

160 PAGINE!

Mensile di progettazione elettronica, attualità scientifica, novità tecnologiche.

€ 6,00

Elettronica In

oltre l'elettronica

www.elettronica.in.it

Anno XVII - n. 154
Marzo 2011

**SMART
GRID:
l'elettricità
diventa
intelligente**

UAV, la versione 2.0

- Localizzatore su rete GSM
- Shield IN/OUT per Arduino
- Avvisatore multiuso codificato in UHF

- AVR Digital Frame, il firmware
- Multimetro digitale
- Convertitore DC/DC miniatura e regolabile
- La condivisione dell'USB negli smartphone
- Una scala luminosa
- Corso .NET Micro Framework
- Corso HW/SW Arduino

Robot Area

- Nessie, il nuovo che avanza
- News ed Eventi



A tu per tu con
**Massimo
Banzi**



• NOVITÀ •
Corso NI Multisim





Accetta in ingresso da 4,5 a 14 Vcc e restituisce in uscita da 0,59 a 5,5 Vcc, con una corrente massima di 6 A; la tensione erogata può essere variata applicando una resistenza ad uno dei contatti.

CONVERTITORE DC/DC MINIATURA E REGOLABILE

di PAOLO GASPARI

Regolare una tensione allo scopo di ricavare un valore stabilizzato, è una delle esigenze che abbiamo più di frequente; ottenere livelli di 3,3 o 5 V, poi, è ancora più comune, in quanto si tratta delle tensioni con cui usualmente lavorano i circuiti logici, i microcontrollori e microprocessori che troviamo praticamente dappertutto, in ogni circuito, anche in quelli che fino a qualche anno fa si avvalevano di semplice logica discreta o di circuitazioni

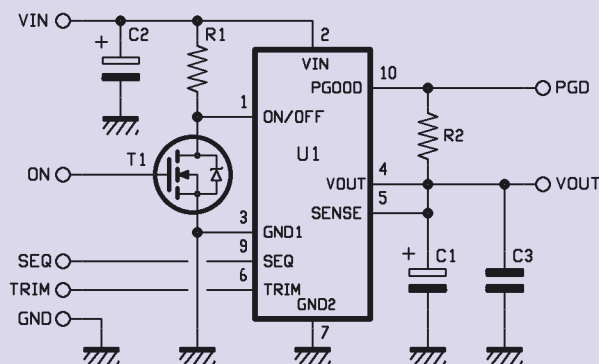
analogiche basate su operazionali e comparatori. Per questo motivo, componenti come i riduttori e gli stabilizzatori di tensione sono sempre più richiesti e, di conseguenza, la loro offerta da parte dei produttori di semiconduttori incrementa continuamente. Senz'altro più appetibili, accanto ai regolatori lineari (i classici 78xx e 79xx, LM317, LM723 ecc.) spuntano numerosissimi moduli e circuiti integrati che sono regolatori a commutazione; il perché di

- Tensione d'ingresso: da 4,5 a 14 Vcc
- Tensione d'uscita: da 0,59 V a 5,5 Vcc
- Corrente d'uscita: 6 A max
- Basso ripple e disturbo
- Temperatura di funzionamento: da -40°C a 85°C
- Protezione da sovracorrenti e surriscaldamento

re, proprio in virtù dell'elevatissima efficienza di conversione che caratterizza il componente usato. Il convertitore è quindi l'ideale per equipaggiare circuiti digitali che impiegano logica funzionante a 3, oppure 3,3 o 5 volt e che devono essere inseriti in apparati dove l'alimentatore esistente fornisce valori sensibilmente più alti, come 9 o 12 volt.

Il regolatore qui descritto è semplice ed essenziale, dato che consta di un semplice modulo miniaturizzato ed un circuito stampato di ridottissime dimensioni, ospitante i condensatori elettrolitici di filtro, un MOSFET per il comando del modulo ed alcune resistenze, e che realizza tutte le connessioni del caso ed è predisposto per ospitare dei

[schema **ELETTRICO**]



R1: 22 kohm
R2: 100 kohm
C1: 100 μ F 25 VL elettrolitico

C3: 100 nF multistrato
T1: BSS138
U1: modulo 12 V TLynx

5 V, basta inserire una resistenza da 1,34 kohm (i valori standard sono tra 1,2 ed 1,5 kohm). Una resistenza da 2,2 kohm permette di ottenere circa 3,3 V. Senza alcuna resistenza, il modulo fornisce in uscita 0,591 V.

La formula che permette di determinare la resistenza da collegare al contatto TRIM in base al valore della tensione voluta all'uscita, è la seguente:

$$R = \frac{5,91}{(V_u - 0,591)}$$

Il valore resistivo ottenuto è espresso in kohm, a patto che la tensione d'uscita (indicata come V_u) sia in volt.

La **Tabella 1** riepiloga i valori della resistenza necessari ad ot-



Il converter è basato su un modulo che contiene l'integrato regolatore, la bobina e i transistor.



tenere le tensioni principalmente utilizzate nei circuiti elettronici; le resistenze è bene siano scelte con tolleranza dell'1 %, della serie E96, all'interno della quale è possibile trovare valori molto prossimi o uguali a quelli esposti nella tabella.

Lo schema elettrico illustrato in queste pagine descrive il circuito e mostra come viene collegato ed utilizzato il modulo della

Lineage Power (siglato U1): l'ingresso dell'alimentazione del circuito raggiunge il pin 2 (Vin) e la massa il 7 (GND2); l'uscita viene filtrata da due condensatori che livellano gli impulsi prodotti dal modulo, ottenendo una componente continua che risulta disponibile tra i contatti Vout e GND dello stampato. La stessa tensione d'uscita viene retrocessa al contatto SENSE, affinché il

Sulla corrente soffia un vento nuovo

Sarà l'energia eolica a fornire nei prossimi anni il maggior contributo alla nuova potenza elettrica installata nell'Unione Europea: questa previsione è contenuta nell'ultima pubblicazione che la Commissione Europea ha dedicato ai nuovi scenari energetici del proprio territorio (Eu Energy trend sto 2030). Secondo il documento, dei 333.000 MW della nuova potenza elettrica che si stima verrà installata nella UE nel decennio 2011+2020, ben il 41% deriverà da fonte eolica. In particolare, si prevede che il complesso delle fonti rinnovabili fornirà il 64 % della nuova capacità, mentre l'apporto dalle altre fonti viene indicato nella misura del 17 % per il gas, del 12 % per il carbone, del 4 % per il nucleare e del 3 % per il petrolio. La quota dell'eolico si porterà dall'attuale 5 %, assicurato dagli 80.000 Mw installati in Europa, al 14% entro il 2020. Per il 2030, la Commissione ha praticamente raddoppiato le sue aspettative per questo settore, passando dai 146.000 MW previsti ai 280.000 MW.

Questi dati confermano la sempre maggiore considerazione del vento come fonte energetica, sfruttabile ad un prezzo sicuramente più contenuto del fotovoltaico e con un'occupazione di spazi nettamente minore, sebbene solo in zone dove le correnti d'aria sono sufficientemente costanti.



Tabella 1 - Tensioni d'uscita in funzione della resistenza.

Vout	RTRIM (kohm)
1	14,45
1,2	9,7
1,5	6,5
1,8	4,89
2,5	3,1
3	2,45
3,3	2,18
5	1,34

regolatore switching possa verificare se sta funzionando correttamente ed operare le correzioni del caso.

Il modulo può essere mantenuto alimentato ma a riposo, quindi acceso mediante un segnale elettrico: quando il piedino ON/OFF è lasciato aperto il componente resta in standby, mentre se viene portato a zero logico il convertitore DC/DC si accende ed eroga la sua corrente d'uscita. Nel circuito qui descritto, il comando esterno si opera mediante la linea ON, la quale termina sul gate di un MOSFET enhancement-mode (T1); dunque, per accendere il modulo bisogna applicare il livello logico alto (vanno bene da 4 a 15 V) alla linea ON, allorché il MOSFET passa in stato di ON e trascina a massa il pin ON/OFF dell'U1, il quale è normalmente (a MOSFET interdetto) mantenuto a livello alto da un resistore di pull-up interno al modulo.

Il comando esterno è comodo quando il convertitore debba alimentare una sezione di un circuito che si vuole attivare o spegnere con un comando logico: ad esempio con una linea di un microcontrollore.

Il modulo della Lineage Power presenta un'altra caratteristica: può essere comandato mediante il piedino SEQ, in modo che la tensione erogata ne segua il valore, limitatamente al massimo impostato; in pratica applicando una tensione al SEQ la differenza di potenziale tra Vout e GND2 cresce proporzionalmente

fermandosi in ogni caso quando viene raggiunto il valore impostato con la resistenza su TRIM. Volendo utilizzare tale funzione bisogna collegare in serie al piedino SEQ una resistenza il cui valore va determinato con la formula:

$$R_{seq} = \frac{24.950 \text{ ohm}}{V_{in} - 0,05}$$

Completiamo la panoramica sullo schema elettrico con la linea PGD (Power Good) che corrisponde ad un'uscita open-drain del modulo (al cui pull-up provvede la resistenza R2) grazie alla quale la logica interna segnala lo stato del convertitore; in pratica quando la tensione di uscita viene correttamente erogata, la linea PGD rimane a livello logico alto, mentre in caso di anomalia, quale surriscaldamento del convertitore, eccesso di corrente richiesta (sovraccarico tra Vout e GND2) o incapacità nel mantenere la tensione impostata, si porta a zero logico. PGD serve ad informare eventuali dispositivi di diagnosi o comandare gestori di reset o watch-dog di microprocessori in caso di anomalia nel regolatore, ovvero quando questo non può erogare la giusta tensione o la corrente richiesta; serve ad esempio per disattivare sezioni di circuito che hanno causato l'eccesso di assorbimento o che in caso di tensione non regolata non funzionano bene o possono essere danneggiati.

Il circuito è molto semplice e può quindi essere montato su un pezzo di basetta millefori, dove prenderanno posto il modulo 12V Pico TLynx, il MOSFET, i condensatori e le resistenze. Per chi non ha l'attrezzatura o non vuole realizzare da sé il circuito, diciamo che il regolatore qui proposto si trova in commercio in versione già montata e collaudata, in quanto è un prodotto della Sparkfun (www.sparkfun.com) e viene commercializzato dalla ditta Futura Elettronica (www.futurashop.it) già pronto e collaudato, privo della resistenza che imposta la tensione di uscita, la quale dovrete essere voi a calcolare e collegare in base alle vostre esigenze.

La basetta va preferibilmente completata con due file di pin-strip maschio a passo 2,54 mm, una da 3 e l'altra da 5 contatti, che serviranno per montarla nei circuiti che richiedono la tensione stabilizzata. ■



per il MATERIALE

Questo eccezionale modulo di conversione, in grado di fornire una corrente di uscita massima di ben 6A, è disponibile già montato e collaudato (cod. 7300-BOB09370) al prezzo di Euro 24,00 IVA compresa.

Il dispositivo va completato con un trimmer esterno o con una resistenza fissa da scegliere in funzione del valore di tensione di uscita che si vuole ottenere come indicato nell'apposita tabella.

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

LOCALIZZATORE SU RETE GSM

di BORIS LANDONI

Permette di stabilire la posizione di un automezzo, una persona o un oggetto che montano o tengono in tasca questo dispositivo, sfruttando le coordinate delle celle della rete telefonica GSM agganciate; funziona con una batteria che può essere ricaricata mediante una presa USB.

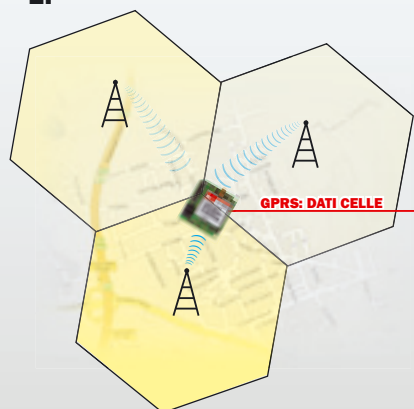
Abituati e circondati da navigatori e localizzatori satellitari, siamo ormai convinti che per individuare una persona sia necessario avvalersi del sistema di posizionamento globale, conosciuto come GPS; in realtà ciò si può evitare usando un cellulare e conoscendo la posizione delle celle cui è connesso, desunta dall'identificativo che esso può ricavare da queste ultime. I dati geografici sulle celle sono disponibili accedendo ad appositi database, che però in Italia non

sono pubblici (il loro utilizzo è a pagamento); va meglio in altri paesi, dove i gestori telefonici rendono disponibili le coordinate delle celle, la qual cosa rende molto più semplice la procedura di localizzazione. Tuttavia possiamo disporre di tali informazioni indirettamente, appoggiandoci ad esempio a Google; il noto portale probabilmente li ha ricavati da utenti i cui smartphone dotati di ricevitore GPS, mentre accedevano ai servizi di Google Maps Mobile, trasmettevano al server

l'identificativo della cella cui erano connessi e la loro posizione geografica. Ma ci sono anche dei siti che si ispirano alla filosofia dell'open-content, i quali utilizzano le informazioni inviate volon-

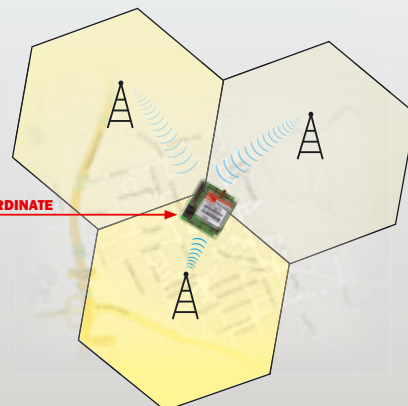
1.

Come funziona?



Interrogato da una chiamata o un SMS, oppure attivato dal pulsante di allarme o dal movimento, il circuito invia a Google gli identificativi della cella cui è collegato e ne ricava la posizione geografica, che ritrasmette via SMS.

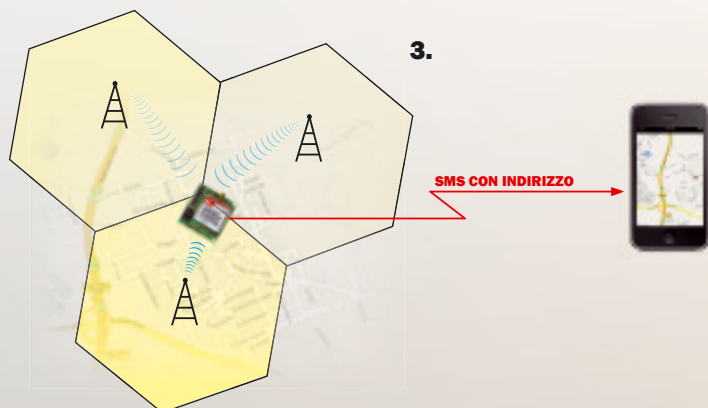
2.



tariamente dai propri utenti per realizzare database di questo tipo. Con tali sistemi vengono automaticamente ricavate ed aggiornate le coordinate delle varie celle, che sono alla base della localizzazione tramite rete GSM.

Nel fascicolo n° 140, per la prima volta abbiamo affrontato l'argomento proponendo il progetto di un localizzatore basato sulla rete cellulare; oggi presentiamo un altro localizzatore, più prestante e dotato di un nuovo modulo GSM ed un caricatore integrato per la batteria al litio con cui funziona, il quale prende energia direttamente dall'interfaccia USB.

Prima di entrare nel merito del progetto, spieghiamo in breve come funziona la localizzazione tramite GSM, partendo dalla premessa che la rete radiomobile è composta da tante celle radio adiacenti, ognuna delle quali è caratterizzata da un identificativo, composto da quattro dati: un numero progressivo (Cell ID), un codice dell'area in cui si trova (LAC, ossia Local Area Code), il codice della rete nazionale di appartenenza (MCC, acronimo di Mobile Country Code) ed infine il codice del gestore (MNC, ossia Mobile Network Code) che, ovviamente,



contraddistingue il gestore telefonico. Noti il nome della cella e le sue coordinate, considerata la distanza alla quale la cella permette la comunicazione con il telefono prima che quest'ultimo ne aggranci un'altra, si può sapere approssimativamente a quale distanza massima si trova il cellulare. Ad esempio, se questa ammonta a 1 km, il cellulare può trovarsi in

Reperire i dati sulle celle

Per ottenere i dati sulla posizione partendo dagli identificativi delle celle (MNC, LAC, CID) bisogna appoggiarsi a siti Internet che dispongono di database dove per ogni CellID sono indicate le coordinate geografiche; non potendo contare sui gestori telefonici e su organizzazioni (ad esempio OctoTelematics -www.octo-telematics.it) che offrono tali dati a pagamento, possiamo ricorrere a Google o attingere a siti in cui i database sono stati compilati con la collaborazione di volontari e vengono resi disponibili gratuitamente. Tra questi, www.opencellid.org e www.cellid.eu; qui ottenete la posizione di una scrivendone l'identificativo (Cell ID). L'identificativo delle celle potete ricavarlo dal cellulare, installando

programmi come Nokia Net Monitor, WM Cell Catcher o Cell Track. Il sito www.opencellid.org rende disponibili delle API, tra cui quelle per scaricare le informazioni sulle celle per ottenere la localizzazione, ma anche per dare al sito informazioni ricavate dagli utenti sulla posizione di nuove celle. Ad esempio, se abbiamo un cellulare con GPS e ci agganciamo a una cella, possiamo desumere il

codice identificativo e la posizione e trasmetterle al sito, che poi le verificherà e pubblicherà a favore di tutti quelli che lo visiteranno. Nello stesso sito troviamo un file che, sfruttando Google Earth, ci fa vedere zona per zona quante e quali celle ci sono, distinte per MCC ed MNC. Proprio riguardo ai codici di gestore, ricordate che MNC 1 identifica TIM, 10

Vodafone, 88 Wind e 99 Tre (H3g). Il file si scarica dalla sezione Browse cells del sito www.opencellid.org, facendo clic sul simbolo di Google Earth. Perché tutto funzioni, bisogna prima aver scaricato ed installato Google Earth dal sito www.google.com; installato il plug-in, nella sezione Luoghi di Google Earth appare, sotto

Luoghi temporanei, la voce View Cells. Per il nostro localizzatore abbiamo preferito avvalerci dei dati di Google Maps Mobile, perché rispetto agli altri siti fornisce informazioni non solo sulla posizione presunta ma anche sull'approssimazione, ovvero, indicando un punto, completa l'informazione dicendo a quale distanza massima il cellulare può trovarsi da esso e qual è l'indirizzo presunto.



una zona di 1 km di raggio. Da ciò deriva che più numerose sono le celle in una zona, più precisa può essere la determinazione della posizione del telefono (fino a meno di 100 metri); è il caso dei grandi agglomerati urbani. Come nel localizzatore del fascicolo n° 140, anche in questo sfruttiamo il collegamento a Google Maps Mobile, ormai accessibile ai cellulari comuni purché capaci di supportare la modalità dati GPRS o UMTS; ebbene, il fatto che basti una connessione dati a rilevare la posizione del telefono che la instaura implica che il server cui questo si connette sa dirgli, in base all'identificativo delle celle che sfrutta, dove esse si trovano, il che permette anche

di stimare dove il cellulare possa trovarsi rispetto a quella che fornisce il segnale più forte. Durante il normale funzionamento, il cellulare si collega alla cella che gli garantisce il miglior collegamento possibile ma rileva anche i segnali delle celle vicine (*neighbour cells*) in modo da essere pronto a commutare su una di esse quando quella in uso, ad esempio perché ci stiamo spostando, non permette la copertura necessaria a garantire una buona comunicazione. Ogni cella trasmette, nel messaggio di comunicazione con il cellulare, il proprio identificativo, che è univoco. Per questa ragione ogni cellulare, quando è connesso con la rete di un determinato gestore,

conosce esattamente la cella cui si appoggia.

Per sviluppare il nostro progetto, non potendo contare sulla volontaria diffusione da parte di Google dei dati di localizzazione delle celle, ci siamo ingegnati connettendo un cellulare GPRS dotato di interfaccia USB (sul quale avevamo caricato Google Maps Mobile) a un computer a sua volta connesso ad Internet via LAN: andando a verificare i dati che transitavano sulla rete (mediante programmi come Microsoft Network Monitor) durante il colloquio tra cellulare e server di Google, abbiamo trovato l'URL cui il servizio punta per ottenere i dati di localizzazione. Chiaramente questa metodica consente una localizzazione grossolana, la cui accuratezza dipende dai dati forniti dal server di Google circa la copertura di una data cella.

IL NOSTRO PROGETTO

Rispetto ai tradizionali localizzatori basati sul GPS, questo presenta numerosi vantaggi perché innanzitutto è più leggero e meno ingombrante, ha un costo minore e garantisce maggiore autonomia di esercizio, dovuta al fatto che siccome in un localizzatore quel che consuma di più è il ricevitore GPS, mancando questo l'assorbimento si riduce. Ciò significa che con una batteria a ioni di litio, ad esempio da 1 Ah, il nostro localizzatore può rimanere in funzione per parecchi giorni (tutto dipende dal numero di chiamate che deve fare). A ciò va aggiunto il fatto che un localizzatore basato sulla rete cellulare può essere più immediato nella risposta di uno convenzionale, dato che in particolari situazioni il ricevitore GPS può richiedere alcuni minuti per determinare la propria posizione,

mentre con i dati della rete cellulare si tratta solo di attendere la risposta alla richiesta effettuata al server di Google. Senza contare che il cellulare, diversamente da un ricevitore GPS, riceve anche in molte gallerie autostradali e in seminterrati.

Il nostro localizzatore funziona a batteria e quindi può essere portato addosso da persone che possono avere l'esigenza di chiedere aiuto o di venire rintracciate, ma anche sistemato a bordo di autoveicoli (senza necessità di installazione) o semplicemente introdotto in merci viaggianti. Per evitare di scaricare inutilmente la batteria, abbiamo previsto che il localizzatore fornisca la propria posizione su richiesta inoltrata via SMS, con una semplice telefonata o periodicamente.

Tra le funzioni implementate spicca l'SOS: premendo il pulsante di cui è dotato invia un SMS di richiesta di aiuto, contenente le coordinate della posizione; l'invio può essere fatto ad un massimo di otto numeri telefonici.

Quando viene interrogato, invia un SMS con l'esito della localizzazione e prevede una modalità chiamata *autoreport*, nella quale invia periodicamente SMS con i dati di localizzazione ad uno o più numeri di quelli in memoria, in base alle impostazioni fatte dall'utente.

Per conoscere la posizione del dispositivo remoto dobbiamo inviargli un SMS di richiesta, allorché esso verifica i dati della cella a cui è connesso ed invia una richiesta (via GPRS) al sito di Google; quest'ultimo risponde con le coordinate ed il dato relativo alla precisione. Non appena ricevute queste informazioni, l'unità remota ce le "gira" mediante un SMS. Il tutto avviene in pochi secondi.

SCHEMA ELETTRICO

Il circuito che realizza il localizzatore è formato da un modulo GSM/GPRS siglato SIM900 e prodotto dalla SIMCom (che conosciamo perché ha equipaggiato la serie di telecomandi GSM descritta nei fascicoli dal 146 al 149) qui gestito da un microcontrollore Microchip. L'insieme è diviso in due schede, una contenente il SIM900 e l'altra che ospita il micro ed il caricabatteria per l'accumulatore al litio.

Analizziamo distintamente le due unità, partendo dallo schema generale che riguarda la scheda base (quella contenente il micro) e l'unità cellulare, contrassegnata nello schema con la sigla GSM; quest'ultima verrà descritta tra breve. La scheda base ospita il micro e il caricabatteria; il microcontrollore è un PIC18LF6722 (U1) che, inizializzati gli I/O e l'UART di cui dispone, fa girare in loop il main-program attendendo un evento, che può essere l'arrivo di un SMS (o di una chiamata telefonica) o la pressione del pulsante P1 (ovvero la transizione 1/0 logico della linea RB1, internamente provvista di resistore di pull-up). Nel caso di ricezione di un SMS, agisce diversamente a seconda che si tratti di un messaggio di configurazione o richiesta di posizione. Soffermiamoci un istante sulla procedura che consente di ricavare la posizione, che è prati-

camente uguale per la richiesta manuale e la pressione del P1 (allarme o S.O.S.): all'arrivo della richiesta, il PIC comanda il modulo cellulare affinché si connetta in modo dati (via GPRS, quindi su Internet) al server di Google Maps e invii la richiesta di posizione corredata dell'identificativo della cella cui il modulo SIMCom è collegato; quindi si dispone a ricevere sul canale RX del proprio UART i dati in arrivo da Internet. Ricevuti i dati con la posizione presunta (latitudine e longitudine approssimativi) e l'accuratezza, prepara un SMS e lo trasmette al numero che ha fatto la richiesta da SMS o a quello memorizzato abbinato alla funzione di allarme. Il modulo GSM viene gestito dal micro mediante le linee RF1 (con la quale legge l'RI, ossia il Ring Indicator che il modulo usa per segnalare l'arrivo di una chiamata) ed RC7/RX1 e RC6/TX1, che sono le linee, rispettivamente, di ricezione e trasmissione dell'UART, usato per la lettura e l'invio degli SMS, oltre che per gestire le funzioni del cellulare, eccezion fatta per il reset e l'alimentazione; quest'ultima viene controllata dalla linea RC2, che agisce sul contatto 1 dello stampato del GSM (dove interviene sulla linea ON/OFF) per comandare l'accensione e lo spegnimento del GSM1 e che viene usata per avviare il cellulare dopo l'inizializzazione.



Le basette componenti il localizzatore: a sinistra la base e a destra il GSM.



Symbian è alla terza generazione

Non si placa la guerra tutta elettronica fra i sistemi operativi che vanno per la maggiore nei cellulari: di recente Nokia ha annunciato il suo X7, uno smartphone basato sul Symbian di terza generazione, destinato a contrastare l'ascesa di Android. Con 256 MB di RAM, una fotocamera da 8 megapixel con doppio flash LED, attacco micro-USB e 4 altoparlanti, il nuovo nato di Nokia punta a farsi largo in un mercato sempre più agguerrito. La casa finlandese punta anche sul recentissimo accordo con Microsoft, secondo il quale Nokia adotterà Windows Phone, e contribuirà a guidare e definire il futuro di Windows Phone. Bing migliorerà i servizi di ricerca offerti da Nokia, dando ai clienti l'accesso alle funzionalità di ricerca Bing di nuova generazione. Microsoft adCenter fornirà servizi di search advertising on line sugli apparecchi Nokia. L'integrazione fra Nokia Maps, le cui mappe sarebbero integrate con il motore di ricerca Bing di Microsoft e la piattaforma pubblicitaria adCenter, formerà un unico ambiente di ricerca ed esperienza pubblicitaria.

L'insieme, ossia il cellulare e il microcontrollore, è alimentato mediante l'interruttore SW1 dai 3,6 volt applicati ai punti + e - PWR, ottenibili da una batteria Li-Ion. Notate i numerosi condensatori di filtro dislocati lungo la linea positiva di alimentazione, indispensabili a filtrare i disturbi prodotti dall'assorbimento del cellulare in trasmissione, che altrimenti creerebbero fluttuazioni dell'alimentazione tali da bloccare il microcontrollore.

Per risparmiare l'energia delle batterie, il firmware del microcontrollore prevede la possibilità, mediante un apposito comando, di mettere il localizzatore in stato di "ibernazione" per un periodo di massimo 240 secondi; ciò consente di limitare il consumo del micro e di porre in standby anche il cellulare, che rimane acceso ma solo in ricezione e

funzionando con un clock ridotto. Il tutto contiene l'assorbimento del localizzatore in standby in soli 6 mA. Per quanto riguarda il SIM900, la riduzione della frequenza di clock (Slow Clock) viene ottenuta mediante il comando AT esteso AT+CSCLK=2; con questo comando, la modalità di riduzione del clock viene impostata automaticamente dal modulo quando mancano dati sulla seriale e disattivata (il clock torna normale) non appena la seriale riceve nuovamente dati. Per uscire dall'ibernazione, il microcontrollore attende una chiamata, ovvero lo scadere del periodo impostato (che, come detto, può arrivare a 4 minuti): scaduto l'intervallo verifica se sono arrivati SMS con richiesta di coordinate o di configurazione, quindi in mancanza di essi o espletate le operazioni corrispon-

denti, torna in ibernazione. In pratica il micro al massimo ogni 240 secondi riattiva la comunicazione seriale, perché se arriva un messaggio deve riattivarsi.

Dato che quando si trova in ibernazione non si può accorgere degli SMS eventualmente giunti (perché il modulo cellulare se è in standby non comunica sulla seriale) se dovete richiederli urgentemente le coordinate e non volete attendere che il dispositivo esca dallo standby, potete far precedere l'SMS di richiesta da una telefonata (alla quale ovviamente non vi risponderà): basta uno squillo. Infatti il cellulare anche a riposo genera il segnale di chiamata entrante (sul contatto RI) e risveglia i microcontrollore dall'ibernazione, quindi esso torna al normale funzionamento, interroga il SIM900 e si accorge dell'arrivo del messaggio, allorché provvede ad espletare le operazioni del caso.

Concludiamo la descrizione delle funzionalità del localizzatore, dicendo che riesce a rilevare il movimento ed attivare le azioni programmate in tale evenienza, pur non disponendo di ricevitore GPS o accelerometro o altro sensore di movimento: il rilevamento si basa sul cambiamento delle celle cui il SIM900 si aggancia. A causa di ciò potrebbero determinarsi sporadici falsi allarmi laddove il cellulare fosse in una zona folla di celle e le condizioni ambientali facessero cambiare l'intensità dei segnali ricevuti dal modulo anche se questo rimanesse immobile. Per evitare ciò, il microcontrollore tiene memoria delle celle vicine e verifica se, quando il cellulare commuta da una all'altra, tali celle rimangono le stesse o ne sono state agganciate altre; nel primo caso non c'è stato spostamento e quindi non viene dato l'allarme, mentre nel

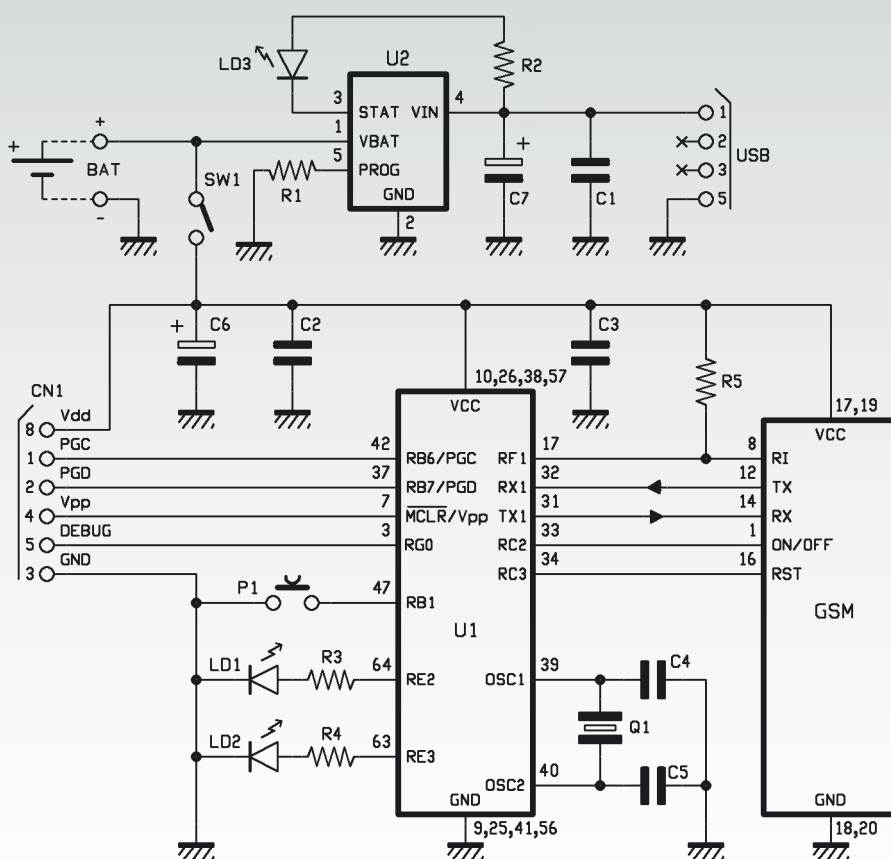
[schema elettrico UNITÀ BASE]

secondo parte la segnalazione di avvenuto movimento.

In ogni caso considerate che in zone con poche celle affinché avvenga il passaggio dall'una all'altra bisogna che il localizzatore compia uno spostamento significativo (anche un centinaio di metri); perciò la funzione va usata quando non si ha bisogno di essere avvertiti se il localizzatore compie minimi spostamenti.

IL MODULO GSM

Dopo aver analizzato il circuito nell'insieme, concentriamoci sul modulo che contiene il cellulare: si tratta di uno stampato che dispone di un connettore maschio a 20 contatti (due file da 10 l'una) a passo 2 mm che permette di inserirlo nel connettore femmina corrispondente sul circuito base ospitante il microcontrollore. Il connettore veicola l'alimentazione (Vcc, contatti 17 e 19) oltre alla linea di controllo dell'accensione (PWR) tutti i segnali e le linee di comunicazione seriale da e verso il modulo GSM, ma anche la massa (GND, contatti 18 e 20). Diamo, adesso, uno sguardo allo schema elettrico, che mostra le connessioni del cellulare: la linea PWR serve al microcontrollore del circuito per comandare l'accensione e lo spegnimento del modulo GSM, che comunque rimane sempre sotto tensione, fornita dalla linea Vcc ai piedini 55, 56, 57; la linea provvista di una resistenza di pull-up interna ed è attiva a zero logico, quindi il microcontrollore, per accendere il modulo cellulare, pone a livello logico alto PWR (contatto 1 del pin-strip) e manda in saturazione il transistor T2, il quale pone a livello basso la linea PWR del GSM. Il controllo del reset avviene mediante una linea dedicata (il SIM900 prevede un ingresso di reset -NRESET- attivo a zero logi-



co e provvisto di resistore interno di pull-up) e si ottiene facendo inviare al micro l'1 logico sulla linea RST, allorché il transistor T3 va in saturazione e pone a livello basso la linea NRESET del GSM1; contestualmente, RST viene portata ad 1 logico.

Procediamo ora con le linee di controllo dell'UART, ossia TXD, RXD, DTR, RTS, CTS, DCD, che vanno all'esterno mediante i contatti, rispettivamente, 12, 14, 10, 2, 4 e 6 del connettore; lo stesso dicasi per VRTC (contatto 5) e ADC0 (25). L'audio, facente capo a due contatti per il microfono (ad ingresso differenziale) ed altrettanti per l'altoparlante, transita dai contatti 19, 20, 21, 22, che corrispondono rispettivamente a MIC1P e MIC1N (positivo e negativo del microfono) ed SPK1N e SPK1P (rispettivamente nega-

tivo e positivo dell'altoparlante). Il segnale di RI (indicatore di arrivo chiamata) esce dal contatto 18 del connettore e viene usato per informare il microcontrollore della scheda base.

L'antenna del modulo GSM in questo caso si collega direttamente sullo stampato del cellulare. La presa d'antenna è localizzata al piedino 60. Proseguiamo con il transistor T1, che qui usiamo per comandare localmente il LED di campo del cellulare: la sua base viene polarizzata dal livello logico presente sul piedino 52 (NETLIGHT) del GSM1. Dal collettore del transistor parte la linea che porta al contatto 3 del connettore che fa capo alla linea LED, con la quale il microcontrollore si informa sull'eventuale presenza della rete GSM e sullo stato di connessione del modulo

Il modulo SIM900

Il cellulare GSM utilizzato nel nostro progetto è un Quad-band compatibile GSM/GPRS, operante sulle frequenze di 850/900/1800/1900 MHz, utilizzabile sia per accessi a Internet, sia per comunicazione vocale (a patto di collegargli un microfono e un piccolo altoparlante) che SMS. Si presenta in un package di 24x24x3 mm con contatti posti sui quattro lati. Il modulo, compatibile GSM fase 2/2+, è di classe 4 (2 W) a 850/ 900 MHz e di classe 1 (1 W) a 1800/1900 MHz; è internamente gestito da un processore AMR926EJ-S che coordina la comunicazione telefonica, quella dati (mediante uno stack TCP/IP integrato) e il dialogo (tramite un UART e un'interfaccia seriale TTL) con il circuito cui il cellulare si interfaccia. Il processore gestisce anche la SIM da 3 o 1,8 V da montare all'esterno del modulo. Oltre a quanto detto, il GSM900 inte-



gra un'interfaccia analogica, un A/D converter, un RTC, un bus SPI, uno I²C, ed un modulo PWM. La seriale TTL serve sia a comunicare i dati relativi agli SMS ricevuti che quelli in arrivo durante le sessioni TCP/IP in GPRS (il data-rate è max. 85,6 kbps), sia a ricevere i comandi dal circuito (nel nostro caso, dal PIC che governa il telecomando) che possono essere sia gli AT standard, sia quelli del set proprietario SIMCom enhanced AT. Il modulo si alimenta a tensione continua di valore compreso fra 3,4 e 4,5 V ed assorbe un massimo di 0,3 A (continuativi) in trasmissione.

(assenza segnale di rete, presenza rete ecc.) utile al microcontrollore della scheda base per sapere come comportarsi.

Concludiamo l'analisi del circuito occupandoci delle linee di gestione della SIM, che si interfaccia al modulo GSM1 tramite il bus formato da SIM_CLK (clock) SIM_RST (reset) e SIM_DATA (canale dati); la linea SIM_VDD viene usata per accendere e spegnere la SIM ed è gestita dal modulo GSM. Notate la presenza del quadruplo Zener D1, i cui diodi proteggono la SIM-Card da eventuali picchi di tensione dovuti a disturbi, che possono arrivare dalla linea di alimentazione a quelle del bus di comunicazione tra SIM e cellulare.

LA SEZIONE CARICABATTERIA

Il localizzatore carica da sé la batteria Li-Ion utilizzata mediante uno stadio caricabatterie che prende l'alimentazione dal connettore USB: da qui i 5 volt raggiungono il regolatore di carica SMD di tipo MCP73831T prodotto dalla Microchip e capace di provvedere alla carica completa di elementi agli ioni di litio da 3,6÷3,7 volt; oltre

all'integrato, sul circuito trovate il connettore per collegare la batteria. L'MCP 73831T è un chip per SMD incapsulato in contenitore SOT-23, che accetta all'ingresso tensioni dell'ordine dei 3,75÷6 volt e fornisce all'uscita la corrente occorrente alla carica di elementi agli ioni di litio o ai polimeri di litio (Li-Po) erogando fino a 550 milliampere. Nel nostro caso fornisce un massimo di circa 220 milliampere, corrente che può essere prelevata da una connessione USB.

L'integrato funziona nella configurazione canonica che vede il diodo luminoso alimentato (mediante la resistenza R1 che ne limita la corrente) dall'uscita STAT, la quale si pone a livello logico basso durante la carica, mentre rimane a livello alto a fine carica; la stessa si pone ad alta impedenza (open) quando l'MCP73831T è in shutdown o all'uscita VBAT non risulta attualmente collegata alcuna batteria. Quest'ultima uscita (piedino 1) è la linea cui si connette la batteria.

L'integrato opera la carica a corrente e tensione costante. La corrente di carica (*I_{reg}*) viene impostata con la resistenza colle-

gata al piedino 2, al cui valore è legata dalla relazione:

$$I_{reg} = 1.000/R$$

dove il valore della *R* risulta in ohm se la corrente *I_{reg}* è espressa in A. Ad esempio, con 4,7 kohm si ottiene una limitazione a 212 mA, mentre con una *R* di 2,2 kohm la corrente vale circa 454 mA. Se il piedino 5 viene aperto, l'integrato si porta a riposo ed assorbe appena 2 μA (shutdown); il piedino può quindi essere usato come enable.

La carica si svolge in diverse fasi: una volta connessa la batteria all'uscita Vbat (piedino 1) se la tensione rilevata risulta minore di una certa soglia (tipicamente intorno al 70 % del valore di piena carica) viene avviato il precondizionamento ad un valore di corrente modesto, che serve a preparare la carica rapida; quest'ultima si avvia quando la tensione ai capi della batteria supera la predetta soglia e viene effettuata ad un valore di corrente stabilito dalla resistenza collegata al piedino 5. Non appena la batteria raggiunge una tensione pari a quella di regolazione impostata nel chip, smette la carica rapida e inizia il mantenimento, che punta a stabilizzare la tensione acquisita a 4,2 V, 4,35 V, 4,4 V o 4,5 V a seconda della versione dell'integrato, che nel nostro progetto è la MCP73831T-2, la quale regola a 4,2 V. Le diverse tensioni di regolazione dell'integrato sono state volute dal costruttore per venire incontro alle esigenze di carica dei nuovi modelli di batterie al litio, che possono lavorare con tensioni anche superiori a 4 volt. Completano il circuito del carica-batteria, i condensatori C1 e C2, utili il primo per filtrare eventuali impulsi e disturbi ad alta frequenza (ad esempio il residuo

dello switching interno all'alimentatore del PC o di quello usato per caricare il localizzatore) ed il secondo finalizzato a filtrare il ripple dell'alternata ed a stabilizzare l'alimentazione a 5 volt.

REALIZZAZIONE PRATICA

Il nostro localizzatore è decisamente semplice, tuttavia costruirlo richiede una certa attenzione, in quanto tutti gli elementi che lo compongono sono in versione SMD. Una volta ottenute le basette (sono entrambe a doppia faccia e le due tracce di ciascuna possono essere scaricate dal nostro sito www.elettronica.in.it) armatevi di un saldatore da non più di 20 watt, a punta molto sottile, e di filo di lega saldante da 0,5 mm di diametro (o meno, se l'avete). Per prima prendete la scheda base e sistemate il microcontrollore, stagnando un paio di piedini su due lati opposti per fermarlo, quindi procedendo col resto dei piedini; completate le saldature ed accertato che non vi siano piazzole unite accidentalmente, si può passare a tutti gli altri componenti partendo dai passivi, quindi al quarzo e al transistor, proseguendo con il micro-pulsante da c.s. e con l'interruttore SW1: questi ultimi vanno montati sul bordo della basetta. Non dimenticate il connettore miniatura ad 8 contatti per connettere il programmatore in-circuit, la presa mini-USB ed il connettore bipolare per la batteria.

Per unire i due circuiti stampati dovete saldare su di essi il connettore a 20 contatti a passo 2x2 mm, rammentando che il maschio va su quello che ospita il SIM900 e che la femmina si colloca sull'altro. Per il modulo GSM/GPRS non è previsto alcuno zoccolo o connettore: bisogna saldarlo direttamente al rispettivo circuito stampato,

1	3	5	7	9	11	13	15	17	19
ON/OFF	LED	VRTC	ADC	SPK1N	SPK1P	MIC1N	MIC1P	VCC	VCC
2	4	6	8	10	12	14	16	18	20
RTS	CTS	DCD	RI	DTR	TXD	RXD	RST	GND	GND

Piedinatura del connettore che unisce le due schede.

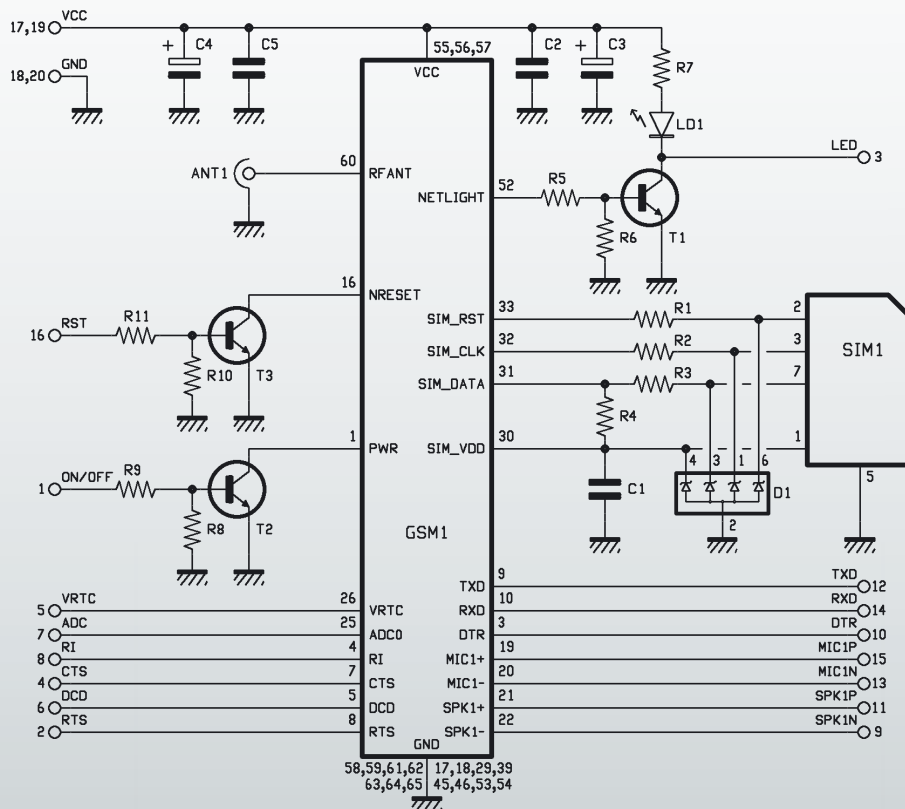
magari aiutandosi con una lente d'ingrandimento per verificare che non ci siano "baffi" di stagno a cortocircuitare piazzole vicine; inserito il modulo, sul lato opposto dello stampato montate il porta-SIM. Concludete con il connettore per l'antenna. Completato il montaggio, attraverso il connettore CN1 (8 poli) si può provvedere alla programmazione in-circuit del micro.

IMPOSTAZIONI E COMANDI

Bene, costruito il localizzatore dovete sapere come usarlo; diciamo allora che tutte le configura-

zioni e le richieste di coordinate si effettuano da remoto mediante un qualsiasi telefono cellulare, nel modo che spieghiamo qui di seguito. Partiamo con la lista dei numeri di telefono abilitati ad effettuare la richiesta dei dati di posizionamento: sono previsti 8 numeri, che poi sono gli stessi cui il localizzatore invia gli SMS con i dati di localizzazione. Alcuni comandi possono essere eseguiti solamente se corredati di password; per altri, basta che giungano da telefoni i cui numeri siano presenti nella lista degli otto abilitati al comando e alla

[schema elettrico del MODULO CELLULARE]



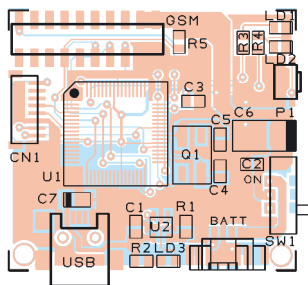
[piano di montaggio dell'UNITÀ BASE]

Elenco Componenti:

R1: 10 kohm (0805)
R2÷R4: 330 ohm (0805)
R5: 10 kohm (0805)
C1÷C3: 100 nF multistrato (0805)
C4, C5: 15 pF ceramico (0805)
C6: 470 µF 6,3 VL tantalio (CASE-D)
C7: 4,7 µF 6,3 VL tantalio (CASE-P)
U1: PIC18LF6722 (MF908)
U2: MCP73831T-2ACI/OT
LD1: LED rosso (0805)
LD2: LED verde (0805)
LD3: LED verde (0805)
Q1: quarzo 20 MHz (12SMX)
SW1: Deviatore a slitta
P1: Microswitch 90° SMD

Varie:

- Connettore Mini-USB
- Connettore molex 2 poli 90° passo 1,25 mm
- Connettore 8 poli passo 1 mm
- Connettore 2x10 poli passo 2 mm femmina
- Circuito stampato S908



richiesta delle coordinate. La password predefinita (e reimposta automaticamente dopo ogni reset totale del sistema) è 12345; è possibile sostituirla con una a piacimento (purché di cinque cifre) impartendo il comando del caso con un SMS contenente il testo *PWDnewpwd;pwd*, dove *newpwd* è la password da memorizzare e *pwd* l'attuale.

Detto ciò, iniziamo a vedere i comandi riguardanti la lista dei numeri telefonici: per memorizzare un numero bisogna inviare un SMS contenente il testo *NUMxnnnnnnnnnnnnnnnnnn;pwd*, dove *x* indica la posizione (1÷8) in cui salvare il numero rappresentato da *nnnnnnnnnnnnnnnnnn* e *pwd* è l'attuale password, necessaria se il comando SMS proviene da un telefono non in lista (serve comunque quando si memorizza il primo numero...) o se, pur arrivando da un telefono di quelli già memorizzati, la posizione specificata è già occupata da un altro numero.

Ogni numero può essere lungo al massimo 19 caratteri e deve comprendere il prefisso di nazionalità, che per l'Italia è +39.

Per cancellare il numero in una posizione della lista, il comando da usare è *NUMx;pwd*, dove *x* è la posizione (1÷8) e *pwd* l'attuale password; è anche possibile interrogare il dispositivo per conoscere tutti i numeri memorizzati: basta impartire il comando da *SMS NUM?;pwd*.

Una volta salvati i numeri, bisogna definire quali sono quelli cui il localizzatore deve inviare gli SMS contenenti le coordinate della posizione; come accennato, a seguito di una chiamata il dispositivo risponde al numero corrispondente, mentre in autoreport e in SOS (cioè quando si preme il pulsante di allarme P1) possiamo fare in modo che l'invio avvenga anche a tutti gli otto i numeri.

Il comando corrispondente è *SMSxxxxxxx:ON*, dove *xxxxxxx* sono le posizioni della lista dove

si trovano memorizzati i numeri. Ad esempio, *SMS1346:ON* imposta che a ricevere gli SMS siano i numeri nelle posizioni 1, 3, 4 e 6. Per disabilitare l'invio di SMS ad uno o più numeri bisogna inviare il comando *SMSxxxxxxx:OFF*, che dice al localizzatore di disabilitare le posizioni specificate. Ad esempio, *SMS2346:OFF* disattiva l'invio degli SMS ai numeri memorizzati nelle posizioni 2, 3, 4, 6. Alla prima accensione e dopo ogni reset, in autoreport o in allarme il localizzatore invia SMS a tutti i numeri che verranno memorizzati.

Un tipico messaggio generato in risposta alla pressione del tasto SOS è questo:

Probable position:

LOCALIZZATORE

LAT: 45.2441

LON: 8.1228

Range: 1148 m

via Adige 21

21013 Gallarate VA, Italy

SOS

Per quanto riguarda l'interrogazione da chiamata telefonica, si può definire come il localizzatore debba comportarsi, mediante il comando *ARI:x*, dove *x* vale *M* se desiderate che il dispositivo risponda inviando un SMS con le coordinate al solo numero chiamante, ovvero *S* se volete che mandi SMS a tutti i numeri della lista abilitati con il comando *SMSxxxxxxx:ON*.

Vediamo ora i comandi che definiscono i parametri identificativi del localizzatore: un SMS contenente il testo *NAME:xxxx* permette di assegnare un nome all'apparecchio, cosa utile quando si devono gestire più localizzatori. *NAME?* interroga l'apparecchio per ottenere il suo nome identificativo. Tutti questi comandi possono essere inviati da numeri

presenti in lista, ovvero da qualsiasi altro purché corredati di password. Per quanto riguarda la modalità di autoreport, abbiamo previsto un apposito set di comandi: *AUTO?* permette di verificare a distanza lo stato delle impostazioni per l'autoreport. *AUTOC:of* abilita o disabilita l'invio periodico delle coordinate alla lista dei numeri memorizzati e abilitati alla ricezione di SMS; per abilitare l'autoreport scrivete ON al posto di *of*, mentre per disattivarlo dovete mettere OFF. Il comando *AUTOC:hh/mm* permette di definire la frequenza dell'invio periodico; in esso, al posto di hh/mm dovete scrivere ogni quante ore e minuti il localizzatore deve mandare i messaggi con la propria posizione ai numeri definiti nella lista con il solito comando *SMSxxxxxxx:ON*. Ad

esempio, se volete un invio ogni 15 minuti dovete impartire il comando *AUTOC:00/15*. Questo localizzatore permette di rilevare il movimento ed eventualmente avvertire una o più persone via SMS, ovvero comunicare i dati (Cell ID) delle celle agganciate di volta in volta; i comandi corrispondenti sono *TMA:of* e *CELL:of*. Per quanto riguarda il primo, se al posto di *of* scriviamo ON il dispositivo notifica ai numeri presenti nella lista e abilitati a ricevere gli SMS il cambio di cella cui è agganciato il modulo GSM; tale funzione è disabilitata con il comando *TMA:OFF*. Il comando *TMA* può anche essere impartito localmente premendo e mantenendo premuto il pulsante P1 e disattivato sempre con analoga manovra; a conferma dell'attivazione

il LED verde lampeggerà 5 volte rapidamente, mentre per confermare la disattivazione il circuito fa emettere altrettanti lampeggi ma al LED rosso. L'attivazione e disattivazione manuale del TMA serve ad esempio quando si ha il localizzatore nell'auto: attivandola, se il veicolo viene rubato parte l'allarme; rientrando a bordo, bisogna disattivare il TMA, altrimenti quando partiamo riceveremo l'SMS di allarme.

