di progettazione di applicazione con la rete GSM risalgono a parecchi anni fa, quando ancora non c'erano i moduli e si usavano i telefonini con degli appositi cavi dati per accedere alla rete cellulare. Con i moduli l'interfacciamento e il controllo si sono semplificati e già nel numero di Aprile 2011 abbiamo affrontato il tema dell'interfacciamento dei moduli GSM con Arduino. La soluzione pubblicata era basata su uno shield dedicato al SIM900 (TDGSM_900) mentre gli Arduino erano ancora alla versione Duemilanove. Nell'articolo, oltre a illu-



Fig. A

strare la nostra scheda su cui il modulo GSM era montato e reso più facilmente interfacciabile, venivano anche illustrate le funzioni accessibili tramite una ricca serie di comandi - implementati tramite libreria - da impartire con la comunicazione seriale alla scheda (Fig. A). Le nostre attività di progettazione e presentazione dei moduli GSM ha però radici più lontane: il modulo SIM900 è stato utilizzato anche nell'articolo che abbiamo pubblicato a Marzo 2011; era il localizzatore GSM basato proprio su questo modulo e per il quale sono state fornite ampie spiegazioni sull'interfacciamento e il funzionamento. Sempre di SIM900 abbiamo parlato nel numero Maggio/Giugno 2010 con il Telecontrollo GSM Low Cost che iniziava la serie dei nostri articoli basati su questo modulo applicandolo a un funzionale apricancello tanto innovativa quanto economica. La tecnologia dei moduli si è evoluta con rapidità e quindi a Marzo 2012 abbiamo realizzato un nuovo localizzatore basato sul modulo SIM908 che,

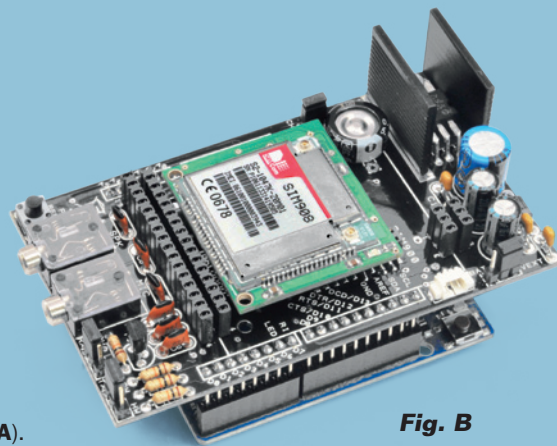


Fig. B

rispetto al precedente, dispone anche di un ricevitore GPS integrato. Nel progetto si spiega l'impiego di entrambe le funzioni del modulo per ottenere un posizionamento molto preciso e la trasmissione del dato con le varie modalità possibili.

Ad Aprile 2012, infine, abbiamo presentato uno shield universale (8190-GSMGPRSSHIELDV2) su cui montare le schede GSM basate su SIM900 e SIM908 di SIMcom per interfacciarle con schede Arduino UNO o compatibili (Fig. B).

Potete scaricare in versione elettronica tutti gli articoli citati utilizzando il nostro servizio per l'accesso e la consultazione degli arretrati disponibile su www.elettronica.in.it.

leggere il messaggio salvato nella posizione indicata, salvandone il mittente ed il corpo (fino a un massimo di caratteri indicato con l'ultimo parametro).

In seguito a tale azione non ci resterà che controllare il testo del messaggio per interpretare i comandi inviati.

Per prima cosa verificheremo che all'interno del messaggio sia presente la password (in questi esempi è stata utilizzata come password 'testpwd').

Per fare ciò ricorriamo alla funzione `strstr(char* stringa, char* sotto_stringa)` che verifica se la seconda stringa passata è contenuta all'interno della prima, restituendo un puntatore al primo carattere della sottostringa.

```
p=strstr(message,"testpwd");
```

In caso non sia presente la sotto

stringa, verrà restituito un puntatore a NULL.

Dunque per verificare la corretta autenticazione sarà sufficiente inserire la parte restante di codice all'interno della condizione:

```
if(p){ [...] }
```

Ricordiamo che `if(p)` è equivalente a scrivere `if(!=NULL)`.