Quanto all'interrogazione riguardante le celle agganciate al momento, si effettua inviando il comando *CELL:ON*; in questo caso ogniquale volta il localizzatore invia un SMS con le coordinate (in autoreport, SOS o su richiesta manuale fatta con il comando *COO*) allega anche il CID e il LAC delle celle attualmente agganciate. Con il comando

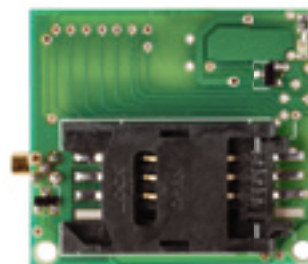
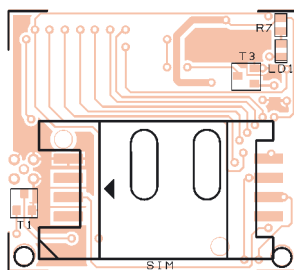
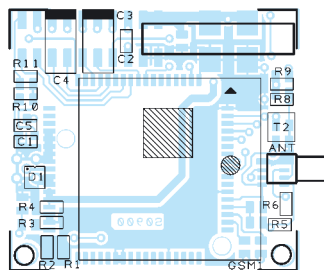
[piano di montaggio del MODULO CELLULARE]

Elenco Componenti:

C1: 220 nF multistrato (0805)
C2: 100 nF multistrato (0805)
C3: 470 μ F 6,3 VL tantalio (CASE-D)
C4: 470 μ F 6,3 VL tantalio (CASE-D)
C5: 100 nF multistrato (0805)
LD1: LED verde (0805)
D1: vedi testo
R1÷R3: 15 ohm (0805)
R4: 10 kohm (0805)
R5: 4,7 kohm (0805)
R6: 10 kohm (0805)
R7: 330 ohm (0805)
R8: 10 kohm (0805)
R9: 4,7 kohm (0805)
R10: 10 kohm (0805)
R11: 4,7 kohm (0805)
T1÷T3: BC817
GSM1: Modulo GSM SIM900
SIM: Slot per SIM-CARD

Varie:

- Connettore Antenna MMCX 90°
- Strip maschio 2x10 poli 2mm
- Circuito stampato S900



Funzioni e comandi

FUNZIONE	COMANDO SMS	PARAMETRI	VALORE PREDEFINITO
CAMBIARE PASSWORD	PWDxxxx;12345	xxxx=nuova password	12345
MEMORIZZARE UN NUMERO (massimo 8 numeri) (MAX 19 CIFRE PER NUMERO)	NUMx+39nnnnnnnnn;12345	x=posizione del numero nnnnnnn=numero da memorizzare	
CANCELLARE UN NUMERO	NUMx;12345	x=posizione del numero	
VERIFICARE I NUMERI MEMORIZZATI	NUM?;12345		
RESET COMPLETO	RES;12345		
IMPOSTARE I NUMERI A CUI VERRANNO INVIATI GLI SMS PER AUTOREPORT	SMSxxxxxxx:of	xxxx=posizioni in cui si trovano i numeri da avisare of=può valere ON o OFF	tutti i numeri verranno avisati
IMPOSTARE IL NOME	NAME:xxxx	xxxx=nome del localizzatore (massimo 15 caratteri)	FT908
RICHIEDERE IL NOME	NAME?		
INVIO NOTIFICA SU MOVIMENTO (CAMBIO CELL). PUÒ ESSERE ATTIVATO/DISATTIVATO MANTENENDO PREMUTO IL PULSANTE. LAMPEGGIO ROSSO DISATTIVATO, LAMPEGGIO VERDE ATTIVATO	TMA:of	of=può valere ON o OFF	
MODALITÀ DI INVIO NOTIFICA IN CASO DI RICHIESTA TRAMITE SQUILLO	ARI:x	x=M chiamante - x=S SMS	S
SELEZIONARE IL FORMATO COORDINATE PER L'INVIO TRAMITE SMS	FORS:x	1 = testo; 2 = link	testo
RICHIEDERE FORMATO	FORS?		
AUTOREPORT CONTINUO	AUTO:of	of=può valere ON o OFF	OFF
INTERVALLO DI TEMPO PER L'INVIO DI UNA NOTIFICA (CONTINUO)	AUTO:hh/mm	hh=ore mm=minuti (impostare minimo 00:03)	01:00
IMPOSTAZIONE APN	GPRSAPN:apn	apn=apn del provider (esempio web.omnitel.it)	
CANCELLAZIONE APN	GPRSAPN		
IMPOSTAZIONE USERNAME GPRS	GPRSUSR:xxxxxxx	xxxxx=username (solitamente non necessario)	
CANCELLAZIONE USERNAME	GPRSUSR		
IMPOSTAZIONE PWD GPRS	GPRSPWD:xxxxxxx	xxxxx=password (solitamente non necessaria)	
CANCELLAZIONE PASSWORD	GPRSPWD		
VERIFICARE IMPOSTAZIONI GPRS	GPRS?;12345		
DISABILITARE LA RISPOSTA PER QUEL MULTIMESSAGGIO	RISP		
RICHIESTA COORDINATE	COO		
INVIARE DATI SULLE CELLE	CELL:of	of=può valere ON o OFF	ON
SLEEP PER XXX SECONDI (OGNI XXX SECONDI VIENE LETTO SE C'È UN SMS E LA CHIAMATA VIENE GESTITA IMMEDIATAMENTE)	SLEEP:xxx	xxx da 000 a 240 secondi	60
INDICAZIONE INDIRIZZO	REV:of	of=può valere ON o OFF	ON
VERSIONE	VER?		

CELL:OFF tale funzione si disattiva e i messaggi inviati successivamente a ciò conterranno solo le coordinate. Un esempio di SMS generato a seguito dell'SOS e contenente sia le coordinate che i parametri delle celle è:

Probable position:
LOCALIZZATORE
LAC 271B

CID 39DF
LAT: 45.2441
LON: 8.1228
Range: 1148 m
via Adige 21
21013 Gallarate VA, Italy
SOS

Ora diamo uno sguardo al set di comandi riguardante le impostazioni Internet, indispensabili

per accedere al server di Google durante la richiesta della posizione: l'APN si definisce con *GPRSAPN:xxxxxxx*, dove al posto delle *x* va inserito il parametro corrispondente al gestore del servizio Internet cellulare. All'occorrenza, *GPRSAPN* cancella l'impostazione esistente. *GPRSUSR:xxxxxx* serve a impostare il nome utente

(username, da scrivere al posto delle *x*) e *GPRSUSR*, invece, cancella l'impostazione attuale. *GPRSPWD:xxxxxxxxxx* definisce la password di accesso ad Internet e *GPRSPWD*, invece, cancella la password esistente.

Infine, chi desidera verificare le impostazioni Internet può usare il comando *GPRS?*, che in risposta produce un SMS diretto al numero che l'ha richiesto, contenente i dati del caso. Resta inteso che il localizzatore risponde al comando se proveniente da un numero di quelli in lista o da uno qualsiasi a patto che l'SMS inviato contenga la password. In ogni momento, con il comando *COO* è possibile richiedere le coordinate dell'attuale posizione; esso produce un SMS diretto al numero del telefono che l'ha impartito. Gli SMS inviati dal localizzatore possono avere vari formati, ognuno dei quali si imposta con il comando *FORS:x*, che stabilisce la rappresentazione degli SMS. I formati impostabili (1 o 2) danno i seguenti risultati:

1:

**Probable position:
LOCALIZZATORE**

LAT: 45.2441

LON: 8.1228

Range: 1148 m

**via Adige 21
21013 Gallarate VA, Italy
SOS**

2:

**<http://maps.google.it/maps?f=q&hl=it&q=45.643830,+8.814598>
Gallarate VA, Italy
SOS**

Il primo comunica il nome del localizzatore (in questo caso è LOCALIZZATORE) le coordinate geografiche della cella, la distanza presunta da essa (espressa in metri) che corri-

sponde alla copertura della cella stessa e l'indirizzo presumibile sulla base di questi. Se è attivata la comunicazione dei parametri della cella (CellID) il messaggio contiene anche LAC e CID. Il secondo formato consiste nella stringa che in Google Maps corrisponde all'indirizzo presunto dal localizzatore: inserendolo in un browser Internet vedrete la posizione stimata del localizzatore. In entrambi gli esempi, SOS indica che il messaggio è stato inviato a seguito dell'attivazione della funzione SOS; se fosse dovuto ad autoreport vedreste AUTO_CONT al posto di SOS, mentre se invece la richiesta venisse effettuata con uno squillo o con un SMS, apparirebbe POLLING. Se la segnalazione deriva dal movimento, riporta la dicitura MOVE.

In ogni momento si può interrogare il sistema con il comando *FORS?*, per conoscere quale formato è attualmente impostato. Con il comando *REV:1* si visualizza anche l'informazione riguardante l'indirizzo presunto, che se il comando stesso vale *REV:0* non viene visualizzato. E passiamo al comando che attiva l'ibernazione del dispositivo, ossia *SLEEP:xxx*; impartendolo, il localizzatore va a riposo (consumando meno di 6 mA) per il tempo, in secondi, scritto nel rispettivo SMS al posto delle *x*. Il valore dell'intervallo deve essere espresso sempre a tre cifre, quindi se volete l'ibernazione per 40 secondi dovete inviare il comando *SLEEP:040*. L'intervallo si sceglie tra 000 e 240 s. Il reset completo, che ripristina le impostazioni predefinite e cancella la lista dei numeri telefonici, si ottiene con il comando *RES:pwd*, dove *pwd* è la password attuale.

Il sistema accetta messaggi multipli, ossia SMS contenenti

più comandi ciascuno purché separati da una virgola; volendo, per risparmiare denaro è possibile disattivare le risposte ad alcuni comandi inserendo, nell'SMS, il comando *RISP*, il quale impedisce la risposta ai comandi che prevedono la conferma tramite SMS.

Il localizzatore prevede una procedura di memorizzazione automatica del primo numero telefonico della lista (*Easy Setup*): dopo l'accensione e per circa tre minuti attende una telefonata, al sopraggiungere della quale salva in memoria il numero del telefono chiamante ed entra nel normale esercizio, dove si porta comunque anche se il tempo passa senza che giunga alcuna chiamata. Per rientrare nell'*Easy Setup* occorre staccare l'alimentazione e riattaccarla. ■



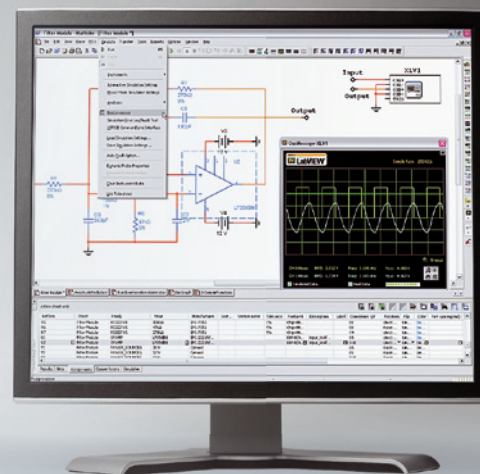
per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono facilmente reperibili in commercio. Tuttavia, in considerazione della miniaturizzazione spinta del progetto, abbiamo ritenuto opportuno proporre il localizzatore anche nella versione montata e collaudata. Il dispositivo già funzionante (cod. FT908M) costa 98,00 Euro e viene fornito completo di antenna ANTGSMSTL-F01 e adattatore CVANT. Non è ovviamente compresa la SIM-CARD del gestore telefonico. Sono disponibili separatamente la batteria ricaricabile al litio (cod. BAT-TWEBT al prezzo di 24,50 Euro) e l'alimentatore cod. FT681M-36 al prezzo di 19,00 Euro. Tutti i prezzi si intendono IVA compresa.

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

Conoscere e usare NI Multisim

Progettazione più semplice,
migliore simulazione.



Dopo aver imparato a realizzare uno schema elettrico, iniziamo a conoscere i tanti strumenti di misura virtuali di Multisim e ad effettuare le prime simulazioni: quelle più semplici, in corrente continua. Seconda puntata.

di Enrico Ambrosini

Dopo una prima puntata propedeutica alla conoscenza di Multisim, è venuto il momento di entrare nel vivo del lavoro con questo prezioso strumento di progettazione ed emulazione, vedendo come condurre la simulazione circuitale in continua. Per iniziare dedichiamo qualche paragrafo alla spiegazione degli strumenti virtuali di Multisim, che possiamo considerare suddivisi in quattro categorie, ovvero:

- 1) quelli appositamente creati per Multisim;
- 2) quelli corrispondenti a veri strumenti commerciali (Agilent e Tektronix) e simulabili all'interno di Multisim;
- 3) quelli della piattaforma NI ELVIS II, importabili per la simulazione all'interno di Multisim, caricando gli appositi driver e disponibili anche con la più economica piattaforma NI MyDAQ;
- 4) quelli creati con NI LabVIEW e resi disponibili all'interno di Multisim.

Facendo clic sul pulsante corrispondente con il tasto sinistro del mouse e tenendo premuto quest'ultimo, possiamo trascinare nel circuito gli strumenti corrispondenti e procedere ai collegamenti con le stesse modalità che abbiamo già visto nella realizzazione dello schema elettrico. In alternativa è possibile impartire il comando dal percorso **Simulate\Instruments**, quindi scegliendo lo strumento voluto.

Successivamente, con un doppio clic del mouse e con il cursore sullo strumento scelto possiamo ingrandire i singoli strumenti in modo da vederne tutti i comandi e funzioni e procedere così alle misure, con le stesse modalità usate nella realtà. Notate che non esiste limite teorico al numero di strumenti inseribili (non costano nulla, quindi possiamo scialare...) ma c'è un limite legato al buon senso e alla potenza del nostro PC: tirare in ballo troppi strumenti rallenta l'esecuzione della

La barra degli strumenti.

	Multimetro.
	Generatore di funzioni.
	Wattmetro.
	Oscilloscopio a doppia traccia.
	Oscilloscopio a quattro tracce.
	Bode plotter (per tracciare i diagrammi di Bode).
	Frequenzimetro.
	Generatore di parole binarie.
	Analizzatore di stati logici.
	Convertitore logico (per creare il circuito logico dalla tabella della verità e viceversa).
	Tracciacurve (di diodi, transistor, triac...).
	Distorsimetro (per la misura della distorsione armonica).
	Analizzatore di spettro.
	Analizzatore di reti (prevalentemente per applicazioni in alta frequenza).
	Generatore di funzioni Agilent 33120A (simulazione di uno strumento commerciale).
	Multimetro Agilent 34401A (simulazione di uno strumento commerciale).
	Oscilloscopio Agilent 54622D (simulazione di uno strumento commerciale).
	Oscilloscopio Tektronix TDS2024 (simulazione di uno strumento commerciale).
	Sonda di misura (da posizionare dove si desidera per misurare tensioni e correnti in cc e in ca, la frequenza, la fase e i guadagni tra due nodi).
	Strumenti provenienti da LabVIEW.
	Strumenti di Elvis e MyDAQ (disponibili solo installando i relativi driver).
	Sonda per la misura della corrente senza interrompere il circuito.

simulazione. Tanto per darvene un'idea, nella **Fig. 1** trovate l'oscilloscopio a quattro tracce di Multisim, mentre nella **Fig. 2** vedete il multimetro di Elvis e MyDAQ; la **Fig. 3** mostra, invece, l'analizzatore dinamico di Elvis e MyDAQ.

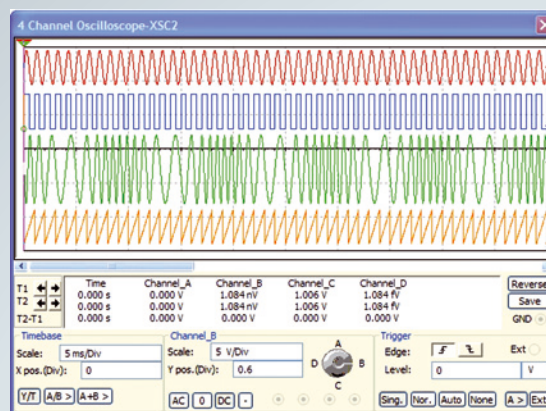


Fig. 1 - Oscilloscopio a 4 tracce di Multisim (sinusoide, onda quadra, segnale FM, segnale triangolare).



Fig. 2 - Multimetro digitale di Elvis e MyDAQ.

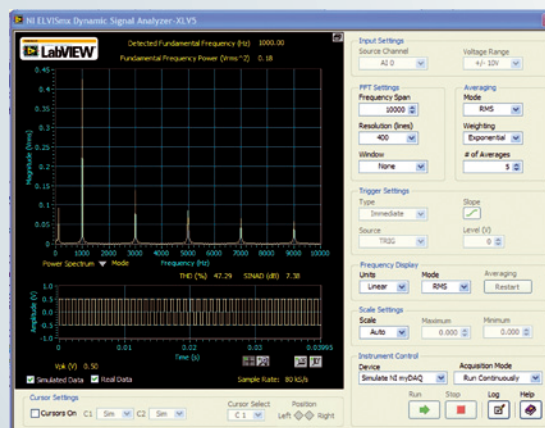


Fig. 3 - Analizzatore dinamico di Elvis e MyDAQ (onda quadra: analisi armonica e nel tempo).

L'analisi dettagliata di tutti questi strumenti sarebbe molto lunga e andrebbe oltre le intenzioni di questo corso, comunque attraverso le diverse analisi proposte potremo, nelle diverse puntate, apprendere almeno le più importanti nozioni riguardanti ognuno di essi.



Fig. 4
Il modulo NI ELVIS II.

I SUPPORTI DIDATTICI NI ELVIS II E NI MYDAQ

Prima di entrare nel vivo nell'uso di Multisim, è opportuno evidenziare le caratteristiche salienti di due supporti hardware e software NI simulabili al suo interno, che offrono la possibilità di confrontare i risultati simulati e quelli misurati sul prototipo usando i medesimi strumenti di misura in entrambi i casi. La simulazione è possibile, prescindendo dal loro effettivo possesso, a condizione di installare il file, con i relativi driver, scaricabile (gratuitamente previo registrazione) dal sito NI (NI ELVISmx Software 4.2.3, valido per entrambi i dispositivi).

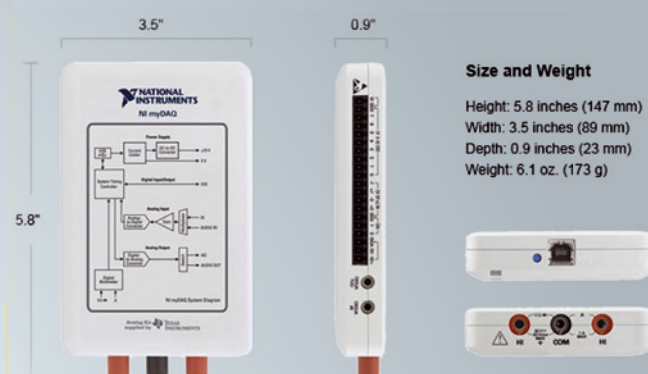
L'unità ELVIS II (**Fig. 4**) interfacciata ad un PC tramite la porta USB dispone di:

- dodici strumenti integrati (multimetro, oscilloscopio, alimentatore, generatore di funzioni, generatore di forme d'onda arbitrarie, analizzatore di segnali, generatore digitale e lettore digitale, analizzatore di impedenze, Bode-plotter, traccia curve a due e tre vie) con i relativi pannelli visualizzati sullo schermo del PC;
- otto ingressi analogici differenziali o dodici non differenziali, due uscite analogiche e ventiquattro I/O digitali;
- piano breadboard per la realizzazione circuiti sperimentali.

Il MyDAQ (**Fig. 5**) può essere considerato il fratello minore di ELVIS e, in virtù del suo basso costo, rappresenta una soluzione ottimale per realizzare un completo laboratorio elettronico per uso personale. Anch'esso opera in abbinamento ad un PC, ma è molto più compatto e il circuito corrispondente va realizzato su una piastra breadboard a parte. MyDAQ dispone di:

- nove strumenti integrati (multimetro, oscilloscopio, alimentatore, generatore di funzioni, generatore di forme d'onda arbitrarie, analizzatore di segnali, generatore

Fig. 5 - Il modulo MyDAQ.



digitale e lettore digitale, Bode-plotter) con i relativi pannelli visualizzati sullo schermo del PC;

- due ingressi analogici di tipo differenziale;
- due uscite analogiche;
- otto I/O digitali;
- un ingresso e un'uscita audio.

Approfondiremo l'uso del MyDAQ e il suo uso simulato con Multisim in un'apposita puntata di questo corso.

LA SIMULAZIONE CON MULTISIM

Si possono considerare due tipi di simulazioni effettuabili:

- 1) usando gli strumenti virtuali;
- 2) con il menu **Simulate**.

La prima modalità usa gli strumenti virtuali, precedentemente elencati, come fossero strumenti reali, simulando così anche l'attività di laboratorio. Una volta inseriti gli strumenti nel circuito, si deve avviare la simulazione con il tasto  o con l'interruttore . A simulazione attiva potremo interromperla, sempre con lo stesso interruttore o con il pulsante . Possiamo anche introdurre una pausa nella visualizzazione della simulazione con uno dei pulsanti  o  senza però interromperla; una volta ripresa, la simulazione non ripartirà daccapo ma da dove l'avevamo lasciata.

Nel secondo caso l'attivazione della simulazione avviene tramite il menu **Simulate**, secondo le modalità descritte nei successivi paragrafi per i diversi tipi di simulazioni. In questo caso si rinuncia alla simulazione degli strumenti, senza quindi le inevitabili limitazioni delle misure reali e con la possibilità

di valutare meglio le specifiche situazioni; la contropartita è che la simulazione è più distante dalla realtà sperimentale.

Nelle simulazioni che proporremo di seguito useremo inizialmente gli strumenti di Multisim e il menu Simulate. Delle altre categorie di strumenti e delle simulazioni con MyDAQ e ELVIS II ce ne occuperemo in specifiche puntate di questo corso, ad esclusione degli strumenti Agilent e Tektronix, che non considereremo in quanto utili solo per un apprendimento mirato a questi specifici strumenti (chi li possiede, usando i relativi manuali, non avrà comunque problemi).

L'ANALISI IN DC CON MULTISIM

Vediamo ora come analizzare, tramite simulazione, un circuito in corrente continua. Per farlo, iniziamo con il realizzare il circuito corrispondente allo schema elettrico riportato nella Fig. 6 (ricordatevi di mettere la massa

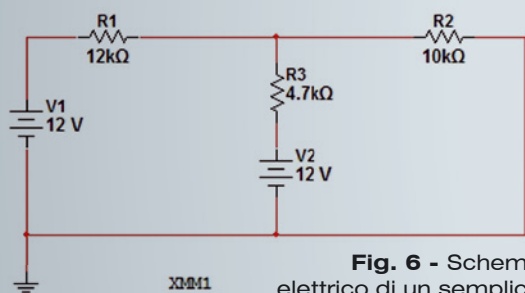


Fig. 6 - Schema elettrico di un semplice circuito in continua.

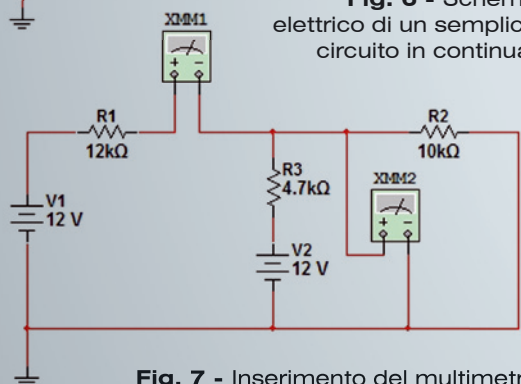


Fig. 7 - Inserimento del multimetro per misurare correnti e tensioni.

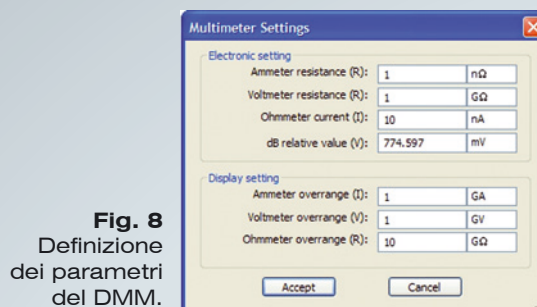


Fig. 8
Definizione
dei parametri
del DMM.

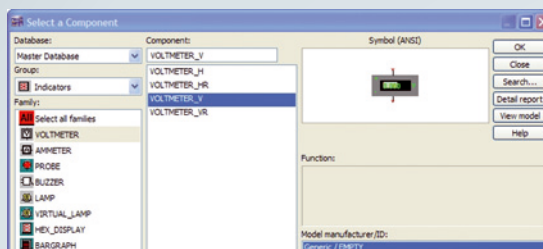


Fig. 9 - Scelta di un voltmetro tra gli strumenti indicatori disponibili.

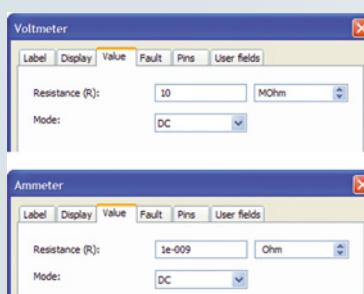


Fig. 10 -
Definizione
dei parametri
degli strumenti
indicatori.

altrimenti la simulazione non funziona; questa regola dovete applicarla sempre, in ogni circuito) e, per sicurezza, salviamolo subito con un nome che può essere, ad esempio, **circuito_DC_1**. Per farlo, impartite il comando **File\Save**, con il quale, a meno di non specificare un percorso differente, salverete il circuito nella cartella predefinita da Multisim; in alternativa, con il comando **File\Save As...** deciderete voi la cartella dove salvare il modello.

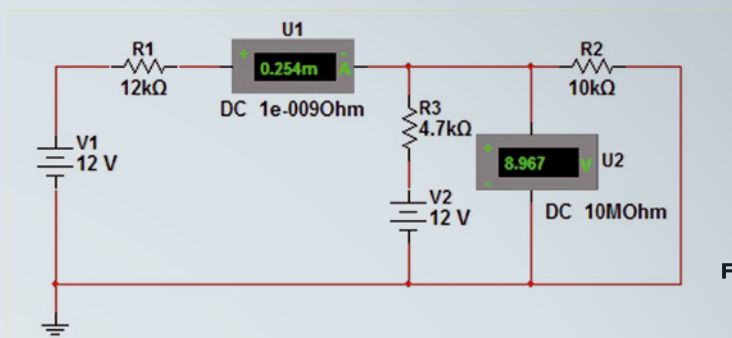
Per la realizzazione del circuito da simulare tenete presente che la massa (**GROUND**) e la batteria (**DC_POWER**) li trovate nel gruppo **Sources** famiglia **POWER_SOURCES**, mentre il resistore (**RESISTOR**) si trova nel gruppo **BASIC**.

Come già detto, possiamo procedere all'analisi del circuito con due modalità diverse.

USO DEGLI STRUMENTI DI MISURA

Sono disponibili voltmetri, amperometri e sonde di misura. Per i voltmetri e gli amperometri si può scegliere il multimetro digitale

Fig. 11 - Uso degli strumenti indicatori.



Name	Show	Minimum	Maximum	Precision
V	No	1.000e-012	1.000e+012	3
V(p-p)	No	1.000e-012	1.000e+012	3
V(rms)	No	1.000e-012	1.000e+012	3
V(dc)	Yes	1.000e-012	1.000e+012	3
Freq.	No	1.000e-003	1.000e+012	3
I	No	1.000e-012	1.000e+012	3
I(p-p)	No	1.000e-012	1.000e+012	3
I(rms)	No	1.000e-012	1.000e+012	3
I(dc)	No	1.000e-012	1.000e+012	3
Vgain(dc)	No	1.000e-012	1.000e+012	3
Vgain(ac)	No	1.000e-012	1.000e+012	3
Phase	No	1.000e-012	1.000e+012	3

Fig. 12 - Definizione dei parametri della sonda.

selezionabile tra gli strumenti di misura con il tasto (Multimeter) oppure con il percorso **Simulate\Instruments\Multimeter**.

Nel nostro esempio (Fig. 7) abbiamo previsto una misura di corrente e una di tensione (fate doppio clic sui multimetri per ingrandirli e stabilire le misure con i pulsanti disponibili). Notate come il multimetro si possa usare sia in corrente continua (è il nostro caso) che in alternata e che con tale strumento si possano misurare tensioni, correnti, resistenze e decibel.

Avviando la simulazione potremo leggere i valori numerici delle tensioni e delle correnti così come si vede nella Fig. 7. Notate altresì come non sia necessario fare una misura alla volta, come faremmo nella realtà se avessimo a disposizione un solo multimetro: infatti qui possiamo inserire contemporaneamente più strumenti.

Salviamo, al solito, il nostro file (diciamo **cir-cuito_DC_1.1**).

Il pulsante **Set...** permette di aprire una finestra (Fig. 8) per l'impostazione dei parametri dello strumento; comunque i valori predefiniti sono, nella maggioranza dei casi, adatti. Se però, per esempio, si volessero fare delle valutazioni sugli autoconsumi degli strumenti, si potrebbe decidere, per renderli più evidenti,

di ridurre le resistenze dei voltmetri e aumentare quelle degli amperometri.

Riguardo all'uso degli strumenti virtuali ed alle indicazioni che forniscono, facciamo notare che:

- **Ammeter resistance** è la resistenza interna dell'amperometro che deve essere piccola;
- **Voltmeter resistance** è la resistenza interna del voltmetro che deve essere grande;
- **Ohmmeter current** è la corrente attraversata dalla resistenza sotto misura;
- **dB relative value** indica il valore di tensione considerato a 0 dB; il valore di default (774,597 mV) determina una potenza di 1 mW su un carico di 600 Ω e quindi di default il valore letto è in dBm.

In alternativa all'uso del multimetro (DMM) possiamo usare degli strumenti dedicati recuperabili tra i componenti con il tasto (oppure impartendo il comando **Place\Component\Master Database\Group\Indicators**) e scegliere i voltmetri e gli amperometri che ci servono (Fig. 9).

Procedendo al solito modo, possiamo inserire gli strumenti nel nostro circuito; anche in questo caso con un doppio clic sui singoli strumenti possiamo definire le relative caratteristiche (Fig. 10): lasciamo le resistenze interne

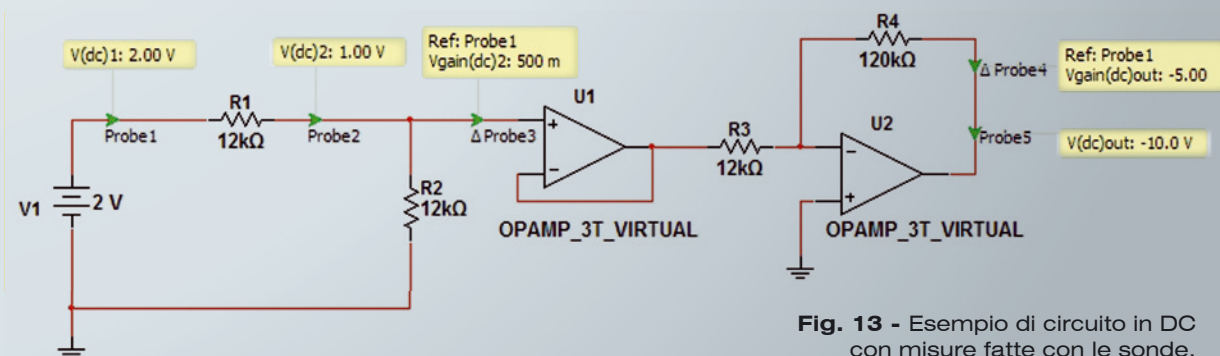
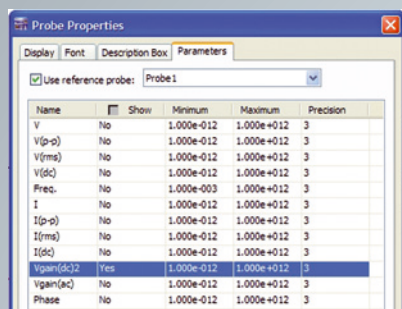


Fig. 13 - Esempio di circuito in DC con misure fatte con le sonde.

Fig. 14
Impostazione
della sonda 3.



ai valori di default e fissiamo il funzionamento in continua (DC).

A circuito realizzato, possiamo procedere alle misure avviando la simulazione e otterremo, ovviamente, gli stessi risultati di prima (Fig. 11). Salviamo al solito il nostro file (diciamo... **circuito_DC_1.2**).

Un ultimo modo strumentale disponibile per simulare il circuito è quello di utilizzare lo strumento sonda (**Probe**). La sonda più completa si ottiene, a simulazione ferma, semplicemente facendo clic sul pulsante  nella barra degli strumenti (oppure impartendo il comando accessibile dal percorso **Simulation\Instruments\Measurement Probe**). Se invece, con lo stesso procedimento, operiamo a simulazione attiva, otteniamo una sonda che misura solo tensioni ma che si può spostare con il mouse nei vari punti di misura (come faremmo nella realtà).

Esistono però anche alcune opzioni alternative (raggiungibili con la freccina posta sotto il simbolo della sonda o con il comando che tro-

vate sul percorso **Simulation\Instruments\Preset Measurement Probes**):

- **From dynamic probe setting:** si comporta esattamente come già descritto;
- **AC voltage:** a simulazione inattiva permette di applicare nel punto desiderato una sonda per misurare la tensione in alternata;
- **AC current:** a simulazione inattiva permette di applicare nel punto desiderato una sonda per misurare la corrente in alternata;
- **Instantaneous voltage and current:** a simulazione inattiva permette di applicare nel punto desiderato una sonda per misurare i valori istantanei di corrente e tensione.
- **Voltage with reference to probe:** permette di fissare il punto di inserimento di una sonda come punto di riferimento per le misure delle altre sonde.

Le sonde virtuali utilizzabili dal programma misurano le seguenti grandezze:

- **V:** valore istantaneo della tensione;
- **V(p-p):** valore picco-picco della tensione;
- **V(rms):** valore efficace della tensione;
- **V(dc):** valore continuo della tensione;
- **I:** valore istantaneo della corrente;
- **I(p-p):** valore picco-picco della corrente;
- **I(rms):** valore efficace della corrente;
- **I(dc):** valore continuo della corrente;
- **Freq:** frequenza della tensione misurata;
- **Vgain (dc):** guadagno in corrente continua

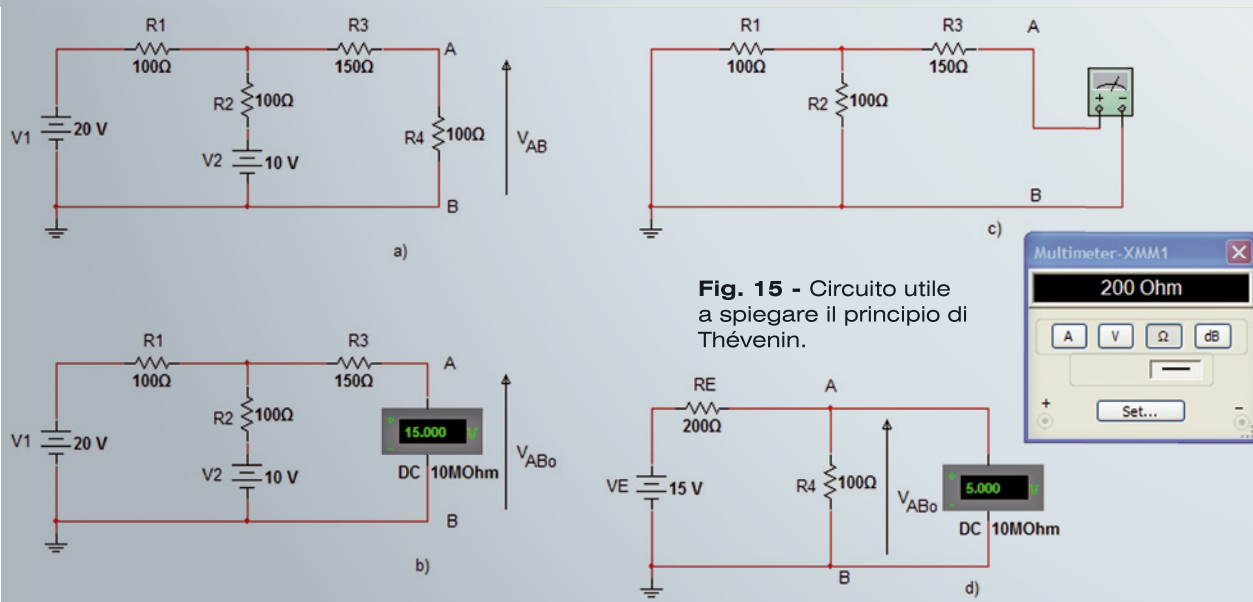


Fig. 15 - Circuito utile
a spiegare il principio di
Thévenin.

rispetto al riferimento scelto;

- **Vgain (ac):** guadagno in regime alternato rispetto al riferimento scelto;
- **Phase:** misura di fase rispetto al riferimento scelto.

L'impostazione delle singole sonde inserite può essere effettuata con il solito doppio clic del mouse o con il comando accessibile dal percorso **Simulate\Dynamic Probe Properties**. Nella Fig. 12 ne vedete un esempio. Cliccando su **Parameters** possiamo definire le proprietà della sonda; per impostazione predefinita sono attive tutte le possibili misurazioni. Togliendo lo spunto a *Show* si disattivano tutte le misure e quindi facendo clic sulle singole righe attiviamo solo quelle che ci interessano (noi abbiamo attivato solo la misura di tensione in continua).

Notate anche che con un doppio clic sulle singole voci possiamo cambiare il nome che verrà visualizzato per la misura (esattamente come si fa in Windows per rinominare un file), gli estremi di misura e il numero di cifre significative. Per impostazione predefinita, le misure di tensione sono valutate rispetto a massa. Concretizziamo il tutto con un esempio, realizzando il circuito indicato nella Fig. 13, che al solito salveremo con un nome: abbiamo scelto **circuito_DC_2**.

Per realizzare questo circuito tenete presenti le seguenti considerazioni:

- gli operazionali scelti sono di tipo virtuale; trattandosi in questo caso di un circuito puramente didattico, possiamo evitare di scegliere un componente commerciale, così non dobbiamo preoccuparci di collegarlo all'alimentazione;
- per impostazione predefinita, l'operazionale è alimentato a ± 12 V; ciò nel nostro caso va bene, ma se prevedessimo tensioni di uscita superiori ai limiti dell'alimentazione, basterebbe accedere ai parametri dell'operazionale con il solito doppio clic e modificare le voci **Positive Voltage Swing** e **Negative Voltage Swing**;
- abbiamo assegnato i nomi alle misure delle diverse sonde e come le sonde 3 e 4 forniscano valori relativi alla tensione della sonda 1, secondo le formule:

$$V_{\text{gain(dc)1}} = \frac{V_{\text{(dc)2}}}{V_{\text{(dc)1}}} = 0,5 \text{ (500m)} \quad V_{\text{gain(dc)out}} = \frac{V_{\text{(dc)out}}}{V_{\text{(dc)1}}} = -5$$

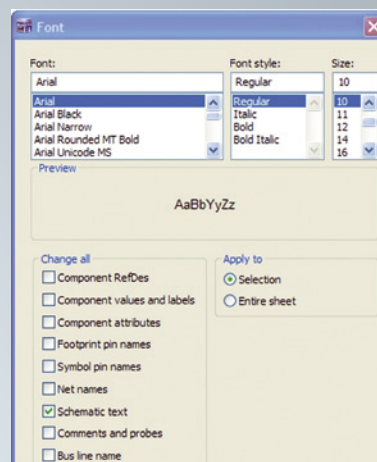


Fig. 16 - Scelta del carattere e del suo corpo per le scritte di testo (Schematic text).

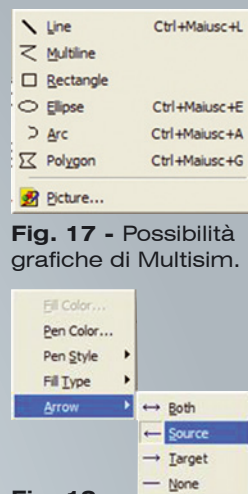


Fig. 18 Scelta del tipo di freccia.

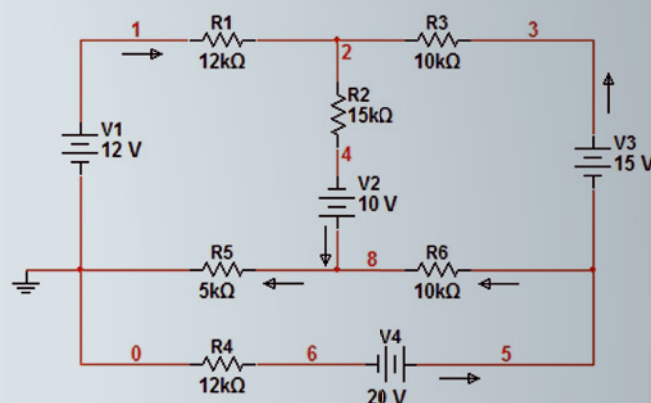


Fig. 19 - Circuito da analizzare in DC.

- la misura di corrente avvenga senza dover aprire il circuito per inserire lo strumento, come invece avverrebbe se usassimo un amperometro;
- per maggiore chiarezza sull'impostazione delle sonde, osservate la Fig. 14, relativa alla sonda 3 e notate, in particolare, lo spunto alla voce **Use reference probe** per assegnare la sonda 1 come riferimento.

Facciamo adesso un breve commento al circuito per verificare teoricamente la correttezza delle misure: la sonda 2 rileva la tensione rispetto a massa, ovvero quella localizzata ai capi della resistenza R2; siccome R1 ed R2 sono in serie (l'operazionale U1, in configurazione voltage follower, ha resistenza di ingresso idealmente infinita e quindi non carica il circuito) e sono uguali, la tensione misurata è la metà di V1, ovvero 1 V. Il tutto equivale

a considerare il guadagno in questo punto rispetto a V_1 pari a 0,5. Il primo operazionale ha un guadagno unitario mentre il secondo, in configurazione invertente, ha guadagno -10 e quindi il guadagno in uscita è $0,5 \cdot 1 \cdot (-10) = -5$. Per comprendere pienamente la valenza non solo progettuale ma anche didattica di Multisim, vediamo l'esempio proposto dalla Fig. 15, il cui file corrispondente è stato da noi chiamato **circuito_DC.3**; in questo esempio supponiamo di voler valutare la tensione V_{AB} del circuito di Fig. 15a applicando il principio di Thévenin. Nella realizzazione dei circuiti non preoccupatevi se, passando da un circuito all'altro, sempre nella stessa videata (come fatto da noi per meglio tenere d'occhio il tutto), le resistenze e i generatori assumono numerazioni crescenti, anziché ricominciare daccapo. E non preoccupatevi neanche delle altre scritte e delle frecce (ce ne occuperemo dopo). Ora seguite il ragionamento sull'analisi circuitale: la teoria dice che per calcolare V_{AB} si deve staccare R_4 , misurare la tensione ai terminali A e B a vuoto V_{AB0} , che rappresenta la forza elettromotrice del generatore equivalente V_E , (in termini sperimentali basta misurare la tensione ai terminali A e B della Fig. 15b); per ricavare la resistenza interna del generatore reale bisogna misurare la resistenza vista dai morsetti A e B con i generatori annullati (cortocircuitati, perché sono generatori di tensione). Il valore così ottenuto rappresenta la resistenza R_E del generatore equivalente (sperimentalmente, è la situazione di Fig. 15c); infine, bisogna applicare la resistenza R_4 , che avevamo tolto, ai morsetti del generatore equivalente. Così la tensione ai capi di R_4 è uguale a quella del circuito iniziale (è il caso

di Fig. 15d). Se poi non siete convinti, per farvene una ragione vi basterà che applichiate un voltmetro ai terminali A e B del circuito iniziale di Fig. 15a. Per assegnare ai componenti i valori voluti, abbiamo disattivato, tra le proprietà del circuito, la voce **RefDes** (come spiegato nella prima puntata) e scritto con il path **Place\Text**: posizionato il cursore nel punto desiderato si può scrivere usando la tastiera del PC; cliccando sulla scritta con il tasto destro del mouse e scegliendo **Font**, possiamo anche definire il carattere e le sue dimensioni (*Size*), come indicato nella Fig. 16. Naturalmente allo stesso modo possiamo agire anche sui caratteri dei valori dei componenti, sulle eventuali etichette, ecc. Le scritte possono poi essere spostate con il mouse o con i tasti freccia del PC. Per creare le frecce, con il percorso **Place\Graphics\Line** decidiamo di tracciare una linea: notate, come indicato nella Fig. 17, che esistono molte altre possibilità, compresa quella di importare con la voce **Picture**, un'immagine in formato Bitmap. Dopo aver tracciato la linea e averla evidenziata con il mouse, possiamo definire le sue caratteristiche con **Edit\Graphic\Annotation\Arrow\Source**: notate, anche in questo caso (Fig. 18), l'esistenza di molte alternative, non solo sul tipo di freccia ma anche sullo stile e il colore della linea.

USO DEL MENU SIMULATE

Per conoscere questa modalità, realizziamo un circuito un poco più complesso, che ci permette di meglio valutare la potenzialità del metodo. Nella Fig. 19 trovate il circuito che proponiamo; in esso la posizione della massa è arbitraria: basta solo che ci sia (le frecce le abbiamo

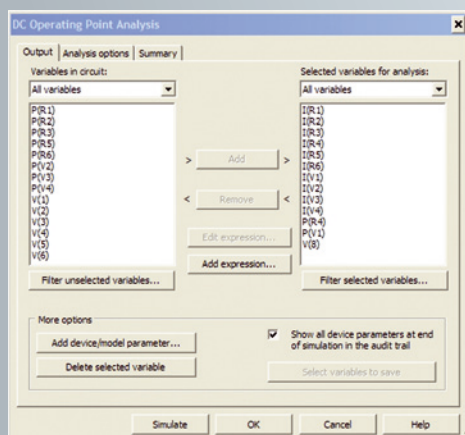


Fig. 20
La scelta delle grandezze da misurare.

DC Operating Point	
1 I(R1)	38.46155 u
2 I(R2)	1.05325 m
3 I(R3)	-1.01479 m
4 I(R4)	1.10947 m
5 P(R4)	14.77102 m
6 I(R5)	-1.14793 m
7 I(R6)	-94.67455 u
8 P(V1)	-461.53071 u
9 V(R1)	5.73964
10 V(V1)	-38.46156 u
11 V(V2)	-1.05325 m
12 V(V3)	-1.01479 m
13 V(V4)	-1.10947 m

Fig. 21 - I risultati cercati con l'analisi in DC.

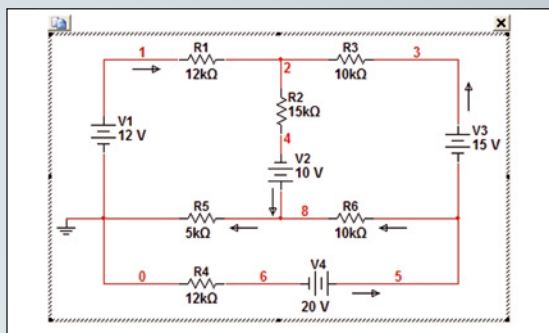


Fig. 22 - La funzione Capture Screen Area.

disegnate con il metodo visto nel precedente esempio). Il classico metodo di analisi è quello di scrivere (e risolvere) un sistema di sei equazioni in altrettante incognite, tante quante sono le correnti incognite, applicando i principi di Kirchhoff ai nodi e alle maglie. Noi possiamo farlo in modo molto più semplice: una volta realizzato e salvato il circuito (**circuito_DC.4**) è necessario, se già non avviene come in **Fig. 19**, visualizzare i nodi (**Option\Sheet Properties** e spuntare **Show All** per visualizzarli tutti o **Use Net-specific Setting** per visualizzare solo quelli che ci interessano (rileggete quanto spiegato -a riguardo- nella prima puntata); noi li abbiamo evidenziati tutti. Notate anche che, mentre nella teoria classica di Kirchhoff si dicono nodi i punti di connessione almeno tripli (cioè tra tre o più rami di circuito) qui sono nodi anche i punti di connessione tra due soli componenti.

A questo punto con **Simulate\Analyses\DC operating point** potremo accedere alla schermata per la simulazione in DC e scegliere le variabili da valutare (in **Fig. 20** a sinistra vedete quelle non considerate e a destra quelle scelte). In questo caso abbiamo scelto tutte le correnti nei resistori ($I(R1)$, $I(R2)$...) e nei generatori ($I(V1)$, $I(V2)$...), la tensione tra il nodo 8 e massa ($V(8)$), la potenza dissipata in $R4$ ($P(R4)$) e quella della batteria $V1$ ($P(V1)$). Con il tasto **Simulate** possiamo ora ottenere tutti i risultati, come mostrato nella **Fig. 21** (con il tasto  possiamo anche stabilire di avere lo sfondo nero o bianco). Per quanto concerne le convenzioni, tenete presente che le tensioni sono valutate rispetto a massa, le correnti nei resistori sono positive se vanno da sinistra a destra o dall'alto verso il basso, le tensioni nei generatori sono positive se entranti nel polo +. I versi delle correnti, indicati nella **Fig. 19** sono quelli corretti (valutati a posteriori dalla lettura dei dati di **Fig. 21**).

LA FUNZIONE CAPTURE SCREEN AREA

Con il tasto , oppure con il comando **Tools**

Capture Screen\Area, è possibile aprire un rettangolo di dimensioni regolabili che permetta la cattura, come immagine negli appunti di Windows, dell'area racchiusa dal rettangolo. In questo modo potremo facilmente copiare circuiti e grafici di simulazione o parti di essi e trasferirli, ad esempio, in Word per realizzare una relazione, ma anche in Paint per manipolare l'immagine. Nella **Fig. 22** vedete un esempio con il nostro ultimo circuito.

LA FUNZIONE CONVERGENCE ASSISTANT

La convergenza è il punto di arrivo di ogni algoritmo che utilizza l'iterazione o la ricorsione per elaborare una serie di dati. Può succedere che i parametri predefiniti usati nella simulazione non permettano la convergenza, ovvero il raggiungimento di risultati univoci; quando accade ciò (errore di convergenza) durante la simulazione in Multisim, appare un messaggio che chiede all'utente se desidera eseguire il **Convergence Assistant** per tentare di risolvere automaticamente il problema. Difficilmente vi sarà successo con le semplici simulazioni sin qui proposte, ma comunque, siccome prima o poi vi capiterà, è utile sapere subito come procedere se si verifica tale evenienza. Per l'esattezza, dovete fare clic su **Yes** quando appare il messaggio in questione; allora inizierà in automatico la procedura di risoluzione del problema attraverso una sequenza di passaggi per ricercare la convergenza attraverso l'ottimizzazione di diversi parametri di simulazione. A convergenza raggiunta, apparirà in una finestra di dialogo una relazione di sintesi con il dettaglio sulle modifiche apportate (attenzione che si tratta di modifiche sui parametri usati da Multisim ma non riguardano il circuito). Nel caso la convergenza non avvenga, ricontrollate il circuito perché potrebbe presentare errori e leggete le indicazioni fornite da Multisim sul tipo di errore riscontrato. ■

I file delle simulazioni presentati in questo articolo sono scaricabili gratuitamente dal sito www.elettronica.in.it.

Il software Multisim fa parte del pacchetto NI Circuit Design Suite Education (cod. 8100-NICDSE) disponibile presso www.futurashop.it. La versione per utenti privati e studenti costa 99,95 Euro.

I moderni smartphone integrano un processore per le comunicazioni, uno applicativo ed un gestore dell'alimentazione, che condividono la porta USB 2.0; scopriamo le soluzioni progettuali che consentono di condividere questa risorsa.

LA CONDIVISIONE DELL'USB NEGLI SMARTPHONE

di OSCAR FREITAS



I dispositivi mobili necessitano di molteplici circuiti integrati per riuscire a gestire le numerose funzionalità richieste dagli utenti; ad esempio, un tipico smartphone dispone di un processore per le comunicazioni, un processore applicativo e un integrato per la gestione dell'alimentazione, i quali devono tutti condividere un'unica porta USB comunicando attraverso essa a velocità di 480 Mbps; ciò può creare qualche problema nella condivisione di

tale risorsa. In queste pagine vogliamo prendere in considerazione le soluzioni adottabili in pratica nella progettazione e nell'uso di dispositivi con USB condiviso, analizzando i pro e i contro delle varie strade percorribili, che spaziano da un hub USB fino a un semplice switch analogico. Un metodo comunemente utilizzato nella progettazione degli smartphone prevede di fare in modo che il dispositivo preposto alla gestione dell'alimentazione all'interno dell'apparecchio controlli

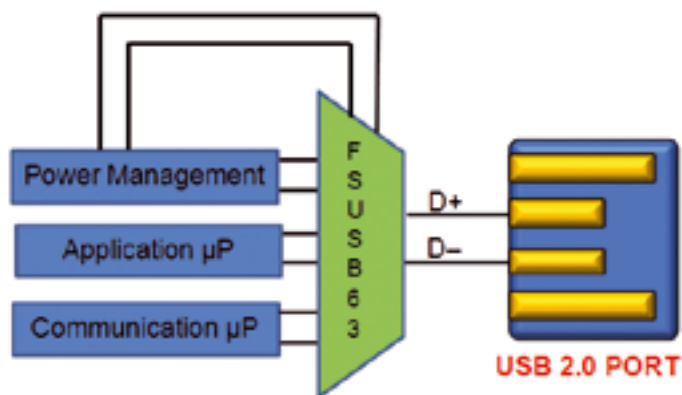


Fig. 1 - Switch multiplexer USB 2.0 utilizzato per condividere un'unica porta USB 2.0.

L'unica porta USB 2.0 presente. A tale scopo viene utilizzato uno switch multiplexer USB 3:1, che commuta la porta USB per svolgere la funzione indicata. Per impostazione predefinita, lo switch commuta la porta USB anche a beneficio sia del processore applicativo utilizzato per funzionalità multimediali quali la riproduzione di file MP3 o l'elaborazione video, sia del processore preposto alle comunicazioni radio, tanto per l'accesso dati quanto per le telefonate (si veda la Fig. 1). Quest'architettura offre il vantaggio di permettere l'attivazione della modalità sleep sull'apparecchio quando la funzione non è richiesta. L'unità preposta alla gestione dell'alimentazione può inoltre attivare il processore pertinente quando viene rilevata un'attività sulla porta USB 2.0, oppure quando uno dei processori ne richiede l'accesso. Quando viene rilevato il collegamento di un dispositivo alla presa USB, il circuito che controlla la gestione dell'alimentazione può anche determinare se si tratta di un caricabatterie oppure di un host con capacità di carica, in modo tale da ricaricare la batteria direttamente tramite il segnale VBUS. Il layer fisico (PHY) del processore di comunicazione utilizza integralmente la banda dati ad alta velocità (480 Mbps) dello switch USB

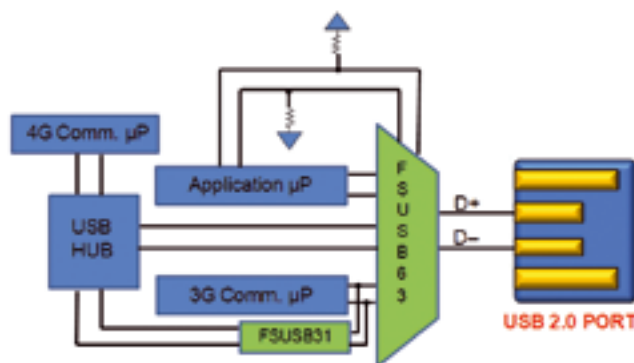
in fase di comunicazione con un host USB, quale, ad esempio, un PC.