A questo punto, ad autenticazione avvenuta, si verificherà, come effettuato in precedenza per la password, la presenza di stringhe contenenti comandi prestabiliti. Una volta riconosciuto il messaggio si procede con l'interpretazione di questo, attivando o disattivando l'uscita stabilita. Infine, cancelleremo il messaggio, per evitare di riempire la memoria a disposizione sulla SIM con il comando:

```
sms.DeleteSMS((int)pos);
```

ESEMPIO 3

Come accennato in precedenza, l'autenticazione è possibile effettuarla anche analizzando il mittente del messaggio.

Così come possiamo controllare se il mittente di una chiamata entrante è autorizzato, allo stesso modo analizzando se il suo numero appartiene a uno dei contatti salvati in rubrica in determinate posizioni, possiamo effettuare la stessa operazione, ma con il mittente di un SMS ricevuto.

```
sms.GetAuthorizedSMS((int)pos,number,message,180,8,10);
```

sarà sufficiente salvare sulla SIM i contatti autorizzati, all'interno di uno specifico intervallo e controllare cosa ci viene restituito dalla funzione in cui andremo a passare tale intervallo attraverso gli ultimi due parametri.

In questo caso il numero autoriz-

Listato 2

```
#include "SIM900.h"
#include <SoftwareSerial.h>
#include "sms.h"

SMSGSM sms;
char number[]="3921234567";
char message[180];
char pos;
char *p;

void setup()
{
  Serial.begin(9600);
  if (gsm.begin(2400))
    Serial.println("\nstatus=READY");
  else Serial.println("\nstatus=IDLE");
};

void loop()
{
  pos=sms.IsSMSPresent(SMS_UNREAD);
  Serial.println((int)pos);
  if((int)pos>0&&(int)pos<=20){
    Serial.print("NUOVO MESSAGGIO, POS=");
    Serial.println((int)pos);
    message[0]='\0';
    sms.GetSMS((int)pos,number,message,180);
    p=strstr(message,"testpwd");
    if(p){
      Serial.println("PSWD OK");
      p=strstr(message,"LEDON");
      if(p){
        Serial.println("LED ON");
        digitalWrite(13,HIGH);
      }
    }
    else{
      p=strstr(message,"LEDOFF");
      if(p){
        Serial.println("LED OFF");
        digitalWrite(13,LOW);
      }
    }
  }
  sms.DeleteSMS((int)pos);
  delay(5000);
};
```

zato è stato salvato tra la posizione 8 e la posizione 10 (entrambe incluse) della SIM.

```
if(stat==GETSMS_AUTH_SMS){
```

Attraverso la condizione precedente entreremo nella routine associata alla condizione in cui l'SMS ricevuto appartiene a uno dei numeri autorizzati. A differenza del caso precedente, in questo esempio (**Listato 3**) ci occuperemo di analizzare una stringa contenente (tramite 1 o 0) gli stati logici che dovranno essere impostati sulle uscite digitali

Listato 3

```
#include "SIM900.h"
#include <SoftwareSerial.h>
#include "sms.h"

SMSGSM sms;
char number[]="3921234567";
char message[180];
byte stat;
char pos;
char *p;
int i=0;

void setup()
{
  Serial.begin(9600);
  if (gsm.begin(2400))
    Serial.println("\nstatus=READY");
  else Serial.println("\nstatus=IDLE");
  pinMode(9,OUTPUT);
  digitalWrite(9,HIGH);
  pinMode(10,OUTPUT);
  digitalWrite(10,HIGH);
  pinMode(11,OUTPUT);
  digitalWrite(11,LOW);
  pinMode(12,OUTPUT);
  digitalWrite(12,LOW);
  pinMode(13,OUTPUT);
  digitalWrite(13,LOW);
};

void loop()
{
  pos=sms.IsSMSPresent(SMS_UNREAD);
  Serial.println((int)pos);
  if((int)pos>0&&(int)pos<=20){
    Serial.print("NUOVO MESSAGGIO, POS=");
    Serial.println((int)pos);
    message[0]='\0';
    stat=sms.GetAuthorizedSMS((int)pos,number,message,180,8,10);
    if(stat==GETSMS_AUTH_SMS){
      p=strstr(message,"LEDCONF");
      if(p){
        p=p+strlen("LEDCONF");
        i=9;
        while(p[0]!='\0'&&i<13){
          Serial.print("P= ");
          Serial.write(p[0]);
          Serial.println("");
          if(p[0]=='1'){
            Serial.print(i);
            Serial.println(" HIGH");
            digitalWrite(i,HIGH);
          }
          else if (p[0]=='0'){
            Serial.print(i);
            Serial.println(" LOW");
            digitalWrite(i,LOW);
          }
          Serial.println("");
          p++;
          i++;
        }
      }
    }
    //Serial.println(message);
    sms.DeleteSMS((int)pos);
  }
  delay(5000);
};
```


(ad esempio: "LEDCONF0000" oppure "LEDCONF1001" e così via).