Una recente tendenza riguardante la progettazione dei telefoni 4G implica la presenza nell'apparecchio di due elementi di elaborazione, che devono accedere contemporaneamente alla porta USB; per questo genere di esigenza, un hub potrebbe costituire l'opzione più adatta. Il problema è che le connessioni che caratterizzano l'hub USB sono generalmente caratterizzate da un'elevata capacità e da una richiesta di potenza considerevole, a fronte del fatto che l'accesso contemporaneo alla porta USB è infrequente. Un tipico layout per questo genere di apparecchi è illustrato nella Fig. 2, dove il processore di comunicazione 3G può accedere alla porta USB distintamente dal processore 4G. Mediante uno switch isolato (FSUSB31) e tracce estremamente brevi sulla scheda PCB, è possibile minimizzare la

capacità, tanto che la maschera dei segnali di trasmissione USB ad alta velocità, nell'ambito del path dal processore di comunicazione 3G verso l'host USB, offre ampi margini per rispettare le specifiche USB. Nell'esempio indicato, il processore applicativo controlla lo switch multiplexer USB 3:1, il quale può incorporare nelle proprie linee di controllo resistenze pull-up e pull-down, in modo tale da commutare normalmente sul processore applicativo quando quest'ultimo viene disattivato e posto nella modalità standby.

La funzionalità sopra descritta è resa possibile dalla disponibilità di switch che assorbono quantità minime di corrente quando attivati e presentano consumi praticamente pari a zero se disabilitati. Nell'esempio dello smartphone 4G illustrato nella Fig. 2, il dispositivo FSUSB63 è sempre attivo e consuma soltanto pochi microampere di corrente. Anche nell'esempio mostrato nella Fig. 1, quando il circuito di gestione dell'alimentazione può spegnere il dispositivo FSUSB63 in modalità standby, il consumo di corrente è inferiore a un microampere. Nella maggior parte dei circuiti integrati, alte velocità e bassi consumi sono generalmente due caratteristiche in contrasto fra loro e molto spesso la soluzione per ottenere entrambe queste ca-

Fig. 2
Esempio di switch multiplexer USB 2.0 e hub con switch isolato.



ratteristiche consiste nel ridurre la tensione in gioco e utilizzare processi geometrici accurati, tuttavia le specifiche USB richiedono segnali a tensione elevata. Inoltre, volendo progettare un apparecchio robusto, in cui le linee D+ e D- sono in grado di sopravvivere a un cortocircuito della linea VBUS a 5 V, non è possibile utilizzare basse tensioni. Nonostante ciò, in tempi recenti sono comparsi gli switch USB a pompa di carica a bassa tensione, che sono in grado di fornire una risposta al problema, soddisfacendo gli stringenti requisiti della maschera dei segnali USB ad alta velocità, come illustrato nella Fig. 3. In effetti, la maggioranza di questi circuiti è in grado di supportare velocità dati USB più che doppie (> 1 Gbps) senza necessità di modifiche all'architettura o ai dispositivi prescelti.

Sovente, il passaggio da un processore a un altro tramite uno switch USB viene effettuato innanzitutto scollegando tutti i path e lasciando trascorrere un tempo sufficiente, in modo tale che il controller della porta dell'host USB possa riconoscere l'interruzione della connessione e passare all'altro path, permettendo quindi all'host di resettare e rinumerare il nuovo dispositivo USB. Ciò viene fatto via software sul cellulare e richiede una debita attenzione da parte dei progettisti, che nella stesura del software devono tenere conto delle limitazioni imposte dall'hardware prescelto.

Gli switch USB più recenti, tuttavia, permettono un più ampio ricorso al software nei dispositivi portatili, rendendo questa situazione un problema ormai superato. Ciò è possibile perché qualunque cambiamento nella selezione dei segnali di controllo di tali switch USB avvia automatica-

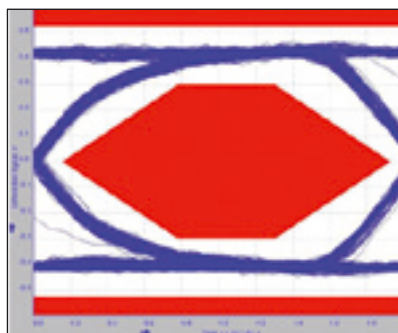


Fig. 3 - Maschera dei segnali ad alta velocità USB 2.0.

mente il processo di interruzione della connessione per un tempo predeterminato; tale capacità è imposta dalle specifiche USB 2.0 ad alta velocità, che richiedono che un host USB sia in grado di riconoscere l'interruzione della connessione, predisponendo una nuova connessione interna.

Come accennato precedentemente, il risparmio di corrente è un imperativo per la maggior parte dei dispositivi portatili, pertanto le tensioni di alimentazione dei processori vengono spesso mantenute su valori di 1,2V o inferiori. Ne consegue che in fase di interfaccia con dispositivi operanti a tensioni più elevate, nei quali l'alimentazione è collegata direttamente alla tensione della batteria, la corrente assorbita dalla batteria da parte degli input operanti a tensioni inferiori è considerevole. Un metodo per ridurre il consumo di corrente consiste nell'impiegare trasformatori di tensione. Poiché la commutazione USB è un processo intrinsecamente lento, in tali circostanze gli switch USB sopracitati assorbono un quantitativo di corrente irrilevante impostando le soglie di ingresso in base alla tensione di alimentazione più bassa. Inoltre, i buffer di ingresso sono progettati in modo da conservare potenza nei casi di differenza di tensione peggiori, facilitando il compito dei progettisti di sistema.

Costi e ingombri delle schede PCB (Printed Circuit Board) sono due aspetti che devono essere tenuti sempre in considerazione in tutti i dispositivi portatili di dimensioni contenute. Oggi gli switch sono realizzati in package compatti aventi un pitch dei pin di 0,4mm, occupando una porzione tanto contenuta della scheda PCB da surclassare totalmente le alternative costituite da grandi connettori proprietari che richiedono dongle specializzati per la connettività USB. È inoltre il caso di rammentare che il costo di uno switch è di gran lunga inferiore a quello di un hub USB.

Talvolta può essere necessario collegare in cascata più switch USB, per esempio in applicazioni dove un unico side loading controller utilizza lo standard Media Transfer Protocol per consentire il download di filmati da un PC host a una scheda di memoria SD senza sovraccaricare il processore applicativo dello smartphone. In questi casi la capacità dello switch USB nello stato OFF rappresenta un fattore critico per consentire il traffico USB ad alta velocità riducendo al minimo i tempi di download. La OFF capacitance dei nuovi switch USB è, inverso, talmente bassa (tipicamente 2 pF) da consentire di aggiungere più switch anche negli stadi finali di un design allo scopo di integrare ulteriori funzionalità nello smartphone. Allo scopo di ridurre al minimo il numero di connettori presenti su uno smartphone, talvolta la porta USB è progettata per essere collegata a un auricolare analogico, oltre che a un dispositivo USB high speed e/o full speed. Speciali switch USB permettono di gestire questa situazione istradando il segnale del microfono incorporato nell'auricolare attraverso la linea VBUS del

FIERE ELETTRONICA INFORMATICA e RADIANTISMO

**& Mercatino
Radio Amatoriale**

**26-27
MARZO
BOLOGNA**

**PALANORD
PARCONORD**

Da stazione FS - Bus n.25 ogni 15'
Uscita Tangenziale n. 7 bis

**ORARIO CONTINUATO 9.00 - 18.30
PARCHEGGIO GRATUITO**

**PROSSIMI
APPUNTAMENTI**

**2
0
1
1**

**CIRCUITO FIERE
mondo
elettronica**

**10-11 SETTEMBRE
PIACENZA**
FIERA DI PIACENZA - LOC. LE MOSE

**24-25 SETTEMBRE
RAVENNA**
PALAZZO MAURO DE ANDRE'

**8-9 OTTOBRE
FERRARA**
FIERA DI FERRARA - VIA DELLA FIERA 11

**22-23 OTTOBRE
BOLOGNA**
PALA NORD - PARCONORD

Tel. 0545 27548 - Cell. 348 9030832
www.mondoelettronica.net

VALE COME RIDOTTO ANCHE IN FOTOCOPIA

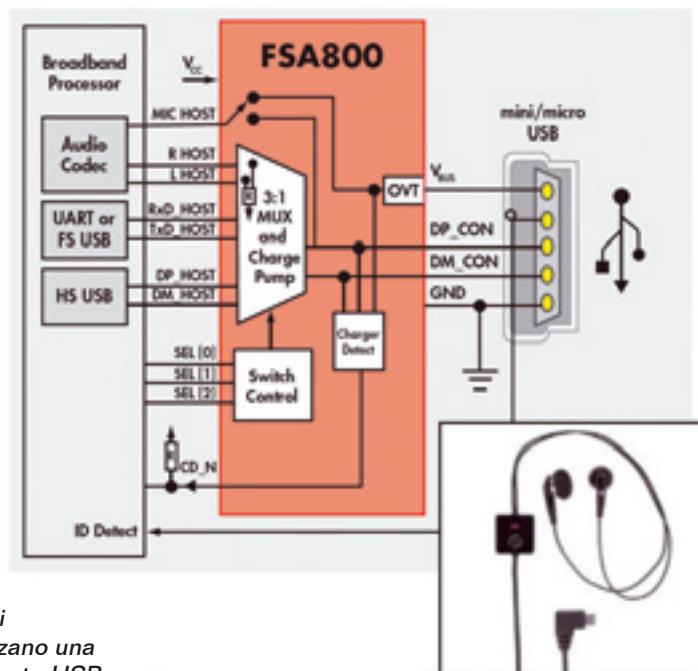


Fig. 4
*Auricolari
che utilizzano una
singola porta USB.*

connettore USB, mentre i segnali D+ e D- vengono indirizzati alle linee rispettivamente degli altoparlanti destro e sinistro. Un tipico esempio di switch USB è l'FSA800 di Fairchild, il quale è in grado di determinare, al momento della connessione, se alla porta USB sia stato collegato un caricabatterie utilizzando l'algoritmo specifico per la carica della batteria via USB. Ciò permette a un processore di controllare la selezione del path dello switch in base allo stato del segnale del pin ID del connettore USB.

Altri switch USB utilizzano il pin ID per il rilevamento e la configurazione automatica, rendendo possibile l'uso di innumerevoli accessori avanzati, tra cui figurano una vasta gamma di dispositivi quali speciali cavi di collaudo che sfruttano al meglio le procedure di test in fabbrica per ottimizzarne i costi, sofisticati auricolari dotati di tutti i controlli per la riproduzione dei file MP3 e accessori per l'ascolto delle radio e l'effettuazione/ricezione delle chiamate che consentono di passare liberamen-

te da una funzione all'altra. Gli switch USB ad alta velocità hanno permesso di integrare nei moderni smartphone un'enorme varietà di funzioni, permettendo nel contempo di contenere lo spessore degli apparecchi a poco più di quello dei connettori compatti ad alta velocità micro-USB. Tali connettori sono in grado di supportare un processore applicativo, un processore di comunicazione, un hub USB (quando è necessario comunicare contemporaneamente con più dispositivi), un driver audio e, se desiderato, un'intera gamma di altre funzioni aventi velocità inferiori. Tutto ciò è possibile in considerazione della circuiteria avanzata integrata nei più recenti switch USB, senza sottostare a compromessi per quanto riguarda consumi di corrente, ingombri sullo stampato e costi, e comunque fornendo la funzione di rilevamento della connessione dei caricabatterie USB, contribuendo attivamente all'impegno globale volto a fare della tecnologia USB lo standard per tutti i dispositivi portatili.

AVR DIGITAL FRAME, IL FIRMWARE

Completiamo la descrizione della nostra cornice digitale, spiegando il firmware e l'interfaccia di programmazione ICSP per Personal Computer.

di ANDREA CHIAPPORI

Dopo aver esaminato in linea di massima il funzionamento del nostro programma, possiamo analizzare più in dettaglio alcune righe del codice, scaricabile dal nostro sito www.elettro-nicain.it. Il progetto è diviso in più file (Fig. 5) raggruppabili in due sezioni: la gestione del display e quella della FAT. Iniziamo dal display, scendendo subito al livello più basso, ovvero all'invio dei comandi e dei dati che ci permettono di disegnare pixel su determinate zone dello schermo. Il protocollo impiegato per la comunicazione tra LCD e micro è il classico SPI e può essere realizzato in software o in hardware (controllando il modulo SPI interno al micro), tramite la define SPI_HW, che nel nostro caso è attiva. Tutto si basa sull'invio di un comando e dati da parte del micro verso il display. Alcuni comandi devono essere seguiti da uno stream di dati che rappresenta l'immagine, altri vengono usati per configurare i registri interni al controller del display. Ad esempio, nella fase di init basterà inviare comandi senza alcun dato, mentre per visualizzare un pixel o un'immagine, ai comandi



Listato 1

```
static void sendByte(unsigned char value)
{
#ifdef SPI_HW          // hardware
    spiTransferByte(value);
#else                  // software

    unsigned char mask;

    for(mask=0x80; mask!=0; mask>>=1)
    {
        PIN_CLK_LOW();
        if(mask & value)
        {
            PIN_DAT_HIGH();
        }
        else
        {
            PIN_DAT_LOW();
        }
        PIN_CLK_HIGH();
    }
    PIN_CLK_LOW();
#endif
}

static void sendCommand(unsigned char reg, unsigned char value)
{
    PIN_DATACOMMAND_HIGH(); //cmd

    PIN_CS_LOW();

    sendByte(reg);
    sendByte(value);

    PIN_CS_HIGH();
}

void LS020_Init(void)
{
#ifdef SPI_HW
    SPI_MasterInit(FOSC_2);
#endif

    // configuration port as output
    DDRB |= (1<<CS);
    DDRB |= (1<<RST);
    DDRD |= (1<<DC);

    // start condition
    PIN_CS_HIGH();
    PIN_DATACOMMAND_HIGH();

    // reset
    PIN_RST_LOW();
    _delay_ms(50);
    PIN_RST_HIGH();
    _delay_ms(50);

    // display software reset
    sendCommand(0xFD, 0xFD);
    sendCommand(0xFD, 0xFD);

    _delay_ms(50);

    // init for internal voltage setting
    sendCommand(0xEF, 0x00);
    sendCommand(0xEE, 0x04);
    sendCommand(0x1B, 0x04);
    sendCommand(0xFE, 0xFE);
    sendCommand(0xFE, 0xFE);
    sendCommand(0xEF, 0x90);
    sendCommand(0x4A, 0x04);
    sendCommand(0x7F, 0x3F);
    sendCommand(0xEE, 0x04);
    sendCommand(0x43, 0x06);

    _delay_ms(7); //need 7ms!

    // init 2
    sendCommand(0xEF, 0x90);
    sendCommand(0x09, 0x83);
    sendCommand(0x08, 0x00);
    sendCommand(0x0B, 0xAF);
    sendCommand(0x0A, 0x00);
    sendCommand(0x05, 0x00);
    sendCommand(0x06, 0x00);
    sendCommand(0x07, 0x00);
    sendCommand(0xEF, 0x00);
    sendCommand(0xEE, 0x0C);
    sendCommand(0xEF, 0x90);
    sendCommand(0x00, 0x80);
    sendCommand(0xEF, 0xB0);
    sendCommand(0x49, 0x02);
    sendCommand(0xEF, 0x00);
    sendCommand(0x7F, 0x01);
    sendCommand(0xE1, 0x81);
    sendCommand(0xE2, 0x02);
    sendCommand(0xE2, 0x76);
    sendCommand(0xE1, 0x83);

    _delay_ms(50);

    // display on
    sendCommand(0x80, 0x01);
    sendCommand(0xEF, 0x90);
```

seguiranno i dati che permetteranno di disegnare i pixel sullo schermo. Prima di ricevere i dati, il display deve anche essere informato sul numero di pixel da disegnare e quindi sulla quantità di dati che seguiranno; informazione, questa, passatagli andando a scrivere queste coordinate su determinati registri, elencati nella **Tabella 2** (puntata precedente). Per scrivere un valore in un registro, basterà inviare due byte consecutivi, il primo dei quali

specifica il valore dell'indirizzo e il secondo il dato. Per dire al nostro LCD se quelli che riceve sul bus SPI sono dati o comandi, si usa il piedino RS del display LS020, che va mantenuto a livello basso quando si invia un dato e a livello alto quando si invia un comando. Tutto questo è racchiuso nella funzione **sendCommand()** del **Listato 1**, in cui la macro **PIN_DATACOMMAND_LOW()** si occupa di portare a massa il piedino RS. Da notare anche le

macro **PIN_DAT_LOW()** e **PIN_DAT_HIGH()** usate per forzare a livello basso il Chip Select del display durante il trasferimento dei 2 byte. Ricordiamo anche l'LS020 non può essere letto, quindi i dati e comandi viaggiano sempre dal micro ad esso. Sempre nel **Listato 1**, è visibile anche la **Init()**, che va a scrivere in una serie di registri per consentire la corretta inizializzazione del controller; è opportuno sottolineare l'importanza dei ritardi inseriti (funzio-


```

sendCommand(0x00, 0x00);          //control registrer

//display options ( ROTATE / MIRROR)
sendCommand(0xEF, 0x90);

sendCommand(0x01, 0x00);          // YDIR XDIR * BGR * * * *

sendCommand(0x05, 0x00);          // X X X X      X ROT X X

return;
}

void LS020_PrepareImage(char width, char height, char posX, char posY, char colMode)
{
    sendCommand(0xEF, 0x90);

    sendCommand(0x06, posX);        // set x cursor pos
    sendCommand(0x07, posY);        // set y cursor pos

    sendCommand(0x08, posX); //x1    // set area
    sendCommand(0x09, posX + (width-1)); //x2

    if (colMode == 24) {
        sendCommand(0x0B, posY);        //y1
        sendCommand(0x0A, posY + (height-1)); //y2

        sendCommand(0xEF, 0x90);
        sendCommand(0x01, 0x50); // XDIR YDIR * BGR * * * *
    }
    else {
        sendCommand(0x0A, posY); //y1
        sendCommand(0x0B, posY + (height-1)); //y2
    }

    PIN_DATACOMMAND_LOW();
    PIN_CS_LOW();
}

void LS020_SendColor(int color)
{
    PIN_CS_LOW();

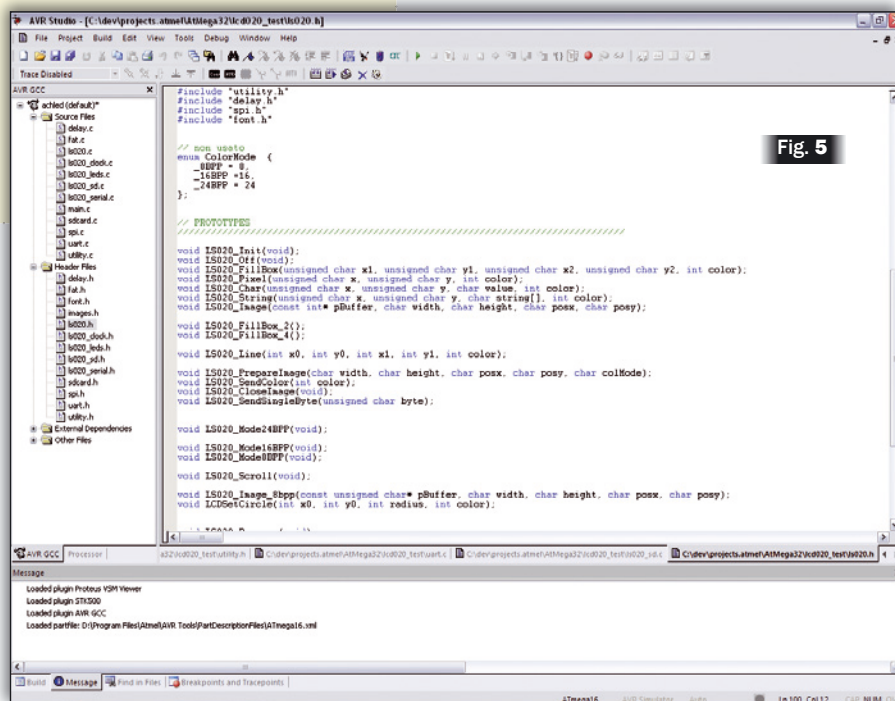
    sendByte(color >> 8); // high
    sendByte(color);       // low

    PIN_CS_HIGH();
}

```

ne **delay_ms()** tra vari blocchi di comandi. Questi sono, infatti, proprio necessari: un ritardo minore, ma anche uno maggiore, potrebbe non avviare il display in modo corretto o non avviarlo proprio! Ok, abbiamo già inizializzato il display e ora siamo in grado di impartirgli qualche comando; se ad esempio volessimo lo schermo tutto nero, senza per forza inviare le informazioni di colore di tutti i pixel, potremmo settare

semplicemente il bit BLK del registro 0x00, come mostrato in **Tabella 2**, ed il gioco è fatto. In realtà va detto che ogni registro appartiene ad un banco e prima di poter essere modificato occorre selezionare il banco di appartenenza con un apposito comando, che utilizza il registro 0xEF. Siccome per la nostra operazione dobbiamo intervenire nel registro 0x00 e questo, così come la maggior parte dei registri, appartiene al banco 0x90, prima di settare il bit BLK, dovremo accedere al banco 0x90 in questo modo: **sendCommand(0xEF, 0x90)**; solo dopo potremo settare il bit con **sendCommand(0x00, 0x40)**. A questo punto abbiamo tutto per fare un po' di prove; manca la parte principale e cioè disegnare un'immagine o una porzione di essa sullo schermo. Ormai, però, non dovrebbe essere un problema, perché, sempre da **Tabella 2**, possiamo utilizzare i registri per identificare una regione dello schermo (in particolare i registri 0x06, 0x07, 0x08, 0x09, 0x0A



Listato 2

```
void LS020_Scroll(void)
{
    sendCommand(0xEF, 0x90);
    int i = 1;
    while(1)
    {
        sendCommand(0x0F, 0); //scroll x start position (y)
        sendCommand(0x11, 30); //scrolling value width

        sendCommand(0x10, 30+i); //scrolling value end position

        i++;

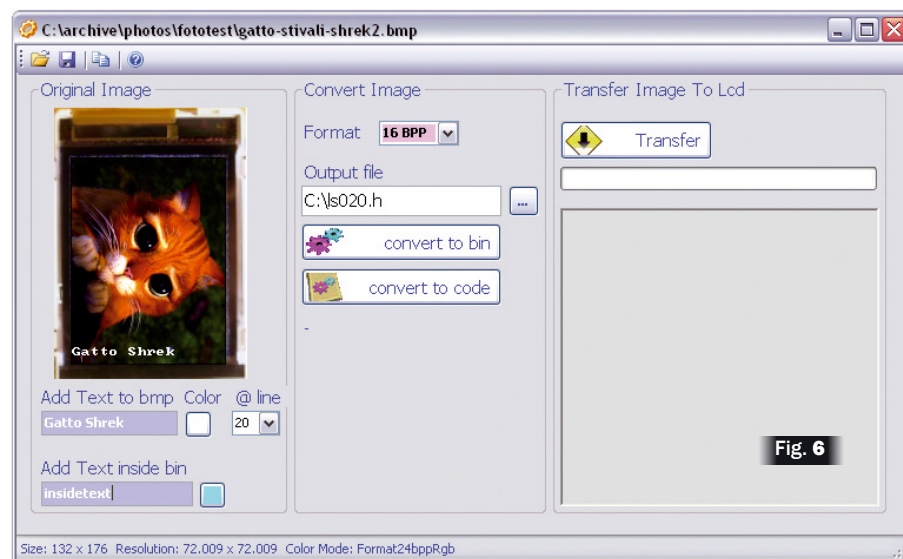
        if ((30+i)> 176)
            break;;
        delay_ms(30);
    }

    int val = (30+i);
    for(i=val;i>0;i--)
    {
        sendCommand(0x0F, 0); //scroll x start position (y)
        sendCommand(0x11, 30); //scrolling value width

        sendCommand(0x10, i);
        _delay_ms(30);
    }
}
```

e 0x0B) tramite la funzione **LS020_PrepareImage()** ed inviare successivamente i dati con la funzione **LS020_SendColor()**. Più in dettaglio, possiamo esaminare la funzione **showImageFromSD()** del **Listato 3**, che legge un file immagine dalla SD e lo visualizza utilizzando il meccanismo descritto.

Ora passiamo alla visualizzazione di stringhe e quindi di font, se non altro per rimarcare la facilità con cui si riesce, grazie all'uso di particolari registri interni, ad ottenere il risultato voluto senza scrivere ulteriore codice. I font possono essere, infatti, considerati come immagini di dimensione 8x8 pixel (8 byte definiscono un font) memorizzate in una tabella. Finora abbiamo considerato l'origine degli assi nell'angolo superiore sinistro, ma se volessimo scrivere testo lungo l'asse y (ovvero il lato più lungo) prima di ridisegnare lo stesso font potremmo settare opportunamente i registri 0x01 e 0x05 (più in particolare, i bit ROT, X-R e Y-R) e lo vedremo magicamente ruotato! Ciò vale anche per le immagini.



Ora il più è fatto; resta ancora da provare un'altra funzione che ci regala il display LS020: lo scrolling. Con poche righe di codice, come mostrato nel **Listato 2**, è possibile stabilire la linea (coordinata Y) di inizio dello scrolling, la larghezza dell'area di scrolling e variando il valore del registro 0x11 (che contiene la linea corrente di spostamento) vedremo la nostra immagine o parte di essa scendere e salire con una velocità

che fisseremo a nostra scelta. Nel progetto sono presenti anche file di più alto livello che, utilizzando le funzioni base, riassunte brevemente nella **Tabella 5**, permettono la creazione di qualche semplice animazione come stringhe scorrevoli su LED e un orologio a lancette. In particolare, nel **Listato 3** viene presentata la funzione **showImageFromSD**, che interagisce con la memoria SD e permette appunto di visualizzare l'immagine contenuta. Ora possiamo dedicarci alla sezione riguardante la FAT e la SD la cui gestione non è banale ed occupa una parte rilevante di codice. La versione utilizzata nel progetto è una base, in modo da

avere ancora spazio nella Flash da utilizzare per il proprio codice, ma nonostante tutto gestisce FAT16 e FAT32 e anche se non vengono utilizzate nel Digital Frame, ha funzioni molto utili come quelle che permettono di creare un file, rinominarlo, salvarlo, o cancellarlo. Noi useremo solo quelle di lettura e ricerca all'interno di una directory. Per semplicità, utilizzeremo tutti i file caricati nella directory prin-

Listato 3

```
void showImageFromSD(char filename[]) // binary images
{
    fatFopen(filename);
    delay_ms(1000);

    //check file kiand
    unsigned char k = fatFgetc();
    if(k != 'K') {
        UART_PutString("Error file");
    }

    // check file type
    unsigned char imgI = fatFgetc();
    unsigned char imgM = fatFgetc();

    if( (imgI!= 'I') || (imgM!= 'M') ) {
        UART_PutString("Error type file");
        return;
    }
    unsigned char mode = fatFgetc();

    if (mode == 8) {
        LS020_Mode8BPP();
    }
    if (mode == 16) {
        LS020_Mode16BPP();
    }
    delay_ms(50);

    unsigned char posx = fatFgetc();
    unsigned char posy = fatFgetc();
    unsigned char width = fatFgetc();
    unsigned char height = fatFgetc();

    int size = width * height;

    LS020_PrepareImage(width, height, posx, posy, mode);

    unsigned int h = 0;
    unsigned int l = 0;

    while(size)
    {
        if(mode == 16)
        {
            h = fatFgetc();
            l = fatFgetc();

            PIN_CS_LOW();

            int color = (h<<8) | l;
            LS020_SendColor(color);

            PIN_CS_HIGH();

        }

        if(mode == 8)
        {
            h = fatFgetc();

            PIN_CS_LOW();

            LS020_SendSingleByte(h);

            PIN_CS_HIGH();

        }
        size--;
    }

    LS020_CloseImage();
}
```

cipale della SD, ma nulla vieta di creare altre directory e spostarsi all'interno di esse. In particolare useremo la **FATFirstDirEntry()**, che restituisce il primo file "valido" e dopo chiameremo tante volte la **FATNextDirEntry()** fino ad analizzare tutta la directory. A livello di file di progetto, il firmware relativo alla lettura della SD-Card occupa due file: uno di basso livello col quale siamo in grado di leggere e scrivere un settore della card tramite il protocollo SPI (si utilizza il modulo *spi hardware* interno al micro) ed uno di alto livello che usa questo primo file e che ci permette di muoverci da una settore all'altro in base alle tabelle FAT. La dimensione di un settore di una SD-Card è 512 byte e si potrebbe anche pensare di utilizzare solo il primo file, che a livello di codice e spazio in memoria di programma occupa

Funzione	Descrizione
LS020_Init ()	Inizializza il controller display, con diverse sequenze di comandi
LS020_Pixel	Disegna un pixel di un determinato colore in una determinata posizione
LS020_Char()	Utilizza una look-up table interna al micro per disegnare su LCD un carattere ad una determinata posizione
LS020_String()	Stampa una stringa a schermo in posizione determinata e con un determinato colore
LS020_Image()	Carica i dati dell'immagine contenuti in un buffer e li visualizza a schermo a partire da una determinata posizione e con larghezza ed altezza specificate
LS020_PrepareImage()	Invia dimensioni al display e attende l'immagine
LS020_Off()	Spegne il display
LS020_FillBox()	Disegna un box sullo schermo di un determinato colore
LS020_Mode24BPP()	Imposta modalità a 24 bpp
LS020_Mode16BPP()	Imposta modalità a 16 bpp
LS020_Mode8BPP()	Imposta modalità a 8 bpp
LCD_Circle()	Disegna un cerchio
LS020_Line()	Disegna una linea
LS020_Reverse()	Attiva modalità reverse
LS020_Scroll()	Effettua lo scroll di una data area dello schermo

Tabella 5 - Funzioni per la creazione delle animazioni.

decisamente molto meno; però si tratterebbe di scrivere i nostri dati (nel nostro caso, immagini)

in formato **raw**, ovvero in modo binario scrivendo su settori consecutivi e leggerlo successivamente

Listato 4

```
int scanRootDirectory(void)
{
    struct diriterator di;
    struct dirent *de;

    de = fatFirstDirEntry(0, &di);

    while (de != NULL)
    {
        if (*de->deName == SLOT_EMPTY)
        {
            return 0;
        }

        if ((*de->deName != SLOT_DELETED) &&
            (de->deAttributes != ATTR_LONG_FILENAME))
        {
            /*unsigned*/ char filename[13];
            memset(filename, ' ', 13);
            filename[8] = '.';
            filename[12] = '\0';

            memcpy(filename, de->deName, 8);

            int fs=0;
            for(fs=0;fs<8;fs++)
            {
                if (filename[fs] == ' ')
                    break;
            }
            filename[fs] = '.';
            memcpy(&(filename[fs+1]), de->deExtension, 3);
            filename[fs+1+4] = '\0';

            int intervalMs = 0;

            if ( (de->deExtension[1] == 'm') || (de->deExtension[1] == 'M') )
            {
                // clear
                LS020_FillBox(0, 0, LCD_WIDTH-1, LCD_HEIGHT-1,
                // RGB(255,255,255));

                //LS020_String(0, 10,filename, RGB(0,0,0));
                LS020_String(0, 168,filename, RGB(0,0,0));

                showImageBmpFromSD(filename);

                if (_fApplyMask) {
                    applyMask("mask.bin");
                }

                intervalMs = _intervalImages*1000;
                delay_ms(intervalMs);
            }
            if ( (de->deExtension[1] == 'i') || (de->deExtension[1] == 'I') )
            {
                // clear
                LS020_FillBox(0, 0, LCD_WIDTH-1, LCD_HEIGHT-1,
                //RGB(255,255,255));

                //LS020_String(0, 10,filename, RGB(0,0,0));
                LS020_String(0, 168,filename, RGB(0,0,0));
                showImageFromSD(filename);

                if (_fApplyMask) {
                    applyMask("mask.bin");
                }

                //delay_ms(2000);
                intervalMs = _intervalImages*1000;
                delay_ms(intervalMs);
            }
        } // end if
        de = fatNextDirEntry(&di);
    } //end while
    return 1;
}
```

nella stessa modalità. Questo, da un lato ci permette di risparmiare codice, ma dall'altro ci costringe a scrivere la SD tramite programmi fatti su misura per scrivere in determinati settori, senza contare che in questo modo la memoria SD non sarà riconosciuta da un normale lettore per PC. Per maggior comodità diventa pertanto necessario il secondo file (**FAT.c**) che occupa molto di più, ma ci permetterà di caricare i nostri file immagine anche nel noto formato standard bitmap (estensione .bmp) e renderli riconoscibili e visualizzarli nella cornice digitale. Nel **Listato 4** è possibile vedere il codice che esegue la scansione di tutti i file presenti nella root e che disegna l'immagine quando trova file con estensione .bmp o .bin.

UTILIZZO DEL SOFTWARE

Il Digital Frame potrebbe funzionare semplicemente caricando immagini bitmap in formato 132x176 o inferiore, col formato 24 bit per pixel. Esistono molti tool per convertire immagini che possono risultare utili: uno su tutti, per la semplicità di utilizzo e nello stesso per la completezza, è senz'altro *Paint.Net*, valido prodotto di casa Microsoft, scaricabile gratuitamente dal sito www.getpaint.net/download.html, da cui è possibile aggiungere anche numerosi plugin. Ad ogni modo, anche senza ulteriori plugin è possibile aprire un'immagine e selezionarne una porzione o rimpicciolirla in modo che entri nel nostro display, quindi salvarla come bitmap; il nostro digital frame la leggerà senza problemi. Uno screenshot di esempio è illustrato nella **Fig. 6**.

Per entrare un po' più nel codice e magari sviluppare le nostre applicazioni con questo circuito, rendendole più carine, migliorando i nostri menu aggiungendo

effetti grafici o piccole animazioni o icone originali, il software Paint.NET non basta, ma diventa necessario un programma realizzato su misura per il display LS020: si tratta di **LS020_Image-Tool**. Questo software, scritto in linguaggio C#, permette di convertire un'immagine (in formato bitmap, png o jpeg) in un file di testo o meglio di codice (un file.h) in cui avremo un array con i valori del colore di ogni pixel dell'immagine. Basterà allora includere questo file nel progetto di AVR Studio o semplicemente copiare i valori e inserirli nel nostro codice sorgente, per visualizzare su display la stessa immagine nella posizione a piacere, tramite la funzione **LS020_Image()** già vista nella **Tabella 5**. Prima della conversione si potrà scegliere la profondità di colore, ovvero se utilizzare il formato a 8, 16 o 24 bit per pixel, con un occhio di riguardo allo spazio occupato nella Flash (per non impegnare tutta la memoria di programma a discapito del codice); questo perché l'array dell'immagine verrà scritto proprio in tale area (grazie alla direttiva **PROGMEM** del compilatore **gcc-avr**). Vale perciò la pena di dare un occhio alle dimensioni che può occupare un'immagine cablata nel codice. Ad esempio un'immagine di dimensioni 20x40 a 24 bpp richiede 3 byte per pixel, per un totale di 2.400 byte, che potrebbero essere già troppi se siamo a corto di spazio. La stessa immagine, a 16 bpp ne occuperebbe 1.600, mentre a 8 bpp l'occupazione scende ad 800 byte. Ovviamente non avremo la stessa nitidezza, ma in caso non sia così determinante potremo risparmiare un terzo dello spazio e, come solito in questi casi, occorre giocare sui compromessi. Tramite lo stesso software, come visibile nella **Fig. 6**, è possibile

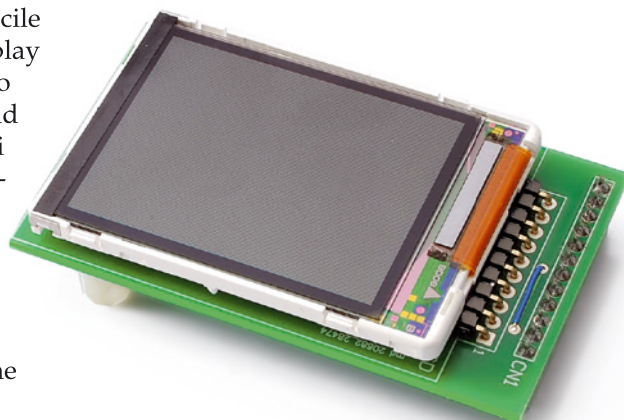


OLED si fa largo fra i display

Al Nokia World 2010 sono stati presentati i nuovi smartphone Symbian 3 con display AMOLED; in entrambi viene usato un OLED da 4 pollici di diagonale, però di tipo touch-screen, con risoluzione di ben 640x360 pixel. Sony ha presentato al SID 2010 un nuovo OLED da 11,7", destinato al mercato dei notebook, che a quanto pare ha superato uno dei limiti della tecnologia: la durata. Infatti Sony assicura una vita del display di 10 anni. Samsung, dopo aver sviluppato display AMOLED e SuperAMOLED, ha realizzato l'OLED Max, pensato per display di piccole dimensioni; è un'evoluzione del SuperAMOLED, che oltre a colori più vicini alla realtà, porta con sé maggiori dettagli e un contrasto "intelligente", con una risoluzione massima di 800x1024. Insomma, sembra che l'OLED stia decollando.

convertire l'immagine in un formato binario praticamente identico e sempre leggibile dal PhotoFrame, ma questa volta conterrà un header iniziale di 8 byte per informare il display sul tipo di file, la modalità di colore, la posizione dell'immagine e le sue dimensioni. Perché dotare il PhotoFrame di questa funzionalità quando legge già le immagini bitmap e non abbiamo problemi di spazio con le capienti SD-Card odierne? Intanto può sempre venire bene per i test, perché in questo caso l'immagine non è rovesciata come nei file bitmap e poi è più facile sfruttare la capacità del display di scrivere parzialmente solo su aree definite per creare, ad esempio, piccole animazioni come una successione di immagini in posizioni diverse e magari contenute in un unico file binario. Inoltre (**Fig. 6**) il terzo pannello prevede una piccola interfaccia per la connessione

diretta tra la porta seriale (RS232) del PC e il nostro digital frame, in modo da trasferire un'immagine sul display senza dover passare attraverso la SD. Ciò può venire utile sia in fase di test, sia per verificare la resa finale di una foto sul display, prima di effettuare il trasferimento. Avendo ormai tutti i tasselli al loro posto e modificando poche righe di codice, diventa pensabile aggiungere la possibilità di salvare l'immagine sulla SD (inserita direttamente nel Digital frame) durante il trasferimento seriale e ritrovarla nel nostro album



fotografico senza aver utilizzato un lettore di SD Card!

USARE IL DIGITAL FRAME

Premendo il pulsante P2 in qualsiasi istante, sarà possibile accedere ad un menu con diverse voci che potremo scorrere utilizzando i tasti P1 (scorrimento verso l'alto) e P3 (scorrimento verso il basso) e infine confermare la nostra scelta ripremendo P2. A questo punto la successione delle foto proseguirà con le modifiche apportate. Di seguito vediamo i comandi che possiamo impartire. **Reverse:** ridisegna l'immagine corrente e quelle successive con i colori invertiti.

Interval: permette di impostare (tramite P1 e P3) l'intervallo di tempo (espresso in secondi) che intercorrerà tra un'immagine e la successiva.

Stop: permette di bloccare il display sull'immagine corrente.

Continue: forza la ripresa dello scorrimento delle immagini.

Apply Mask: applica una maschera all'immagine corrente e alle successive, in modo da cre-

are effetti come cornici virtuali, puzzle, vetro rotto. Questo file di maschera deve essere generato in formato binario ed occorre sia presente nella root della SD-Card con il nome **mask1.bin**. I pixel bianchi nell'immagine "maschera" vengono considerati come "trasparenti".

Normal: torna alla normalità nel caso siano stati applicati gli effetti sopraelencati.

Transfer: mette il micro in stato di attesa, pronto a ricevere comandi seriali dal PC, dove sarà attivo il software **LS020_Image-Tool** analizzato in precedenza.

Leds: visualizza una stringa scorrevole a tutto schermo realizzata su una matrice di 8x8 LED di 20x20 pixel ciascuno.

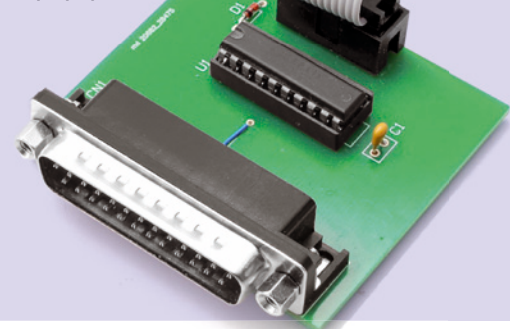
Clock: mostra un classico orologio a lancette realizzato con le funzioni viste prima creare cerchi e linee.

Turn-off: spegne il display con effetto dissolvenza.

TOOL DI SVILUPPO

Abbiamo analizzato alcuni frammenti di codice utilizzato nel

L'ATmega32, così come altri micro AVR, può essere programmato in-circuit tramite una vasta serie di programmatori, facilmente reperibili e semplici da autocostruire. Utilizzando la porta parallela è possibile costruirsi con pochi componenti il noto STK200, che è stato usato per questo progetto e che utilizza il buffer 74HCT244 per dare maggior robustezza ai segnali. Esistono programmatori che non utilizzano buffer, dove i pin della porta parallela vengono collegati direttamente al micro ed altri che al posto di utilizzare la parallela, ormai sempre più in disuso, sono controllati via USB. Nello schema di un STK200 si nota come l'alimentazione dell'U1 venga presa direttamente dalla scheda target, che deve pertanto essere alimentata durante la programmazione. Il cortocircuito tra i pin 12-2 ed 11-3 della parallela è solo una sorta di "chiave" per far capire al software il tipo di programmatore collegato. Il software usato per la programmazione



Digital Frame, abbiamo realizzato fisicamente la nostra scheda e l'abbiamo vista funzionare con alcune foto e immagini di test; ora siamo pronti ad aggiungere funzionalità o a migliorare quelle già esistenti, dando sfogo alla nostra fantasia o alle nostre esigenze. Atmel fornisce gratuitamente un IDE (Integrated development environment) per sviluppare su tutta la famiglia di microcontrollori a 8 bit, tra cui il nostro ATmega32: è il famoso **AVR Studio 4**. La versione usata in questo progetto è la 4.13 e nasce principalmente come Assembler, ma è possibile aggiungere un plugin per permettere anche la compilazione di sorgenti in linguaggio C. Si tratta di **WinAvr**, un progetto

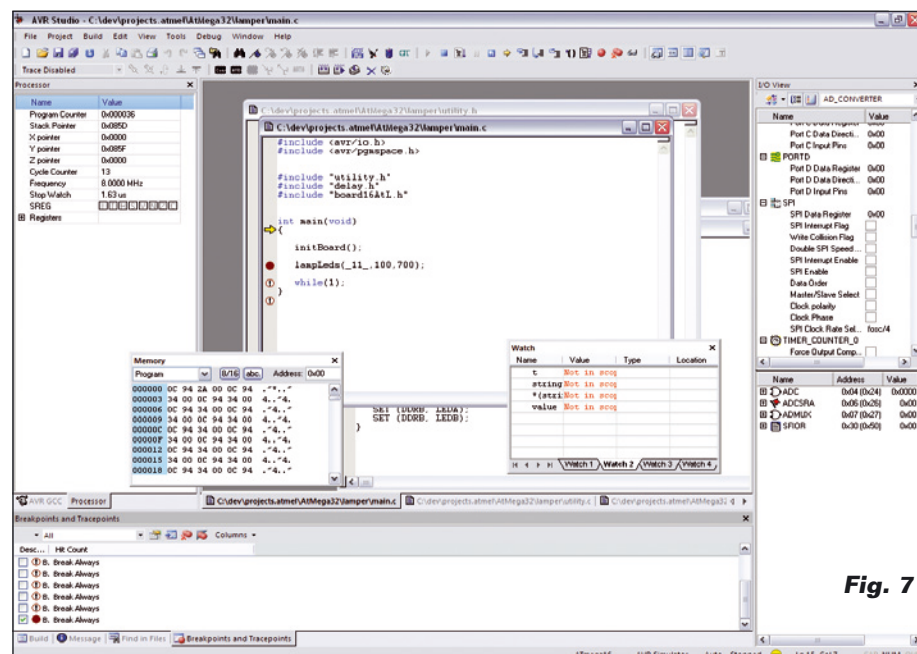


Fig. 7

è AVRDUDE, che funziona su PC da linea di comando; è gratuito, molto potente e gestisce praticamente tutti i tipi di programmatori. Rimandiamo l'analisi dettagliata di tutti i comandi alla lettura del manuale in formato pdf, elencando però i più usati:

-c <interface> è usato per specificare il tipo di programmatore hardware usato, nel nostro caso abbiamo usato STK200;

-p <microcontroller> è usato per specificare il tipo di micro da programmare (m32 nel caso di atmega32);

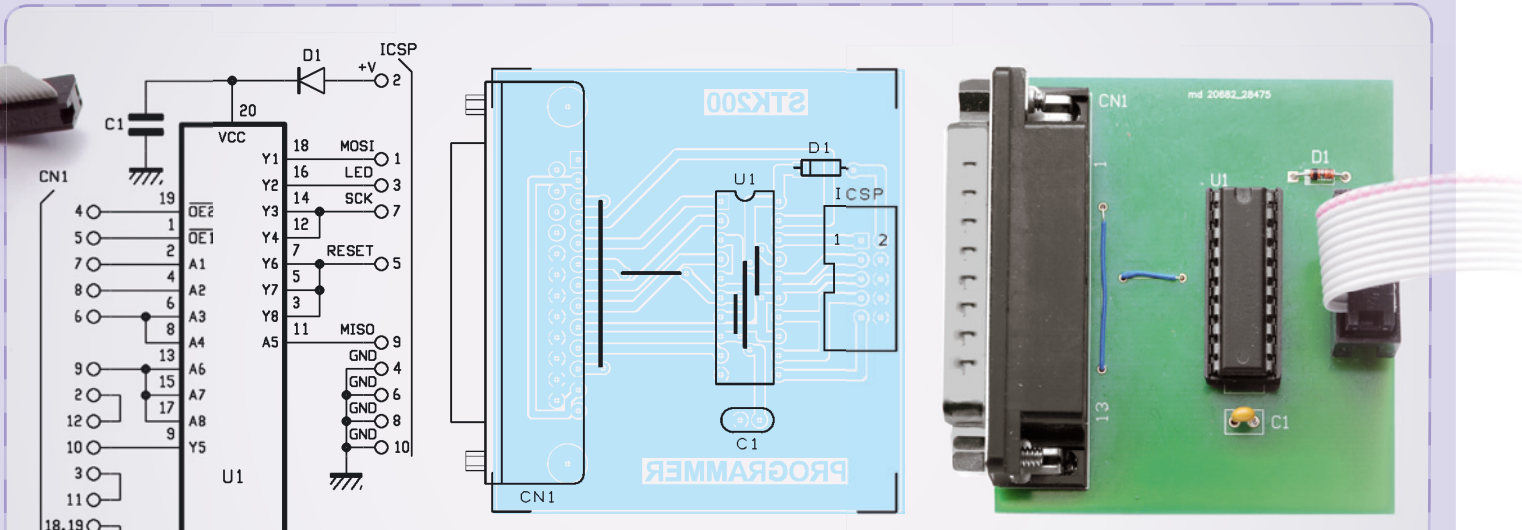
flash:w: <file hex> specifica il file hex da caricare nel micro;

-e cancella la Flash prima della programmazione.

Un esempio di comando sarà quindi del tipo:

```
avrdude.exe -p m32 -c stk200 -e flash:w:avrphotoframe.hex
```

Inoltre AVRDUDE consente anche una modalità "console" attivabile tramite comando -t (es. Avrdude.exe -p m32 -c stk200 -t) tramite cui si può risalire a utili informazioni sul micro come ad esempio la lettura e scrittura dei fuses. La programmazione dei micro atmel avviene tramite il bus SPI impiegando gli stessi pin usati per il display e la SD-Card. In realtà i micro AVR supportano anche una programmazione parallela, che però è generalmente poco usata per una questione di risparmio di linee di I/O.



Elenco componenti:

D1: 1N4148	Varie:
C1: 100nF ceramico	Connettore 5x2 poli da CS
U1: 74HCT244N	Porta parallela maschio da CS

creato da terze parti e scaricabile gratuitamente dal sito winaavr.sourceforge.net, che si integra perfettamente con AVR Studio. Basterà installarlo dopo l'installazione di AVR Studio e, senza ulteriori settaggi, alla successiva apertura dell'IDE, quando creeremo un nuovo progetto, ci verrà chiesto se intendiamo usare l'assembler Atmel o il compilatore C. Quest'ultimo, specifico per i microcontrollori AVR, si basa sul più classico e conosciuto GCC ed è infatti chiamato *avr-gcc*. AVR Studio non ha nulla da invidiare ad altri IDE presenti in circolazione: contiene un simulatore che gestisce ogni tipo di microcontroller AVR ed ha una finestra di I/O in cui sono visibili tutti i

registri dei moduli hardware interni al micro (timer, porte di I/O, convertitori analogici, spi, I²C, seriale) e il cui valore viene continuamente aggiornato, in modo da farci verificare il corretto funzionamento del nostro codice. Il simulatore permette di inserire breakpoint nel nostro codice e consente di fermarsi in una linea, modificare manualmente questi registri e continuare con l'esecuzione del programma verificando i risultati del cambiamento. Ciò è utile soprattutto in fase di test, per verificare le varie funzionalità offerte dai moduli hardware del micro senza essere costretti a riprogrammare il chip numerose volte, risparmiando anche in tempi di attesa. Analogamente,

sono disponibili da simulatore anche i vari registri interni dell'Atmel, lo Stack Pointer, il Program Counter ed altre utili informazioni come il Cycle Counter e lo Stopwatch; il primo di questi due indica i cicli macchina eseguiti e il secondo ci informa sul tempo trascorso (espresso in microsecondi) dal suo ultimo azzeramento. Occorre preventivamente inserire il valore del quarzo usato. In questo modo è possibile ad esempio, verificare la bontà delle proprie routine di ritardo, che servono praticamente in tutte le applicazioni. Nel nostro caso, le fasi di inizializzazione del display devono rispettare strettamente alcune tempistiche e non possono essere eseguite

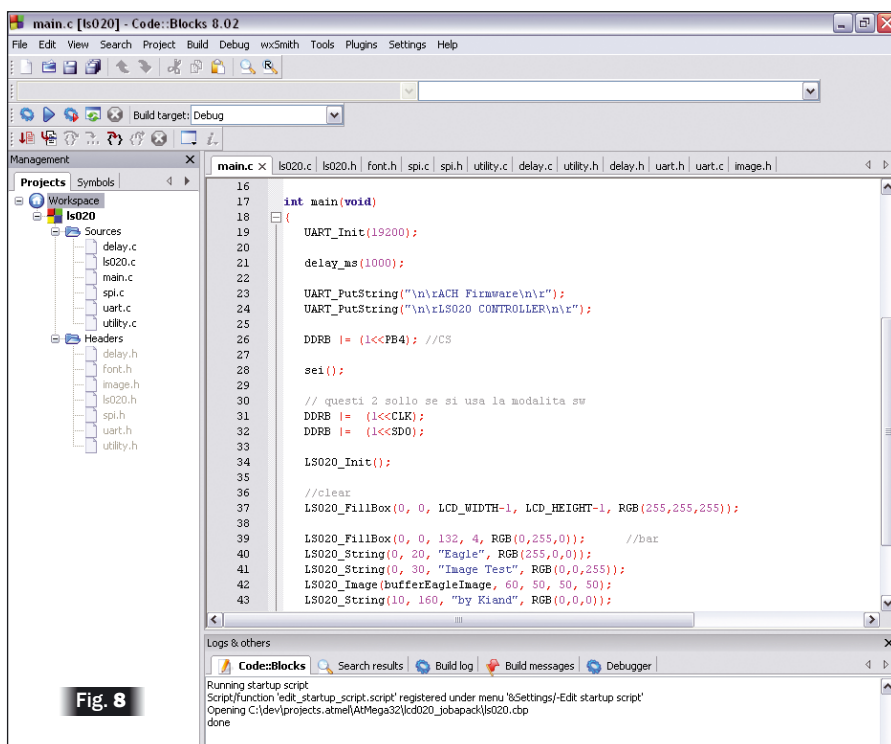


Fig. 8

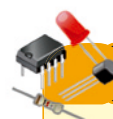
dopo troppo tempo (altrimenti non visualizzeremo alcunché) e a volte un errore nella gestione dei delay potrebbe pregiudicare il corretto funzionamento della nostra applicazione. Rimanendo in ambito simulazione, AVR Studio gestisce anche il plugin che consente l'integrazione con Proteus (tool di simulazione visuale di dispositivi elettronici) a fianco delle finestre con il codice. Per la programmazione vera e propria del micro, abbiamo utilizzato il software AVRDUDE, che gestisce diverse interfacce standard (utilizzanti le porte seriale, parallela e USB) ed è richiamabile direttamente da AVR Studio. Se abbiamo un debugger (come ad esempio JTAGICE MKII o AVR Dragon), oltre alla semplice programmazione del nostro dispositivo, possiamo anche inserire breakpoint hardware direttamente nel micro (e non più solo su simulatore) e spostarci di una linea per volta nel codice di pari passo all'esecuzione sul micro. Esistono comunque anche altri tool di sviluppo ed IDE che meritano attenzione; uno è *Code-*

blocks, giunto alla versione 10.05 e scaricabile gratuitamente dal sito www.codeblocks.org. Presenta un editor di testo molto curato con sintassi colorata, richiamo al corpo delle funzioni, apertura automatica dei file di header, completamento automatico del codice e tanto altro. È forse un tantino ostico da configurare, perché al contrario di AVR Studio è customizzabile anche per compilatori ed assembler diversi, ma vale sicuramente la pena e in fatto di usabilità e comodità di scrittura ha qualcosa in più rispetto ad AVR Studio. Potete vedere una schermata di CodeBlocks nella Fig. 8.

CONCLUSIONI

Il digital frame può essere utilizzato come una piccola e versatile piattaforma di sviluppo per testare altri tipi di display e altri modelli di microcontrollori (vista la compatibilità a livello di pin tra molti micro della famiglia Atmel e la compatibilità del connettore CN1 con il display nokia LCD6100); inoltre, la disponibilità di quasi tutti i pin del micro

sui diversi connettori facilita l'espandibilità tramite plugin aggiuntivi autocostruiti o di terze parti. Sarà pertanto possibile aggiungere un piccolo buzzer sui pin di I/O ancora disponibili, piuttosto che un accelerometro, LED supplementari o ancora un piccolo schermo touch, utile ad esempio per disegnare "a mano" sul nostro display. Inoltre, l'integrazione del connettore SD sulla basetta principale, che a questo punto diventerebbe una vera e propria demoboard, favorisce la possibilità di scrivere un bootloader sul micro che legga, all'avvio, il nostro codice binario, in formato hex, da una scheda SD. Addirittura, si potrebbe caricare sulla card più di un firmware e offrire la possibilità all'utente di scegliere quale utilizzare di volta in volta. Ancora, si potrebbe integrare nel bootloader il codice che riconosca il display collegato al CN1 (LS020 o Nokia 6100) e carichi nella Flash del microcontrollore il relativo codice dalla SD-Card. Insomma il digital frame è solo un inizio; a voi scoprire come sviluppare il progetto. ■



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 firmware utilizzato per programmare il microcontrollore. Quest'ultimo (ATmega32-16PU) può essere acquistato presso Futura Elettronica al prezzo di 18,00 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>



INTERVISTA A MASSIMO BANZI IDEATORE DI *Arduino*

..... di ARSENIO SPADONI

A margine di NIDays 2011 abbiamo incontrato Massimo Banzi che ci ha spiegato il senso profondo del progetto Arduino nonché i suoi programmi per il futuro, specie nel campo della Robotica Educativa.

Quando mi hanno detto che una delle keynote di NIDays 2011- l'annuale Forum Tecnologico sulla Progettazione Grafica di Sistemi di National Instruments che si è svolto di recente a Milano - sarebbe stata tenuta da Massimo Banzi, in un primo momento ho pensato: *"che ci fa uno dei guru dell'open source in una manifestazione come questa?"*

La risposta me l'ha fornita la brochure di presentazione dell'evento che quest'anno (ma non solo) aveva come tema centrale la necessità di inventare e innovare. E chi se non un innovatore come Massimo Banzi poteva fornire una personalissima risposta a questa esigenza? Una risposta che anche National Instruments con i propri prodotti

Numeri da record per la 18^a edizione di NIDays

Si è svolta mercoledì 23 febbraio al Centro Congressi Milanofiori di Assago la 18^a edizione italiana di NIDays. Quasi 600 tra ingegneri, sviluppatori, professori, studenti ed espositori (record di presenze di sempre) sono stati i veri protagonisti di una giornata fitta di appuntamenti, celebrata nel segno dell'Innovazione.

Durante la plenaria del mattino, Owen Golden, Vice President, Global Energy Segment di National Instruments Corporation, e lo staff tecnico della filiale italiana ci hanno accompagnato in un viaggio entusiasmante tra le sfide tecnologiche del futuro, come la gestione del tempo e le soluzioni con le quali NI intende rispondere. Ospite d'onore della fase conclusiva della Keynote mattutina è stato Massimo Banzi, co-fondatore di Arduino, la scheda italiana per la prototipazione elettronica più famosa al mondo. Durante la sua appassionata presentazione, Banzi ha introdotto l'innovativo concetto di hardware open source mostrando decine di applicazioni realizzate con la piattaforma Arduino, conquistando letteralmente il pubblico in sala. La keynote si è conclusa con l'annuncio dei tre vincitori del concorso "Premio Nicola Chiari per la migliore applicazione di misura e automazione 2011". Quest'anno si sono classificati ai più alti gradini del podio:

1°: DEPAArray™: Progettazione e sviluppo di un sistema automatico per l'isolamento di cellule rare, Silicon Biosystem, Sky Technology:

uno strumento capace di individuare e isolare cellule tumorali circolanti (CTC) con lo scopo di studiare cure personalizzate in campo oncologico o cellule fetali nel sangue materno, per consentire una diagnosi prenatale non invasiva.

2°: KITEnergy: energia eolica d'alta quota a costi inferiori del petrolio, Politecnico di Torino, University of California- Santa Barbara: permette la conversione dell'energia dei venti di alta quota (tra 200 e 1000 metri dal suolo) in elettricità.

3°: Monitoraggio wireless strutturale a lettura remota del duomo di L'Aquila, Giorgio Sforza, Essebi: un sistema integrato hardware e software di diagnostica remota durante le fasi di risanamento e ricostruzione parziale del Duomo di L'Aquila.

Il pomeriggio di NIDays ha visto i propri partecipanti dividersi tra le varie sale del centro congressi, per seguire liberamente la sessione di conferenze più vicina ai propri interessi.

Energy Technology Summit, fiore all'occhiello della seconda parte della giornata, ha ospitato i principali esperti di settore in una serie di sessioni tecniche dedicate all'efficienza energetica in ambito industriale, al monitoraggio della power quality nelle smart grid e alle nuove frontiere delle energie rinnovabili in un'ottica di ricerca avanzata e innovazione tecnologica.

Durante tutta la giornata gli ospiti di NIDays hanno potuto visitare gli stand di oltre 30 aziende espositrici, che hanno mostrato le ultime novità di prodotto e le applicazioni tecniche nei settori del controllo, del design, della misura e dell'acquisizione dati.



e le applicazioni presentate dai numerosi partner tecnologici ha messo in mostra nel corso della manifestazione.

Durante il suo intervento, Banzi ha introdotto l'innovativo concetto di hardware open source mostrando decine di applicazioni realizzate con la piattaforma Arduino e conquistando letteralmente il pubblico in sala.

Prima del suo intervento, ho avuto l'occasione di intervistare Massimo Banzi al quale ho rivolto una serie di domande facendomi portavoce anche di molti nostri lettori.

In questi giorni, Phillip Torrone ha scritto su Make un editoriale dal titolo "Why the Arduino Won and Why It's Here to Stay" innescando un animato dibattito su vari siti web sui motivi del successo di Arduino. A questo punto non resta che sentire il suo parere: Arduino ha realmente "vinto"? E se sì, quali sono le ragioni principali di questo successo?

"Credo che Arduino non abbia vinto nulla, nel senso che non credo ci sia nulla da vincere; penso invece che ci sia un percorso naturale che tutta la tecnologia fa, di continuo avvicinamento alle esigenze degli uomini. Posso immaginare che 40, 50 o 60 anni fa l'idea che una persona normale potesse programmare un computer era una cosa inconcepibile, mentre adesso anche il mio commercialista usa le macro di Excel mentre quarant'anni fa si sarebbe dovuto rivolgere all'IBM con dei costi imbarazzanti. Nel mondo della tecnologia c'è un percorso naturale verso questa direzione mentre spesso nel mondo della tecnologia, diciamo così, "classica", dell'ingegnere "classico", c'è una certa resistenza a proporre degli strumenti che



siano facili per gli esseri umani normali, forse per mantenere un sorta di aurea, per fare pesare la propria sapienza, quando in realtà i concetti di base sono abbastanza semplici. D'altra parte è evidente che un mese speso a realizzare applicazioni con Arduino non è in grado di trasformare nessuno in un ingegnere elettronico, così come non è vero che ci vogliono 5 anni di ingegneria elettronica per essere in grado di realizzare dei progetti, delle cose anche interessanti. Io ho frequentato l'ITIS e già allora fabbricavamo di tutto, e sicuramente nessuno di noi era un genio...

Secondo me quello che in qualche maniera "ha vinto" è il concetto di semplificazione della tecnologia: se prendiamo un pezzo di tecnologia ritenuto, a torto o a ragione, complesso, e lo semplifichiamo lavorando non sulla tecnologia ma sull'esperienza d'uso, ecco che tutto si semplifica e si aprono nuove possibilità che prima non esistevano. Arduino ha consentito a tantissime persone di avvicinarsi con semplicità alla programmazione dei microcontrollori, persone che mai avrebbero pensato di fare nulla del genere."

Quali ritiene siano i principali vantaggi in termini pratici di questa apertura?

"In un modello tradizionale per mettere in pratica un'idea o per risolvere un problema, ad esempio se fossi un allevatore di polli e mi venisse in mente un'idea per aumentare l'efficienza o il grado di sicurezza del mio allevamento,

dovrei rivolgermi ad un'ingegnere per realizzare quello che ho in mente e per fabbricare quello che mi serve. Con Arduino, invece,

"Arduino ha consentito a tantissime persone di avvicinarsi con semplicità alla programmazione dei microcontrollori, persone che mai avrebbero pensato di fare nulla del genere."

anche chi si intende di polli ma non di elettronica e programmazione è in grado di realizzare da solo e in pochissimo tempo l'accrocchio che gli serve, ovvero di risolvere il suo problema pratico. Si tratta in qualche modo di spostare la tecnologia verso quello che potremmo chiamare un *domain expert*, perché comunque, alla fine, chi alleva polli è sicuramente molto più esperto di polli di quanto lo possa essere un ingegnere o un estraneo che sicuramente non ne sa nulla. Tutto ciò permette all'innovazione di svilupparsi dal basso, dando ai *domain expert* nei vari campi gli strumenti per sviluppare da soli le proprie applicazioni, per mettere in pratica le proprie idee. Nel mondo del design dal quale provengo, succedono cose di questo tipo: quando un designer va a discutere con gli ingegneri che debbono realizzare gli aspetti più tecnologici dei prodotti

chiedendo certe funzioni, spesso si sente dire "ma tu sei scemo, sparisce..." I casi, invece, dove il design comanda sulla tecnologia sono veramente pochi. Uno di questi è Apple: quando chi comanda decide che l'antenna, ad esempio, va fatta in un certo modo, gli ingegneri in un primo momento si incavolano ma poi,

alla fine, realizzano l'antenna come la volevano i boss, e guarda caso l'antenna funziona perfettamente anche se nel realizzarla non hanno fatto ricorso a delle soluzioni classiche."

Ci troviamo al raduno annuale degli utilizzatori dei sistemi National Instruments e questa domanda è d'obbligo: è possibile - secondo lei - paragonare il supporto che la programmazione grafica NI offre da anni a schiere di tecnici e ingegneri con la semplicità di utilizzo che la piattaforma Arduino offre ad un pubblico molto più vasto, specialmente a designer, hobbysti, artisti e studenti che vogliono realizzare rapidamente e con ridotte competenze di elettronica e informatica i propri prototipi?

"In entrambi i casi si tratta di strumenti che cercano di rendere più accessibile la tecnologia. Io, ad esempio, lo vedo con i miei studenti che sono designer e artisti: loro la programmazione classica non la vogliono fare. Per questo motivo nel mondo dell'arte esistono una serie di piattaforme che sono concettualmente simili a LabVIEW, che hanno una

grande fortuna e che consentono di realizzare in maniera grafica i propri progetti, specialmente quando si tratta di persone alle prime armi. Un

"Credo che se un giorno decidessi di arrivare fino alla Gelmini per spiegare questi concetti, sprecherei un sacco di energie."

sistema di programmazione grafica porta sicuramente quel tipo di tecnologia ad un pubblico che magari non ha voglia di programmare in C.

Per quanto ci riguarda, abbiamo cercato di prendere il C e semplificarlo, togliendo tutte quelle parti che fanno spavento, togliendo tutti i pulsanti dal software,

cercando la semplicità assoluta. Anche nel mondo dell'educazione, dove sono i ragazzini ad utilizzare questi strumenti, i linguaggi di programmazione grafica hanno una grande importanza; lo vediamo, ad esempio, con quello più famoso, il linguaggio *Scratch* del MIT, col quale gli insegnanti sono riusciti a far fare ai più piccoli delle cose che magari col *Logo*, il linguaggio utilizzato in precedenza, non riuscivano a fare. Lo stesso avviene con *LabVIEW* che è "dentro" al *LEGO* e col quale molti ragazzini programmano i loro robot senza magari rendersene conto..."

Da tempo si parla dell'introduzione della Robotica Educativa nelle scuole italiane, anche in quelle elementari e medie. Proprio qui al NIDays è in corso un seminario sull'argomento in quanto LabVIEW viene utilizzato per la programmazione di uno dei primi robot programmabili disponibili, il LEGO NXT. Ora, da più parti, si sta proponendo l'introduzione proprio di Arduino come supporto didattico innovativo in numerosi corsi di studio, dalle medie alle superiori, e

non solo nei corsi di elettronica e informatica. Magari insieme ai corsi di robotica. Pensa che una cosa del genere possa mai accadere nel nostro Paese?

"Questa cosa in Italia si fa, nonostante il sistema è all'oscuro dal sistema. Da parte nostra, da quasi un anno, abbiamo cominciato un progetto del genere, partendo dal Piemonte, coinvolgendo i professori delle scuole superiori ai quali abbiamo tenuto dei corsi di tre giorni sulla piattaforma Arduino e sul linguaggio di programmazione Processing che è il "gemello" di Arduino. Poi abbiamo fatto vedere loro come realizzare con questi strumenti e con dei pezzi di recupero dei piccoli robot e cose del genere; ormai abbiamo formato una settantina di insegnanti che operano prevalentemente negli Istituti Tecnici, cercando anche di fargli capire che gli strumenti ci sono, che è possibile dare un kit a tutti i ragazzini. I risultati non si sono fatti attendere, ad esempio in un corso serale di un ITIS di Ivrea uno studente ha costruito un "Controllo per pannello solare via web", un altro ha realizzato una rete di sensori wireless: tutte cose che sono state realizzate nell'ambito di un corso di elettronica abbastanza breve. Credo che succederà proprio in questo modo, con professori intelligenti che porteranno queste idee nella scuola..."

Non in maniera istituzionalizzata, dunque? Magari a loro spese...

"Nel nostro caso i professori

si sono portati da casa la loro scheda Arduino che ha un costo decisamente molto basso.

Da questo punto di vista noi siamo abbastanza specializzati in tutte queste cose che avvengono

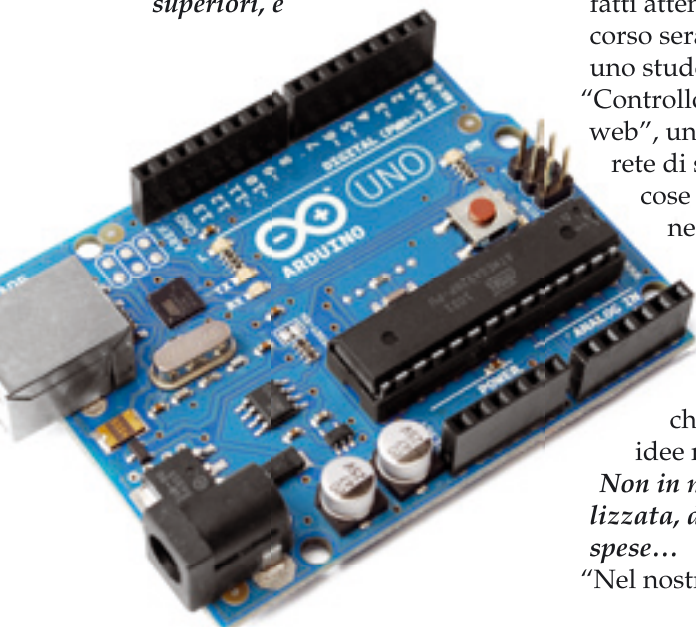
dal basso, *bottom-up*, *grass-roots*, come si dice...

Credo che se un giorno decidessi di arrivare fino alla Gelmini per spiegare questi concetti sprecherei un sacco di energie; preferisco convincere un certo

numero di persone... Un altro modello che funziona in Italia è, purtroppo, quello di imitare i successi stranieri: sono convinto che se uno dei progetti che stiamo portando avanti in Spagna o in Messico o negli Stati Uniti o in Svezia produrrà qualcosa di molto bello, allora gli italiani diranno "dobbiamo farlo anche noi", perché lo hanno fatto gli americani o gli spagnoli. Però, impegnarsi a realizzare per primi qualcosa del genere nel nostro paese, non sembra che per gli italiani abbia molto senso."

... e tutto ciò nonostante Arduino sia un prodotto italiano, con il logo della nostra penisola sulla basetta e la scritta - ben evidente - Made in Italy. A questo proposito: quanto questi elementi hanno contribuito al successo all'estero di questa piattaforma?

"Abbastanza, è una classica questione di *branding*. Siccome Arduino è un open source sia nella parte hardware che in quella software, ha lo stesso problema che hanno i marchi di moda... ad esempio, il disegno del maglione che indosso non è proteggibile da alcun copyright, se però lo riempio di loghi diventa un *trademark* che invece è proteggibile. Ecco, noi per certi versi ci comportiamo come un'azienda di *fashion*. Una



Tutto **Arduino** in 275 pagine

cosa sicuramente importante è il *brand* ma noi cerchiamo di dare anche un prodotto che è progettato meglio, che è un po' più carino e che spinge un po' anche su quest'idea dell'Italia che piace molto. Da questo punto di vista abbiamo anche fatto l'esperimento di fabbricare delle schede in Corea, per il mercato asiatico, ma i coreani ci dicevano che non le volevano, loro volevano le schede fatte in Italia: queste datele a qualcun altro. E, come nel campo della moda, abbiamo anche noi il segmento di quanti comprendono il messaggio, che vogliono aiutare il progetto e che comprano il prodotto originale perché è Made in Italy; ma anche di quanti si comprano sempre la roba meno costosa e acquistano il clone cinese. E questo comportamento riguarda anche alcuni italiani che abitano ad un chilometro dalla fabbrica di Arduino e che non capiscono che continuando a comprare il clone cinese su eBay risparmiano due euro, una cosa che non ha molto senso, anche perché molti cloni lasciano alquanto a desiderare.

Anche uno dei professori che si è presentato ai nostri corsi aveva un clone di Arduino: era così brutto, ma così brutto... In conclusione, il fatto che Arduino sia Made in Italy ha sicuramente contribuito al suo successo."

Quale contributo potrebbe eventualmente dare la piattaforma Arduino al lavoro di tecnici e ingegneri che quotidianamente

"Abbiamo già iniziato l'anno scorso a focalizzarci sul mercato dell'educazione, con l'intenzione di fare un sacco di prodotti destinati alle scuole superiori, medie ed elementari e con una serie di progetti in giro per il mondo."

di fare capire come funzionano le tecniche di acquisizione, può servire allo scopo. Noi lo vediamo nei progetti che facciamo nei paesi in via di sviluppo dove costruisco-

Volume adatto sia ai principianti che agli esperti, il nuovo libro di Maik Schmidt pubblicato da "The Pragmatic Programmers" consente ai primi - anche se digiuni di qualsiasi nozione di elettronica e programmazione - di iniziare ad utilizzare subito questa piattaforma ed ai secondi di approfondire alcuni argomenti quali la comunicazione seriale sfruttando vari linguaggi. Nel mezzo, otto progetti completi in differenti campi, analizzati in maniera dettagliata in modo da risultare facilmente comprensibili a chiunque. Il volume è aggiornato con la nuova versione Arduino UNO. Nel nostro paese il libro (del costo di 35 dollari) può essere acquistato anche presso i vari store on-line, ad esempio IBS (www.ibs.it).



utilizzano i prodotti National Instruments?

"Sui vari forum di Arduino è possibile vedere che ci sono molte persone che utilizzano LabVIEW

con Arduino, ad esempio c'è un ragazzo, Alessandro Ricco, che ha collegato Facebook con Arduino usando LabVIEW. Sicuramente Arduino non si pone il problema di campionare a 20 gigabit, anche perché non ce la farebbe mai: evidentemente non è questo il suo mestiere. E soprattutto le misure che possiamo fare noi non sono certificate. C'è però da dire che come prodotto entry-level o come sistema in grado

no con Arduino degli strumenti equivalenti, ovviamente non così precisi ma che sono in grado di risolvere molti problemi, perché, ad esempio, lavori in una zona povera del Brasile, o la tua piccola società non ha i soldi per acquistare strumentazione più precisa o ti devi realizzare da solo una sonda... Da questo punto di vista è un modo per coinvolgere le persone, per avvicinarle alla tecnologia: è chiaro che poi, quando si iniziano a fare seriamente le cose, non si può più usare Arduino ma, magari, si inizia a pensare agli strumenti della National Instruments... Però, nel frattempo, molte persone, gli studenti, eccetera, hanno risolto i loro problemi di misura."

Come vede l'evoluzione del sistema Arduino, su quali campi punterete maggiormente in futuro?

"Abbiamo già iniziato l'anno scorso a focalizzarci sul mercato dell'educazione, con l'intenzione di fare un sacco di prodotti destinati alle scuole superiori, medie ed elemen-

La filosofia dell'open source si estende anche all'hardware



Arduino è stato uno dei primi sistemi ad adottare la filosofia dell'open source oltre che nel software anche nell'hardware. Sei anni fa, tuttavia, quando Arduino è stato sviluppato, non esisteva ancora una precisa definizione di hardware open source né tantomeno di "Opere Culturali Libere" ("Free Cultural Works"); per dare una definizione precisa dell'open source hardware (OSHW) e stabilire quali criteri debba rispettare per definirsi tale, un gruppo di esponenti di questa scuola di pensiero, compreso Massimo Banzi, alcuni mesi fa ha iniziato a lavorare attorno a questo tema e da pochissimo è giunto alla stesura della prima versione (1.0) della Open Source Hardware Definition, aggiornata di recente con la release 1.1 (<http://freedomdefined.org/OSHW>). Come richiamato nel preambolo, è convinzione diffusa che i progressi sociali e tecnologici rendono possibile a una crescente parte dell'umanità di accedere, creare, modificare, pubblicare e distribuire vari tipi di lavori (opere d'arte, materiali scientifici ed educativi, software, articoli): in pratica tutto ciò che può essere rappresentato in forma digitale. Molte comunità si sono formate per sperimentare queste nuove possibilità creando una grande quantità di opere riutilizzabili dalla collettività. Molti autori, quale che sia il loro campo di attività, qualunque sia il loro status - amatoriale o professionale - hanno un profondo interesse a favorire un ecosistema in cui le opere possano essere diffuse, riutilizzate e derivate in modo creativo. Più facile è riutilizzare e derivare opere, più ricca diventa la nostra cultura. Per assicurare il buon funzionamento di questo ecosistema, le opere devono essere libere e, per libertà, i sostenitori di questa iniziativa intendono essenzialmente:

- la libertà di usare l'opera e di goderne i benefici derivanti dall'uso;
- la libertà di studiare l'opera e di impiegare la conoscenza acquisita da essa;

tari e con una serie di progetti in giro per il mondo: abbiamo nel Bronx un gruppo di ragazzi che realizzano dei robottini, abbiamo fatto dei workshop in Messico, cercando ovviamente di fare un robottino quanto più economico possibile... Dall'altro lato abbiamo lavorato e stiamo cercando di lavorare su piattaforme più potenti: stiamo facendo una serie di progetti (che per adesso non abbiamo ancora reso disponibili al pubblico) per capire se è possibile utilizzare la logica con cui Arduino semplifica l'approccio a certe piattaforme e trasportarla su qualcosa di molto più potente.

"Stiamo facendo una serie di progetti (che per adesso non abbiamo ancora reso disponibili al pubblico) per capire se è possibile utilizzare la logica con cui Arduino semplifica l'approccio a certe piattaforme e trasportarla su qualcosa di molto più potente."

La prima cosa che mi passa per la mente riguarda Linux che io utilizzo normalmente e che mi piace molto ma che, ad esempio, i miei studenti non userebbero mai perché pensano sia troppo complicato... però se si riuscisse ad applicare la logica di Arduino nella programmazione Linux potremmo dare a queste persone l'accesso a delle piattaforme Linux che hanno una grande potenza e caratteristiche di un certo livello. C'è poi, sempre per quanto riguarda l'area educational, il discorso dei linguaggi visuali: con un gruppo di ragazzi stiamo lavorando ad una sorta di LabVIEW, ma molto più semplice, che è fatto tutto in html e che gira su un browser che si chiama *Modkit*. Stiamo lavorando con *Modkit* per consentire anche all'utente che è a digiuno di tutto di poter iniziare a lavorare... *Modkit* è un linguaggio simile a

Scratch di MIT, una cosa grafica molto semplice con dei mattoncini che si incastrano e questo permette anche ai principianti di poter fare i primi progetti. Un altro vantaggio di *Modkit* è che è una piattaforma online, non devi installare nulla e anche questo è molto importante... come nel caso di quanti possiedono un Mac e che possono iniziare a lavorare con Arduino dopo appena 15 minuti perché non c'è driver da installare... Insomma, stiamo lavorando su queste cose: cerchiamo di aumentare la potenza delle piattaforme e cerchiamo di capire cosa fare per il mercato educational."

Nel software e nei sistemi operativi l'open source si è ormai affermato da tempo. Lei che è stato uno dei primi a proporre un progetto open source anche nell'hardware, quale ritiene sarà il futuro dell'open source in questo settore, ovvero nell'hardware? Lo so che è una domanda da un milione di dollari...

"Secondo me c'è grande futuro, specialmente nel campo delle

- la libertà di creare e ridistribuire copie, in tutto o in parte, dell'informazione o espressione;
- la libertà di fare modifiche e miglioramenti, e di distribuire opere derivate.

Se gli autori non intervengono, le loro opere sono coperte dalle leggi sul copyright esistenti, che limitano gravemente ciò che gli altri possono o non possono fare. Gli autori possono rendere libere le loro opere scegliendo tra una serie di documenti legali denominati licenze. Per un autore, scegliere di porre la propria opera sotto una licenza libera non significa perdere i propri diritti, ma dare a chiunque le libertà sopra elencate.

È importante che qualsiasi opera che voglia appartenere a questa categoria - ovvero che sostenga di essere libera - fornisca, praticamente e senza alcun rischio, le suddette libertà. Nel dettaglio, la versione 1.1, approfondisce i quattro punti e chiarisce che un'opera affinché venga riconosciuta "libera" in base a questa definizione deve garantire le seguenti libertà senza limitazioni:

- a) La libertà di usare e rappresentare l'opera: al licenziatario deve essere permesso di poter fare qualsiasi uso, privato o pubblico, dell'opera. Per le tipologie di opera dove ciò è rilevante, questa libertà deve comprendere tutti gli usi derivati ("diritti correlati") come la rappresentazione o interpretazione dell'opera. Non deve esserci alcuna eccezione riguardante, ad esempio, considerazioni politiche o religiose.
- b) La libertà di studiare l'opera e impiegarne le informazioni: al licenziatario deve essere permesso di esaminare l'opera e di usare la conoscenza ottenuta

da essa in qualsiasi modo. La licenza non può, ad esempio, limitare il "reverse engineering".

- c) La libertà di ridistribuire copie: le copie possono essere vendute, scambiate o cedute gratuitamente, come parte di un'opera più grande (es.: una raccolta), o indipendentemente. Non devono esserci limiti al quantitativo di informazione che può essere copiato. Non deve inoltre esserci alcun limite su chi possa copiare l'informazione o su dove l'informazione possa essere copiata.
- d) La libertà di distribuire opere derivate: al fine di dare a chiunque la possibilità di fare miglioramenti ad un'opera, la licenza non deve limitare la libertà di distribuire una versione modificata (o, per opere fisiche, un'opera in qualche modo derivata dall'originale), indipendentemente dall'intento e scopo di tali modifiche.

Solo alcune restrizioni possono essere applicate per proteggere queste libertà essenziali o l'attribuzione degli autori. Infatti non tutte le restrizioni sull'uso o distribuzione delle opere ostacolano le libertà essenziali. In particolare, requisiti per l'attribuzione, la collaborazione simmetrica (ovvero il "copyleft"), e per la protezione delle libertà essenziali sono considerati restrizioni ammissibili.

Sicuramente nelle prossime versioni alcuni argomenti verranno chiariti meglio e la discussione coinvolgerà anche i non addetti ai lavori; in ogni caso, come ha convenuto anche Banzi, il futuro dell'open source hardware è ancora tutto da scoprire.

piattaforme; in altre applicazioni, invece, ad esempio per realizzare un televisore, non ha molto senso. Una piattaforma open source può sicuramente essere uno strumento utile, ad esempio, per gruppi di piccole aziende; 4 o 5 società che portano avanti lo sviluppo di un'unica piattaforma che poi utilizzano per fare prodotti differenti. Una sorta di ricerca di gruppo tanto più valida in paesi, come l'Italia, dove ci sono migliaia di microaziende, e dove ciascuno di solito porta avanti in maniera autonoma e chiusa i propri progetti. Secondo me ha molto senso raggruppare in qualche modo gli sforzi di R&D in alcune aree, perché poi tutti ne traggono vantaggio. Una delle cose belle dell'open source software è che se io prendo Linux e voglio sapere, ad esempio, come funziona il driver di una certa scheda video, mi scarico il sorgente, vado dentro nel kernel e capisco come funziona, così la prossima volta il driver lo faccio io. Con Arduino è la stessa cosa: posso vedere lo schema elettrico, ce l'ho disponibile in formato CAD

(tra l'altro noi usiamo un CAD che è free fino a una certa dimensione), cosa che per un tecnico o un ingegnere ha un valore educativo importantissimo in quanto consente di entrare nel dettaglio di ogni cosa, puoi sapere come è fatto quel pezzo di software o di hardware, lo puoi modificare e lo puoi rifare tu. Personalmente ho imparato l'elettronica leggendo "Nuova Elettronica", quando sono andato all'ITIS sapevo già un sacco di cose... alla fine il metodo è sempre quello: guardi cosa fanno gli altri e impari."

Nel maggio dell'anno scorso, al Maker Faire di San Mateo, in California, ho potuto constatare

di persona quanto Arduino abbia contribuito alla crescita di nuovi gruppi di DIY americani, soprattutto di quelli legati alle tecnologie più avanzate. Pensa

che anche nel nostro paese Arduino possa essere l'elemento trainante di una nuova generazione di auto-costruttori, smanettoni e hacker?

"Attualmente l'Italia rappresenta per Arduino solo il 5% del mercato, contro il 50% degli Stati Uniti dove il successo della nostra piattaforma si basa su comunità molto forti, di persone che collaborano attivamente e seriamente tra di loro. Nel nostro paese sento spesso discorsi del gene-

re: "Questo che ha fatto l'open source è un idiota...prendiamo quello che possiamo e scappiamo". Non è certo un modo di fare community, non so..."

"Una piattaforma open source può sicuramente essere uno strumento utile, ad esempio, per gruppi di piccole aziende; 4 o 5 società che portano avanti lo sviluppo di un'unica piattaforma che poi utilizzano per fare prodotti differenti."

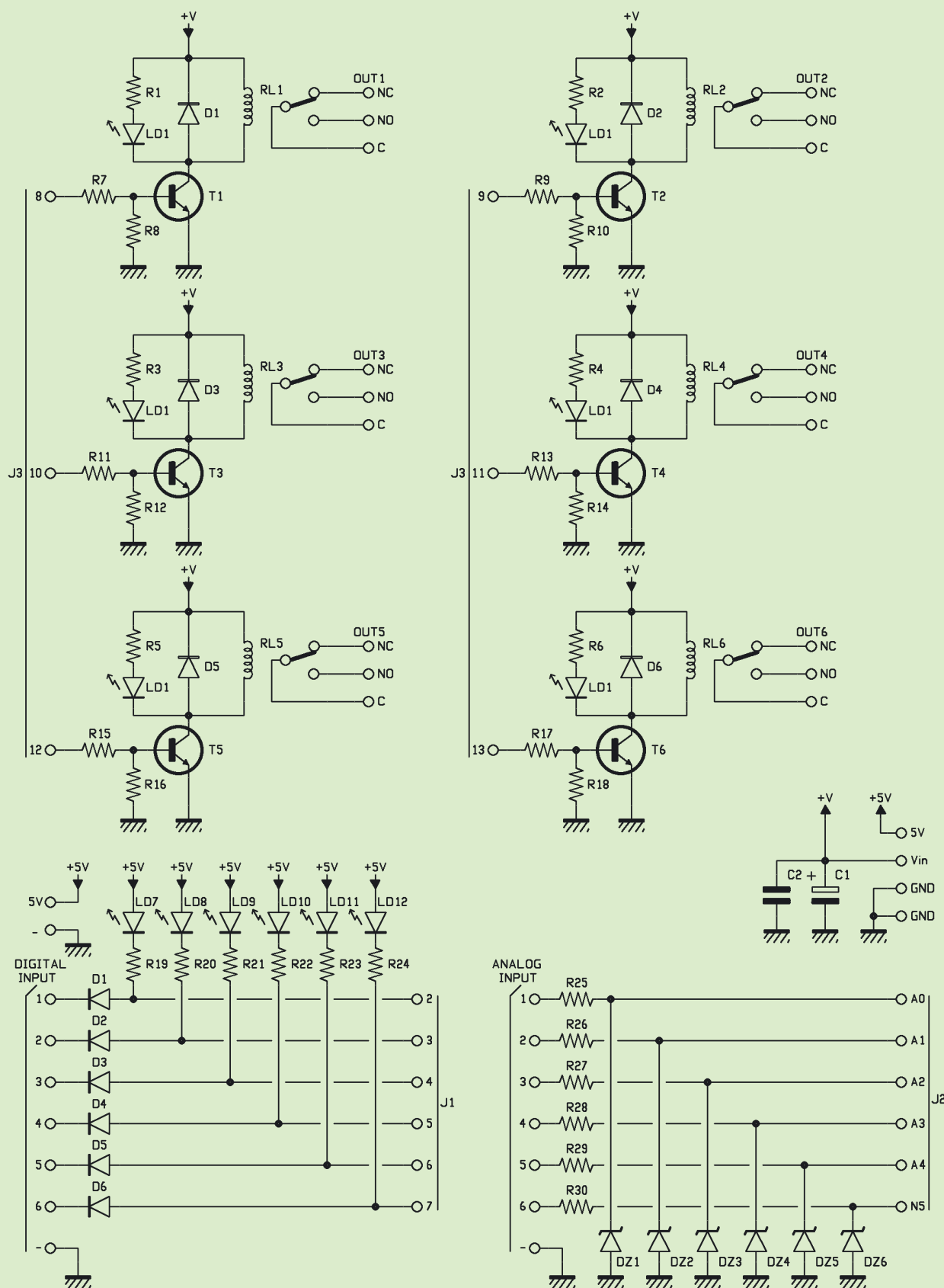
Compatibile con svariate versioni della popolare scheda basata su ATmega, permette di interfacciare gli input analogici e digitali e di dotare le uscite di un relé ciascuna.

di BORIS LANDONI

SHIELD IN/OUT PER ARDUINO

Non si può negare che da un po' di tempo il prodotto più gettonato nel settore dell'elettronica hobbistica e semi-professionale sia la scheda nota come Arduino; ciò perché a bordo di quest'ultima, insieme ad un potente microcontrollore Atmel ATmega si trova un certo numero di periferiche di I/O, porte di comunicazione ed altro ancora, che consentono di sfruttare Arduino come generico dispositivo base per una innumerevole varietà di progetti. Infatti la scheda è sostanzialmente un microcontrollore interfacciato di cui possiamo utilizza-

re praticamente tutte le periferiche interne e gli I/O, ma anche l'A/D converter. Tutti gli I/O e le periferiche possono essere programmati per essere usati a piacimento del progettista, ed è qui il successo di Arduino. Siccome in realtà la scheda è un microcontrollore di cui abbiamo accesso ai pin, per ogni applicazione bisogna realizzare apposite interfacce; queste vengono chiamate *shield* (schermo) perché stanno intorno all'unità e permettono il transito dei segnali proteggendone le linee. Gli shield sono di solito compatibili con varie versioni di Arduino. In



queste pagine vogliamo presentarvi uno shield che abbiamo sviluppato per consentire ad Arduino Duemilanove, Arduino Uno e Seeeduino, di comandare sei utilizzatori mediante relé ed acquisire lo stato di sei ingressi analogici e altrettanti digitali. Per ciascun ingresso digitale è previsto un LED che ne segnala lo stato; anche le uscite hanno ciascuna un LED che indica la condizione del relé corrispondente.

La scheda può essere utilizzata in tantissime applicazioni e in svariati modi; per utilizzarla e provarla in pratica abbiamo realizzato un piccolo sketch di esempio per gestire gli I/O tramite comandi seriali.

Ma procediamo con ordine, dando uno sguardo per primo all'hardware.

SCHEMA ELETTRICO

La nostra scheda è un insieme di relé con rispettivi driver, nonché linee di I/O collegate alle corrispondenti di Arduino mediante pin-strip a passo 2,54 mm; la sezione delle uscite consta di sei driver a transistor abbinati ciascuno ad un relé miniatura a 12 volt e un diodo luminoso, ognuno pilotato da una linea di output del modulo Arduino. Le sezioni sono tutte uguali, perciò ci limitiamo a descriverne una sola: quella facente capo a T1. La base di questo transistor NPN è polarizzata mediante il partitore resistivo R7/R8 quando all'ingresso corrispondente (pin 8 del pin-strip di collegamento ad Arduino) si trova il livello logico alto, livello che può essere 3,3 o 5 volt; in caso ci sia lo zero logico, il transistor è a riposo. Quando T1 è polarizzato in base, va in saturazione e il suo collettore si porta ad un potenziale di poche centinaia di millivolt, lasciando che fluisca corrente nella

bobina del relé; la conduzione dell'NPN fa anche accendere il diodo luminoso LD1, perché lo polarizza attraverso la resistenza R1. All'interdizione del T1, RL1 ritorna a riposo ma la sua bobina, venendo privata della corrente, tende a mantenere le condizioni dell'istante precedente generando un'extratensione inversa che, se non soppressa, danneggerebbe il transistor; perciò abbiamo inserito D1, che sopprime tale tensione.

Dello scambio del relé rendiamo accessibili tutti i contatti, che sono riportati sulle morsettiere di uscita: ogni morsettiera corrisponde ai tre contatti C, NC ed NO.

Passiamo adesso al circuito d'ingresso, partendo con l'analisi della sezione digitale: le sei linee di input, ognuna delle quali corrisponde ad un contatto di della morsettiera siglata DIGITAL INPUT, sono protette da un diodo che si lascia attraversare dalla corrente solo quando vengono portate a zero logico; lasciando un input aperto o portandolo a livello alto, il resistore di pull-up corrispondente manterrà la relativa linea di collegamento ad Arduino allo stato 1. I diodi servono ad impedire che applicando al circuito tensioni maggiori di

quelle previste dal modulo Arduino, le linee di input di quest'ultimo restino danneggiate; ciò consente di applicare livelli logici di 5 o anche 12 volt.

Ciascun resistore di pull-up ha in serie un LED, che si accende quando il relativo ingresso viene trascinato a zero logico, allorché il diodo corrispondente, trovandosi il catodo a massa, va in conduzione; l'accensione del LED indica zero logico all'ingresso, mentre lo spegnimento significa che l'input è ad 1 logico o si trova privo di tensione.

Tutti gli ingressi digitali hanno la massa in comune, che corrisponde al contatto "—" della relativa morsettiera; inoltre, accanto alla morsettiera degli input digitali ne abbiamo collocata una bipolare che porta in uscita i 5 volt (+ e -).

Vediamo adesso le linee analogiche, le quali non sono altro che delle protezioni degli ingressi analogici del modulo Arduino; infatti ciascuno dei sei input si trova fra il contatto della morsettiera ed il corrispondente contatto di Arduino una resistenza in serie e un diodo Zener in parallelo: il diodo limita la tensione se dovesse superare i 5,1 volt e la resistenza limita la corrente che altrimenti scorrerebbe illimitata nello Zener quando questo interviene. Ogni diodo protegge il



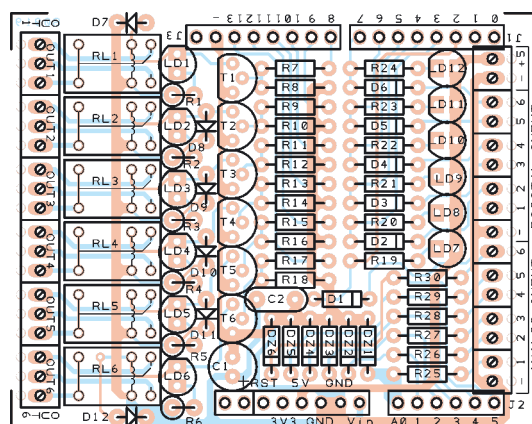
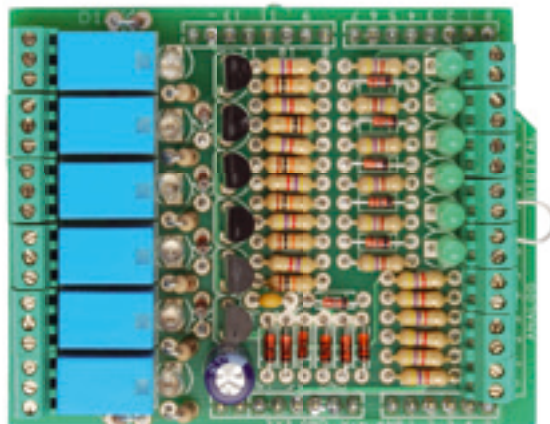
La nostra scheda si inserisce nei connettori I/O dei moduli Arduino.

Elenco Componenti:

R1÷R6: 1 kohm
 R7: 4,7 kohm
 R8: 10 kohm
 R9: 4,7 kohm
 R10: 10 kohm
 R11: 4,7 kohm
 R12: 10 kohm
 R13: 4,7 kohm
 R14: 10 kohm
 R15: 4,7 kohm
 R16: 10 kohm
 R17: 4,7 kohm
 R18: 10 kohm
 R19÷R24: 470 ohm
 R25÷R30: 4,7 kohm
 C1: 100 μ F 25 VL elettrolitico
 C2: 100 nF multistrato
 D1÷D12: 1N4148
 DZ1÷DZ6: Zener 5,1V 400 mW
 LD1÷LD6: LED 3 mm rosso
 LD7÷LD12: LED 3 mm verde
 T1÷T6: BC547
 RL1÷RL6: Relé sub-miniatura 12 V, 1 scambio

Varie:

- Morsetto 2 poli passo 2,54 mm (8 pz.)
- Morsetto 3 poli passo 2,54 mm (6 pz.)
- Strip maschio 8 poli 12 mm (3 pz.)
- Strip maschio 6 poli 12 mm
- Circuito stampato



relativo input di Arduino anche nel caso in cui la linea analogica della nostra scheda riceva una tensione negativa rispetto a massa: in tal caso lo Zener si comporta come un comune diodo e praticamente va in cortocircuito, limitando a circa 0,6 V la tensione negativa presentata alle linee di Arduino, valore, questo, tollerabile.

La nostra scheda prende l'alimentazione dal modulo Arduino, che fornisce i 5 volt ricavati dal proprio regolatore fra i contatti 5V e GND; tali contatti sono (almeno nelle Arduino Duemilano-ve, Uno e Seeeduino) guardando la Arduino dal lato dei componenti e tenendo in alto le file di

connettori di interfaccia, collocati nella fila in basso. Praticamente la massa (GND) corrisponde ai contatti 8 e 9 dall'alto, mentre il +5 V è sul decimo contatto, sempre partendo dall'alto a destra. Oltre ai 5 V, Arduino somministra anche la tensione d'ingresso che l'alimentatore gli applica al plug di alimentazione; ciò, mediante il contatto 9 V.

Notate però che i mini-relé adottati hanno la bobina funzionante a 12 volt, mentre Arduino fornisce solo 5 volt; quindi affinché i relé funzionino bisogna alimentare il modulo Arduino con 12 volt, anziché con i 9 che basterebbero a farlo funzionare. Ciò consente di prelevare dal contatto di Arduino

situato nel connettore delle alimentazioni sopra i pin di massa, la tensione presente sul plug e quindi quanto basta a far funzionare i relé.

REALIZZAZIONE PRATICA

Per costruire la nostra scheda occorre innanzitutto preparare il circuito stampato, che è del tipo a doppia faccia e si ottiene per fotoincisione usando come pellicole due buone fotocopie su carta da lucido o acetato delle tracce lato rame pubblicate nel nostro sito www.elettronica.in. Una volta in possesso della basetta disponete i componenti a partire dalle resistenze e dai diodi al silicio (Zener compresi) dei quali racco-

mandiamo il rispetto della polarità indicata nel disegno di montaggio che trovate in queste pagine; alcuni di questi componenti vanno disposti verticalmente, quindi saldateli dopo. Fatto ciò sistemate i transistor NPN (da orientare con il lato piatto rivolto a dove staranno i relé) e i diodi luminosi, che dovranno tutti essere tondi da 3 mm di diametro; procedete con l'elettrolitico e i sei relé miniatura, quindi inserite e saldate le morsettiere miniatura a passo 2,54 mm. Concludete il montaggio inserendo e stagnando due file da otto pin-strip ciascuna nei contatti relativi ad input ed output digitali di Arduino (le due file che si trovano dal lato opposto all'elettrolitico C1) e due da sei pin ciascuna nelle file di piazzole per la connessione ad Arduino, situate dal lato dell'elettrolitico C1. I pin-strip devono essere alti 20 mm. Se pensate di impiegare la nostra scheda con Seeeduino, prevedete altri due pin-strip lunghi 35 mm da saldare nelle ultime due piazzole che si trovano proprio accanto a C1: questi due collegamenti servono a portare fuori dal modulo Seeeduino due linee a beneficio di altri shield che possono eventualmente essere applicati sopra il nostro. Infatti gli shield per Arduino sono costruiti in modo da poter essere montati a catasta; quindi se intendete montare sul nostro altri shield, dovete usare tutti i pin-strip da 35 mm invece che da 20.

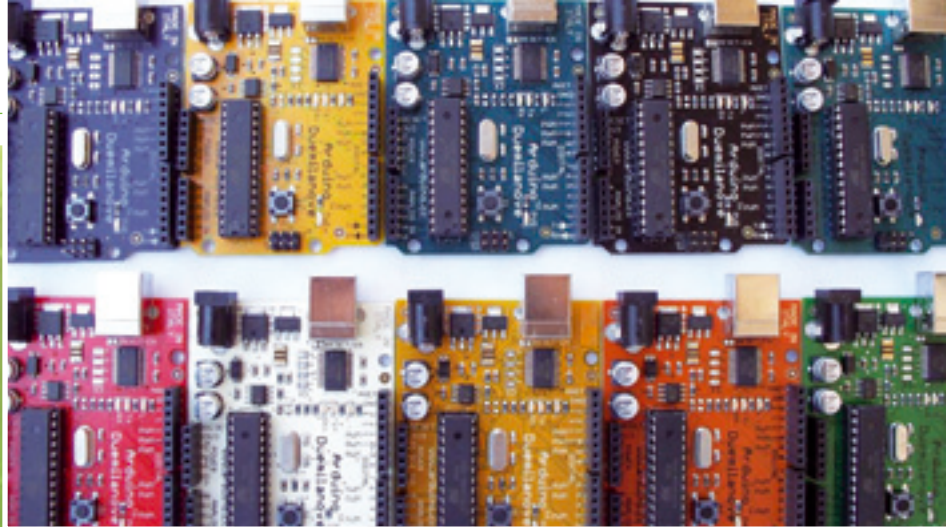
Terminato il montaggio e verificata l'esattezza mediante il piano di montaggio illustrato in queste pagine, potete inserire la scheda sul vostro modulo Arduino. Non è prevista alcuna taratura o regolazione.

USARE LO SHIELD I/O

La scheda qui descritta può essere utilizzata per collegare Arduino ad utilizzatori che devono essere gal-

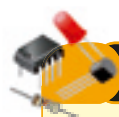
vanicamente separati da essa (ad esempio piccoli motori elettrici) e leggere segnali di sensori o contatti che possono presentare disturbi impulsivi o picchi di tensione tali da danneggiare il microcontrollo-

re ATmega senza un'adeguata protezione; i relé possono pilotare servo relé di maggiore potenza per gestire carichi che assorbono correnti molto intense o lavorano ad alta tensione. La gestione dei relé e



Arduino si fa strada

Non si arresta il successo di Arduino, la scheda a microcontrollore open-source che, partita da Ivrea, è entrata nella cassetta degli attrezzi di progettisti e appassionati di elettronica; in pochi anni è diventata la piattaforma di prototipazione per eccellenza e ha richiamato l'interesse anche di multinazionali come Microsoft (che l'ha impiegata in un esperimento per ridurre le vibrazioni nelle foto con gli smartphone) ma anche di Apple, Hitachi, Panasonic, Asus. La filosofia open-source che ha ispirato Arduino ha facilitato la nascita di start-up per l'elettronica hobbistica come la cinese Seeed Studio Bazaar (che realizza Seeeduino) e Sparkfun, azienda che ha raggiunto un fatturato di dieci milioni di dollari con la vendita di moduli per l'elettronica fai da te. Le applicazioni si moltiplicano anche nei paesi in via di sviluppo, dove sorgono iniziative per costruire strumenti scientifici a basso costo da utilizzare per misure di laboratorio. Per l'anno prossimo Arduino, già utilizzato per la didattica dei corsi di master e durante le lezioni universitarie, potrebbe entrare nei programmi educativi delle scuole secondarie.



per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono di facile reperibilità ed il master del CS può essere scaricato gratuitamente dal sito della rivista così come lo sketch di esempio per gestire gli I/O tramite comandi seriali. Questo shield è anche disponibile in scatola di montaggio (cod. FT919K) al prezzo di 22,00. Le schede di sviluppo Arduino UNO (Euro 32,00) e Arduino MEGA (Euro 61,00) possono essere acquistati presso Futura Elettronica. Tutti i prezzi sono comprensivi di IVA.

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

Il linguaggio Processing

Il nostro modulo può essere gestito con Processing, il linguaggio di programmazione basato su Java, che consente di sviluppare diverse applicazioni come giochi, animazioni e contenuti interattivi. Basandosi su Java, ne eredita completamente la sintassi, i comandi e il paradigma di programmazione orientata agli oggetti; inoltre mette a disposizione numerose funzioni ad alto livello per gestire facilmente l'aspetto grafico e multimediale.

È distribuito sotto licenza Open Source ed è supportato dai sistemi operativi GNU/Linux, Mac OS X e Windows.

Il pacchetto, scaricabile gratuitamente dal sito ufficiale <http://processing.org>, mette a disposizione un ambiente di sviluppo integrato (IDE), e le varie creazioni (sketch) vengono organizzate in uno sketchbook e possono essere esportate come Java applet.

Le funzionalità di processing possono essere espanse tramite delle librerie aggiuntive: ad esempio, è possibile utilizzare in Processing protocolli di comunicazione Seriale, TCP/IP, UDP, RDP, OSC, riuscendo in questo modo a gestire una moltitudine di applicazione hardware esterne.

Lo stesso ambiente di sviluppo Software di Arduino è derivato da Processing, con il quale condivide sia la struttura che il linguaggio di programmazione.

L'acquisizione dei segnali digitali e delle tensioni degli input analogici possono essere effettuate utilizzando semplici routine software o firmware generici già pronti; è anche possibile procedere da Personal Computer, infatti l'IDE di Arduino dispone di un emulatore di terminale (una sorta di Hyper Terminal di Windows) che permette l'invio e la ricezione di dati seriali alla scheda e quindi la gestione diretta dal PC dei relé e degli input. Inviando tramite seriale (impostata per 9.600, 8, N, 1, ossia velocità di 9.600 bps, 8 bit, nessuna parità e un bit di stop) il comando **O** seguito dal numero del canale, si attiva il relé corrispondente; ad esempio,

con **O2** si attiva il relé 2. Il funzionamento previsto per le uscite è di tipo bistabile (toggle) quindi ogni comando inverte lo stato del relativo relé. Col comando **Ix** (dove *x* è il numero del canale) si richiede lo stato di un ingresso digitale, mentre con **Ax** si interroga un ingresso analogico. Ad ogni comando, il modulo Arduino risponde all'emulatore di terminale con un messaggio che segnala la condizione dell'uscita dovuta al comando stesso, ovvero lo stato dell'ingresso oggetto del comando. Ad esempio, impartendo il comando **O2** si può ottenere "Out 2 = 1". Analogamente, per gli ingressi possiamo avere messaggi del

tipo "In 3 = 0" se trattasi di input digitale o Analog 1=126 se si parla di input analogico; in quest'ultimo caso il valore indicato dopo il simbolo di uguale è il corrispondente dell'A/D converter. È anche possibile gestire gli I/O caricando nel microcontrollore dell'Arduino il firmware FIRMATA, scaricabile dalle pagine web <http://arduino.cc/en/Reference/Firmata> oppure http://firmata.org/wiki/Main_Page.

Con esso potete gestire gli ingressi e le uscite direttamente da un'interfaccia grafica, grazie a Processing; quest'ultimo è un linguaggio di programmazione Java Based e open-source di cui abbiamo parlato nella quinta puntata del corso su Arduino, pubblicata nel fascicolo numero 151 (novembre 2010). Processing è un linguaggio che permette di realizzare moltissime applicazioni (sketch con Arduino) e in questo caso lo possiamo usare per elaborare lo sketch *firmata*, che si trova direttamente tra gli esempi dell'IDE di Arduino e che ci servirà a gestire input e relé della nostra scheda.