Utilizzando la funzione di ricerca di sottostringhe mostrata nell'esempio precedente potremo identificare il comando per l'assegnazione degli stati e, successivamente, ci muoveremo analizzando carattere per carattere, impostando lo stato opportuno alla relativa uscita.

In questo esempio la stringa "LEDCONF0000" - inviata per disattivare tutte le uscite - è composta dalla prima parte LEDCONF che ci suggerisce la tipologia di comando impartito (potremmo infatti inviare comandi diversi all'interno di un singolo messaggio) e la seguente parte numerica 0000 dove sono contenuti gli stati delle uscite dalla 9 alla 12.

Ricordando che la funzione di ricerca, in caso di esito positivo, restituisce un puntatore al primo carattere della sottostringa, sarà necessario andare avanti di un numero sufficiente di caratteri per raggiungere il primo valore contenente l'informazione sulla prima uscita.

Per fare ciò utilizziamo la funzione `strlen(char* stringa)` che restituisce un intero con la lunghezza della stringa passata come parametro.

Trasliamo dunque il puntatore di un numero di caratteri pari alla lunghezza del comando:

```
p=p+strlen("LEDCONF");
```

In questo modo il puntatore `p` farà riferimento al primo carattere numerico. Sarà sufficiente controllare se questo sia pari ad 1 o 0 per attivare o meno l'uscita ad esso associata.

```
if(p[0]=='1'){
```

Per comodità inseriremo tutto in un ciclo con qualche messaggio diagnostico (non strettamente ne-

Listato 4

```
##include "SIM900.h"
#include <SoftwareSerial.h>
#include "sms.h"
#include "call.h";

CallGSM call;
MSGSMS sms;
char number[]="3921234567";
char message[180];
byte state;
int i=0;
char pin_str[2];
char pin_val[1];
void setup()
{
  Serial.begin(9600);
  Serial.println("GSM Shield testing.");
  if (gsm.begin(2400))
    Serial.println("\nstatus=READY");
  else Serial.println("\nstatus=IDLE");
  pinMode(9, INPUT);
  pinMode(10, INPUT);
  pinMode(11, INPUT);
  pinMode(12, INPUT);
  pinMode(13, INPUT);
};

void loop()
{
  number[0]='\0';
  state=call.CallStatusWithAuth(number,8,10);
  if (state==CALL_INCOM_VOICE_AUTH){
    Serial.println("Chiamata in corso da numero autorizzato");
    Serial.println(number);
    call.HangUp();
    message[0]='\0';
    for (i=9; i<=13; i++){
      strcat(message,"PIN");
      itoa(i,pin_str,10);
      strcat(message,pin_str);
      strcat(message,"=");
      itoa(digitalRead(i),pin_val,10);
      strcat(message,pin_val);
      strcat(message," ");
    }
    delay(2000);
    Serial.println("INVIO SMS");
    sms.SendSMS(number,message);
  }
  delay(1000);
};
```

cessario) sulla seriale hardware. Come in precedenza, a fine ciclo oltre a cancellare l'SMS analizzato inseriamo un ritardo di qualche secondo per evitare di effettuare continue interrogazioni allo shield.

ESEMPIO 4

Come ultimo esempio è stato scelto uno sketch (**Listato 4**) che utilizzasse entrambe le classi relative alle chiamate e agli SMS.

Il programma dovrà, una volta ricevuta una chiamata da un numero autorizzato, controllare lo stato degli input (sono stati scelti quelli digitali ma può essere facilmente esteso a quelli analogici) e comunicare tali informazioni attraverso un SMS al numero indicato. Utilizzando funzioni già spiegate in questo e nel precedente articolo, riporteremo semplicemente il codice senza spiegare dettagliatamente ogni passaggio.