Questo sketch va modificato leggermente, semplicemente per "dire" ad Arduino quali pin sono collegati alle uscite e quali agli ingressi; poi tutto funzionerà senza difficoltà. ■

Elettronica In
Rivista mensile di progettazione elettronica

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 Elettronica In 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

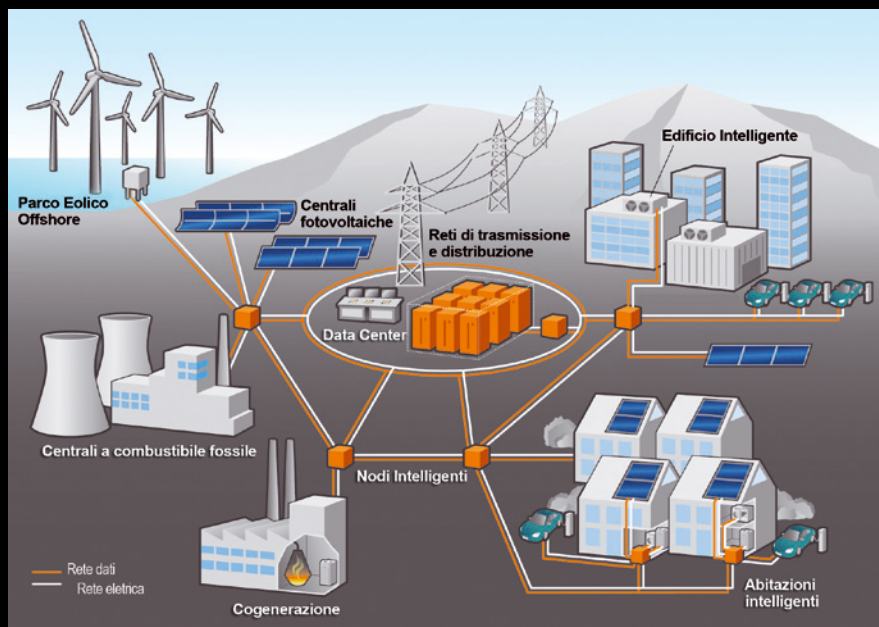
Non perdere questa opportunità!
Contattaci oggi stesso!



Le nuove esigenze della distribuzione, la liberalizzazione della produzione, l'incremento dell'apporto delle fonti rinnovabili, le piccole centrali private, impongono alle reti elettriche di innovarsi ed affidarsi all'informatica.

SMART GRID L'ELETTRICITÀ DIVENTA INTELLIGENTE

di DAVIDE SCULLINO



Negli ultimi anni la produzione e la distribuzione dell'energia elettrica sono state interessate da grandi cambiamenti: la liberalizzazione della produzione e l'entrata in tale settore di grandi soggetti privati, la crescita della quota energetica dovuta alle fonti rinnovabili, la diffusione -grazie anche agli incentivi governativi- dei piccoli impianti fotovoltaici domestici, ma anche l'incremento del consumo civile e industriale. Questo ed altro impone oggi una rivisitazione del sistema elettrico nazionale dei vari Paesi, sia per ottimizzare e meglio dislocare la produzione, sia per limitare le perdite e quindi incrementare l'efficienza delle reti di distribuzione, cosa che si traduce sia in un vantaggio economico, sia in un bene per l'ambiente. E

la revisione del sistema elettrico non può che passare per l'informaticizzazione della rete di distribuzione dell'energia. Tutto ciò si dice in due sole parole: Smart Grid. Questo termine (dall'inglese Reti Intelligenti) significa un'insieme di cose ma in realtà un unico concetto: affidare la gestione, l'operatività, la risoluzione dei problemi, all'Information Technology e quindi ad una rete di computer, ognuno dei quali dovrà governare una porzione del sistema elettrico e tutti quanti, in Rete, coopereranno e risponderanno ad un unico centro nazionale di controllo. Ma perché dovremmo affidare la rete elettrica a dei computer? Sicuramente perché potrebbero mantenere sotto controllo i punti nevralgici, diagnosticare guasti,

prevenire sovraccarichi e intervenire più prontamente di una persona; ma non è solo per questo, anzi, se fosse tutto qui non si vedrebbe l'incombenza di ricorrere alle Smart Grid. In realtà ci sono varie ragioni che appaiono evidenti valutando lo stato attuale ed il futuro del sistema elettrico dei Paesi industrializzati. Il primo aspetto è la dislocazione sul territorio nazionale delle centrali elettriche e della produzione, che non sempre rispecchia le necessità locali; questo significa, in pratica, che sovente le zone dove c'è bisogno di molta corrente non hanno centrali a sufficienza e che in regioni dove la produzione è molta il fabbisogno è basso. Certo, ci sono le linee di distribuzione tramite cui la corrente prodotta

Il sistema elettrico nazionale e le Smart Grid

La disponibilità di energia elettrica è fondamentale per lo sviluppo di un Paese, perché dall'elettricità dipende un po' tutto; senza contare che l'elettronica che ausilia e coordina l'attività degli ospedali, degli uffici, della Difesa, della navigazione aerea ecc., dipende dall'elettricità.

Per comprendere la necessità delle Smart Grid occorre rifarsi alla complessità del sistema elettrico, che in Italia, nel 2009, contava:

- 1.161 impianti di generazione termoelettrica;
- decine di migliaia di impianti di generazione da fonti rinnovabili;
- oltre 400.000 km di elettrodotti di trasmissione ad alta e media tensione
- circa 850.000 km di linee di distribuzione a bassa tensione;
- oltre 2.000 stazioni elettriche di trasformazione e distribuzione in alta tensione.

Il consumo giornaliero di

elettricità in Italia raggiunge 45 GWh in inverno e supera i 53 GWh in estate. Tutto questo viene gestito da un imponente sistema finalizzato ad avviare a guasti, variazioni della domanda e degli assetti di rete. Siccome allo stato attuale l'energia elettrica non si può immagazzinare, bisogna produrre, istante per istante, la quantità richiesta dall'insieme dei consumatori (famiglie e aziende) e gestirne la trasmissione in modo che l'offerta e la domanda siano sempre in equilibrio, garantendo così la continuità e la sicurezza della fornitura del servizio. La gestione di questi flussi di energia sulla rete si chiama dispacciamento. Tale attività richiede il monitoraggio dei flussi elettrici e l'esercizio coordinato di impianti di produzione, rete di trasmissione e servizi ausiliari. La gestione in tempo reale del nostro sistema elettrico, interconnesso con quello europeo, viene svolta dal Centro nazio-

nale di controllo, il quale acquisisce, istante per istante, tutti i dati relativi allo stato del sistema elettrico e, in base alle esigenze del momento, mette in atto le opportune azioni correttive.

Il Centro opera in tre fasi: nella fase di programmazione, elabora i piani di esercizio sviluppati sulla base delle previsioni della domanda di energia e di potenza a livello nazionale e della disponibilità dei mezzi di produzione. Le previsioni a breve termine, settimanali e giornaliere, sviluppate in base a quelle a medio termine, consentono di valutare i livelli di produzione, la configurazione di funzionamento della rete e la riserva di potenza.

Nella fase di controllo in tempo reale, il Centro analizza lo stato del sistema elettrico ed interviene sulla produzione della potenza attiva e reattiva e sull'assetto di rete; contemporaneamente opera per l'ottimizzazione del servizio, il ripristino in

caso di disservizi, il controllo di eventuali emergenze ed il coordinamento della manutenzione. Nella fase di analisi dell'esercizio, oltre all'elaborazione delle statistiche, analizza il funzionamento del sistema di produzione e trasmissione. Il Centro nazionale di controllo si avvale di otto centri di ripartizione, che decidono gli interventi sugli impianti in fase di programmazione e controllo in tempo reale. Lo sviluppo di nuove tecnologie di generazione elettrica esige che il sistema si evolva, ad esempio per adeguarsi alla generazione distribuita (basata su tanti piccoli impianti dispersi nel territorio) ed alle nuove centrali, dove a differenza delle termoelettriche (la cui potenza può essere variata cambiando la quantità di combustibile bruciato) e di alcune idroelettriche (modulabili bypassando alcune turbine) non è possibile controllare la produzione. Per ottimizzare l'uso di questi impianti, occorre non solo

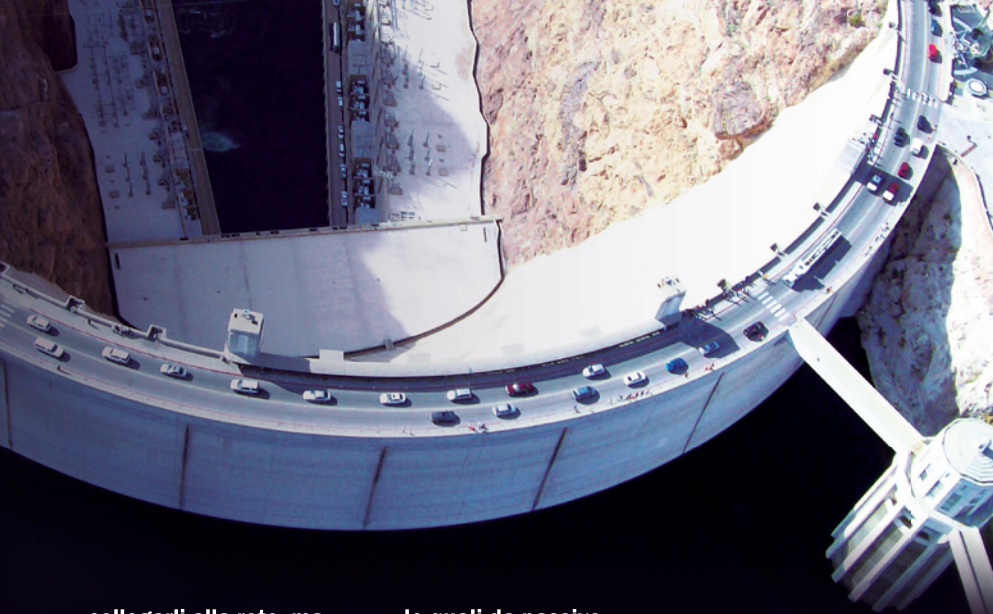
in eccesso può essere trasportata lontano, tuttavia ogni km di linea comporta delle perdite di potenza dovute alla resistenza dei cavi; è forse per questo che, pur avendo sulla carta tutta la corrente che ci occorre, talvolta si verificano blackout localizzati. Ridurre le perdite di energia nelle linee di distribuzione convogliando l'elettricità all'utilizzatore attraverso la scelta di percorsi sempre più corti, è uno dei compiti delle Smart Grid e certamente viene incontro alle esigenze sia economiche, sia ambientali, giacché energia sprecata vuol dire -almeno fin quando a produrre elettricità saranno centrali termoelettriche (negli U.S.A.

si stima che se la rete fosse solo del 5 % più efficiente, si ridurrebbe l'emissione di gas serra pari a quelli prodotti da 53 milioni di automobili) e nucleari- un'inutile offesa all'ambiente. Il risparmio in termini di denaro deriva sia dalla riduzione degli sprechi, sia dall'accorciamento delle linee di distribuzione e dalla possibilità di progettare le centrali della taglia che serve, giacché con le metodiche attuali, per mettersi al riparo dai blackout occorrerebbe avere potenze installate maggiori dell'effettiva richiesta e solo per sopperire a picchi d'assorbimento momentanei. Attualmente la distribuzione si fa monitorando i consumi e dirottando parte della

produzione dove si ritiene che serva di più, ma la ripartizione si ottiene secondo statistiche e non certo in tempo reale; affidando la gestione ai computer che controllano grosse centrali di commutazione, diverrebbe più veloce e semplice compensare picchi di richiesta non prevedibili basandosi sulle semplici statistiche.

GESTIRE LE RINNOVABILI

Il secondo aspetto, probabilmente il più attuale, è la crescente diffusione degli impianti privati, i quali, grazie allo "scambio sul posto", possono cedere elettricità alla rete elettrica; questa realtà è apprezzabile, perché seppure il suo contributo sia minimale e discon-



collegarli alla rete, ma anche integrare, controllare e gestire la loro produzione di energia. Trattandosi di impianti molto diffusi sul territorio, la rete più vicina per collegarli è quasi sempre quella di distribuzione, che oggi non è tecnicamente in grado di integrare una notevole quantità di energia prodotta da un gran numero sorgenti di piccola taglia. Lo sviluppo delle nuove fonti rinnovabili richiede dunque di cambiare le modalità di uso e progettazione delle reti elettriche,

le quali da passive devono diventare attive ed intelligenti, in grado di integrare e gestire in modo flessibile flussi di energia differenti per quantità, provenienza e programmabilità. Le Smart Grid nascono proprio per integrare in rete nuovi servizi e infrastrutture, in modo da ridurre gli sprechi, accrescere l'autosufficienza e la sicurezza energetica, ridurre i costi e, in generale, aumentare l'efficienza del sistema. È questo il concetto di Smart Grid: applicare

alle reti elettriche gli stessi principi che sono alla base di Internet, dove qualsiasi dispositivo connesso può inviare e ricevere contenuti in modo flessibile e distribuito. Allo stesso modo, nelle Smart Grid i produttori locali (piccoli e grandi) devono poter scegliere se attingere dalla rete oppure immettere l'energia che producono, basandosi sulla convenienza economica offerta dalle tariffe orarie e sui momenti in cui consumare costa meno.

tinuo, va a colmare quei vuoti lasciati dalla grande produzione e consente ai grandi produttori di centrare gli obiettivi di ottenimento dei cosiddetti "certificati verdi" attraverso l'acquisizione di energia prodotta da terzi mediante le rinnovabili. Il problema è che non è facile controllare l'attività di miriadi di piccoli punti di immissione e convogliare questa micro-produzione dove il fabbisogno lo impone. Qui, più che in altri ambiti, le Smart Grid possono risultare risolutive, interfacciandosi con i contatori elettronici e ottimizzando il dispacciamento. Analogo è il concetto per i grandi impianti che si basano sull'energia solare ed eolica; quest'ulti-

ma, in particolare, è nota per la sua discontinuità, che rischia di vanificare i grandi pregi dello sfruttamento del vento (gli aerogeneratori si installano in circa 40 giorni e sviluppano ciascuno da 0,5 a 4 MW). Eolico e fotovoltaico, diversamente da termoelettrico, nucleare ed idroelettrico, non producono costantemente; pertanto occorre fare in modo che quando non sono in grado di fornire l'elettricità richiesta, la zona da essi servita venga rifornita da corrente in arrivo da altre centrali, che possono essere tradizionali oppure eoliche e fotovoltaiche di regioni vicine. L'apporto di piccole produzioni sparse e la pluralità dei soggetti

implicati oggi nella produzione e distribuzione di elettricità sono i punti chiave del cambiamento del sistema elettrico nazionale; questo riguarda tanto l'Italia, quanto altri paesi industrializzati dove la partecipazione dei privati è ad uno stadio più avanzato. Se da un lato ciò avvantaggia il mercato e il consumatore, dall'altro pone il problema del coordinamento della produzione, giacché mentre prima c'era un unico soggetto a pianificare il settore, oggi si rischia che ognuno viaggi per conto proprio e che si crei o si acuisca un certo squilibrio tra domanda ed offerta a livello regionale e locale. Inoltre accadrebbe che sebbene la produzione potrebbe superare la richiesta, in realtà intere zone verrebbero a trovarsi senza la corrente di cui necessiterebbero, perché ci sarebbe una produzione poco mirata.

Un ulteriore aspetto che evidenzia l'importanza delle Smart Grid è l'evoluzione della tipologia di consumo e l'indubbio incremento dell'elettricità richiesta quando le automobili elettriche sostituiranno le attuali: se è vero che avremo l'aria più pulita in città, è vero altrettanto che sarà necessario produrre più corrente, dato che i MWh attualmente sviluppati dai motori endotermici dovranno essere sostituiti da MWh elettrici. Ma non solo: le auto elettriche a batteria si caricano molto lentamente e quindi è preferibile metterle in carica la notte a casa propria o nelle autorimesse degli autonoleggi o delle aziende, il che significa spostare più elettricità in certe zone la notte, dove il giorno ne serve meno. Inoltre in un futuro prossimo le auto elettriche verranno alimentate non da batterie, ma da supercondensatori da decine di farad, i quali possono essere caricati in pochi minuti, quindi da distributori tali e quali

NI per le Smart Grid

La stima dello stato delle reti elettriche di distribuzione attive, ossia basate su generazione distribuita controllabile, può essere facilitata dalla misura sincronizzata dei fasori (sincrofasori) delle tensioni nodali, mediante dispositivi denominati Phasor Measurement Unit o PMU. Il concetto, derivato da quello di misura su vasta area (Wide Area Monitoring System) che ha aperto nuove prospettive nel monitoraggio e nella gestione dei sistemi elettrici di grandi dimensioni, può essere applicato anche alle reti elettriche con generazione distribuita. Conoscere i fasori sincronizzati delle tensioni in diversi punti della rete consente, infatti, di fornire misure estremamente utili per la stima dello stato di tali reti (il fasore è un numero complesso composto da modulo e fase della corrente e

permette di valutare le alterazioni della forma d'onda), fondamentale per l'implementazione dei cosiddetti Distribution Management Systems (DMS) in grado di implementare diversi algoritmi di gestione sia della rete sia delle risorse di produzione in essa presenti. In quest'ottica si colloca una sperimentazione effettuata presso l'Università degli studi di Bologna, riguardante la misura di sincrofasori realizzata tramite lo sviluppo di una unità basata sull'implementazione di algoritmi di analisi su un sistema basato su DSP ed FPGA. Tale sistema è in grado di monitorare, acquisire ed elaborare grandezze caratteristiche (in genere le tensioni di un nodo del sistema) in maniera autonoma. Gli algoritmi sono stati sviluppati su una piattaforma CompactRIO della National Instruments.

quelli delle attuali stazioni di servizio; perciò serviranno (visto che la carica sarà rapida) grandi quantità di elettricità e la produzione dovrà essere localizzata o decentrata anche vicino alle strade di grande comunicazione.

DA PASSIVA AD ATTIVA

Le reti elettriche convenzionali sono passive: il sistema produ-

ce corrente indipendentemente e la immette in rete, da dove attinge l'utenza. Realizzare una Smart Grid significa far partecipare, interagire tutti i soggetti coinvolti, dal produttore al distributore, fino al consumatore. In altre parole, una rete diventa "intelligente" quando integra tecnologie elettriche ed elettromeccaniche con soluzioni

digitali, in modo da consentire una gestione flessibile, un controllo proattivo e automatizzato, un dialogo sinergico tra i vari componenti della rete e anche nuovi sistemi e dispositivi in una rete sempre più complessa e articolata. In altre parole in una Smart Grid la produzione non va per conto proprio ma si adegua, insieme alla distribuzione, alle informazioni attinte dall'utenza, ad esempio attraverso i contatori intelligenti tradizionali e a quelli per lo scambio sul posto: nel primo caso il contatore invia i dati sul consumo alla centrale di controllo che, insieme a quelli dei contatori degli altri utenti di un'area, valuta quanta corrente trasferirvi. Nel secondo, il sistema valuta la differenza tra energia consumata e prodotta e se il saldo è negativo si regola sapendo di poter contare sull'apporto di elettricità fornita al momento dallo scambio sul posto. Gli elementi base di una Smart Grid sono:

- un sistema informatico che monitorizza la rete da remoto attraverso sensori e apparecchiature elettroniche intelligenti collocati presso le utenze, allo scopo di analizzare consumi e trend;
- la trasmissione di dati in tempo reale, attraverso reti di comunicazione;
- un sistema decisionale in real-time che include modelli, simulazioni, visualizzazioni e capacità analitiche, capace di intervenire sulla rete e di operare in remoto, interagendo con i carichi attivi e i parametri di esercizio.

Le reti intelligenti cambieranno il modo di operare della trasmissione e della distribuzione dell'energia elettrica, estenden-



I produttori di semiconduttori si contendono il mercato delle soluzioni di Smart Metering; STM, con i suoi STarGRID™ ST75xx (SoC per la comunicazione lungo le linee elettriche) partecipa in Spagna al progetto STAR Project della IBERDROLA, che prevede entro l'anno l'installazione di 100.000 contatori intelligenti, conformi allo standard PRIME, nella città spagnola di Castellon.

do le capacità di controllo e di misura ad un maggior numero di componenti della rete. La maggiore complessità richiede nuovi strumenti e tecnologie, tra cui software dedicati; in tal senso va segnalato l'impegno dell'IBM, la quale ha sviluppato una varietà di software per rendere la rete elettrica più intelligente. Nel 2007 l'azienda ha creato l'Intelligent Utility Network Coalition, che comprende un gruppo di servizi di pubblica utilità interessati a portare il computing alla rete elettrica (per approfondimenti, visitate www.ibm.com/smarterplanet/it/it/smart_grid/ideas/). Per vincere la sfida lanciata dalle Smart Grid occorrerà introdurre nuove funzionalità di analisi e di supporto alle decisioni degli operatori, nonché contenere le conseguenze degli errori umani e i tempi di reazione.

CHI GESTISCE LE SMART GRID

L'interesse a realizzare un sistema intelligente di gestione della rete elettrica è certamente dei gestori della rete elettrica, ma indubbiamente il progetto è di più ampio respiro e si sviluppa sotto il controllo e il patrocinio dei governi nazionali; infatti l'elettricità è la fonte energetica più importante, in quanto facilmente trasportabile e convertibile in altre forme di energia con un solo passaggio, cosa impossibile per i combustibili, ad esempio. Infatti l'elettricità permette di ricavare calore nelle stufe, luce con le lampade, movimento con i motori, magnetismo con le bobine, ma anche di far funzionare tutta l'elettronica senza la quale oggi saremmo praticamente retrocessi al medioevo. Questo bene vitale per il funzionamento di settori quali la sicurezza e la salute pubblica, ma anche dei trasporti e la



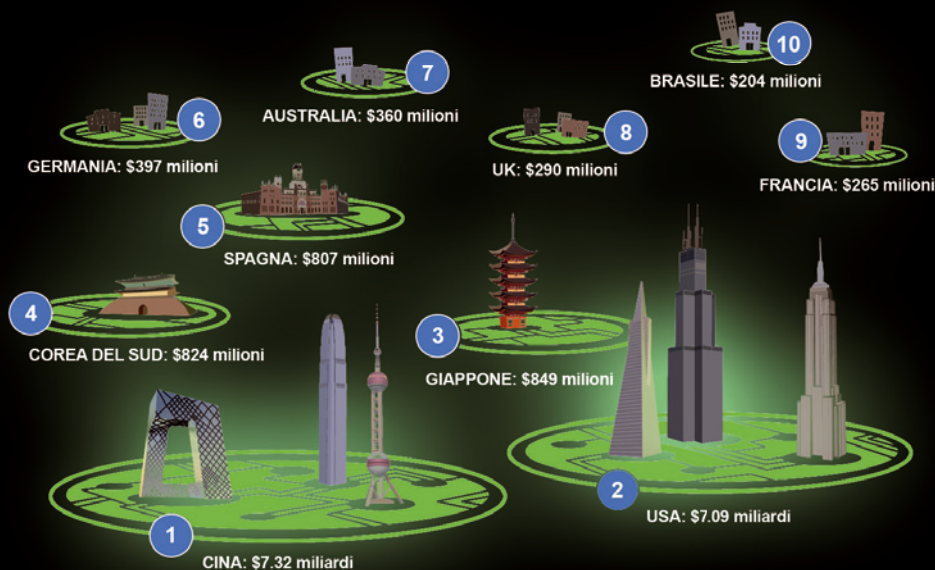
Da tempo l'ENEL studia tecnologie innovative di monitoraggio e controllo delle reti e guarda alle Smart Grid: in ambito europeo, l'azienda italiana fa parte dell'Advisory Council della Technology Platform Smart Grids, una piattaforma tecnologica nata nel 2005 per creare una visione condivisa delle reti europee del futuro. Gli altri progetti in cui l'ENEL è impegnata sono:

- **ADDRESS**, co-finanziato dall'Unione Europea con l'intento di favorire la partecipazione attiva dei piccoli e medi consumatori al mercato dell'energia; mira a sviluppare apparati per la gestione dei carichi e dei sistemi di generazione, in modo da offrire servizi al mercato in funzione delle necessità della rete;
- **Smart Metering**; consiste nella gestione della connessione e acquisizione remota delle misure dell'energia elettrica sia prelevata, sia immessa nella rete del distributore;
- **Smart Info**, che ha l'obiettivo di rendere disponibili direttamente all'utente le informazioni dello Smart Meter, aumentando la consapevolezza del consumo e promuovendo un uso razionale dell'energia;
- **Smart Distribution Network Operator** che punta a sviluppare, realizzare e verificare un sistema di protezione, controllo e automazione per la gestione delle reti in media tensione in presenza di elevata quantità di

flussi di energia provenienti da impianti a fonti rinnovabili diffusi sul territorio;

- **e-mobility Italy**, progetto finalizzato a realizzare un modello integrato e capillare di mobilità elettrica; prevede la diffusione di tecnologie di ricarica d'avanguardia a supporto della penetrazione e utilizzo efficiente di veicoli elettrici.

Il più importante progetto dell'ENEL, che ha preparato la strada alla realizzazione delle Smart Grid, è il Telegestore, cioè l'adozione del contatore elettronico, che ha comportato la sostituzione di circa 31 milioni di vecchi contatori elettromeccanici con apparecchi digitali di nuova generazione progettati e prodotti dall'ENEL. Oltre ai contatori, l'infrastruttura di telegestione prevede anche il concentratore, che installato nelle cabine secondarie della rete ENEL raccoglie i dati registrati dai contatori ad esso collegati; e poi il sistema centrale, che gestisce l'acquisizione dei dati di misura e le operazioni contrattuali, scambiando le informazioni con i concentratori attraverso la rete pubblica di telecomunicazione GSM. Enel ha esportato la propria tecnologia, tant'è che Endesa ne adotterà il Contatore Elettronico per i propri 13 milioni di clienti. Il contatore ha ricevuto l'European Utility Award nella categoria Business Performance al Metering Europe 2009.



La classifica dei primi 10 paesi al mondo per investimento nelle Smart Grid, aggiornata al 2010: svettano Cina e Stati Uniti.

difesa, sta sicuramente a cuore agli Stati, i quali non a caso vedono di buon occhio le Smart Grid e, chi prima e chi dopo, sviluppano piani e stanziando ingenti somme di denaro, con la collaborazione di partner capaci di apportare fondi e tecnologie. I primi a muoversi nella direzione delle Smart Grid, sono stati gli U.S.A., dove da anni si sta lavorando al progetto, che coinvolge grandi multinazionali del calibro di Siemens ed IBM (la quale ha siglato un accordo da 70 Milioni di euro per realizzare una Smart Grid a Malta entro il 2012) sono stati investiti denari per ben 1 miliardo di dollari nella creazione di Start-Up, due delle quali -GridPoint e Silver Spring Networks- hanno raggiunto quota, rispettivamente, 220 e 170 milioni. Nei prossimi anni gli investimenti saranno consistenti (ad oggi gli U.S.A. hanno stanziato oltre 7 miliardi di \$) e la Siemens conta di aggiudicarsi ordinativi per 6 miliardi di dollari. Anche la Cina, ad oggi ha messo in campo investimenti pari a 7,32 miliardi di dollari! Un giro d'affari imponente che ci fa capire l'importanza non solo tecnica e strategica delle Smart Grid, ma

anche il rilievo che avrà nello sviluppo economico degli anni a venire. Le Smart Grid costituiscono una delle frontiere per il futuro dei sistemi elettrici, un futuro verso il quale si sta indirizzando l'impegno internazionale, pur nella consapevolezza che la strada da percorrere è ancora lunga; occorre, infatti, modificare radicalmente infrastrutture che hanno dimensioni enormi. E occorre farlo in presenza di vincoli tecnici molto complessi. Le attività di ricerca e i progetti di dimostrazione si moltiplicano. Importanti passi in tale direzione vengono fatti quasi ogni giorno, come ha dimostrato in modo palese l'ENEL, che è oggi riconosciuta leader indiscusso a livello mondiale in questo processo di rinnovamento. Il primo passo è stato compiuto con la realizzazione del Progetto Contatore Elettronico, che non ha paragoni al mondo per estensione e complessità strutturale e che in Italia sta già dando tangibili risultati in termini di maggiore "intelligenza" della rete e di efficienza dell'intero sistema. Oltre a misurare l'energia utilizzata, il contatore elettronico ha consentito

l'introduzione di tariffe multiple e di preparare il terreno all'avvento di centri di gestione elettronica delle reti, dato che ciascun contatore invia -con la tecnica delle onde convogliate- dati sul consumo sfruttando la stessa linea elettrica che lo alimenta.

Nel quadro dei progetti di sviluppo delle Smart Grid, l'ENEL ha avviato a fine 2009 il Programma "Reti intelligenti MT", per la sperimentazione di modalità innovative di esercizio della rete in Media Tensione. Il programma, che si articola in quattro Progetti regionali e beneficia di un finanziamento dell'UE pari a 77 milioni di euro, riguarda regioni del Sud Italia interessate al cosiddetto "Obiettivo convergenza" UE: Puglia, Sicilia, Campania e Calabria. I fondi utilizzati sono quelli del Programma Operativo Interregionale (POI), che si prefigge di aumentare la quota di energia consumata proveniente da fonti rinnovabili. L'obiettivo specifico del Programma "Reti intelligenti MT" è facilitare la diffusione della generazione distribuita sperimentando, su alcuni siti pilota, l'evoluzione della gestione delle reti verso un tipo misto attivo/passivo. Gli interventi sono mirati a rendere l'assetto della rete MT più favorevole all'inserimento di impianti di generazione fotovoltaica di taglia compresa tra 100 kW e 1 MW.

SICUREZZA E CYBER-ASSALTI

Affidare la gestione dell'intera rete elettrica di una nazione ad una serie di computer pone una serie di interrogativi, primi fra tutti quelli sulla sicurezza nazionale: infatti molti Stati -U.S.A. in testa- ritengono che tra le maggiori minacce per il

L'Europa ci chiede le Smart Grid

mondo industrializzato negli anni a venire, figurino gli attacchi ai sistemi informatici; non possiamo escludere che il terrorismo non tenti l'assalto alle Smart Grid con virus e roba simile.

La possibilità che un attacco da Internet possa mettere fuori uso il sistema elettrico di una nazione non va comunque sottovalutata: più volte in America è stata denunciata la vulnerabilità del sistema ed è recentissimo un rapporto del Dipartimento dell'Energia degli Stati Uniti, in cui si denuncia la vulnerabilità della rete elettrica, la cui causa principale sarebbe l'insufficiente implementazione degli standard di sicurezza informatica definiti nel 2008 dalla North American Electric Reliability Corporation, ai quali le compagnie elettriche avrebbero dovuto adeguarsi. Ebbene, secondo l'Ispettore generale del dipartimento dell'Energia, non è andata così: ogni compagnia ha agito con proprie metodiche. Il report sostiene anche che mancano i requisiti di sicurezza per log-in e password, facilmente penetrabili dagli hacker. Il tema è stato affrontato anche in *Cyber War*, il libro di Richard Clarke, ex funzionario della Sicurezza alla Casa Bianca, che definisce quella elettrica come una "infrastruttura critica" e che mette in guardia contro le armi informatiche, capaci di distruggere le infrastrutture controllate dai computer, come il caso del virus Stuxnet ha dimostrato negli impianti nucleari iraniani; Clarke individua nella Cina la potenziale ideatrice dei cyber-attacchi, in grado, nel giro di un quarto d'ora, di causare un blackout all'ora di punta in 157 città statunitensi, gettandole nel panico.

La legislazione europea impone a tutti gli stati membri di migliorare l'efficienza energetica entro il 2016, puntando, rispetto al consumo energetico registrato tra il 2001 e il 2005, ad un risparmio del 9 %, anche grazie al contributo delle Smart Grid, il cui elemento cardine sono per ora i contatori elettrici intelligenti (Smart Meter). Per raggiungere lo scopo sono stati avviati programmi di incentivazione statali in tutta Europa (ad esempio in Germania, Francia e Gran Bretagna); secondo Greenbang, l'istituto britannico di ricerche di mercato, nel Vecchio Continente sono già installati quasi 53 milioni di Smart Meter. In dieci anni il loro numero dovrebbe raggiungere i 133-145 milioni di esemplari. In Italia siamo già a posto ed anche la Svezia è a buon punto; in Germania, i contatori intelligenti per energia elettrica e gas devono essere

installati sulle nuove costruzioni e sugli edifici completamente ristrutturati a partire da gennaio 2010. Quanto all'Olanda, gli Smart Meter avrebbero dovuto essere installati entro il 2013 in tutti i sette milioni di abitazioni del paese, ma finora non è andata così. Gli Smart Meter devono soddisfare diversi requisiti tecnici in termini di misurazione e comunicazione, oltre a consumare poca elettricità (altrimenti vanificano i benefici per cui nascono); inoltre devono essere costruiti con un'elettronica facilmente adattabile, visto che per il momento non c'è unificazione nei vari Paesi europei. Per i sistemi Smart Metering sono stati realizzati i moduli Arrow, in grado di supportare sia la trasmissione dei dati misurati ai fornitori di energia elettrica e ai consumatori finali, sia il cablaggio domestico intelligente.



Come ogni innovazione, anche le Smart Grid nascondono qualche falla. Comunque il rischio c'è ma non è elevatissimo, perché se è vero che colpendo il sistema informatico delle Smart Grid si potrebbe bloccare la distribuzione dell'elettricità (fermando il traffico ferroviario e aereo, bloccando l'attività degli ospedali e rallentando quella dei sistemi di difesa militare) o, peggio, causare sovraccarichi che portano alla distruzione di interi tratti della rete elettri-

ca, in realtà si può arginare il pericolo strutturando opportunamente il sistema. Ad esempio ripartendo il controllo su tanti computer (ognuno dei quali governa un tronco) e interconnettendo in rete tutti i tronchi, proteggendo l'accesso a ciascun sistema informatico dalla rete dati e imponendo che alcune decisioni possano essere prese solo dall'uomo (ciò si realizza affidando ad hardware svincolato dal software il controllo dei passaggi più importanti).

APPUNTAMENTO ALLE FALDE DELL'ETNA PER LA **RobotCup Junior**

di Arsenio Spadoni

Numeri da record per la terza edizione della **RoboCup Junior** che si svolgerà ad aprile a Catania: solo per la categoria "under 19" saranno ben 75 le squadre che gareggeranno nelle categorie **Soccer, Rescue e Dance**: team provenienti praticamente da tutte le regioni d'Italia ai quali si aggiungeranno le squadre under 14, per la prima volta presenti a questa manifestazione. Le competizioni della categoria Rescue hanno anche lo scopo di selezionare la squadra italiana che parteciperà alla finale mondiale che si svolgerà dal 5 all'11 luglio a Istanbul, mentre per quanto riguarda le altre categorie sarà la RomeCup - in svolgimento questo mese - che selezionerà i team italiani che parteciperanno alla finale in Turchia. La manifestazione in

programma dal 14 al 16 aprile 2011 è stata presentata ufficialmente sabato 22 gennaio presso l'Istituto Tecnico Archimede durante il Convegno Roboticamente 2: le gare si svolgeranno presso il Centro fieristico "Le Ciminiere" a cura proprio dell'ITIS Archimede.

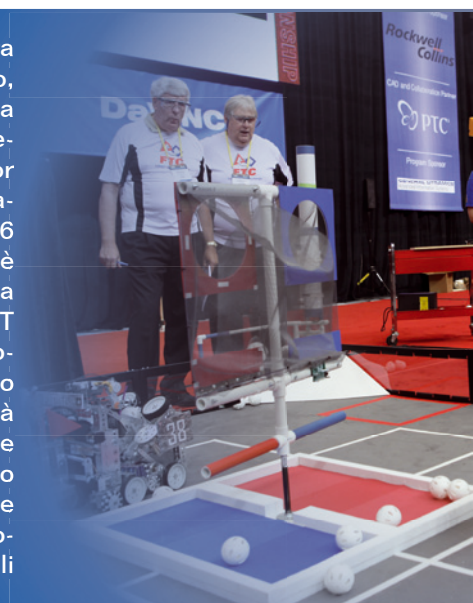
La grande novità di quest'anno è la competizione riservata agli studenti under 14 che saranno seguiti dalla Prof.ssa Simonetta Siega: per la prima volta anche i ragazzi più giovani (tipicamente allievi della Scuola Media Inferiore) potranno cimentarsi in questo genere di gare, come già avviene all'estero (vedi riquadro sulla manifestazione FIRST). Tra le squadre in gara per la categoria Rescue, i due team dell'Istituto Erasmo da Rotterdam di Bollate (MI) guidati dall'Ing. Giulio



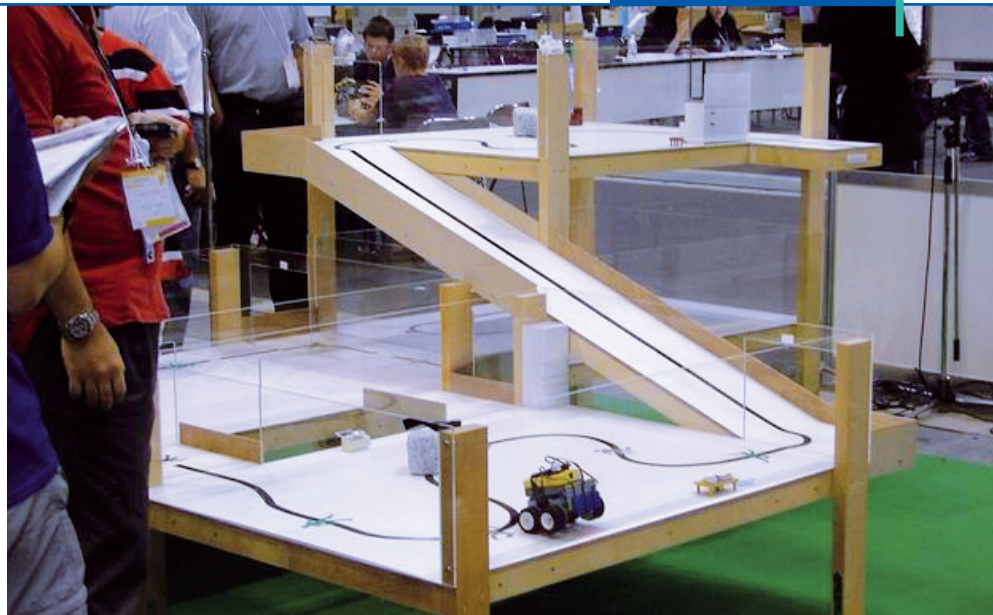
Anche in USA aprile è il mese delle gare di robotica

Dal 27 al 30 aprile si svolgeranno a St. Louis, in Missouri, le fasi finali dell'annuale FIRST Championships (www.usfirst.org). La fondazione FIRST (For Inspiration and Recognition of Science and Technology) è un'organizzazione americana senza fini di lucro nata per volontà di Dean Kamen, inventore e imprenditore (tutti lo conoscono per l'invenzione del Segway) e attiva dal 1989 con lo scopo di diffondere la passione

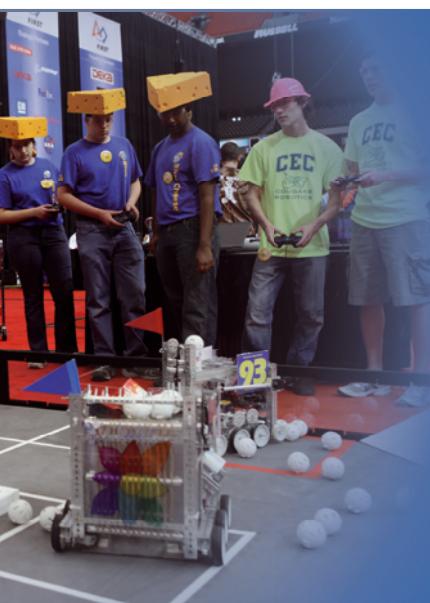
per la scienza e la tecnologia tra le nuove generazioni. A tale scopo, FIRST organizza gare di robotica con differenti livelli di difficoltà destinate a tre fasce d'età: la Junior FIRST LEGO League ha come protagonisti scolari di età compresa tra 6 e 9 anni, la FIRST LEGO League è destinata a ragazzi di età compresa tra 9 e 14 anni mentre nella FIRST Tech Challenge e nella FIRST Robotics Competition i protagonisti sono gli studenti delle scuole liceali di età compresa tra 14 e 18 anni. Le regole delle gare più impegnative cambiano di anno in anno mentre nelle prime fasce d'età i protagonisti sono i robot della Lego, tipicamente quelli



Vitale, uno dei quali scenderà in campo col robot **Nessie**, da noi sponsorizzato ed il cui progetto stiamo presentando all'interno di questa sezione della rivista. In questa categoria i robot devono seguire un percorso segnalato da una striscia disegnata sul terreno, striscia che in alcuni punti si interrompe e in altri è nascosta da ostacoli. In una prima fase il percorso si snoda su un terreno pianeggiante per poi imboccare una salita che porta ad una stanza dove è presente la vittima da recuperare; nel nostro caso questa è rappresentata da una lattina per bibite. Il robot la deve identificare, la deve agganciare con le sue pinze e la deve porre su una pedana presente in un angolo della stanza. Il tutto in maniera completamente autonoma, nel minor tempo possibile e con la strategia più adeguata atta a superare gli ostacoli, che possono essere aggirati o abbattuti. Tra le varie categorie questa è sicuramente quella che necessita di maggior capacità nella stesura del software ed anche quella che richiede grandi capacità dal punto di vista elettronico e meccanico, a meno di non utilizzare



dei modelli già pronti (tipicamente Lego Mindstorms NXT). Sempre a proposito di competizioni tra robot segnaliamo una particolare manifestazione che si è svolta di recente a Osaka, in Giappone: la **Robo Marathon**, una gara tra robot bipedi il cui intento è quello di mettere a punto e selezionare dei robot in grado di competere con gli umani sulla classica distanza dei 42,195 chilometri. Così come l'intento della RoboCup è quello di mettere a punto una squadra di calciatori-robot che possa sfidare i vincitori della Coppa del Mondo



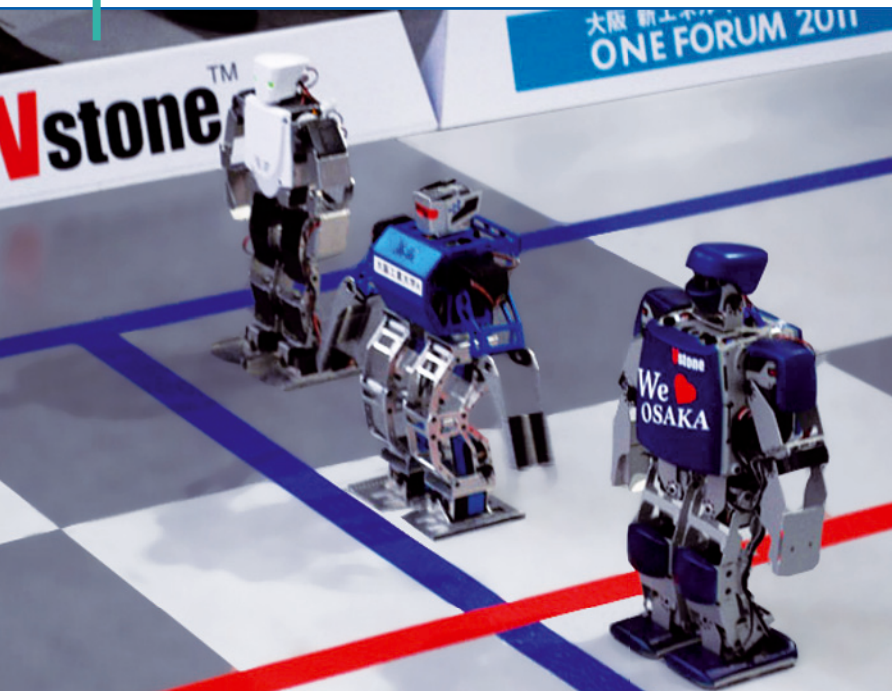
della tecnologia Lego Mindstorms. Alle gare prendono parte ogni anno decine di migliaia di studenti mentre la fondazione si finanzia con le sponsorizzazioni delle più importanti aziende americane.

Oltre alle competizioni, FIRST organizza anche corsi di formazione per insegnanti, conferenze, seminari e fornisce tutto il supporto possibile a quanti vogliono prendere parte a queste manifestazioni.

L'ormai ventennale esperienza ha consentito ai ricercatori di FIRST di verificare come tra i giovani partecipanti alle gare di robotica una percentuale superiore alla media degli studenti della stessa fascia d'età si

sia dedicata allo studio delle materie scientifiche e tecnologiche e, tra i più grandi, i risultati ottenuti durante la carriera universitaria siano stati superiori alla media. Non solo: una volta terminati gli studi, questi ragazzi si sono fatti strada anche nel mondo del lavoro andando ad occupare le posizioni di maggior prestigio.

Anche l'esperienza americana conferma dunque che la partecipazione a questo tipo di competizioni contribuisce a rafforzare non solo l'apprendimento delle materie scientifiche ma anche a formare dei giovani più motivati e caratterialmente meglio attrezzati ad affrontare le sfide della vita.



di Football, lo scopo della Robo Marathon è quello di realizzare un robot bipede che possa battere il più bravo maratoneta in carne ed ossa. Indubbiamente una sfida più facile di quella calcistica ma sicuramente ancora di là da venire; attualmente il divario tra umani e robot è ancora notevole dal momento che il vincitore di questa maratona ha impiegato circa 4 giorni per portare a termine il percorso contro poco più di 2 ore dei campioni in carne ed ossa. Anche in questo caso i robot debbono essere in grado di muoversi autonomamente seguendo il percorso (un anello di 100 metri da percorrere 423 volte) e, in caso di caduta, debbono rialzarsi da soli; l'unica assistenza ammessa riguarda la ricarica delle batterie, similmente ai maratoneti che durante le gare possono bere e nutrirsi.

La gara è stata organizzata dal comune di Osaka e dalla società **VStone**, nota per la

sua piattaforma di robot umanoide Robovie R3 e per una serie di kit robotici più piccoli, destinati agli hobbysti, come il Robovie-PC. VStone si augura che eventi del genere possano diffondersi in tutto il mondo per stimolare la ricerca in questo settore. Sempre a proposito di record di percorrenza, un altro robot giapponese - l'Evolta di Panasonic - ha percorso lo scorso novembre le 300 miglia che separano Tokyo da Kyoto stabilendo un primato difficilmente superabile.

Vogliamo infine segnalare questo mese un'iniziativa particolare denominata **RoboEarth**, (www.robearth.org), una rete Internet destinata esclusivamente ai robot.

Così come gli esseri umani utilizzano Internet per trarre vantaggio dalle esperienze e dalle conoscenze comuni, anche nel campo della robotica una rete Internet dedicata esclusivamente ai robot potrebbe consentire un miglioramento delle performance di queste macchine: una piattaforma open source che permetta a qualsiasi robot connesso a questa rete di inviare, condividere e riutilizzare i dati potrebbe facilitare l'apprendimento di compiti complessi in maniera completamente automatica.

È quanto si propone il progetto europeo RoboEarth al quale lavorerà nei prossimi quattro anni un team di 35 ricercatori. Il progetto si pone due obiettivi:

- velocizzare in maniera significativa l'apprendimento dei robot, in particolare di attività complesse;
- consentire l'esecuzione di compiti che non sono stati esplicitamente previsti in fase di progettazione.

Ma la visione che sta dietro RoboEarth è ancora più ampia: realizzare una piattaforma, metodi standard di comunicazione e un completo data-base che consenta a tutti i robot la codifica, lo scambio e il riutilizzo delle informazioni in un vicendevole scambio di esperienze che va oltre l'esternalizzazione della capacità di calcolo o l'utilizzo di controlli remoti.

A capo del progetto **RoboEarth** è il dottor Markus Waibel dell'Istituto Federale Svizzero per la Tecnologia di Zurigo al quale si affiancheranno ricercatori dell'Università di Saragozza, di Stoccarda, di Monaco di Baviera e di Eindhoven. □



UAV, LA VERSIONE 2.0

I dettagli della realizzazione del nostro UAV al quale, rispetto alla prima versione, abbiamo aggiunto la telemetria, un GPS più performante con 10 letture al secondo e un sensore di velocità a tubo di Pitot.

di ANDREA LETTIERI

Per concretizzare e poter “toccare con mano” gli argomenti trattati nel primo articolo (fascicolo n° 151) dedicato alla costruzione di un UAV, in questo secondo appuntamento descriviamo come è stato realizzato l'aeromodello, con gli accorgimenti e le modifiche apportate per installare tutta la componentistica aggiuntiva rispetto alla pura versione modellistica. Quella che viene descritta è la realizzazione del secondo prototipo di Easystar, in quanto il primo, dopo essere servito alla messa a punto del sistema ed aver ottenuto i primi risultati di volo stabilizzato e ritorno al punto di lancio in autonomia, è andato definitivamente distrutto dopo es-





sere precipitato (vedi foto) a causa, guarda caso, di un grossolano errore di pilotaggio. La parte elettrica ed elettronica si è salvata tutta (come accade quasi sempre) ed è stata reinstallata nel nuovo modello che, con l'occasione, abbiamo arricchito di un nuovo GPS MediaTek MT3329 GPS 10Hz+Adapter Basic ((Fig. 1) che Ardupilot è in grado di utilizzare a 10 Hz di campionamento con protocollo in binario a 38.400 bps, ed un modulo aggiuntivo che comprende un sensore di pressione con relativo tubo di Pitot per il rilevamento di velocità e altitudi-



Fig. 1
Il GPS a 10 Hz.

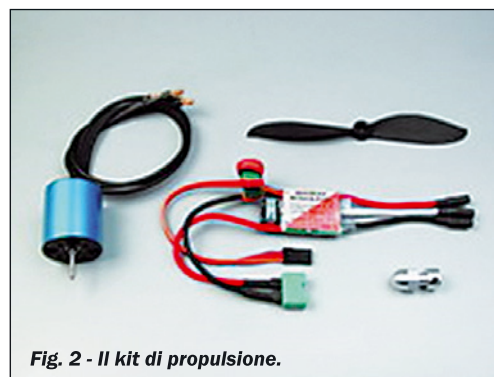


Fig. 2 - Il kit di propulsione.

ne, oltre ad un sistema di telemetria per la trasmissione dei dati di volo a terra.

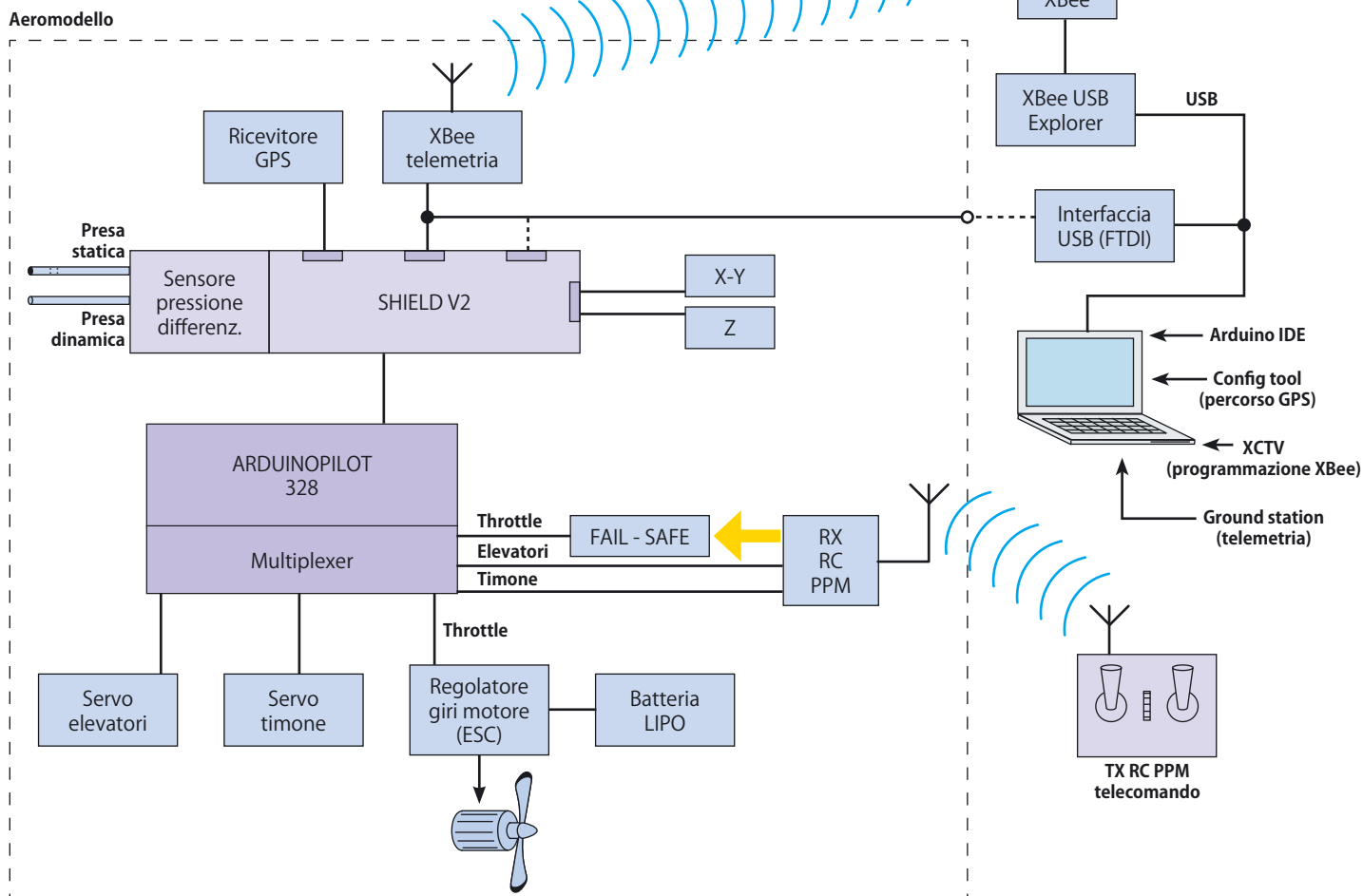
Qui vale la pena ribadire che durante il volo il modello deve essere sempre in vista e sotto il controllo diretto del pilota in tutte le configurazioni di volo e che solo con questi stringenti vincoli è ragionevole operare un modello così equipaggiato, chiaramente in luoghi destinati al volo di modelli, gli unici adatti a tale scopo.

Con questa filosofia è stata scelta la tipologia di componenti utilizzata per le nostre prove. Il software di controllo caricato sull'Ardupilot, un circuito basato sul processore Arduino e personalizzato per accogliere i connettori delle periferiche "modellistiche", è realizzato in modo che in tutte le configurazioni, l'intervento del pilota sugli stick della trasmittente sia sempre prioritario rispetto alle decisioni del sistema autopilota.

Ma andiamo con ordine: la scatola di montaggio dalla quale partire è la versione di Easystar "tutta da montare", in quanto permette di fare una serie di modifiche alla fusoliera prima di incollarla.

La scatola contiene un motore a spazzole con prestazioni non eccelse, che andremo a sostituire procurandoci il kit di motorizzazione brushless "EasyStar BL-TUNING" (prodotto dalla Multiplex e reperibile con il

Il sistema UAV



codice 333646) composto da (Fig. 2):

- Motore brushless Himax B 2810-2150;
- Regolatore BEC MULTIcont BL-20 S;
- Elica da 5,5x4,5 pollici con mozzo porta elica in alluminio.

Il tutto è completato da una batteria LiPo a 3 celle (11,1 V) da 2.000 mAh. La radio utilizzata è una Graupner MC24 a 40 MHz, in PPM, dotata di ricevente in doppia conversione, sempre di produzione Graupner. Il segnale PPM è quello meglio interpretato e gestito dal multiplexer e dal software dell'ArduPilot.

È ovviamente possibile utilizzare anche radio operanti su diverse frequenze (purché omologate); nel caso è opportuno consultare i blog del sito www.diydrones.com per conoscere i problemi di compatibilità e di configurazione del software.

Diydrones è uno dei siti di riferimento delle comunità che lavorano attorno ad alcuni progetti di UAV amatoriali, tra i quali l'ArduPilot; sempre nello stesso sito, sono disponibili i link alle versioni del software di base, dei software di test e configurazione e della stazione di telemetria.

Per la nostra configurazione sono necessari i seguenti componenti hardware:

- il modulo ArduPilot - Arduino Compatible UAV Controller w/ ATmega328 (cod. Sparkfun GPS-08785);
- il kit Vshield (ArduPilot Shield V2 Kit) che integra il sensore di pressione e semplifica il collegamento delle termofili che consentono la stabilizzazione (modulo XY e modulo Z); quest'ultimo è stato recentemente sostituito con un modulo IMU a sei gradi di libertà che esploreremo in una futura versione;

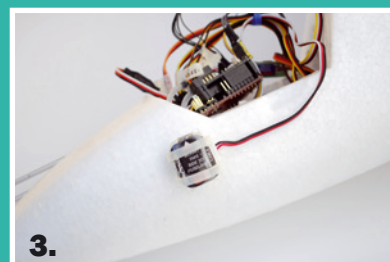
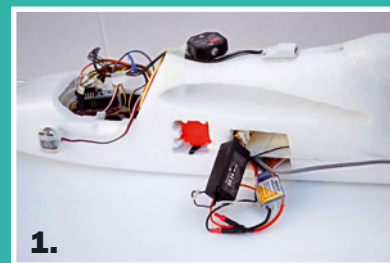
- un sensore basato su termofile per gli assi XY ed uno per l'asse Z; attualmente questi sensori sono sostituiti da un modulo IMU (Inertial Measurement Unit) costituito da tre accelerometri e altrettanti giroscopi (ArduIMU+ V2 - Flat) per il quale si anticipano le note di montaggio in questo articolo, rimandando la descrizione della teoria di funzionamento al prossimo;
- modulo GPS (MediaTek MT3329 GPS 10Hz+Adapter Basic) con relativo cavo di prolunga da 25 cm (EM-406/uBlox/MTK Adapter Cable 25 cm);
- modulo FTDI USB (FTDI Basic Breakout - 3,3 V cod. Sparkfun DEV-09873) per la programmazione e configurazione dell'ArduPilot con il PC;
- modulo USB per la telemetria lato PC (XBee Explorer USB cod. Sparkfun: WRL-08687);
- modulo FTDI per la telemetria lato aeromodello (XtreamBee Board);
- due moduli XBee PRO, lato PC e lato modello, per la trasmissione e ricezione della telemetria (XBee Pro 60 mW - Wire Antenna cod. Sparkfun WRL-08742); sono possibili altre combinazioni di moduli XBee su frequenze diverse e con antenna cablata o esterna; unica accortezza è che i due moduli devono essere compatibili tra di loro;
- un modulo Intelligent Failsafe (Fusion FS-01) da interporre tra la ricevente e il canale Throttle dell'ArduPilot, in modo da poter gestire da software la condizione di perdita del segnale radio.

Nella foto in queste pagine sono visibili le modifiche fatte alla fusoliera (prima di incollarla) per installare in modo razionale l'elettronica di bordo:

- nella semifusoliera di sinistra è stata aperta una finestra per alloggiare la ricevente ed il regolatore del motore brushless;
- all'interno della stessa semifusoliera è stato sistemato ed incollato un bowden per farvi passare l'antenna della radio, in quanto, con la nuova collocazione della ricevente, non è più raggiungibile il bowden previsto originariamente sotto la fusoliera;

Modifichiamo la fusoliera

Per installare l'avionica di bordo del nostro UAV dobbiamo apportare alcune modifiche alla struttura dell'Easystar che si trova in commercio, ovvero aprire una finestra nella metà di sinistra per introdurre il regolatore di velocità del brushless e l'unità ricevente del radiocomando (1) e, sempre nella stessa semifusoliera, sistemare un bowden da cui far passare l'antenna ricevente (2). Poi occorre realizzare ed incollare al fianco della fusoliera il sostegno (perfettamente perpendicolare alle ali) del sensore dell'asse Z (3) e costruire e disporre altri due piani, uno (parallelo alle ali) per il sensore delle direzioni sugli assi X e Y ed un altro per il ricevitore GPS, posti entrambi nella parte alta della fusoliera (4).

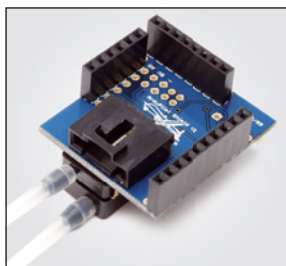


- sono state allargate le scanalature per il passaggio dei cavi del motore;
- sono stati allargati il più possibile i passaggi dei cavi dalla ricevente al modulo ArduPilot e da questo ai servocomandi;
- è stato realizzato un supporto con rinforzo per il sensore sull'asse Z;
- sono stati realizzati (questi dopo la chiusura e incollaggio della fusoliera) tre pianetti per alloggiare il sensore degli assi XY, il modulo GPS ed il modulo della telemetria; bisogna prestare la massima attenzione al fatto che il pianetto del sensore XY sia perfettamente orizzontale rispetto all'assetto di volo del modello perché è rispetto a questo riferimento che agirà il software delegato alla stabilizzazione automatica.

Per i rinforzi e la basetta che supporta la



Fig. 3 - Lo shield V2 per Ardupilot.



**Fig. 4
Le connessioni
dell'Ardupilot.**

ricevente ed i pianetti sono stati utilizzati dei pezzetti di canalina da elettricista, molto facile da tagliare, leggera e che si incolla benissimo con la colla ciano specifica per l'EPP, materiale antiurto (fino ad un certo punto...) con il quale è formato il modello.

Ultime attenzioni prima di chiudere i due semigusci: il motore deve essere incollato con attenzione sulla semi fusoliera con la scanalatura per i cavi (attenzione a non fare entrare colla ciano all'interno del motore) e tenere i cavi in posizione con un pezzetto di carta adesiva da mascheratura. Da questo punto in poi il modello viene completato come da istruzioni del costruttore, facendo particolare attenzione agli allineamenti dei piani di coda: più sono accurate le geometrie delle superfici aerodinamiche e meglio volerà il modello. Provate e riprovate a secco, perché una volta incollato non potrete rimediare.

Una volta completato il modello, fate molta attenzione al corretto posizionamento del baricentro, che si ottiene collocando in modo opportuno la batteria all'interno della fusoliera.

Una volta terminata la cellula si inizia ad assemblare e ad installare l'avionica.

Armati di pazienza e saldatore si comincia ad assemblare il complesso formato da Ardupilot e modulo Vshield; conviene prima saldare il connettore per i sensori XY e Z e successivamente, come suggerito

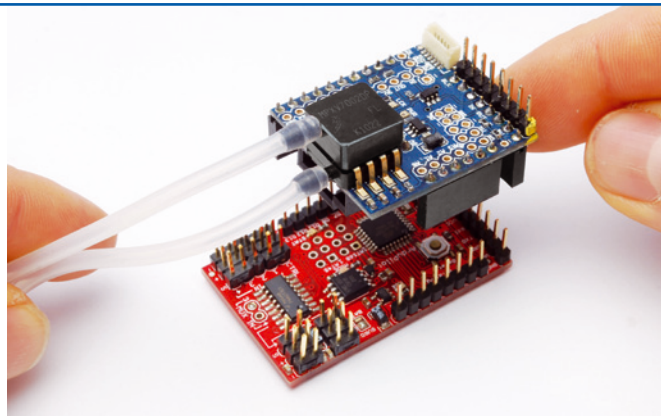
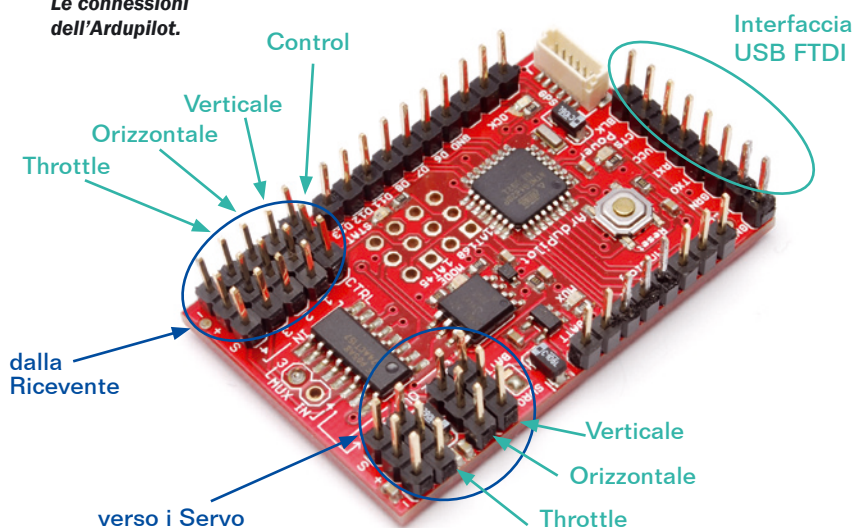


Fig. 5 - L'insieme Ardupilot + shield.

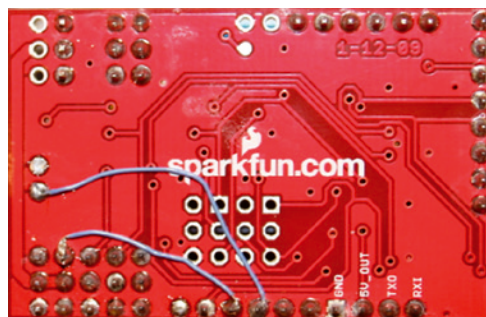


Fig. 6 - I cavallotti da realizzare sotto il modulo Ardupilot.

nel wiki di documentazione (<http://code.google.com/p/ardupilot/wiki/ArduPilot>), tagliare e posizionare le strip di pin assemblando i moduli a secco, quindi verificare e riverificare, tenendo i moduli in posizione (...evviva la carta da mascheratura!); fatto ciò, puntate con il saldatore i pin sugli angoli per rendere solido l'assemblaggio e poi saldate tutti gli altri.

Saldare i pin dei servocomandi e la strip sul modulo Vshield, infine i due cavallotti sotto l'Ardupilot (Fig. 6); questi ultimi non servono a rimediare a una dimenticanza del progettista, ma sono una modifica che permette di gestire diverse configurazioni hardware e software con la stessa basetta. Recuperate o realizzate delle prolunghine dei cavi dei servocomandi con spinette femmina lunghe circa venticinque centimetri (quelle incluse nel kit dello Vshield sono troppo corte per la nostra configurazione).

Inserite le prolunghine nei canali elevatore, direzionale e nel canale scelto per l'interruttore a tre posizioni che permette di cambiare le modalità di funzionamento dell'Ardupilot; conviene marcare ciascuna con dei segnacavo (pezzetti di carta da

mascheratura) su entrambe le terminazioni prima che la situazione diventi piuttosto ingarbugliata.

Fate passare il cavetto del throttle (controllo della velocità) dall'alloggiamento del regolatore a quello della ricevente. Infine, convogliate tutti i cavi verso Ardupilot e la batteria dal passaggio in fondo al muso, quindi collegate i cavi della ricevente e dei servo al modulo Ardupilot.

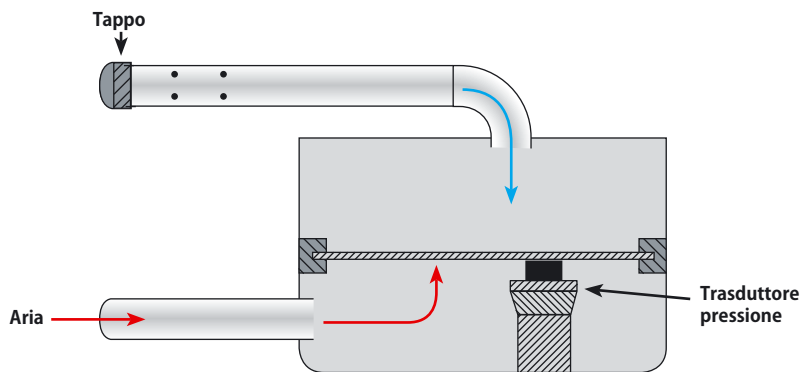
A seconda del complesso trasmittente/ricevente in vostro possesso, dovete procedere a descrivere il modello in termini di:

- tipologia di modello (Aereo);
- associazione degli stick ai comandi; nel nostro caso il throttle allo stick verticale di destra, l'elevatore allo stick verticale di sinistra ed il timone allo stick orizzontale di destra.

Infine dovete assegnare un interruttore a tre posizioni ad un canale che corrisponderà all'uscita della ricevente verso l'ingresso CTRL del modulo Ardupilot.

Nelle fotografie del prototipo vedete i moduli GPS e telemetria (XBee) racchiusi in un involucro di termorestringente, in modo da non attaccare il biadesivo direttamente al modulo, evitare contatti elettrici indesiderati ed aumentare la resistenza meccanica del tutto; l'unica accortezza richiesta consiste nel non esagerare con il riscaldamento del termorestringente, altrimenti potrete danneggiare l'elettronica.

Con biadesivo e fascette installate i sensori ed il modulo GPS, quindi collegate i



Il sensore di velocità a tubo di Pitot effettua la differenza tra la pressione esercitata dall'aria prelevata da un tubo disposto nella direzione del movimento e quella da fermo dovuta all'altitudine.

cavi di questi ultimi ed installate i tubi di Pitot come visibile nella foto.

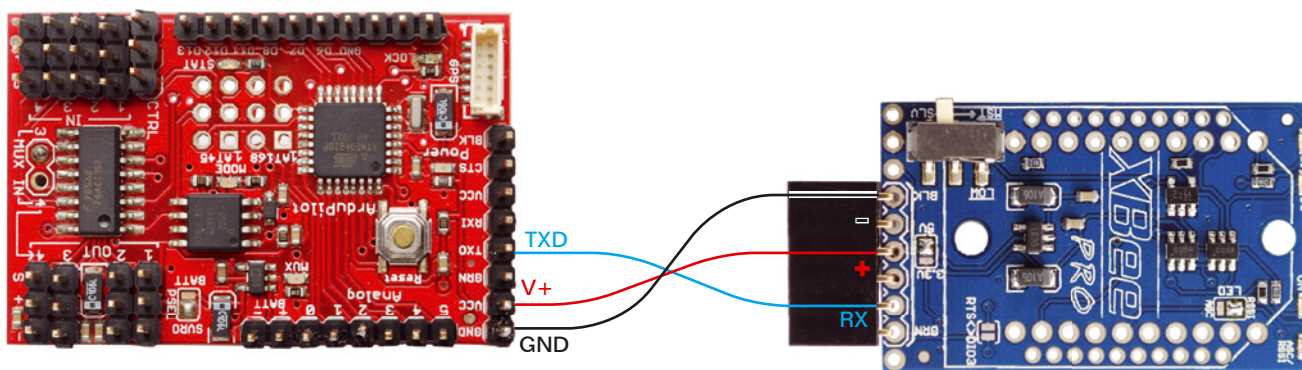
Rimandando ad un momento successivo il montaggio della telemetria, possiamo dedicarci alla parte software e vedere se tutto il complesso dà i primi segni di vita.

Per prima cosa bisogna procurarsi l'IDE di sviluppo di Arduino, disponibile gratuitamente dal sito omonimo (<http://arduino.cc/en/Main/Software>) ed installarlo. Poi occorre scaricare ed installare il driver per il modulo USB dell'FTDI, dal sito (<http://www.ftdichip.com/Drivers/VCP.htm>). Infine, scaricate il software di controllo di Ardupilot dal sito <http://code.google.com/p/ardupilot/downloads/list>; al momento della scrittura di questo articolo è considerata stabile la versione 2.7.1.

Scaricare dal medesimo indirizzo la Utility Config Tool Global, che ci permetterà di caricare il percorso GPS per la modalità AUTO del software di controllo.

Inizialmente modificate il file *Config.h*, come mostrato nel **Listato 1**. A tale proposito facciamo un cenno ai parametri `#define POSITION_1`, `#define POSITION_2` e `#define POSITION_3` del listato; come è intuibile, sono quelli che configurano le

Ecco come collegare al modulo Ardupilot la scheda XBee Pro per la telemetria.



Listato 1

```

/* ArduPilot 2.7.1 Header file : July 23, 2010 */
/* These Flight Modes can be changed either here or directly in events.pde

MANUAL          = Full manual control via the mux - cannot be overridden
CIRCLE          = Fly in a stabilized 'dumb' circle - for planes with no GPS
STABILIZE       = Try and maintain level flight
FLY_BY_WIRE_A   = Autopilot style control via user input, manual throttle
FLY_BY_WIRE_B   = Autopilot style control via user input, aisppeed controlled with throttle
AUTO            = What you bought the thing for
RTL             = Come home - Home is set at first good GPS read from ground startup
LOITER          = Circle around current location
TAKEOFF         = full throttle with desired pitch, stabilized roll
LAND            = You must specify the landing waypoints in this header - read the manual

*/
#define POSITION_1 MANUAL // the default, don't bother changing.
#define POSITION_2 FLY_BY_WIRE_A
#define POSITION_3 RTL
// So why isn't AUTO here by default? Well, please try and run Stabilize first,
// then FLY_BY_WIRE_A to verify you have good gains set up correctly
// before you try Auto and wreck your plane. I'll sleep better that way...

// Debug options - set only one of these options to 1 at a time, set the others to 0
#define DEBUG_SUBSYSTEM 0 // 0 = no debug
// 1 = Debug the Radio input
// 2 = Debug the Servo output
// 3 = Debug the Sensor input
// 4 = Debug the GPS input
// 5 = Debug the GPS input - RAW HEX OUTPUT
// 6 = Debug the IMU
// 7 = Debug the Control Switch
// 8 = Debug the Throttle
// 9 = Radio Min Max values

/*****/
//HARDWARE CONFIGURATION
#define SHIELD_VERSION 1 // Old (red) shield versions is 0, the new (blue) shield version is 1, -1 = no shield
#define AIRSPEED_SENSOR 1 // (boolean) Do you have an airspeed sensor attached? 1 = yes, 0 = no.

#define GPS_PROTOCOL 4 // 0 = NMEA
// 1 = SIRC,
// 2 = uBlox
// 3 = ArduIMU
// 4 = MediaTek,
// 5 = Simulated GPS mode (Debug),
// -1 = no GPS

//Ground Control Station:
#define GCS_PROTOCOL 0 // 0 = Standard ArduPilot (LabVIEW/HappyKillmore),
// 1 = special test,
// 2 = Ardupilot Binary(not implemented),
// 3 = Xplane
// 5 = Jason's GCS,
// -1 = no GCS (no telemetry output)

/*****/
//Thermopile sensors:
#define ENABLE_Z_SENSOR 1 // 0 = no Z sensor, 1 = use Z sensor (no Z requires field calibration with each flight)
#define XY_SENSOR_LOCATION 0 // XY Thermopiles Sensor placement
// Mounted right side up: 0 = cable in front, 1 = cable behind
// Mounted upside down: 2 = cable in front, 3 = cable behind

```

diverse modalità di funzionamento del software di controllo dell'ArduPilot e che si attivano a seconda della posizione dell'interruttore CTRL sulla trasmittente del radiocomando. A ciascun parametro cor-

risponde una posizione dell'interruttore; la prima va lasciata tassativamente su MANUAL, mentre le altre due possono essere scelte in base alle funzionalità che si vogliono ottenere e che analizzeremo più ap-

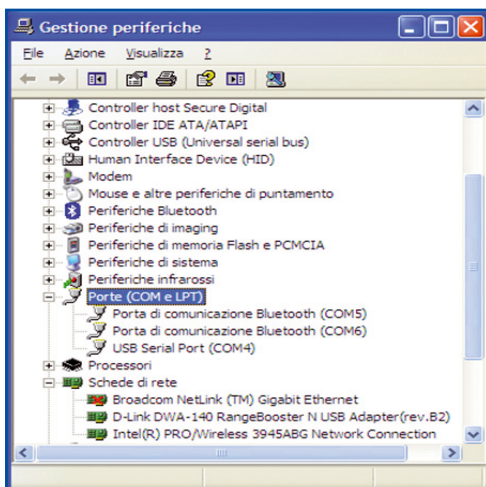
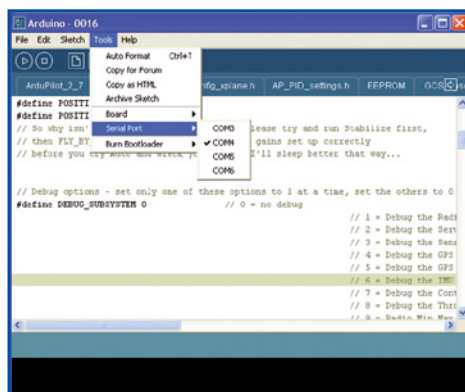


Fig. 7 – Se non ricordate la porta COM cui è collegato Ardupilot, sfogliate Gestione periferiche.



**Fig. 8
Nell'interfaccia grafica dell'IDE, impostate la COM del vostro computer cui è collegato Ardupilot.**

profonditamente nelle prossime puntate. A questo punto possiamo fare una prova di compilazione del software, che non dovrebbe dare errori; allo scopo, collegate l'FTDI con un cavetto (realizzato ad hoc) al modulo Ardupilot e con una prolunga USB al PC. Se necessario, configurate la porta COM generata dall'FTDI e settate i parametri di connessione dell'IDE con la stessa porta.

Disconnettete momentaneamente il modulo GPS, alimentate Ardupilot con la batteria del modello (nelle foto vedete un cavo a Y che permette varie configurazioni di uso del throttle); per le prime prove è meglio avere l'accortezza di non montare l'elica, che non si sa mai: se ci fosse qualche problema, il motore potrebbe avviarsi accidentalmente e fare danni!

Eseguite l'Upload del software da IDE e, al termine, ricollegate il cavetto del GPS e, dopo qualche secondo di "lucine lampeggianti", il modello dovrebbe animarsi muovendo le superfici di coda e i due LED blu (uno su Ardupilot ed uno sul GPS) dovrebbero cominciare a lampeggiare: è il segno che Ardupilot sta cercando di ottenere un segnale GPS valido.

Selezionando il Serial Monitor sull'IDE e cambiando la velocità della porta in 38.400 bps, dovremmo vedere una sequenza di messaggi scorrere nella parte bassa dello schermo (Fig. 9): è il protocollo di telemetria che fa il suo dovere.

Notate che il modulo Ardupilot possiede una sola porta seriale, la quale viene condivisa dal GPS e dalla comunicazione con l'IDE e la telemetria. Ecco perché è necessario disconnettere il GPS durante l'Upload del software, onde evitare interferenze

tra i messaggi. Una volta in funzionamento normale, la porta viene impostata con la velocità di comunicazione del GPS; se sul serial monitor vedete una sequenza interminabile di quadratini e faccine, con molta probabilità è sufficiente modificare la velocità di trasmissione e tutto dovrebbe sistemarsi.

Se siamo arrivati con successo a questo punto, è opportuno realizzare e configurare anche la telemetria, in modo da liberare il modello dal cavo che lo vincola al PC e poter salvare i messaggi a scopo di debug, sia in laboratorio che in fase di volo.

Con la telemetria installata e funzionante sarà possibile, per verificare il funzionamento di base del sistema, stando in laboratorio, portare il modello sul balcone per vedere se acquisisce il segnale GPS

e, in caso positivo, vedere apparire la posizione nella quale ci troviamo, nella finestra di Google Earth della telemetria.

Ancora una volta procediamo con ordine: per rendere funzionante la telemetria è necessario eseguire un serie di passi, il primo dei quali è collegare al PC con il cavetto USB il modulo XBee Explorer USB; se stiamo usando lo stesso PC che usiamo per la programmazione dell'Ardupilot, il modulo dovrebbe essere riconosciuto senza problemi, in quanto entrambi usano lo stesso integrato FT232RL per l'interfaccia USB/seriale.

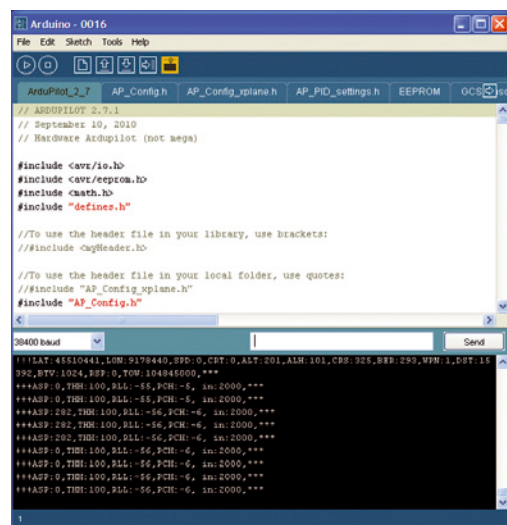


Fig. 9

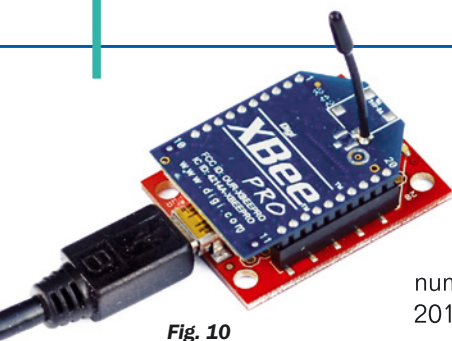


Fig. 10

Poi bisogna recuperare ed installare il software X-CTU per configurare i moduli XBee e, come descritto nel numero 152 di Elettronica In (Dicembre 2010, Gennaio 2011) alla pag. 141 e seguenti, configurare i due moduli XBee in modo da farli comunicare tra loro.

In particolare, per ciascuno dei due moduli XBee bisogna:

- selezionare il tipo di modulo XBee che si vuole inizializzare (si può leggere sul rovescio del modulo stesso);
- modificare la velocità di trasmissione portandola a 38.400 bps;
- leggere i dati dal modulo (pulsante READ);
- impostare un PAN-ID a piacere, ma rigorosamente identico per i due moduli;
- inizializzare il modulo XBee (botone WRITE).

A questo punto occorre lasciare uno dei due moduli XBee installato sul modulo XBee Explorer USB, quindi realizzare un cavo di collegamento tra l'uscita FTDI del modulo Ardupilot ed il modulo XstreamBee Board, come illustrato nella Fig. 10.

Installate, ora, il modulo XBee sulla scheda XstreamBee Board. A questo punto è necessario scaricare ed installare il software di telemetria, sempre dal sito Diydrones, alla voce *Ground Station* sono reperibili due applicazioni compatibili con Ardupilot: la *HappyKillmore GCS* e la *Original Ardupilot GCS*.

La prima Ground Station funziona in ambiente Microsoft dot.Net 2.0 e richiede il

runtime Direct-X e Google Earth. Il tutto è scaricabile gratuitamente sotto varie licenze free. Per funzionare bene è sufficiente un Windows XP, con Service Pack 2 installato e abbastanza aggiornato, e con installato MS dot.Net 2.0. In caso si verificassero errori di funzionamento della Ground Station, fate aggiornare (da *Aggiornamenti del computer*) il framework dot.Net e tutto dovrebbe sistemarsi.

La Ground Station Happy Killmore nasce da un progetto Open Source con l'ambizione di coprire diversi sistemi di autopilota ed è in costante evoluzione, pur essendo già sufficientemente matura per i nostri scopi. Per aggiornarla è sufficiente scaricare la nuova versione e ...installarla. La seconda, Original Ardupilot GCS, è costruita utilizzando l'ambiente di sviluppo LabVIEW ed è particolarmente interessante per chi ha seguito il corso su LabVIEW presentato nella nostra rivista. Per installarla è necessario l'ambiente di sviluppo LabVIEW, oppure il Labview Runtime Engine con i relativi driver per la gestione delle porte seriali, oltre che il già citato Google Earth.

In questo articolo faremo riferimento alla Happy Killmore GCS, mentre lasceremo la descrizione della versione LabVIEW ad un prossimo articolo.

Dopo aver verificato quale porta COM è stata assegnata al nostro modulo XBee Explorer USB, apriamo la Ground Station, scegliamo la porta COM corrispondente, la velocità di comunicazione e facciamo clic sul pulsante *Connetti* (Fig. 11).

Nella finestra a sinistra vengono presentati gli strumenti che rappresentano in modo analogico le grandezze trasmesse dalla telemetria, mentre la finestra a destra dovrebbe contenere la rappresentazione del globo terrestre così come presentato da Google Earth.

A questo punto controlliamo che l'interruttore che controlla la modalità di funzionamento sia in posizione MANUAL, accendiamo la trasmettente del radiocomando (sempre prima la trasmettente della ricevente) poi colleghiamo la batteria per alimentare l'elettronica del modello.

Se tutto funziona correttamente vedremo le lancette cominciare a muoversi; sce-

Fig. 11 - Il pannello di controllo della Ground Station Happy Killmore.

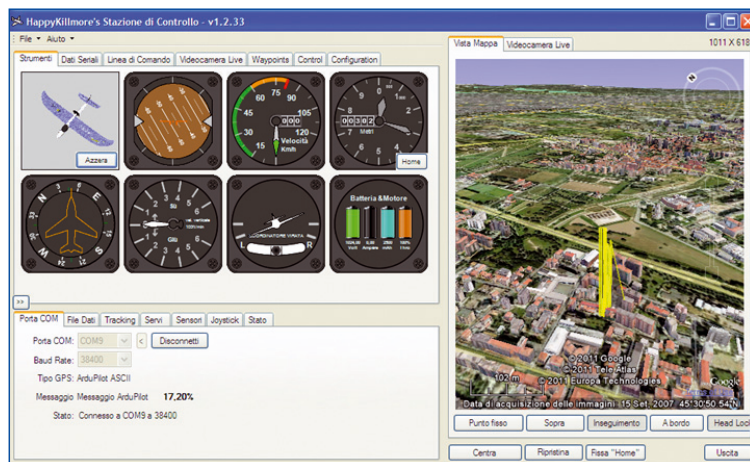


Fig. 12 – I messaggi trasmessi dall'Ardupilot arrivano alla Ground Station Happy Killmore.

gliando la linguetta Dati Seriali vedremo scorrere i messaggi nativi così come trasmessi dall'Ardupilot (Fig. 12).

Sul modello lampeggiano i LED blu del modulo Ardupilot e del GPS, a significare che il sistema è in attesa di un segnale GPS valido.

Per verificare il funzionamento del GPS si può portare il modello in un luogo dove possa essere ricevuto il segnale GPS, anche su un balcone, e dopo qualche minuto i due LED dovrebbero smettere di lampeggiare (rimanere blu) e sulla Ground Station dovremmo vedere la parte Google Earth posizionarsi nel punto della Terra dove ci troviamo.

Se siamo in un luogo aperto, tipo un prato, e proviamo a camminare intorno, vedremo la nostra passeggiata tracciata sullo schermo della Ground Station.

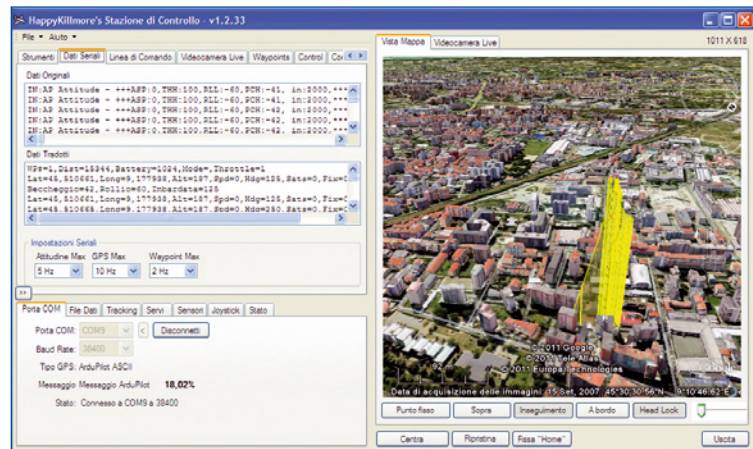
Ora è necessario verificare che i comandi impartiti agli stick producano gli effetti voluti sul modello; per prima cosa facciamo la prova motore: muovendo lo stick dal basso verso l'alto il motore deve girare progressivamente più velocemente.

Se non abbiamo ancora montato l'elica, evitiamo di portare lo stick a fine corsa, onde evitare di sollecitare troppo il motore a vuoto.

Lo spostamento a destra dello stick destro deve produrre un piegamento verso destra del timone di direzione, guardando il modello da dietro; viceversa a sinistra.

Lo spostamento verso il basso dello stick sinistro deve produrre un movimento verso l'alto del timone di profondità e viceversa con il movimento opposto. In caso di comportamento diverso da quello descritto è necessario configurare l'inversione di funzionamento del canale sulla trasmettente che aziona il servo in questione.

A questo punto è necessario verificare il corretto intervento del software di controllo della stabilizzazione sui servo; posizionate l'interruttore CTRL sulla modalità FLY_BY_WIRE_A e ruotate il modello sull'asse trasversale (beccheggio): puntando il muso verso il basso, per reazione il software di stabilizzazione deve fare muovere il piano orizzontale del timone verso l'alto (viceversa puntando il muso verso l'alto). Obiettivo del software di stabilizza-



zione è riportare il modello in volo livellato orizzontale ed i suoi interventi sono tesi a contrastare le azioni che perturbano la condizione ottimale.

Più violenta è la perturbazione e più incisiva sarà l'azione di contrasto, come descritto nell'articolo introduttivo nel numero 151 Novembre 2010 di Elettronica In.

Stessa verifica va fatta ruotando il modello sull'asse longitudinale (rollio); questa volta, abbassando l'ala sinistra il timone verticale deve rispondere muovendosi verso destra e viceversa abbassando l'ala destra.

In caso di risposta contraria a quanto desiderato, devono essere modificate le impostazioni nel modulo *Config.h* del software di controllo, applicando queste:

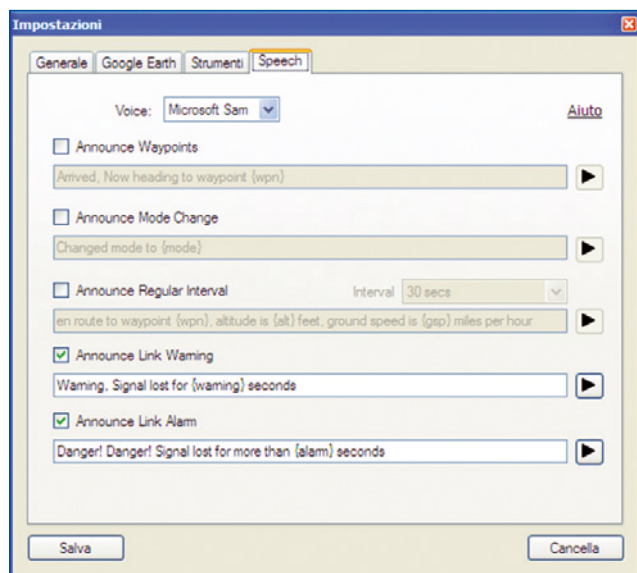
```
#define REVERSE_ROLL 1           // To reverse roll, PUT -1 to reverse it
#define REVERSE_PITCH 1        // To reverse pitch, PUT -1 to reverse it
```

Dopo ogni modifica ricordate di ricompilare e rieseguire l'upload del software.

Prima di ricollegare l'Ardupilot al modulo USB FTDI, ricordatevi di riportare l'interruttore CTRL in posizione NORMAL e di scollegare il modulo GPS; fate molta attenzione a collegare e scollegare il GPS, perché i contatti sono molto delicati e un eventuale danno risulterebbe molto difficile da diagnosticare e rimediare.

Eseguite queste verifiche con la massima scrupolosità e provate e riprovate fino ad avere l'assoluta certezza che tutto funzioni correttamente; qualsiasi errore di comportamento in questa modalità porterà irrimediabilmente ad un crash clamoroso senza possibilità di controllo, infatti è assolutamente impossibile pilotare un modello con i comandi che non rispondono nel

Fig. 13



clic sui pulsanti della finestra principale del software Ground Station, possiamo accedere ad una ricca serie di funzionalità che ci permettono di approfondire il funzionamento e la configurazione del nostro sistema; in particolare possiamo (Fig. 13):

- verificare il corretto funzionamento del modello (ciò si rivela un utilissimo strumento di debug in caso di anomalie);
- salvare i dati di telemetria per successivi utilizzi, come ad esempio rivedere ed analizzare il volo;
- integrare una telecamera con sistema di antenna tracking;
- verificare lo stato del volo in termini di distanza dai waypoint, modalità di funzionamento e stato GPS;
- Attivare allarmi con segnalazione di condizioni anomale (File,>Impostazioni>Generale+Speech);
- Attivare messaggi vocali relativamente alle condizioni di navigazione ed istato del sistema (File,>Impostazioni>Speech).

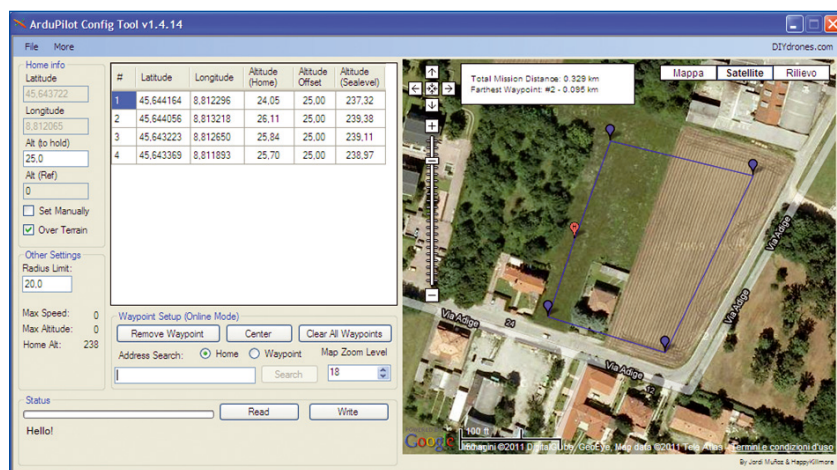


Fig. 14 - La schermata di lavoro dell'applicazione ArduPilotConfigTool.

modo normalmente atteso dal pilota. Infine verifichiamo, sempre in modalità FLY_BY_WIRE_A, che l'intervento del pilota sugli stick sia prioritario rispetto alle funzioni di auto stabilizzazione: agendo sugli stick, in qualunque assetto, i piani di coda devono muoversi coerentemente con gli input del pilota, per poi riprendere il controllo una volta rilasciati gli stick (stick al centro).

In caso di comportamenti non corretti, come risposte inverse ai comandi del pilota, riverificate pazientemente tutto da capo fino a trovare la corretta combinazione di settaggi sulla trasmettente e nel software del modulo Ardupilot.

Non abbiate assolutamente fretta di provare in volo il modello: verificate a terra fino alla nausea e poi ... riverificate ancora. Selezionando le varie linguette e facendo

Dopo l'accensione del sistema, il primo punto GPS valido acquisito viene memorizzato come HOME: questo è il punto che viene utilizzato dalla modalità RTL (Return To Launch) per riportare il modello a casa e metterlo in una rotta circolare sopra lo stesso punto. Altri punti GPS possono essere caricati nella EEPROM del modulo Ardupilot per essere utilizzati come altrettante boe di un percorso nella modalità di volo AUTO.

Per fare questo si utilizza l'applicazione ArdupilotConfigTool (Fig. 14) che si scarica sempre dal sito <http://code.google.com/p/ardupilot/downloads/list>; il file corrispondente è ConfigToolGlobal.

Installata e lanciata, permette di creare un percorso di punti selezionando direttamente le posizioni dalla mappa di Google-Earth nella finestra a destra e poi, dopo aver collegato il modulo Ardupilot al PC come per l'upload dei programmi, di scaricare il percorso nello stesso, con il pulsante WRITE.

UTILIZZIAMO L'IMU

Dato che i sensori basati su termofile sono attualmente abbastanza difficili da reperire, concludiamo l'articolo con alcune indi-

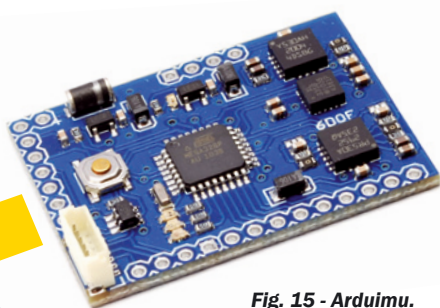


Fig. 15 - Arduimu.

cazioni su come utilizzare, al loro posto, un IMU con tre accelerometri e tre giroscopi, e più precisamente il modulo Arduimu, specificamente realizzato per essere integrato con l'Ardupilot.

L'Arduimu va montato perfettamente orizzontale e orientato come mostrato nella Fig. 15; la sua collocazione ideale è sul pianetto previsto per il sensore XY, davanti al modulo GPS.

Anche l'Arduimu ha un suo software di gestione utilizzabile con l'IDE di Arduino; questo software è disponibile alla pagina web <http://code.google.com/p/arduimu/downloads/list>, dove sono disponibili anche manuali e utility.

Dopo averlo installato, aprite il software e modificate i parametri nel modulo di configurazione, secondo quanto mostrato nel Listato 2. Ora dovete collegare il modulo Ardupilot al PC ed eseguire il download del software, quindi realizzare i cablaggi come indicato nella Fig. 16 ed installare il tutto a bordo del vostro aeromodello.

Per il corretto funzionamento del modulo IMU, è necessario che il modello sia in posizione orizzontale come se fosse in volo livellato e che venga mantenuto immobile fino a quando l'IMU stesso non si sia correttamente calibrato. All'accensione del modello, sempre con l'interruttore CTRL in posizione NORMAL, i tre LED di colore rosso, blu e giallo lampeggiano, segnalando che il modulo IMU è in fase di calibrazione. Ripetiamo che il modello non deve essere assolutamente mosso durante tutta questa fase, ovvero fin quando il LED giallo smette di lampeggiare e rimane acceso in continuazione; solo allora, il modulo è correttamente calibrato.

I LED blu sul modulo IMU e sul ricevitore GPS possono continuare a lampeggiare

Listato 2

```
// *** NOTE! To use ArduIMU with ArduPilot you must select binary output
messages
#define PRINT_BINARY 1 //Will print binary message and suppress ASCII
messages (above)
```

```
// GPS Type Selection - Note Ublox or MediaTek is recommended. Support
for NMEA is limited.
```

```
#define GPS_PROTOCOL 4 // 1 - NMEA, 2 - EM406, 3 - Ublox, 4 --
MediaTek
```

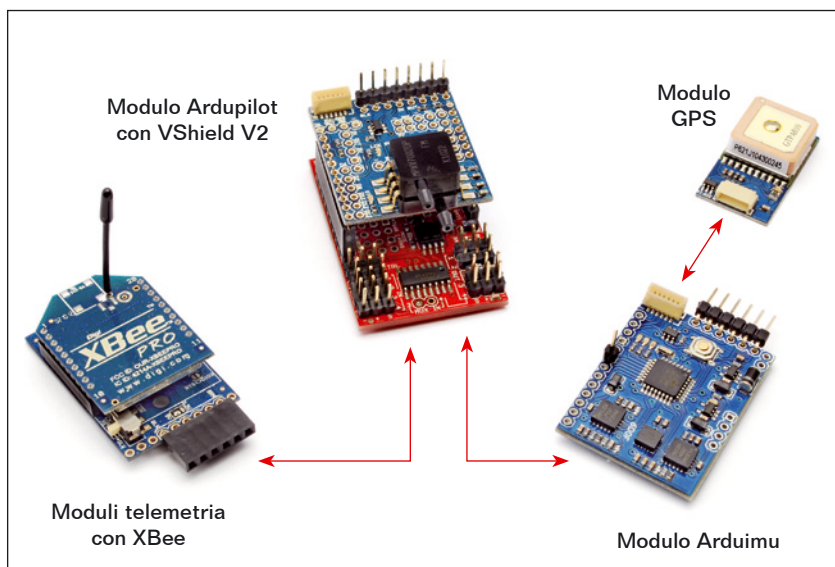
Collegare l'Arduimu al PC con il modulo USB FTDI ed eseguire l'upload del software.

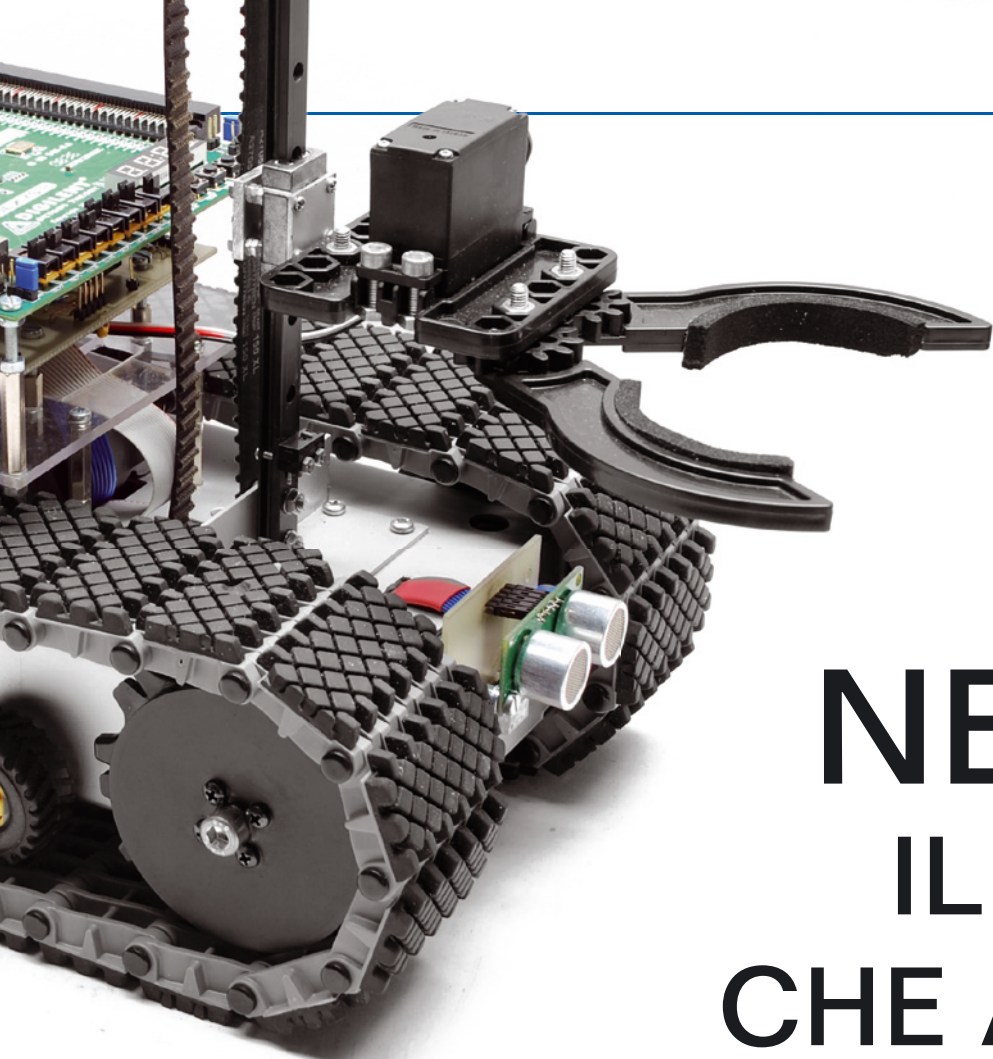
Aprire il software dell'Ardupilot e, nel modulo Config.h modificare il parametro GPS_PROTOCOL come segue.

```
#define GPS_PROTOCOL 3 // 0 = NMEA
// 1 = SIRF,
// 2 = uBlox
// 3 = ArduIMU
// 4 = MediaTek,
// 5 = Simulated GPS mode (Debug),
// -1 = no GPS
```

fino a quando non viene ricevuto un segnale GPS valido, dopo di che restano accesi permanentemente. L'intera operazione può durare anche più di quindici minuti, in base alle condizioni di ricezione ed al periodo di inattività del ricevitore GPS (in cold start occorre un tempo significativo). A questo punto, spostando la levetta dell'interruttore in modalità FLY_BY_WIRE_A possono essere eseguiti tutti i controlli descritti per la configurazione con i sensori termofile. □

Fig. 16 - I componenti dell'avionica nel caso di utilizzo dell'Arduimu, un IMU con tre accelerometri e tre giroscopi.





Proseguiamo l'analisi del progetto del robot, descrivendo le scelte progettuali della versione che parteciperà alla Robocup di Catania.

NESSIE, IL NUOVO CHE AVANZA

Il mese scorso vi abbiamo presentato il modello di Robot "rescue", realizzato nel Laboratorio di Robotica Didattica dell'ITCS Erasmo da Rotterdam di Bollate, che ha partecipato alla manifestazione di Vicenza 2010 della Robocup Jr. Adesso è giunto il momento di affrontare e spiegare le soluzioni tecniche adottate per cercare di risolvere alcuni dei principali problemi riscontrati in precedenza, diventate le linee guida nella preparazione del nuovo modello -chiamato Nessie- che sarà presentato alla manifestazione di Catania, nei giorni dal 14 al 16 di aprile prossimo venturo.

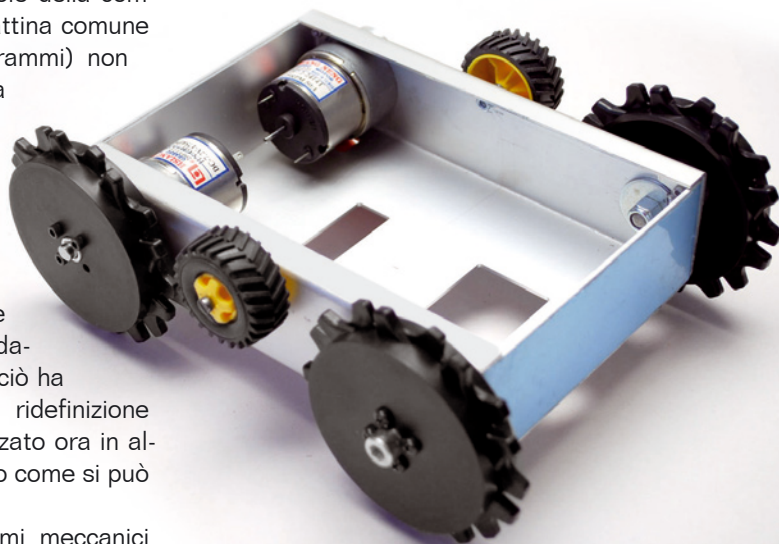
Lo scorso anno all'ITCS avevano progettato, meccanicamente parlando, una sorta di "gigante dai piedi d'argilla", nel senso che i materiali usati e le dimensioni del robot hanno determinato un peso

troppo impegnativo per il sistema di locomozione scelto, basato su un blocco di trazione (composto da motoriduttori più cingoli) della Tamiya. Sempre dal punto di vista meccanico, la pinza per il salvataggio della vittima (costituita, secondo le regole della competizione, da una lattina comune dal peso di 150 grammi) non si era dimostrata all'altezza della situazione, per l'ingombro e le difficoltà di montaggio necessarie a metterla in condizioni di sollevare la lattina nelle modalità richieste. Tutto ciò ha reso necessaria la ridefinizione dello chassis, realizzato ora in alluminio e strutturato come si può vedere nella Fig.1.

Ma oltre ai problemi meccanici

a cura del prof. GIULIO VITALE

Fig. 1 - Lo scheletro del robot.



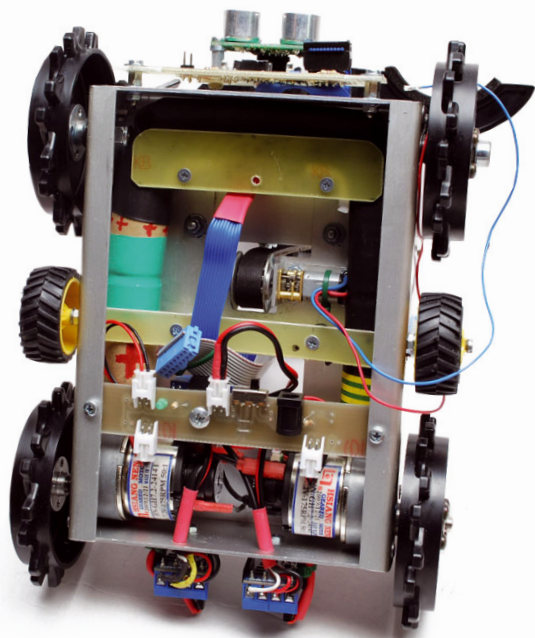


Fig. 2

Fig. 3 - I cingoli adottati.

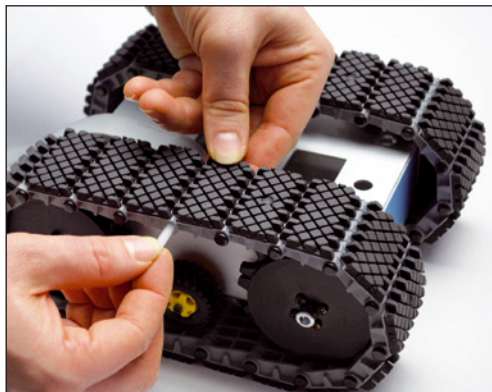


Fig. 4 - Il gruppo motore-motoriduttore che fa muovere il robot.



di deambulazione, ne abbiamo incontrati altri, sia di natura elettrica, riguardanti l'alimentazione, sia di "ingegnerizzazione" dei sensori. Riguardo ai primi, (vedi Fig. 2) la soluzione adottata è stata quella di impiegare pile cilindriche di formato Sub C, al NiMH, da 3.100 mAh di capacità ciascuna, confezionate ad hoc, del tipo impiegato negli avviatori, e che dovrebbero possedere doti di robustezza maggiore rispetto alle normali AA usate in precedenza. Il nuovo pacco batterie è in grado di garantire una più elevata capacità di affrontare lo spunto.

Riguardo ai sensori, le difficoltà sono state di altra natura; in particolare, legate ai problemi di ottica e all'eccesso di "autarchia" nella loro realizzazione: i sensori a infrarosso soffrivano di scarsa affidabilità per problemi di allineamento e quelli di luminosità pativano l'abbagliamento da luce solare non uniforme.

In questo articolo presentiamo, tra le altre, due soluzioni adottate per i sensori: una che interessa gli infrarossi, già implementata e disponibile da subito per il nuovo modello, e l'altra candidata a rappresentare un'evoluzione per quello successivo nel sistema di inseguimento della traccia.

La prima delle due ha, inoltre, fornito lo spunto per inserire "gratuitamente" un giroscopio, fruendo dello stesso canale di comunicazione utilizzato per le misure all'infrarosso.

Cingoli e pinza

Nella scelta dei cingoli, vista la fragilità manifestata dai precedenti, abbiamo voluto "esagerare" ed utilizzare direttamente cingoli più professionali. La scelta fatta dagli studenti, dopo una lunga ricerca, è caduta su uno dei sistemi completi di movimentazione forniti dalla Lynx Motion, composto da:

- cingoli da 2" di larghezza (vedere Fig. 3) che si sono dimostrati senz'altro all'altezza delle aspettative, per robustezza, praticità d'uso e per aderenza;
- motore in corrente continua (vedi Fig. 4) alimentato a 7,2 V e capace di produrre una coppia di 7,1 kg x cm, con un rapporto di riduzione di 50:1; il motore è dotato di encoder ottico incrementale per il controllo di velocità e di posizione, da 100 periodi/giro, su due frequenze in quadratura, da cui sono estraibili 400 impulsi/giro, differenziando i fronti.

Quanto ai cingoli, va detto che in verità sono forse fin troppo robusti, vista l'applicazione, ma lo scopo principale di tutta questa attività è didattico, non commerciale; l'obiettivo è mettere gli studenti in condizio-

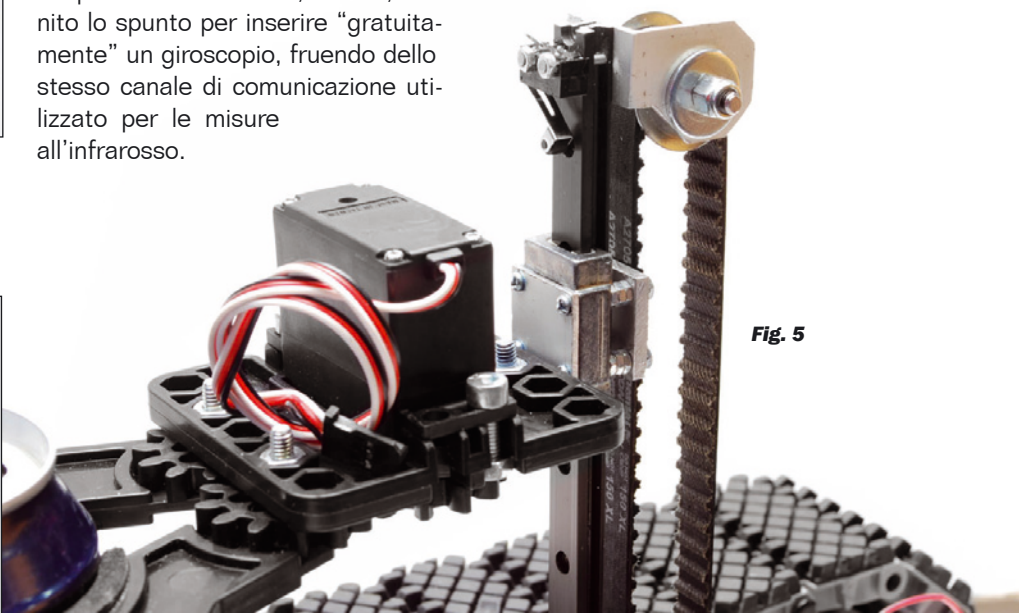
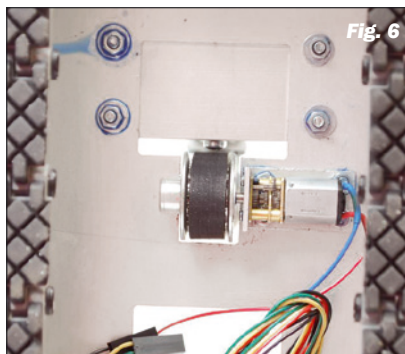


Fig. 5



ni di pensare, cercare e sperimentare soluzioni diverse per convergere, infine, verso una ottimale che risolve il problema. Lo stesso lavoro di indagine via Internet sulle varie possibilità proposte dai costruttori e la necessità di scelta conseguente, risulta essere, di per sé, un processo notevolmente formativo.

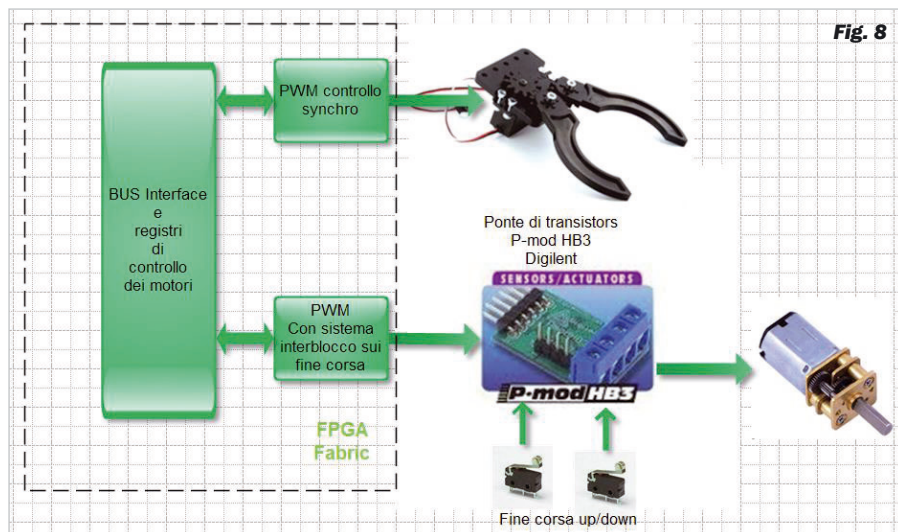
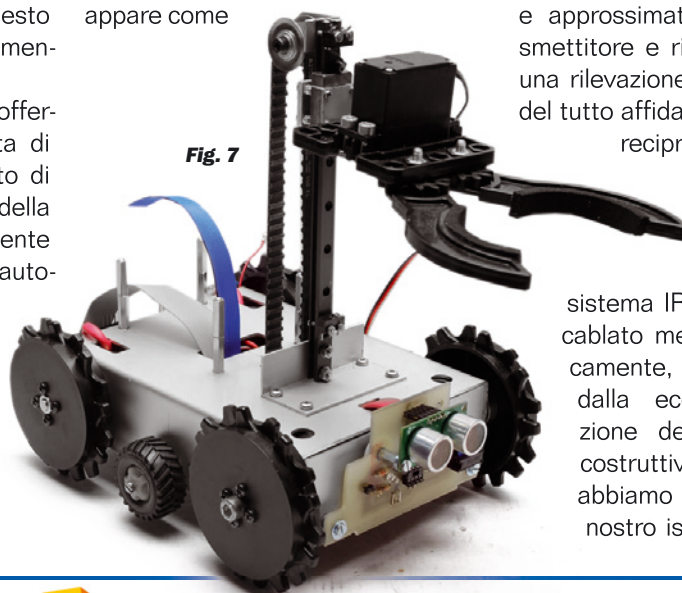
La questione della ricerca della "pinza ideale" è risultata invece più impegnativa, in quanto, nel precedente modello, era stata adottata una soluzione offerta dalla POB Technology, dotata di due classici servo pilotabili in PWM, uno per la rotazione verticale e l'altro per l'apertura/chiusura della pinza. Il movimento di tale pinza non è risultato del tutto adatto a sollevare una lattina e a posizionarla come e dove richiesto; senza contare che da quest'anno questo modello di pinza non è più facilmente reperibile.

Con il determinante supporto offerto dal sig. Gabriele Daghetta di Futura Elettronica, si è pensato di utilizzare una slitta in alluminio della larghezza di 18 mm, normalmente utilizzata nei piccoli sistemi di automazione per gli spostamenti lineari di precisione, e di movimentarla con un minuscolo, ma prestante, motore in c.c. (vedere figure 5 e 6), dotato di un rapporto di riduzione di 1 a 298, che con pochi mA di corrente e 6 volt massimi di

tensione di alimentazione, consente un agevole sollevamento della vittima da salvare. Il motore occupa uno spazio di solo 26x10x12 millimetri e pesa in tutto circa 10 grammi.

Il meccanismo di elevazione è stato realizzato con un sistema formato da una cinghia solidale alla slitta e due pulegge alle estremità di un'asta di sostegno, producendo il risultato finale osservabile in Fig. 7; il controllo dei fine corsa è stato affidato a due micro switch, montati alle due estremità. La pinza utilizzata è ancora fornita dalla Pob Technology ed è risultata più adatta al nostro scopo del modello precedente.

Il controllo elettronico del modulo, infine, è stato affidato alla flessibilità della FPGA Spartan3e, e l'intero sottosistema appare come



raffigurato nello schema a blocchi di Fig. 8.

Un PIC gestisce l'infrarosso, percepisce i dislivelli e informa la CPU

Per il rilevamento degli ostacoli vicini, nel modello precedente si era utilizzato un sistema molto semplice a infrarossi, composto da un diodo emettitore IR, modulato con una portante a 38 kHz, e da un fotodiodo ricevitore, con demodulazione della portante, secondo il modello classico impiegato nei telecomandi. Questo schema era stato replicato quattro volte: due per gli ostacoli anteriori e altrettante per il rilevamento di oggetti laterali. Naturalmente il sistema funzionava, ma il montaggio, un po' troppo casereccio e approssimativo, disallineava trasmettitore e ricevitore, portando a una rilevazione degli IR riflessi non del tutto affidabile, con interferenze reciproche tra i vari moduli.

Si è pensato allora ad una soluzione alternativa, che impiegasse un sistema IR ricetrasmittente già cablato meccanicamente e otticamente, e che ci proteggesse dalla eccessiva approssimazione delle nostre soluzioni costruttive, per le quali non abbiamo a disposizione, nel nostro istituto, una dotazione

di strumenti di lavoro in grado di farci intervenire efficacemente. La scelta è caduta, pertanto, sugli ottimi dispositivi della Sharp GP2D120, che forniscono in uscita un segnale analogico proporzionale alla distanza dell'oggetto riflettente dal sensore, in un campo di valori compreso tra i 4 e i 40 cm. Ovviamente questa scelta ha imposto la necessità di un interfacciamento diverso, per via dell'uscita analogica. D'altro canto, pensando alla possibilità limite di impiegare quattro rilevatori simultaneamente, la soluzione più naturale è stata quella di inserire un microcontrollore della famiglia PIC, con la funzione di collettore intermedio, il quale, convertendo i segnali in digitale, li assembla in una semplice tabella di misure e li trasmette in forma seriale alla FPGA centrale, usando una normale porta UART.

In aggiunta, questa soluzione ha fornito lo spunto per risolvere un altro problema emerso nella stagione precedente, ovvero quello di avvertire la presenza di una rampa e di segnalare l'inizio e la fine, in modo da avere la certezza di aver completato una parte del percorso di gara. Si è perciò aggiunto, vista la disponibilità di una porta SPI sul PIC scelto, un giroscopio a singolo asse, la cui verifica di validità è in corso di

valutazione. Il sistema complessivo cui si è giunti è rappresentato dallo schema a blocchi mostrato nella Fig. 9, con una realizzazione del PCB molto contenuta nelle dimensioni, come mostrato nella Fig. 10. Quest'ultima mostra in primo piano il modulo realizzato, ripreso durante le prove di test con tutti e quattro i sensori installati, e, sullo sfondo, un

ulteriore PIC di controllo che simula temporaneamente la FPGA. Il PIC impiegato è il PIC18F14K22, che offre, in un compatto contenitore da 20 pin, un carnet di risorse molto allettante e buone prestazioni, ottenibili grazie al generatore di clock programmabile, disponibile internamente, il quale può raggiungere i 64 MHz con un PLL che non

Fig. 9 - Interfacciamento dei sensori di distanza.

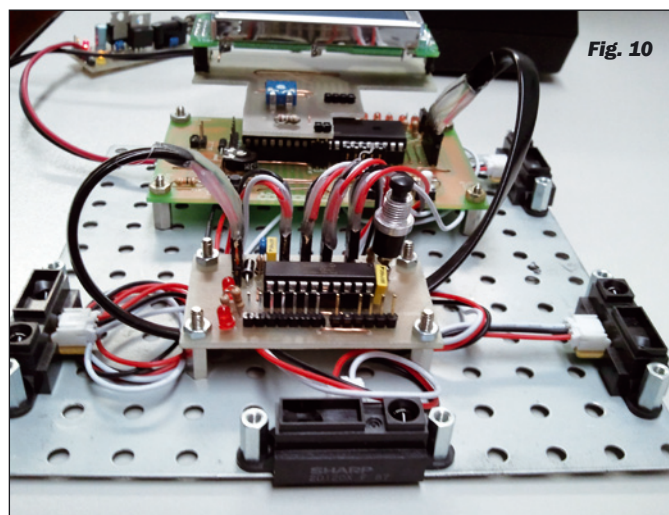
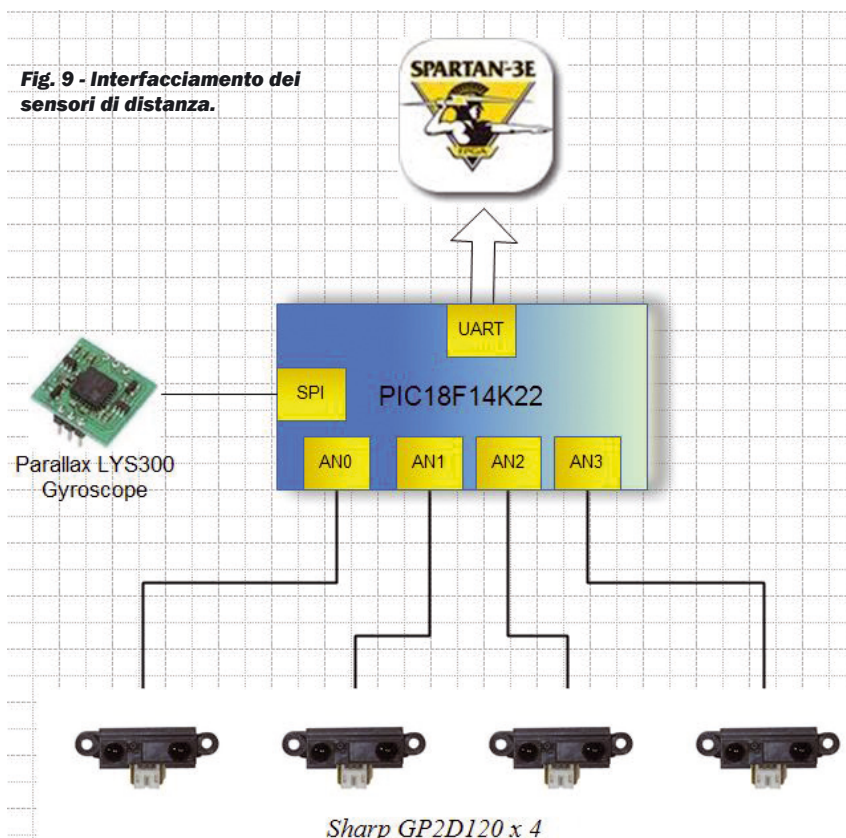


Fig. 10

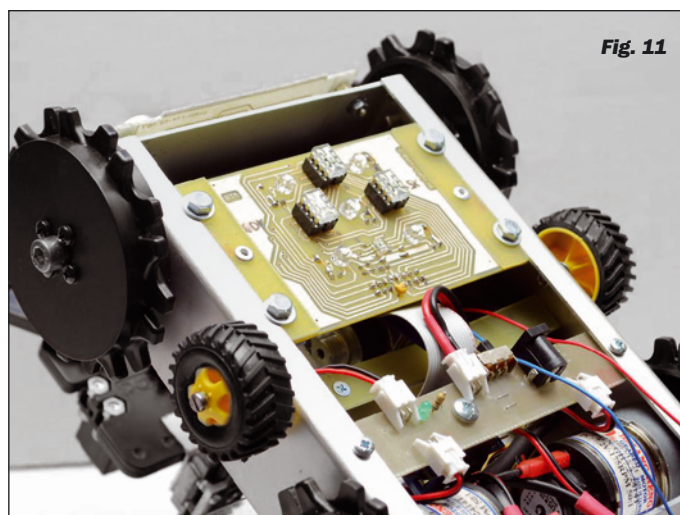


Fig. 11

necessita di alcun componente esterno. In più, il PIC18F14K22 utilizzato nasce ottimizzato per l'uso del linguaggio di programmazione C, grazie al set esteso di istruzioni e ai 16 kB di memoria di programma disponibili; ciò aiuta ad accorciare ulteriormente i tempi di sviluppo.

I sensori di luminosità

Per le gare "rescue" della Robocup Junior, cui Nessie è predestinato, il compito fondamentale da svolgere è l'inseguimento di un tracciato segnato da una linea nera su campo bianco.

L'organo sensoriale delegato allo scopo è il gruppo di sensori di luminosità che era stato presentato nella prima puntata (nel numero di Febbraio 2011 di Elettronica In) e realizzato con quattro componenti della Taos, tre dei quali convertono in frequenza l'intensità luminosa ed uno la scompone nelle tre componenti cromatiche fondamentali, fornendo ancora tre frequenze: una per il rosso, una per il verde e una per il blu. La soluzione adottata è presentata nella Fig. 11 dove si vede la sua "ri-ingegnerizzazione" per Nessie, in un formato più compatto, usando un PCB a doppia faccia.

Questo sistema, potenzialmente efficace nelle prestazioni che è in grado di fornire, soffre di problemi di selettività dell'area inquadrata e di abbagliamenti in presenza di passaggi, più o meno repentini, da luce artificiale a luce solare e viceversa. Per questi ultimi si sta lavorando a livello software per compensare le variazioni di luminosità con opportuni aggiustamenti della sensibilità; mentre per i primi stiamo preparando una maschera artigianale da far indossare a Nessie ma il problema potrebbe essere risolto solo parzialmente, vista la mancanza di un'ottica specifica, realizzata su misura, che consenta



Il gruppo di ragazzi dell'ITCS Erasmus da Rotterdam, che parteciperà col robot Nessie alla competizione Robocup di Catania.

la corretta messa a fuoco dell'area osservata da ognuno dei sensori. A tale scopo è stato individuato un nuovo sensore, offerto sempre dalla Taos, che potrebbe risultare più idoneo a rilevare ed inseguire una traccia, ma questo

potrebbe essere l'argomento di un prossimo articolo sul tema. Per finire, diamo un'occhiata a come appare Nessie ora, a un passo dalla sua alba (Fig. 12). □

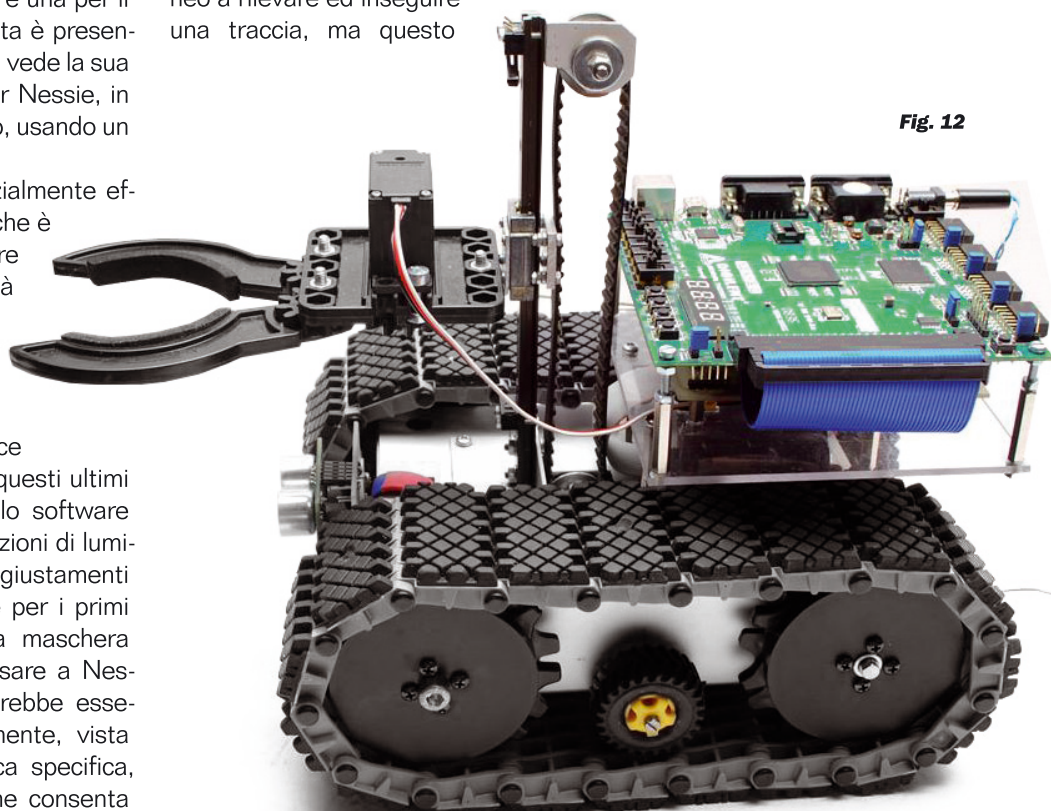
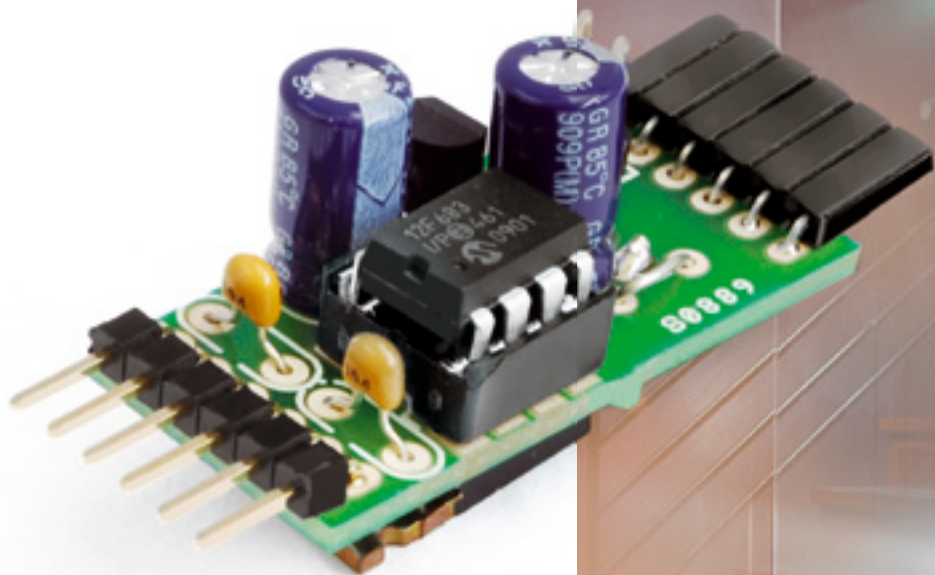


Fig. 12

UNA SCALA LUMINOSA



Realizziamo un suggestivo sistema per illuminare in maniera originale ed efficace le scale della vostra abitazione.

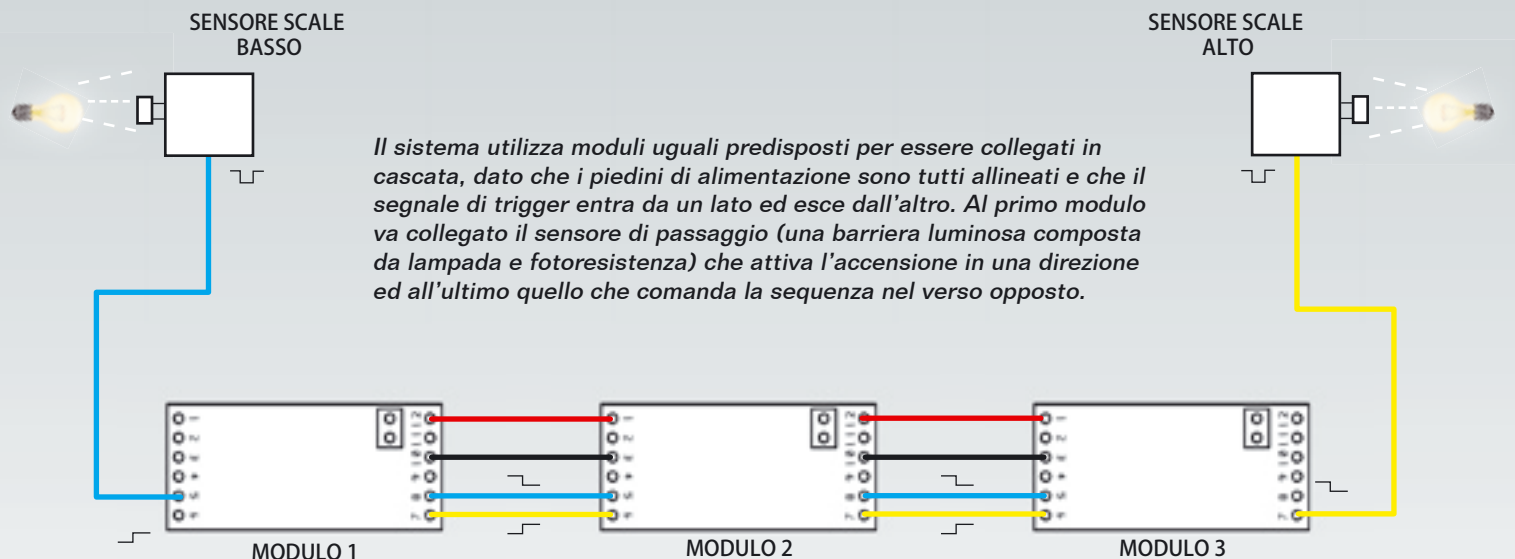
..... di ANDREA LETTIERI

Se qualcuno dovesse chiedervi quali sono gli impianti più importanti presenti in un'abitazione, molti di voi sicuramente risponderebbero: "l'illuminazione interna". E infatti è vero: basti pensare a come staremmo senza la luce in casa e quanto la vita sia cambiata da quando è stata inventata la lampadina elettrica. Illuminare una casa significa certamente mettere luce in ogni vano, ma può anche significare dare un tocco di raffinatezza, ossia usare architettonicamente la luce, collocando lampadine di vario genere e sagoma ad esempio sulle pareti, in angoli

ed archi, ma anche nelle scale. Un esempio di ciò è il progetto che vi proponiamo in queste pagine, ossia una semplice ma funzionale soluzione, finalizzata proprio ad illuminare le scale di casa vostra. Quando si pensa all'illuminazione delle scale, viene subito in mente un temporizzatore che tiene accese più lampade per un certo tempo; ma l'applicazione di queste pagine vi farà conoscere un nuovo concetto di illuminazione, realizzato grazie a strisce di LED con le quali illuminiamo direttamente i gradini, producendo una luce che arriva dal basso. Infatti



Schema funzionale



quando saliamo le scale che cosa ci interessa di più? Illuminare l'ambiente o dove mettiamo i piedi? Certamente la seconda delle due cose. Sfruttiamo quindi le moderne tecnologie per coniugare la drastica riduzione dei consumi di energia elettrica con un tocco di originalità per la vostra casa. Entriamo adesso nel vivo della soluzione che vi proponiamo.

IL NOSTRO SISTEMA

La soluzione consiste nell'utilizzo di un sistema *modulare* per

l'accensione/spegnimento delle luci; è modulare perché realizzato facendo uso di più moduli, ognuno dei quali preposto all'accensione/spegnimento del dispositivo di illuminazione disposto sul singolo gradino. Ci sarà quindi un numero di moduli pari al numero di gradini della scala, ognuno dei quali accenderà il suo "pezzo di illuminazione" e farà da ingresso al modulo successivo, che accenderà il seguente e così via. Ovviamente il primo e l'ultimo modulo (che innescheranno il processo in caso di salita e discesa) avranno come ingresso una fotocellula o un sensore PIR, che rileveranno la presenza della persona e faranno partire il processo di accensione.

Il sistema è schematizzato nella Fig.1.

Il connubio tra illuminazione e tecnologie avanzate è nell'utilizzo di LED (o strisce di LED) come dispositivi di illuminazione dei gradini. Questo sicuramente è un punto a favore della soluzione che vi stiamo presentando, in quanto l'impiego dei LED si basa su consumi ridotti e sull'enorme varietà di colorazioni che essi possono assumere.

SCHEMA ELETTRICO

Partiamo dal singolo modulo, che è basato su un microcontrollore PIC12F683-I/P prodotto dalla Microchip, di cui vengono utilizzati cinque pin, esclusi quelli di alimentazione (5 V) e massa. Oltre ai due pin di ingresso e ai due di uscita, viene sfruttata l'uscita PWM (pin CCP1) del dispositivo, che poi è la prerogativa che ci interessa di più del micro, dal momento che i LED li pilotiamo in PWM; questa modulazione consiste nell'alimentare i diodi luminosi con un'onda rettangolare a frequenza fissa, della quale facciamo variare il duty-cycle: a bassa larghezza degli impulsi corrisponde una ridotta poten-

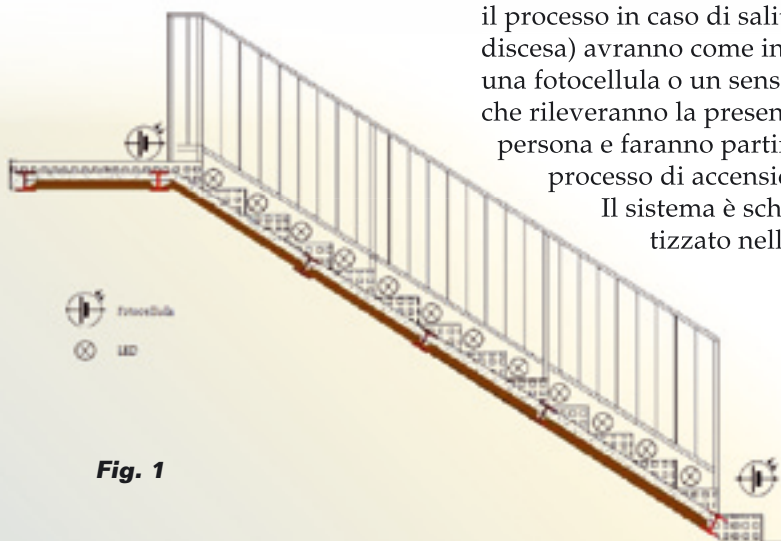
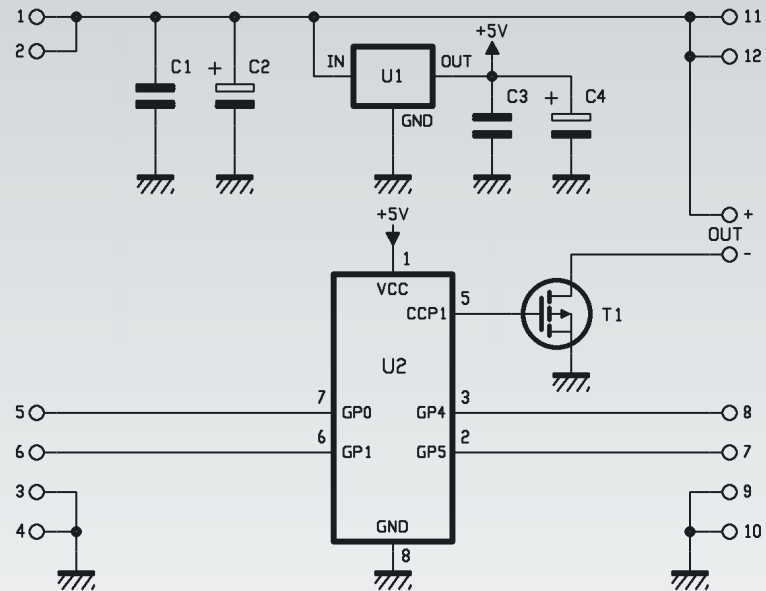


Fig. 1

za e quindi poca luminosità, e viceversa. Il MOSFET in uscita al pin CCP1 piloterà direttamente la striscia di LED.

A completare questo semplice circuito c'è lo stadio di alimentazione, realizzato con un regolatore di tensione 78L05 ed alcuni condensatori di filtro, per fornire al sistema un'alimentazione stabile pari a 5 V.

All'attivazione del primo modulo, che avverrà attraverso il segnale fornito dal sensore e porterà ad un livello logico basso l'ingresso GP0 (contatto 5) il microcontrollore accenderà gradualmente la striscia di LED e quando questa raggiungerà una certa luminosità (decisa da un parametro impostato preventivamente nel PIC) l'uscita GP4 (contatto 8) verrà portata a livello basso in modo da attivare l'ingresso



del modulo successivo, mentre il transistor T1 continuerà a tenere acceso il dispositivo di illuminazione da esso alimentato per un tempo prestabilito in fase di programmazione del microcontrollore. Trascorso questo tempo (e ciò avviene ovviamente per ogni modulo),

il segnale PWM si disattiverà e il transistor si spegnerà, lasciando i LED senza alimentazione. Il risultato sarà che lo spegnimento progressivo gradino per gradino a partire dal primo che avete percorso (figure 2 e 3).

La versatilità del microcontrollore utilizzato permette di usare un solo modulo per ogni gradino, che potrà funzionare percorrendo le scale in ambo le direzioni; infatti, facendo le scale in direzione inversa, il sensore che vi rileverà

porterà a livello basso il segnale all'ingresso GP5, così che il microcontrollore accenderà la striscia di LED ad esso collegata abbassando dopo un certo tempo l'uscita GP1, la quale sarà collegata all'ingresso GP5 del modulo successivo, e così via. Lo spegnimento avverrà esattamente come per l'altro senso di percorrenza, cioè dopo un certo tempo preventivamente impostato si spegneranno i primi LED e lo spegnimento si propagerà per tutti i gradini fino all'ultimo che avrete percorso (figure 4 e 5).

Dovete collegare in cascata tanti moduli quanti sono i gradini, applicando al primo e all'ultimo i sensori di passaggio.

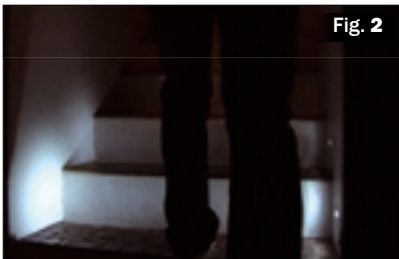


Fig. 2



Fig. 4

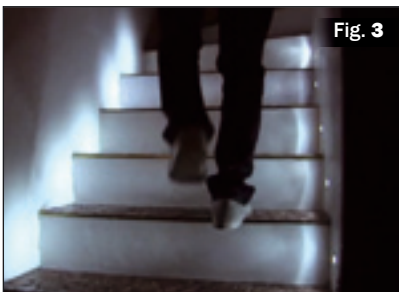


Fig. 3

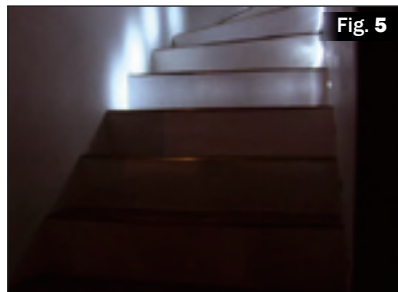


Fig. 5

Listato 1

```

DEFINE OSC 4
@      DEVICE INTRC_OSC
@      DEVICE MCLR_OFF

SYMBOL IN5      =GPIO.0
SYMBOL IN7      =GPIO.5
SYMBOL OUT6     =GPIO.1
SYMBOL OUT8     =GPIO.4

LUCE      VAR BYTE
INGRESSO  VAR BYTE

TEMPO     CON 10
PAUSA     CON 10000
PUNTOIN   CON 50

OPTION_REG=%00000000
CMCON0   =%00000111
ADCON0=%00000000
ANSEL=%00000000
WPU=%11111111

INTCON=%00000000

GOSUB    UP
GOSUB    DOWN

LOW      GPIO.2
HIGH     OUT8
HIGH     OUT6
CLEAR

MAIN:
    GOSUB INGRESSI
    GOTO MAIN

INGRESSI:
    IF IN5=0 THEN
        INGRESSO = 5
        GOSUB UP
        WHILE IN5=0
            PAUSE 500
        WEND
        HIGH OUT8
        PAUSE PAUSA
        GOSUB DOWN
    ENDIF

    IF IN7=0 THEN
        INGRESSO = 7
        GOSUB UP
        WHILE IN7=0
            PAUSE 500
        WEND
        HIGH OUT6
        PAUSE PAUSA
        GOSUB DOWN
    ENDIF

RETURN

UP:
    FOR LUCE=0 TO 255
        IF LUCE=PUNTOIN THEN
            IF INGRESSO=5 THEN
                LOW OUT8
            ELSE
                LOW OUT6
            ENDIF
        ENDIF
        HPWM 1,LUCE,250
        PAUSE TEMPO
    NEXT LUCE
RETURN

DOWN:
    FOR LUCE=255 TO 0 STEP -1
        HPWM 1,LUCE,250
        PAUSE TEMPO
    NEXT LUCE
RETURN

```

IL FIRMWARE DEL PIC

Bene, spiegato lo schema elettrico ed il funzionamento del singolo modulo, passiamo a descrivere qui di seguito il firmware (scritto in PICBasic PRO) che gira nel microcontrollore. Iniziamo ricordandovi che per ottenere il funzionamento descritto deve essere possibile utilizzare i moduli in entrambe le direzioni, in modo da ottenere lo spegnimento sequenziale sia quando salite che quando scendete le scale.

Come si evince dal **Listato 1**, dopo la parte di dichiarazione di variabili e costanti, seguita dall'inizializzazione dell'hardware, il programma principale (main) rimanda subito alla subroutine INGRESSI in cui, quando IN5 (che rappresenta l'ingresso GPI.0 del microcontrollore) è a livello basso, la subroutine UP terrà allo stato logico zero l'uscita GPI.4 del microcontrollore, che andrà ad innescare il "gradino" seguente (ingresso GPI.0 del successivo modulo) mentre l'uscita PWM accenderà la striscia di LED con una frequenza di modulazione pari a 250 Hz. Passato il tempo dato dalla costante PAUSA (nel nostro esempio, pari a 10 secondi), la subroutine DOWN attiverà il processo inverso, che determinerà lo spegnimento del modulo. Nelle subroutine UP e DOWN è chiaro anche come l'accensione del modulo successivo avrà inizio al termine dell'accensione del precedente ovvero al termine del periodo preimpostato, che nel

nostro esempio
si verificherà
quando la



In auto con i fari sempre accesi?

[piano di MONTAGGIO]

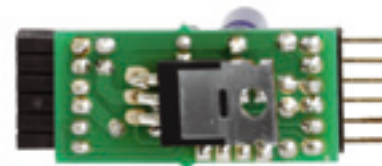
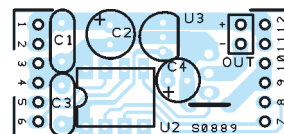
variabile LUCE, durante il suo incremento, raggiungerà l'indice 50 (PUNTOIN). La costante TEMPO darà, una volta moltiplicato il suo valore per 255 (perché si trova all'interno del ciclo *for*), il tempo di "fader up e down" della striscia di LED. Come già detto in precedenza, tutto funziona ugualmente se la scala viene percorsa nell'altro senso e quindi in tal caso i segnali di ingresso andranno ad alimentare GPI.5, mentre quelli di uscita saranno alimentati dal pin GPI.1, e questo è evidente nel **Listato 1**, che mostra il firmware del microcontrollore da noi usato.

Elenco Componenti:

C1: 100 nF multistrato
C2: 100 μ F 25 VL elettrolitico
C3: 100 nF multistrato
C4: 100 μ F 25 VL elettrolitico
U1: 78L05
U2: PIC12F683-I/P (MF889)
T1: IRF540

Varie:

- Zoccolo 4+4
- Strip maschio 2 poli
- Strip maschio 90° 6 poli
- Strip femmina 90° 6 poli
- Circuito stampato



REALIZZAZIONE PRATICA

La costruzione del singolo modulo è semplice, visto l'esiguo numero di componenti che esso richiede; il nostro consiglio è di partire dalla saldatura dei componenti più bassi per arrivare a quelli più alti. Il microcontrollore andrà montato sullo zoccolo 4+4 preventivamente saldato sul circuito stampato. Gli strip maschio e femmina serviranno all'interfacciamento del modulo

con l'utilizzatore e con i moduli precedenti e successivi, mentre lo strip maschio a 2 poli consentirà di alimentare la striscia a LED. Per completare l'intero sistema bisognerà montare due sensori agli estremi della scala, una striscia di LED per ogni gradino, quindi effettuare i collegamenti tra un modulo ed il successivo; i sensori devono avere l'uscita che in caso di attivazione fornisca un

livello basso, ovvero che sia un contatto normalmente aperto che si chiude a massa, sempre in caso di attivazione. I sensori si collegano tra i punti 5 e massa (3, 4, 9 o 10). In luogo della fila di LED si potrà utilizzare per ciascun gradino uno spot composto da un singolo diodo high-power, ad esempio da 1 watt. Fatti i collegamenti, alimentate pure i singoli moduli.

Una recente direttiva dell'UE impone che le auto immatricolate per la prima volta dal gennaio 2011 siano dotate di lampade diurne (DRL, ossia Day Running Light) da tenere accese anche di giorno; si tratta sostanzialmente di luci di posizione potenziate la cui funzione, per le vetture che ne sono sprovviste, sarà assolta tenendo sempre accesi gli anabbaglianti, almeno durante la circolazione. L'obbligo di mantenere accesi i fari pone una serie di problemi, primi fra tutti il consumo (che si traduce in un maggiore carico del motore) e la ridotta durata delle lampade, non sempre economiche. Per questa ragione vari produttori, fra cui la Philips, hanno realizzato e stanno sviluppando gruppi ottici con lampade DRL a LED già omologate ECE R87: ciò consentirà di ridurre nettamente i consumi (in virtù della maggiore efficienza dei diodi luminosi rispetto alle comuni lampadine alogene) e i costi di manutenzione, dato che una lampada a LED dura mediamente 40 volte una a incandescenza. Le luci diurne si accenderanno automaticamente avviando il motore, quindi i distratti non dovranno temere di incorrere nelle multe che certamente verranno introdotte allo scopo...



per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono di facile reperibilità. Il master del circuito stampato può essere scaricato gratuitamente dal sito della rivista così come il firmware utilizzato per programmare il microcontrollore PIC12F683. Quest'ultimo può essere acquistato già programmato (cod. MF889) al prezzo di 8,00 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>



Microsoft®

.NET Micro Framework

di Lorenzo Maiorfi
e Gianluca Ruta
Innovactive Engineering

È giunto il momento di realizzare insieme un'interfaccia utente di tipo grafico per le nostre applicazioni basata su un display; scopriremo l'uso delle primitive, la gestione del touch-screen e tanto altro ancora. Quarta puntata.

Nelle puntate precedenti del corso sulla piattaforma per lo sviluppo di applicazioni per microcontrollori di casa Microsoft, abbiamo affrontato la comunicazione seriale dei dati e l'utilizzo dello stack TCP/IP per comunicazioni su rete Ethernet. In tutto il codice che abbiamo prodotto per realizzare test concreti inerenti agli argomenti trattati, abbiamo utilizzato spesso la finestra di debug dell'ambiente di sviluppo per avere una semplice diagnostica di quanto il microcontrollore operava, mentre in altri casi la nostra diagnostica si è basata sulla comunicazione seriale di dati ad un PC, con i messaggi visualizzati tramite una finestra di emulazione terminale. Per i semplici scopi didattici che ci siamo prefissati

questo poteva bastare, ma non c'è dubbio che la forma di comunicazione più efficace nella direzione che va dal sistema all'utente non può prescindere dall'uso di un display dedicato, possibilmente grafico e ormai sempre più spesso dotato di funzionalità "touch" per diventare periferica di input oltreché di output. L'argomento di questa puntata verte proprio sull'interfaccia utente delle nostre applicazioni per microcontrollori, realizzata per un display grafico.

GRAFICA ED INTERFACCIA UTENTE

Anche se parlare di interfaccia utente di un'applicazione non significa necessariamente fare riferimento ad argomenti legati alla

Listato 1

```

1: int w,h,d,r;
2:
3: HardwareProvider.HwProvider.GetLCDMetrics(out w,out h,out d,out r);
4:
5: Bitmap bmp = new Bitmap(w, h);
6:
7: bmp.DrawRectangle(
8:     Color.White, // colore bordo ininfluyente
9:     0,           // senza bordo
10:    0, 0, w, h,   // parte da 0,0 (in alto a sinistra) ed ha le stesse dimensioni del display
11:    0, 0,         // raggio degli angoli "scondati"
12:    Color.RoyalBlue, 0, 0, // colore e coordinate di partenza del gradient
13:    Color.RoyalBlue, 0, 0, // colore e coordinate di arrivo del gradient
14:    Bitmap.Opaque); // contenuto opaco (non trasparente)
15:
16: bmp.Flush();

```

visualizzazione di informazioni, è innegabile che il mezzo di gran lunga più efficace in tal senso è costituito dall'insieme delle funzionalità e dei dispositivi finalizzati alla rappresentazione grafica e testuale dei dati manipolati da un'applicazione. L'avvento dell'espressione "multimediale" ha in realtà fatto sì che l'attenzione degli utenti ricadesse anche su rappresentazioni diverse da quella visuale, come quella sonora (ad esempio), ma è indubbio che il ruolo giocato dai media non visuali è assolutamente marginale. Per questo motivo, gran parte dell'evoluzione subita dalle tecnologie che orbitano intorno al mondo dell'Information Technology, ivi compreso il mondo embedded, si è concentrata sul potenziamento delle capacità di visualizzazione delle informazioni e della gestione dei comandi impartiti dagli operatori.

Attento alle tendenze del panorama tecnologico e agli evidenti segnali che provengono dal mondo dell'elettronica consumer, il team dei designer del .NET Micro Framework ha speso molte energie nella realizzazione di

un sistema per la messa a punto di interfacce utente particolarmente ricche. Tale sistema, assolutamente inedito in questo settore, è caratterizzato dalla possibilità di esprimere gerarchie complesse di oggetti grafici attraverso un modello astratto ed indipendente dalla geometria del display utilizzato. Inoltre, il .NET Micro Framework supporta in modo nativo la gestione delle funzionalità "touch", supportate ovviamente dai dispositivi abilitati in tal senso, tra i quali figura la scheda di sviluppo "Tahoe II" di Device Solutions, utilizzata e descritta già nelle precedenti puntate.

BITMAP MODE

Addentrando nelle classi del .NET Micro Framework dedicate alla gestione degli oggetti grafici, scopriamo che esistono in realtà due diversi "strati" che separano il nostro firmware dal driver che di fatto pilota, tipicamente attraverso un'interfaccia parallela, il controller del display: un primo strato si occupa del trasferimento (detto "blitting") di un oggetto *Bitmap* sulla superficie del display, mentre lo strato superiore, denominato "Presentation", si occupa dell'astrazione degli oggetti grafici, ottenuta attraverso la definizione di una struttura gerarchica di contenitori ed elementi di interfaccia utente che alimentano il motore di layout e di rendering attraverso l'uso di un'apposita "pipeline" gestita dal framework stesso.

Iniziamo la nostra sperimentazione a partire dal layer di basso livello. Il paradigma utilizzato in questo modello di sviluppo prevede che all'interno del firmware venga definita un'istanza della classe *Bitmap* caratterizzata dalla stessa geometria del display disponibile sul dispositivo. Tale *Bitmap* agisce come un vero e proprio display "offline", sul quale

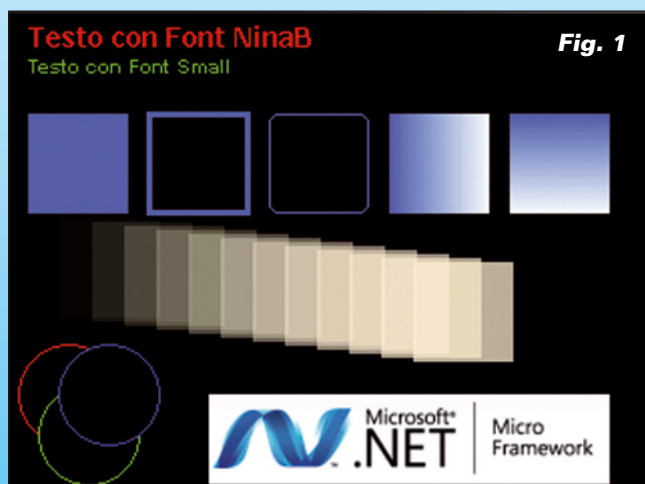


Fig. 1

vengono disegnate delle primitive grafiche attraverso appositi metodi della classe `Bitmap`, fino a quando, a seguito dell'esecuzione del metodo `Flush()`, i pixel che descrivono la superficie della bitmap offline vengono effettivamente disegnati sul display del dispositivo.

Se, ad esempio, vogliamo disegnare un rettangolo rosso che copra l'intera superficie del display, è sufficiente utilizzare la porzione di codice riportata nel **Listato 1**.

Analogamente, è possibile utilizzare le numerose primitive grafiche per disegnare ellissi, bitmap, testo e linee. Facendo riferimento alla **Fig. 1**, in cui è illustrato l'output del firmware che prenderemo a breve in esame, vale la pena di sottolineare che ciascuna primitiva comporta delle peculiarità di cui occorre essere al corrente per evitare sorprese al momento del suo utilizzo.

TESTO

Nella parte in alto a sinistra dell'immagine, vediamo due testi realizzati mediante la primitiva `"DrawText"`. Tale metodo può essere utilizzato secondo quanto illustrato nel **Listato 2**, in cui `"bmp"` rappresenta, come nel listato precedente, la `Bitmap` relativa al display `"offline"`. Alle righe 2 e 7 è possibile notare come selezionare il font da utilizzare per il rendering del testo; infatti, a differenza di quanto avviene all'interno di Windows, nella runtime del .NET Micro Framework non sono presenti dei font residenti. Questo si riflette sull'impossibilità di istanziare la classe `Font` a partire, ad esempio, dal nome della famiglia del carattere e dalla dimensione. L'unica via prevista è quella in cui il font viene caricato attraverso le `"risorse"` definite all'interno dell'applicazione.

Una `"risorsa"` è definita dal framework come un oggetto (binario o testuale) incapsulato all'interno di un'apposita area dell'assembly prodotto come risultato della compilazione di un file sorgente con suffisso `".resx"`. Tale file, strutturato come un xml, riporta il riferimento ai file che verranno effettivamente inclusi come risorsa. In fase di editing del progetto, l'ambiente di sviluppo gestisce le risorse con

Listato 2

```
1: bmp.DrawText("Testo con Font NinaB",
2:   Resources.GetFont(Resources.FontResources.NinaB),
3:   ColorUtility.ColorFromRGB(255, 0, 0),
4:   10, 5);
5:
6: bmp.DrawText("Testo con Font Small",
7:   Resources.GetFont(Resources.FontResources.small),
8:   ColorUtility.ColorFromRGB(0, 255, 0),
9:   10, 20);
```

un apposito designer, dal quale è possibile includere file generici o di tipo specifico, quali immagini, font, testi, ecc. L'editor di risorse genera inoltre un file sorgente per ciascun file `.resx` presente nel progetto. Tale file, denominato con l'estensione `".Designer.cs"`, contiene la dichiarazione della classe parziale `"Resources"`, attraverso la quale è possibile accedere all'effettivo contenuto di una risorsa, utilizzando metodi specializzati per il tipo di risorsa richiesta. All'inclusione di un file di tipo `".tinyfont"`, che descrive un tipo di carattere tipografico utilizzabile nel .NET Micro Framework, viene, ad esempio, generato il metodo `GetFont()` che prevede un parametro di tipo enumerato denominato `FontResources`, anch'esso definito dal designer, in cui sono elencati tutti i font utilizzabili dall'applicazione. L'installazione del .NET Micro Framework SDK crea una cartella denominata `"Fonts"`, tipicamente contenuta in `"C:\Program Files\Microsoft .NET Micro Framework\v4.0"`, in cui sono presenti due file con suffisso `tinyfont`: `'NinaB'` e `'small'`, illustrati nella già citata **Fig. 1**, rispettivamente in rosso e in verde. Altri font possono essere aggiunti mediante l'utilità `TFCConvert`, contenuta nella cartella `"Tools"`. Tale utility consente infatti la conversione di un font TrueType in un `.tinyfont`, con l'unica limitazione rappresentata dal fatto che, essendo i font del .NET Micro Framework non scalabili, in fase di conversione occorre specificare la dimensione del font da convertire.

Listato 3

```
1: // Test trasparenza
2: for (ushort opacity = 0; opacity < 200; opacity += 16)
3: {
4:     bmp.DrawRectangle(Color.White,
5:       0,
6:       10 + opacity, 100 + opacity / 8, 50, 50,
7:       0, 0,
8:       ColorPapayaWhip, 0, 0,
9:       ColorPapayaWhip, 0, 0,
10:      opacity);
11: }
```

AREE**RETTANGOLARI**

La primitiva più sviluppata è senz'altro quella relativa alla produzione di aree rettangolari; ciò perché costituisce il principale strumento per la realizzazione di oggetti grafici complessi quali pulsanti, liste, pannelli, ecc. Analizzando i cinque rettangoli blu riportati nella **Fig. 1** è possibile notare come siano previste diverse varianti implementate dallo stesso metodo `DrawRectangle()`. Nell'applicazione demo sono disegnati, nell'ordine da sinistra verso destra, rispettivamente un rettangolo pieno con colore uniforme, uno vuoto con spessore maggiore di un pixel, uno vuoto con angoli arrotondati, un rettangolo colorato con riempimento 'gradient' (ossia sfumato tra due colori estremi) e in ultimo, ancora un rettangolo con riempimento 'gradient' ma con direzione verticale. La direzione del riempimento 'gradient' è liberamente impostabile specificando i due punti che delimitano gli estremi della interpolazione. Il rendering di aree rettangolari può inoltre avvenire utilizzando una modalità semitrasparente, in cui, attraverso l'ultimo parametro, è possibile specificare il livello di opacità tra 0 e 255. Un esempio di utilizzo di tale funzionalità è riportato nell'area centrale dell'immagine in **Fig. 1**, prodotta tramite i comandi riportati nel codice del **Listato 3**.

CERCHI ED ELLISSI

Attraverso la primitiva `DrawEllipse()` è possibile disegnare all'interno di una `Bitmap` cerchi ed ellissi, analogamente a quanto avviene per le aree rettangolari. Però, a causa di una limitazione nell'attuale implementazione del .NET Micro Framework, tale primitiva non può essere utilizzata per disegnare figure piene, ma solo in modalità "outline" (limitazione aggirabile attraverso l'uso di opportune risorse di tipo "Image"). L'uso della primitiva `DrawEllipse()` è riportato nel codice contenuto nel **Listato 4**, in cui viene riprodotta la parte in basso a sinistra dell'immagine in **Fig. 1**.

Listato 4

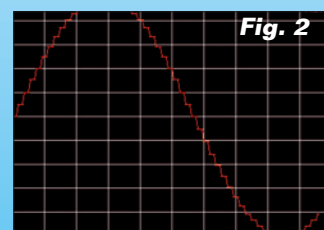
```
1: // Cerchi
2: bmp.DrawEllipse(ColorUtility.ColorFromRGB(255, 0, 0), 30, 190, 25, 25);
3: bmp.DrawEllipse(ColorUtility.ColorFromRGB(0, 255, 0), 40, 210, 25, 25);
4: bmp.DrawEllipse(ColorUtility.ColorFromRGB(0, 0, 255), 50, 190, 25, 25);
5:
6: // NON IMPLEMENTATO IN .NET MF 4.0
7: // bmp.DrawEllipse(
8: //     ColorYellow, 0, 90, 200, 25, 25,
9: //     ColorYellow, 90, 200, ColorMediumSpringGreen, 100, 210, Bitmap.OpacityOpaque);
```

Listato 5

```
1: // Immagini
2: Bitmap bmpIcon1 = Resources.GetBitmap(Resources.BitmapResources.netmf);
3: bmp.DrawImage(100, 190, bmpIcon1, 0, 0, bmpIcon1.Width, bmpIcon1.Height);
```

IMMAGINI BITMAPPED

L'ultima primitiva che prendiamo in considerazione è anche la più potente. Il metodo `DrawImage()` consente infatti di disegnare all'interno della superficie grafica "offline" delle immagini descritte attraverso istanze della classe "Bitmap". Oltre al rendering semplice, è possibile effettuare anche la scalatura, riportando l'immagine con dimensioni diverse da quelle originali, ed utilizzare diversi livelli di trasparenza, come per le aree rettangolari. Le immagini da disegnare possono essere definite attraverso il loro contenuto "binario", espresso in formato Bmp, Gif, Jpeg o TinyCLRbitmap (un formato personalizzato utilizzato all'interno del Framework), oppure, in alternativa, estratte dalle risorse dell'applicazione. Il codice di esempio riportato nel **Listato 5**, disegna sul display l'immagine raffigurante il logo del .NET Micro Framework, precedentemente aggiunto al progetto come risorsa. Come esempio conclusivo della parte riguardante l'approccio a basso livello per la generazione di interfacce utente, ci proponiamo di realizzare un semplicissimo oscilloscopio, attraverso lo sviluppo di un'applicazione che, utilizzando l'ADC presente a bordo della scheda di sviluppo TahoeII, permetta il monitoraggio grafico del livello dell'ingresso analogico "AUX", in una scala 0÷3,3V. L'immagine prodotta dal firmware in questione è riprodotta nella **Fig. 2**. Il codice relativo al sorgente del firmware è riportato nel **Listato 6**, in cui è possibile notare l'utilizzo della primitiva `DrawLine()`, non trattata in precedenza, per la tracciatura di linee



con spessore e colore parametrici.

PRESENTATION

Lo strato di più alto livello per la realizzazione di interfacce utente prende, all'interno del .NET Micro Framework, il nome di "Presentation". In realtà esso è distribuito all'interno di un assembly specifico, denominato "Microsoft.SPOT.TinyCore", che definisce tutte le classi relative ai namespaces "Microsoft.SPOT.Presentation.*". A differenza di quanto visto per il layer di basso livello, lo strato "Presentation" si occupa anche della gestione dell'input dell'utente, per quanto riguarda l'uso sia di pulsanti, sia di display con funzionalità "touch". Prima di addentrarci nei meandri di "Presentation", vale la pena sottolineare che l'ambiente di sviluppo prevede un apposito template per la creazione di applicazioni che ne utilizzano le funzionalità. Tale template prende il nome di "Window Application" (nella Fig. 3 si

vede la finestra di dialogo di Visual Studio durante la creazione di un nuovo progetto con questo modello). L'applicazione così creata dall'ambiente di sviluppo ci dà modo di esaminare alcune delle classi più importanti ed in particolare la maniera in cui collaborano, nell'ambito di un'applicazione di questo tipo, gli oggetti di tipo "Window", "Program" e "GPIOButtonInputProvider".

Premesso che le funzionalità relative al firmware di esempio che verrà illustrato si limitano a riportare sul display una stringa

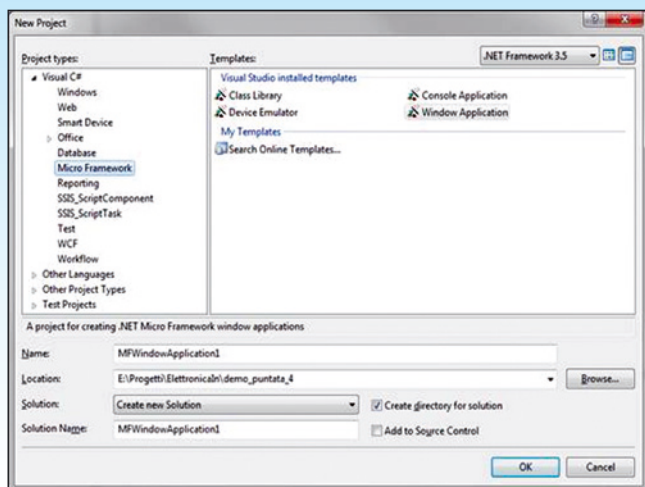
Listato 6

```

1: // Divisione del reticolo...
2: int xdivs = 10;
3: int ydivs = 10;
4: int oldx = 0, oldy = 0;
5:
6: while (true)
7: {
8:     oldx = -1;
9:     oldy = -1;
10:
11:     // Pulisco la bitmap
12:     bmp.Clear();
13:
14:     // Bordo...
15:     bmp.DrawRectangle(ColorGrey, 1, 0, 0, w, h,
16:         0, 0, Color.White, 0, 0, Color.White, 0, 0,
17:         Bitmap.OpaCityTransparent);
18:
19:
20:     // reticolo...
21:     for (int xi = 0; xi < xdivs; xi++)
22:     {
23:         bmp.DrawLine(ColorGrey, 1, xi * (w / xdivs), 0, xi * (w / xdivs), h);
24:     }
25:
26:     for (int yi = 0; yi < ydivs; yi++)
27:     {
28:         bmp.DrawLine(ColorGrey, 1, 0, yi * (h / ydivs), w, yi * (h / ydivs));
29:     }
30:
31:     bmp.DrawText("AUX",
32:         Resources.GetFont(Resources.FontResources.small),
33:         ColorUtility.ColorFromRGB(255, 0, 0), 294, 6);
34:
35:     bmp.Flush();
36:
37:     // Scansione e lettura ADC
38:     for (int xi = 0; xi < w; xi++)
39:     {
40:         float scaledtemp=(TahoeII.Tsc2046.ReadAux()) / 3.3f;
41:
42:         int yt = h-(int)(scaledtemp*h);
43:
44:         if (oldx != -1)
45:         {
46:             // Linea che unisce gli ultimi 2 punti rilevati...
47:             bmp.DrawLine(ColorUtility.ColorFromRGB(255, 0, 0), 1, oldx, oldy, xi, yt);
48:
49:             bmp.Flush();
50:         }
51:
52:         oldx = xi - 1;
53:         oldy = yt;
54:
55:         Thread.Sleep(10);
56:     }
57:
58: }
```

caricata da risorse e ad intercettare la pressione dei pulsanti del dispositivo per riportarne l'identificativo all'interno della finestra di debug, vale la pena di "sezionare" le parti principali dell'applicazione per identificarne responsabilità e peculiarità. L'interfaccia dell'applicazione in questione è illustrata in Fig. 4. L'entry point dell'applicazione, stavolta derivata dalla classe "Microsoft.SPOT.Application", è ancora una volta il metodo statico "Main", nel quale vengono creati un oggetto Window ed un oggetto GPIOButtonInputPro-

Fig. 3



vider, come illustrato nel **Listato 7**.

L'oggetto Window rappresenta un contenitore logico e visuale per la generazione dell'interfaccia utente. Durante il normale funzionamento dell'applicazione ne vengono spesso create più istanze, la cui sovrapposizione viene gestita dal motore grafico del framework. La creazione della prima (ed unica) finestra della nostra applicazione di esempio avviene grazie al codice riportato nel **Listato 8**.

Le sezioni degne di nota sono senz'altro quelle relative alle righe 11÷19, in cui viene creato e configurato un oggetto grafico di tipo "Text" (paragonabile ad un controllo grafico di tipo "etichetta") e alla riga 22, in cui vediamo come configurare il sistema di gestione dell'input in modo tale che gli eventi di tipo "ButtonUp" provenienti dal sottosistema "Buttons" vengano gestiti dal metodo denominato (arbitrariamente) OnButtonUp.

Tale metodo, implementato nella nostra applicazione come semplice funzione di debug che riporta il codice del pulsante premuto, è

Listato 7

```

1: public class Program : Microsoft.SPOT.Application
2: {
3:     public static void Main()
4:     {
5:         Program myApplication = new Program();
6:
7:         Window mainWindow = myApplication.CreateWindow();
8:
9:         // Crea l'oggetto che mappa i GPIO come ingressi di tipo "Button"
10:        GPIOButtonInputProvider inputProvider = new GPIOButtonInputProvider(null);
11:
12:        // Avvia l'applicazione a partire dalla finestra appena creata
13:        myApplication.Run(mainWindow);
14:    }
15:
16:    // [...]
17: }
```

Fig. 4



definito con il codice riportato nel **Listato 9**. Quello che non appare evidente da

questo listato è come sia possibile che gli eventi relativi alla pressione dei pulsanti presenti sul dispositivo target siano effettivamente legati agli ingressi digitali corrispondenti ai pulsanti stessi, o anche come siano coniugati lo stato logico dell'ingresso digitale con quello relativo alla pressione del pulsante, collegabile, quest'ultimo, con una resistenza di pull-up o di pull-down rispettivamente ai livelli di tensione alto e basso. La risposta alla nostra domanda viene dall'analisi di quanto avviene alla riga 10 del metodo Main() illustrato in precedenza: la classe "GPIOButtonInputProvider" si occupa della trasformazione degli interrupt di basso livello generati dalle variazioni di stato degli ingressi digitali collegati ai pulsanti in eventi instradati all'interno del sistema di gestione dell'input del layer "presentation" del Framework. Ciò avviene come illustrato nel **Listato 10**.

Ad una prima analisi, la complessità della struttura della classe GPIOButtonInputProvider può effettivamente spiazzare, ma la cosa effettivamente importante da notare è l'effettiva responsabilità di ciascuna delle classi utilizzate, riassunta nell'elenco che segue.

- **PresentationSource** è una classe che agevola l'utilizzo di sistemi di rendering indipendenti dall'hardware, attraverso l'esposizione della proprietà RootUIElement, radice

logica di una sorgente di contenuti.

- **Dispatcher** è una classe che si occupa della gestione della coda di eventi di un thread.

- **InputManager** è una classe che coordina tutti i sistemi di input attivi nel sistema, trasformando gli eventi di basso livello in eventi consumabili dal sottosistema "Presentation".

- **HardwareProvider** fornisce informazioni sulla

mappatura delle funzionalità per uno specifico hardware. Nel caso appena visto ad esempio viene utilizzata per conoscere la mappatura tra GPIO di ingresso e codici pulsante.

- **ButtonDevice** rappresenta un generico dispositivo hardware dotato di pulsanti.
- **InputProviderSite** rappresenta una classe con funzioni di mediazione tra un input provider ed il InputManager.
- **RawButtonInputReport** rappresenta l'oggetto che veicola l'informazione sull'attivazione di un pulsante che viene passato da un input provider all'InputProviderSite e ancora all'InputManager.

PATTERN D'USO DEL PRESENTATION LAYER

Per approfondire la conoscenza sugli strumenti disponibili all'interno del vasto mondo del layer "Presentation", è opportuno prendere in esame alcuni esempi notevoli che illustrano soluzioni articolate per la rappresentazioni di informazioni anche complesse. Gran parte di quanto vedremo in questa sezione è riassunto nel sample denominato "SimpleWPFApplication", contenuto nel .NET Micro Framework SDK, di cui la **Fig. 5** rappresenta l'interfaccia utente principale. Per dare accesso a ciascun sotto-demo illustrato, l'applicazione in questione utilizza un menu scorrevole animato, azionabile attraverso i tasti "destra", "sinistra" e "centrale", mappati sulla scheda TahoeII che stiamo utilizzando, all'interno del tastierino direzionale riportato nella parte inferiore destra della scheda stessa. Ciascuna finestra relativa ad una parte del programma dimostrativo deriva da una classe comune, definita all'interno della medesima applicazione, denominata

Listato 8

```
1: public Window CreateWindow()
2: {
3:     Window wnd;
4:
5:     // Creazione di una finestra con le stesse dimensioni del display
6:     wnd = new Window();
7:     wnd.Height = SystemMetrics.ScreenHeight;
8:     wnd.Width = SystemMetrics.ScreenWidth;
9:
10:    // Creazione di un oggetto di tipo Text
11:    Text text = new Text();
12:
13:    text.Font = Resources.GetFont(Resources.FontResources.small);
14:    text.TextContent = Resources.GetString(Resources.StringResources.String1);
15:    text.HorizontalAlignment = Microsoft.SPOT.Presentation.HorizontalAlignment.Center;
16:    text.VerticalAlignment = Microsoft.SPOT.Presentation.VerticalAlignment.Center;
17:
18:    // Aggiunta dell'oggetto Text alla gerarchia della finestra
19:    wnd.Child = text;
20:
21:    // Aggiunta di un gestore di evento per l'evento ButtonUp
22:    wnd.AddHandler(Buttons.ButtonUpEvent, new ButtonEventHandler(OnButtonUp), false);
23:
24:    // La finestra viene effettivamente mostrata
25:    wnd.Visibility = Visibility.Visible;
26:
27:    // L'input di tipo "Button" viene ridirezionato su questa finestra
28:    Buttons.Focus(wnd);
29:
30:    return wnd;
31: }
```

'PresentationWindow'. La derivazione comune di più tipi di finestra da una stessa base permette, come avviene in generale per mezzo dell'ereditarietà dei linguaggi orientati ad oggetti, di condividere caratteristiche e comportamenti comuni, come il fatto di inicializzarsi con la massima dimensione disponibile sul display e supportare l'operazione comune relativa alla pressione di un pulsante, nel caso del codice dimostrativo in questione.

I vari sotto-demo illustrati mostrano ciascuno un aspetto peculiare dello sviluppo di interfacce utente attraverso gli strumenti del framework. A titolo di esempio, prendiamo in esame il sotto-demo relativo all'utilizzo del controllo 'Stack Panel', illustrato in **Fig 6**. Il codice corrispondente è nel **Listato 11**.

Vediamo sinteticamente gli aspetti salienti che costituiscono la soluzione appena illustrata.

- (riga 1) La finestra di questa sotto-demo eredita dalla 'PresentationWindow' illustrata in precedenza.
- (righe 3-17) Viene definita una classe an-

Listato 9

```
1: private void OnButtonUp(object sender, ButtonEventArgs e)
2: {
3:     // Visualizza il codice del pulsante nella finestra di debug
4:     Debug.Print(e.Button.ToString());
5: }
```


Listato 10

nidata che eredita dalla classe 'Shape'; l'ereditarietà consente alla nostra forma personalizzata di ereditare tutto ciò che è comune alle altre forme, pur mantenendo la flessibilità di ridefinire l'aspetto in cui è specializzata, effettuando quello che viene descritto come "overriding" del metodo OnRender(), in cui avviene la produzione del disegno vero e proprio.

- (righe 20-36) Nel costruttore della classe StackPanelDemo, viene creato uno StackPanel, ossia un contenitore grafico che implementa un layout secondo il quale gli oggetti figli vengono riportati uno sotto l'altro (in caso di orientamento verticale) o uno accanto all'altro (in caso di orientamento orizzontale), seguendo l'ordine di inserimento; il pannello viene poi aggiunto come primo ed unico figlio dell'oggetto Window corrente.

Nella seconda parte del costruttore viene poi popolato un array di oggetti di tipo Shape, in realtà costituito da forme di tipo diverso (un ellisse, una linea, un quadrato, un rettangolo ed un oggetto del nostro tipo personalizzato, ossia la croce). Come ultima operazione viene scorso l'array in questione per aggiungere ciascuna Shape al pannello. Lo StackPanel

```

1: public sealed class GPIOButtonInputProvider
2: {
3:     public readonly Dispatcher Dispatcher;
4:
5:     private ButtonPad[] buttons;
6:     private DispatcherOperationCallback callback;
7:     private InputProviderSite site;
8:     private PresentationSource source;
9:
10:    public GPIOButtonInputProvider(PresentationSource source)
11:    {
12:        // memorizziamo la sorgente dell'input
13:        this.source = source;
14:
15:        // registriamo questo oggetto come sorgente di "input" e abbiamo
16:        // indietro un "InputProviderSite", che inoltra le segnalazioni
17:        // di input all'InputManager, che a sua volta colloca l'input nell'
18:        // area di "staging"
19:        site = InputManager.CurrentInputManager.RegisterInputProvider(this);
20:
21:        // Creiamo un metodo di callback che, a seguito di un "report"
22:        // di segnalazione di basso livello proveniente da un interrupt,
23:        // inoltra la notifica all'InputProviderSite
24:        callback = new DispatcherOperationCallback(delegate(object report)
25:        {
26:            InputReportArgs args = (InputReportArgs)report;
27:            return site.ReportInput(args.Device, args.Report);
28:        });
29:        Dispatcher = Dispatcher.CurrentDispatcher;
30:
31:        // Creiamo un provider hardware
32:        HardwareProvider hwProvider = new HardwareProvider();
33:
34:        // definiamo i pin relativi ai GPIO collegati ai pulsanti
35:        // con il default dell'emulatore
36:        Cpu.Pin pinLeft = Cpu.Pin.GPIO_Pin0;
37:        Cpu.Pin pinRight = Cpu.Pin.GPIO_Pin1;
38:        Cpu.Pin pinUp = Cpu.Pin.GPIO_Pin2;
39:        Cpu.Pin pinSelect = Cpu.Pin.GPIO_Pin3;
40:        Cpu.Pin pinDown = Cpu.Pin.GPIO_Pin4;
41:
42:        // Se non siamo sull'emulatore, usiamo la mappatura del
43:        // HardwareProvider appena definito
44:        if ((pinLeft = hwProvider.GetButtonPins(Button.VK_LEFT)) ==
45:            Cpu.Pin.GPIO_NONE)
46:            pinLeft = Cpu.Pin.GPIO_Pin0;
47:        else
48:        {
49:            pinRight = hwProvider.GetButtonPins(Button.VK_RIGHT);
50:            pinUp = hwProvider.GetButtonPins(Button.VK_UP);
51:            pinSelect = hwProvider.GetButtonPins(Button.VK_SELECT);
52:            pinDown = hwProvider.GetButtonPins(Button.VK_DOWN);
53:        }
54:
55:        // Definiamo un insieme di oggetti ButtonPad come array
56:        // di oggetti che legano il GPIOButtonInputProvider, il codice "virtuale"
57:        // del pulsante e il pin associato
58:        ButtonPad[] buttons = new ButtonPad[]
59:        {
60:            // Associate the buttons to the pins as discovered or set above
61:            new ButtonPad(this, Button.VK_LEFT, pinLeft),
62:            new ButtonPad(this, Button.VK_RIGHT, pinRight),
63:            new ButtonPad(this, Button.VK_UP, pinUp),
64:            new ButtonPad(this, Button.VK_SELECT, pinSelect),
65:            new ButtonPad(this, Button.VK_DOWN, pinDown),
66:        };
67:
68:        this.buttons = buttons;

```

è solo uno dei contenitori inclusi nel framework e altri possono essere aggiunti anche attraverso la creazione di classi derivate da Panel; l'aspetto più rilevante da tenere a mente è che nell'ambito della gerarchia

```

69:     }
70:
71:     internal class ButtonPad : IDisposable
72:     {
73:         private Button button;           // Codice del pulsante
74:         private InterruptPort port;      // Porta che genera gli interrupt di notifica
75:         private GPIOButtonInputProvider sink;
76:         private ButtonDevice buttonDevice;
77:
78:         public ButtonPad(GPIOButtonInputProvider sink, Button button,
79:             Cpu.Pin pin)
80:         {
81:             this.sink = sink;
82:             this.button = button;
83:             this.buttonDevice = InputManager.CurrentInputManager.ButtonDevice;
84:
85:             // Se stiamo su un hardware vero e non emulato
86:             // creiamo una porta interrupt che notifica
87:             // di entrambi i tipi di variazione
88:             if (pin != Cpu.Pin.GPIO_NONE)
89:             {
90:                 // When this GPIO pin is true, call the Interrupt method.
91:                 port = new InterruptPort(pin, true,
92:                     Port.ResistorMode.PullUp,
93:                     Port.InterruptMode.InterruptEdgeBoth);
94:                 port.OnInterrupt += new NativeEventHandler(this.Interrupt);
95:             }
96:         }
97:
98:         protected virtual void Dispose(bool disposing)
99:         {
100:             if (disposing)
101:             {
102:                 if (port != null)
103:                 {
104:                     port.Dispose();
105:                     port = null;
106:                 }
107:             }
108:         }
109:
110:         public void Dispose()
111:         {
112:             Dispose(true);
113:             GC.SuppressFinalize(this);
114:         }
115:
116:         void Interrupt(uint data1, uint data2, DateTime time)
117:         {
118:             // Intercetto la variazione dell'ingresso digitale
119:             // e identifico il tipo di azione effettivamente svolta
120:             RawButtonActions action = (data2 != 0) ?
121:                 RawButtonActions.ButtonUp : RawButtonActions.ButtonDown;
122:
123:             // Creao un nuovo report di segnalazione
124:             RawButtonInputReport report = new RawButtonInputReport(
125:                 sink.source, time, button, action);
126:
127:             // accodo l'evento relativo all'interrupt verso il nostro InputProviderSite
128:             // utilizzando l'oggetto Dispatcher per poter effettuare
129:             // la chiamata della callback da un altro thread
130:             // (il thread dell'interrupt e quello che gestisce la
131:             // interfaccia utente sono diversi)
132:             sink.Dispatcher.BeginInvoke(sink.callback, new InputReportArgs(buttonDevice,
133:                 report));
134:         }
135:     }

```

del layer Presentation alcuni oggetti grafici non consentono l'inclusione di oggetti figli, altri consentono l'inclusione di un unico elemento figlio (come la Window) ed altri consentono invece l'inclusione di un nume-

ro arbitrario di figli (come per tutti gli oggetti derivati da Panel).

Il codice appena preso in esame viene utilizzato anche nel caso del sotto-demo "Horizontal Stack Panel", in cui gli oggetti Shape sono riportati in orizzontale anziché in verticale, sfruttando la proprietà dello StackPanel valorizzata alla riga 23 del codice appena esaminato.

Per un esempio di personalizzazione di un controllo contenitore di tipo "Panel", è possibile fare riferimento al sotto-demo "Scrollable Panel", la cui interfaccia utente è illustrata nella Fig 7. Tale applicazione illustra come creare una superficie grafica virtualmente molto più estesa del display fisico che la visualizza, tramite la definizione di un controllo personalizzato denominato "TextScrollView".

TOUCH-SCREEN

La gestione del touch-screen all'interno del sistema "Presentation" è

estremamente semplificato dal modello ad eventi di chiara ispirazione Windows. In particolare, come è nel demo denominato "StylusCapture", anch'esso installato dal setup del .NET micro Framework SDK, gli oggetti

Fig. 5

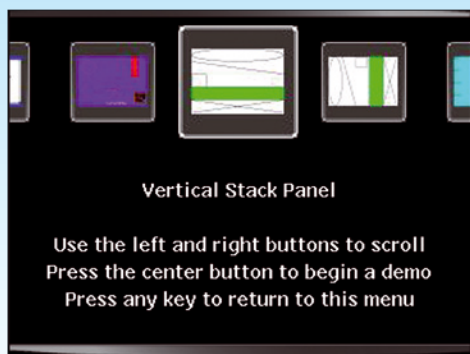


Fig. 6

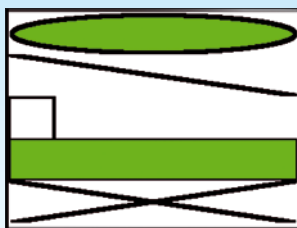
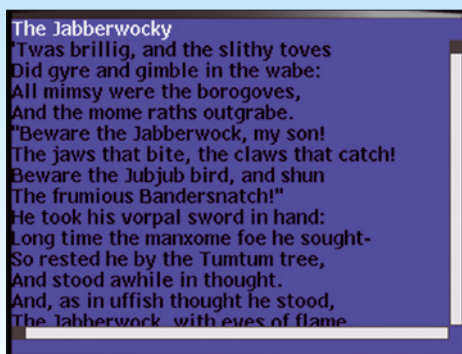


Fig. 7



Listato 11

```

1: internal sealed class StackPanelDemo : PresentationWindow
2: {
3:     private sealed class Cross : Shape
4:     {
5:         public Cross() { }
6:
7:         // Ridefiniamo il metodo OnRender della classe "UIElement"
8:         // ereditata tramite Window
9:         public override void OnRender(DrawingContext dc)
10:        {
11:            // Linea diagonale 1
12:            dc.DrawLine(base.Stroke, 0, 0, Width, Height);
13:
14:            // Linea diagonale 2
15:            dc.DrawLine(base.Stroke, Width, 0, 0, Height);
16:        }
17:    }
18:
19:    // Costruttore
20:    public StackPanelDemo(MySimpleWPFApplication program, Orientation orientation)
21:        : base(program)
22:    {
23:        StackPanel panel = new StackPanel(orientation);
24:        this.Child = panel;
25:        panel.Visibility = Visibility.Visible;
26:
27:        // Array delle forme geometriche da disegnare
28:        Shape[] shapes = new Shape[] {
29:            new Ellipse(0, 0,
30:            new Line(),
31:            // Quadrato
32:            new Polygon(new Int32[] { 0, 0, 50, 0, 50, 50, 0, 50 }),
33:            new Rectangle(),
34:            // Forma personalizzata (croce)
35:            new Cross()
36:        };
37:
38:        // Aggiungo le forme alla gerarchia visuale
39:        for (int x = 0; x < shapes.Length; x++)
40:        {
41:            Shape s = shapes[x];
42:            s.Fill = new SolidColorBrush(ColorUtility.ColorFromRGB(0, 255, 0));
43:            s.Stroke = new Pen(Color.Black, 2);
44:            s.Visibility = Visibility.Visible;
45:            s.HorizontalAlignment = HorizontalAlignment.Center;
46:            s.VerticalAlignment = VerticalAlignment.Center;
47:            s.Height = Height - 1;
48:            s.Width = Width - 1;
49:
50:            if (panel.Orientation == Orientation.Horizontal)
51:                s.Width /= shapes.Length;
52:            else
53:                s.Height /= shapes.Length;
54:
55:            panel.Children.Add(s);
56:        }
57:    }
58: }

```

grafici definiti all'interno della gerarchia visuale espongono una serie di eventi relativi alle singole azioni rilevabili, quali ad esempio "TouchDown" o "TouchUp", rispettivamente sollevati da un elemento visuale quando questo viene "toccato" o "rilasciato". Per sottoscrivere la nostra applicazione alla notifica di questo tipo di eventi, è sufficiente utilizzare i comandi riportati nel **Listato 12**. Una volta intercettato l'evento è possibile compiere qualsiasi tipo di azione, come cambiare lo stato dell'elemento visuale "toccato" o attivare delle specifiche funzioni hardware. Qualora fosse necessario che il controllo che rileva il primo evento di TouchDown continui a ricevere notifiche dal touchscreen anche quando l'area impegnata dal tocco non si sovrappone più con la superficie del controllo, è possibile utilizzare la cosiddetta funzionalità di "cattura", come illustrato nel **Listato 13**. In maniera analoga, è possibile intercettare gli eventi relativi al touch screen all'interno di un oggetto Window o derivato ridefinendo tramite "override" i metodi OnTouchUp() e OnTouchDown(), come illustrato nel **Listato 14**. L'assegnazione fatta a titolo di esempio a riga 7 consente di intervenire nel processo

di “bubbling” degli eventi gestito dal sistema “Presentation”. Per *bubbling* si intende la politica di gestione delle notifiche degli eventi relativi all’interazione con elementi visuali; poiché infatti è possibile che un evento venga rilevato da un elemento ma debba essere notificato anche al suo contenitore, o al contenitore ancora più esterno o all’oggetto Window che fa da involucro “root” della gerarchia, il framework utilizza il concetto di RoutedEvent. Quest’ultimo è un oggetto il cui compito principale è veicolare una notifica legata ad una qualche tipo di azione effettuata su un elemento visuale secondo una delle seguenti tre “strategie”:

- Bubble; in cui l’evento sale a galla della gerarchia, dall’elemento più interno al contenitore più esterno;
- Direct; in cui l’evento non viene effettivamente instradato ma arriva solo al target principale;
- Tunnel; in cui l’evento scende nella gerarchia, dall’elemento più esterno a quello più interno.

Mentre lo scopo delle prime due strategie è abbastanza semplice da intuire, la Tunnel ha una finalità più difficile da cogliere; essa viene utilizzata prevalentemente nelle notifiche di tipo “Preview” in cui a ciascun contenitore viene data l’opportunità di intercettare la notifica dell’evento del proprio “sottoalbero”. L’interazione con il processo di “bubbling” o di “tunnelling” avviene tramite la proprietà *Handled*, esposta dalla classe base che descrive il parametro che accompagna ogni *RoutedEvent*: *RoutedEventArgs*. Come ultimo esempio, vale la pena di soffermarci sulle immagini relative all’applicazione dimostrativa della più potente libreria disponibile per .NET Micro Framework in merito allo sviluppo di interfacce utente: *MFRichMediaExtensions* della Innobedded, società fondata da Jens Kühner, autore del miglior testo in circolazione in materia di .NET Micro Framework, giunto alla sua seconda edizione. Tale libreria, scaricabile all’indirizzo <http://www.innobedded.com/download.html>, è utilizzabile in modalità demo senza limiti di tempo, seppur con frequenti interruzioni da parte di una bitmap che agisce come “nag screen” raffigurante il logo della Innobedded. La licenza d’uso della

Listato 12

```
1: Text text1 = new Text();
2: text1.TouchDown += new TouchEventHandler(Text_TouchDown);
3: text1.TouchUp += new TouchEventHandler(Text_TouchUp);
4:
5: void Text_TouchDown(object sender, TouchEventArgs e)
6: {
7:     // ...
8: }
9:
10: void Text_TouchUp(object sender, TouchEventArgs e)
11: {
12:     // ...
13: }
14:
```

Listato 13

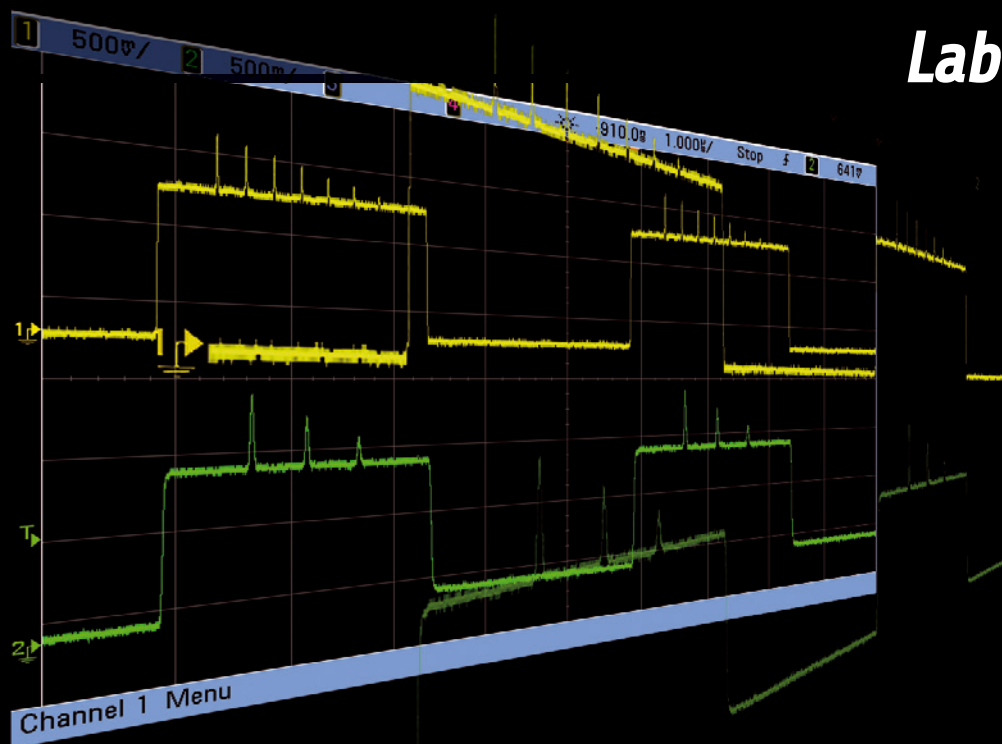
```
1: // Cattura...
2: TouchCapture.Capture(text);
3:
4: // Rilascio...
5: TouchCapture.Capture(text, CaptureMode.None);
```

Listato 14

```
1: protected override void OnTouchDown(TouchEventArgs e)
2: {
3:     base.OnTouchDown(e);
4:
5:     // ...
6:
7:     e.Handled = true;
8: }
9:
10: protected override void OnTouchUp(TouchEventArgs e)
11: {
12:     base.OnTouchDown(e);
13:
14:     // ...
15:
16: }
```

libreria ha un prezzo molto vantaggioso se utilizzata per scopi non commerciali ed in tutti i casi semplifica radicalmente lo sviluppo di un’interfaccia utente che sia al contempo funzionale ed esteticamente appetibile, di cui vedete un esempio nella Fig. 8.





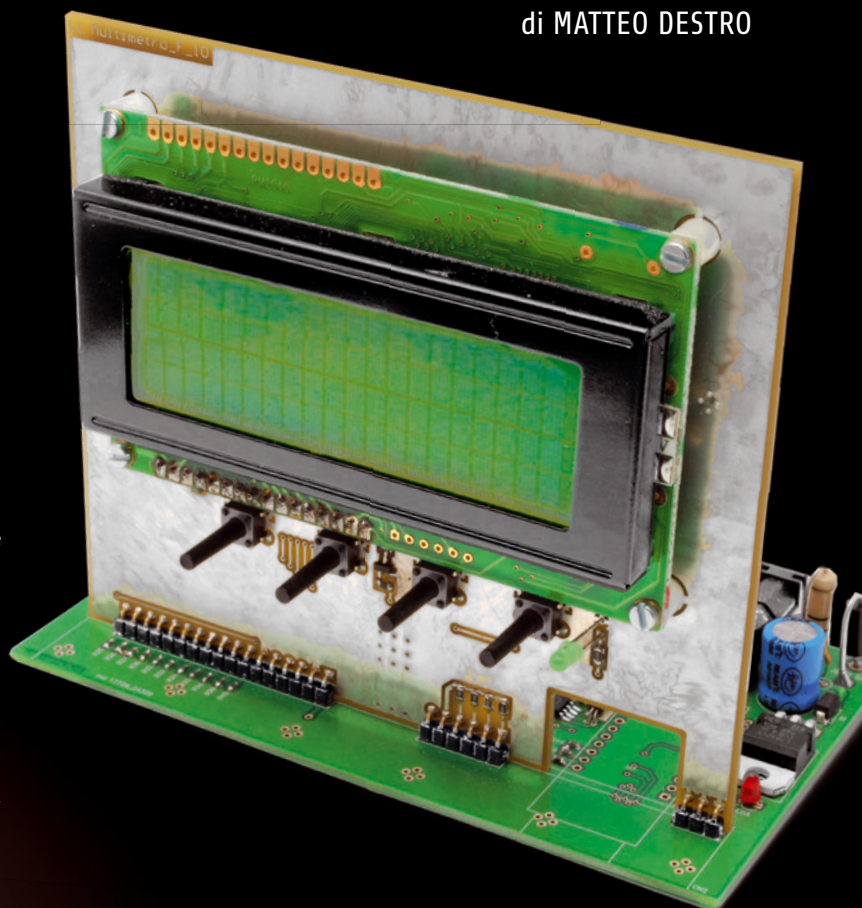
Misura tensioni e correnti ed è in grado di segnalare e memorizzare il superamento di soglie liberamente impostabili mediante pulsanti. I risultati appaiono su un display 20x4. Prima puntata.

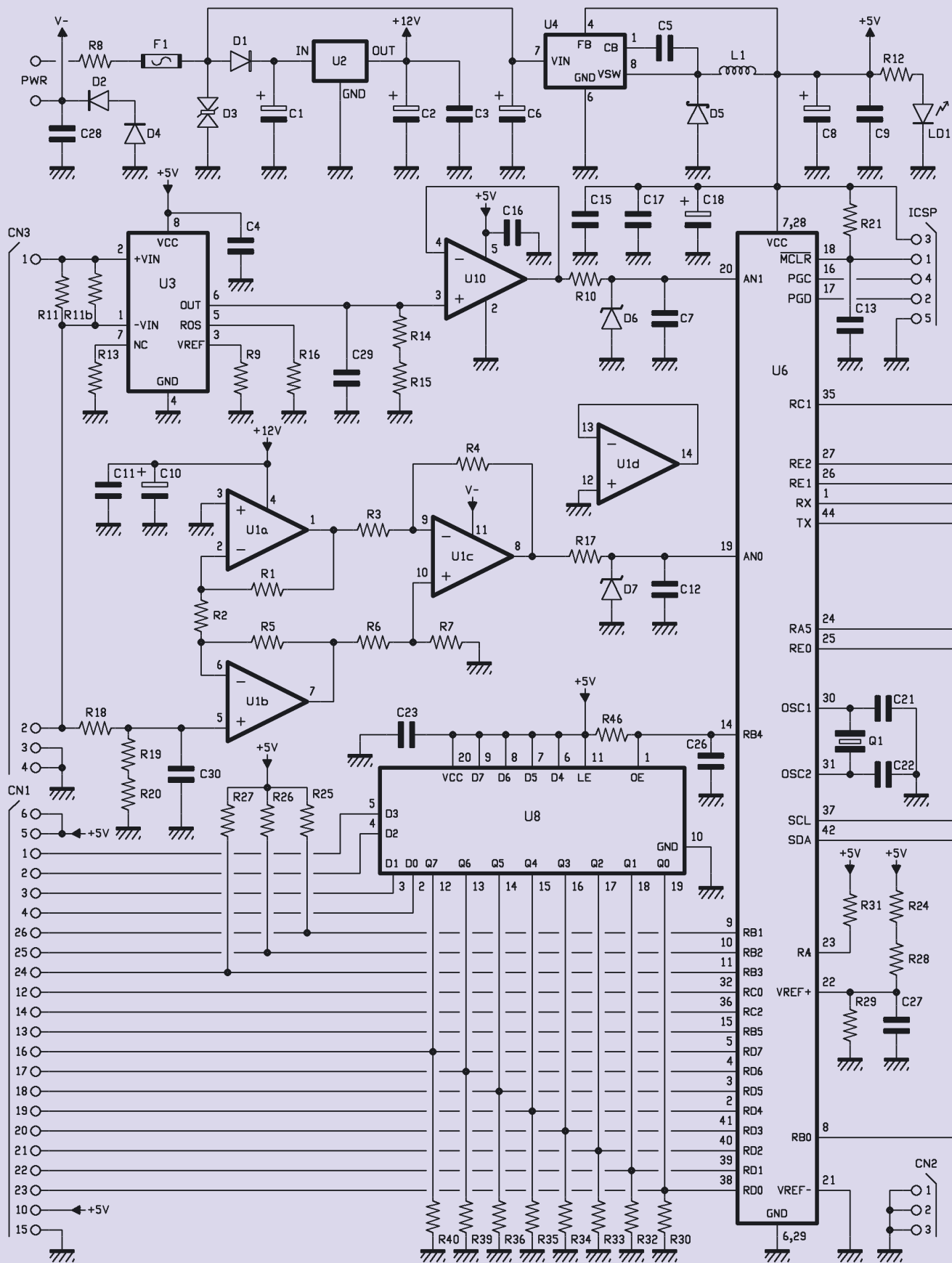
MULTIMETRO DIGITALE

di MATTEO DESTRO

Il multimetro, sia esso palmare o da banco, è uno strumento indispensabile per l'elettrotecnico come per il tecnico elettronico. Se è vero che in commercio oggi si trovano versioni a prezzi ridottissimi, è vero altrettanto che un buon multimetro si paga bene, soprattutto se parliamo di uno strumento professionale da banco provvisto di memoria e capace di rilevare valori di cresta e quant'altro. Quindi pensiamo sia una buona cosa pubblicarne un progetto valido ed al tempo stesso alla portata

di tutti; quello che trovate in queste pagine è uno strumento digitale per la misura di tensione, corrente e potenza di segnali in corrente continua. Praticamente si tratta di un piccolo multimetro con possibilità di memorizzare eventuali allarmi di superamento di soglie di tensione, corrente e potenza. Le soglie possono essere selezionate tramite apposito menu e memorizzate nella EEPROM del microcontrollore che governa l'unità. Il range di tensioni misurabili dallo strumento varia da un minimo di 0





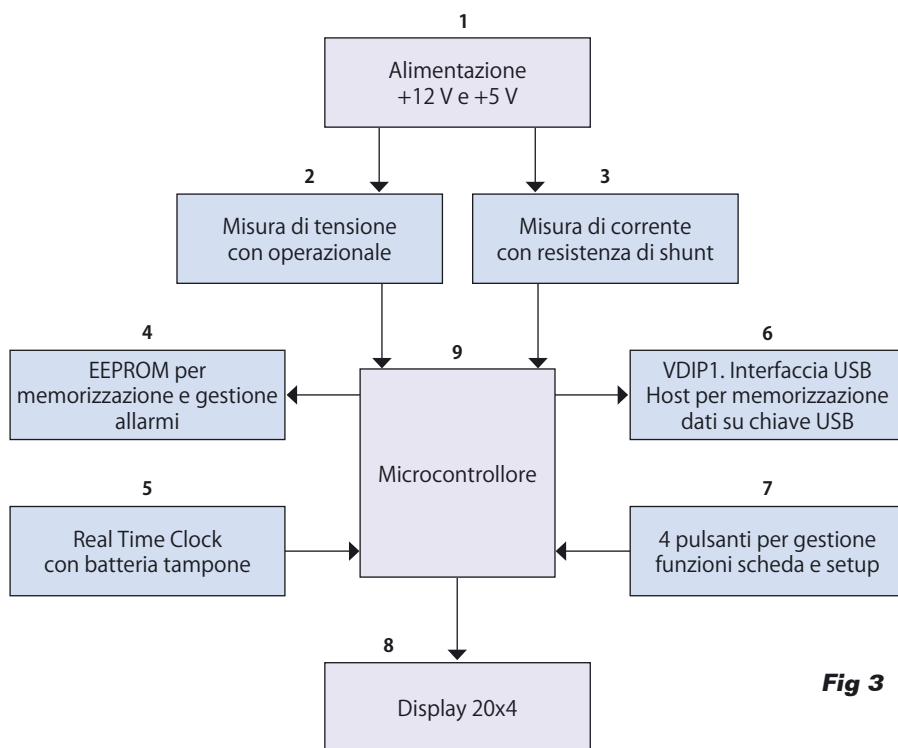
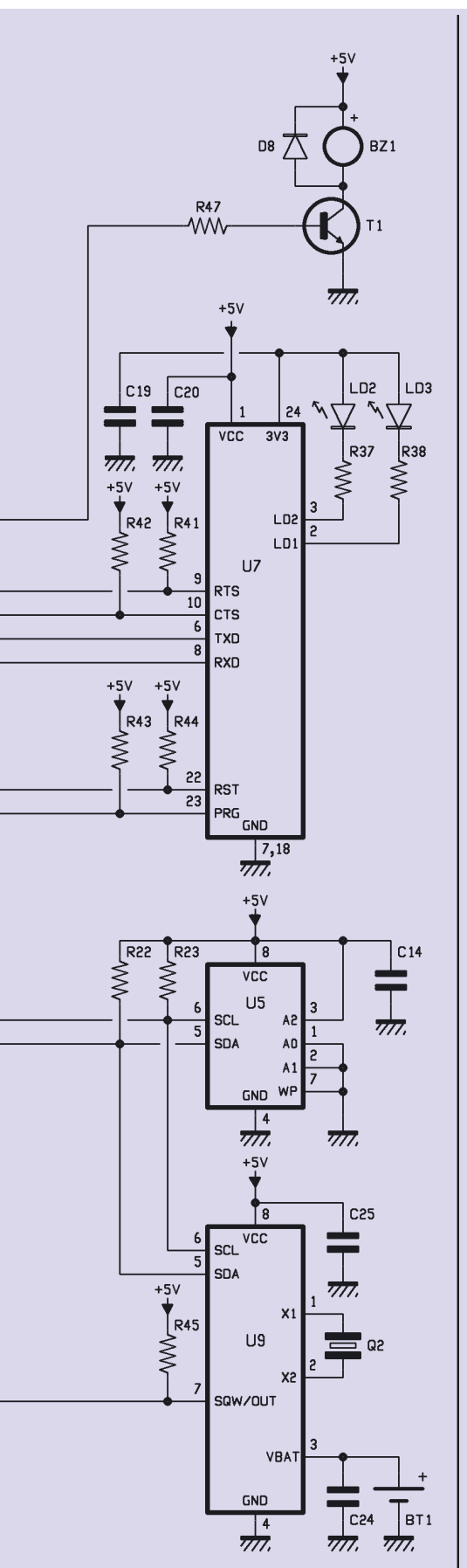


Fig 3

a un massimo di +36 Vcc. Per quanto riguarda la corrente, la massima intensità rilevabile si attesta a +5A. Le misure in corrente e tensione rendono possibile il calcolo della potenza istantanea data dal prodotto della tensione (V) rispetto alla corrente (I).

IL CIRCUITO

Lo schema elettrico del multi-metro è suddiviso in due parti, la prima delle quali è la scheda base dove risiedono i circuiti di alimentazione, i circuiti di misura, l'orologio, la EEPROM e il microcontrollore, nonché lo zoccolo per l'inserzione del circuito VDIP per la gestione dell'interfaccia USB di tipo Host. La seconda scheda (*pannello*) comprende i pulsanti e il display LCD da 20 caratteri per quattro righe.

La **Fig. 1** illustra lo schema a blocchi del multimetro; partiamo dalla trattazione della sezione di alimentazione (blocco 1) la quale deve generare i +12 Vcc necessari ad alimentare l'amplificatore per strumentazione e i +5 V necessari al resto dell'elettronica presen-

te sulla scheda. La tensione di alimentazione +5 V viene ricavata da un regolatore di tipo switching. La scelta dello switching nasce dal fatto che tale tensione, oltre ad alimentare il 90 % dell'elettronica della scheda, deve anche provvedere a un display alfanumerico da 20 caratteri per 4 righe con relativa retroilluminazione, la quale richiede una certa quantità di corrente.

Il blocco 2 esegue una misura della tensione presente sul carico, effettuata utilizzando un operazionale per strumentazione; il 3 si occupa della misura della corrente che fluisce nel carico (la misura si compie con una resistenza di shunt). Il blocco 4 evidenzia la sezione per la memorizzazione e gestione degli allarmi con relativo buzzer. Quando la tensione di alimentazione, la corrente o la potenza, superano o scendono sotto la soglia impostata, si scatena un evento che viene memorizzato in EEPROM. Se impostato, verrà attivato il buzzer ad indicare che una delle soglie è stata superata. Per salvare gli allarmi è neces-

Il microcontrollore PIC18F4523:

Il PIC18F4523 fa parte di una famiglia di microcontrollori che si differenziano gli uni dagli altri per il numero di periferiche disponibili e la quantità di memoria Flash, RAM e EEPROM messa a disposizione. Attualmente, senza la gestione del modulo VDIP, il firmware implementato occupa 13.090 istruzioni di programma (equivalenti a 26.180 byte) e un quantitativo di RAM pari a 646 byte. Le altre caratteristiche del microcontrollore sono i 36 pin di I/O messi a disposizione, i quali possono essere configurati per funzionare da semplici ingressi o uscite digitali, oppure con le periferiche loro associate; tra queste ultime ricordiamo il convertitore ADC da 12 bit su 13 canali, due comparatori, una periferica I²C o SPI, una periferica EUSART, più tre timer a 16 bit ed uno ad 8 bit.

La frequenza massima di lavoro consentita è 40 MHz, che si ottiene usando un quarzo da 10 MHz con il PLL interno che moltiplica tale frequenza per quattro; così facendo, si ottiene una

Device	Program Memory		Data Memory		I/O	12-Bit A/D (ch)	CCP/ ECCP (PWM)	MSSP		EUSART	Comp.	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)				SPI	Master I ² C™			
PIC18F2423	16K	8192	768	256	25	10	2/0	Y	Y	1	2	1/3
PIC18F2523	32K	16384	1536	256	25	10	2/0	Y	Y	1	2	1/3
PIC18F4423	16K	8192	768	256	36	13	1/1	Y	Y	1	2	1/3
PIC18F4523	32K	16384	1536	256	36	13	1/1	Y	Y	1	2	1/3

potenza di calcolo pari a 10 MIPS, più che sufficiente per l'applicazione in esame.

A queste caratteristiche si sommano quelle base di ogni microcontrollore della serie 18F, le quali comprendono un modulo per eseguire le moltiplicazioni in hardware, un watchdog program-mabile, un BrownOut Reset programmabile, la possibilità di programmare e debuggare il microcontrollore in-circuit e, non meno importante, la possibilità di scrivere il firmware utilizzando un linguaggio ad alto livello come il C. Infatti l'architettura della famiglia di microcontrollori 18F è stata progettata per ottenere la massima ottimizzazione possibile dalla programmazione ad alto livello con linguaggio C.

saria l'informazione di data e ora dell'avvenuta rilevazione dell'anomalia. Per fare ciò si utilizza un real-time clock con relativa batteria tampone (Blocco 5). Gli eventuali allarmi possono anche essere salvati su chiavetta USB usando l'interfaccia VDIP evidenziata dal blocco 6; tale funzione non è ancora implementata, ma sarà resa disponibile come espansione futura.

I blocchi 7 e 8 raggruppano la sezione pulsanti più display alfanumerico: la scheda dispone di quattro pulsanti necessari a

gestire tutte le funzioni messe a disposizione. Lo scopo principale dei pulsanti è la navigazione all'interno del menu di gestione della scheda, il quale sarà discusso in dettaglio nella seconda puntata.

L'ultimo blocco (9) è il cuore del sistema, rappresentato dal microcontrollore PIC18F4523, che fornisce a corredo un convertitore ADC da 12 bit più tutte le restanti periferiche tipiche di questa famiglia di microcontrollori. Il compito del microcontrollore consiste nella gestione di tutte le funzioni

messe a disposizione dalla scheda. Possiamo quindi riassumere le proprietà fondamentali dello strumento:

- 1 ingresso analogico per la misura della tensione (massimo 36 Vcc);
- 1 ingresso analogico per la misura di corrente (corrente massima 5 A);
- 1 clock calendar con batteria tampone per gestione di data, giorno e ora;
- 1 buzzer per avviso acustico in caso di anomalia;
- 1 memoria EEPROM, su bus I²C, da 128 kbyte, per la registrazione degli eventi; gli eventi memorizzati possono essere visualizzati sul display nell'apposito menu di cui si discuterà in seguito;
- 1 porta USB di tipo Host gestita dal modulo VDIP1 FTDI (per futuri sviluppi);
- 4 pulsanti per gestione menu e configurazione del multimetro;
- 1 display alfanumerico da 4 righe per 20 caratteri;
- 4 input di diagnostica.

SCHEMA ELETTRICO

Cominciamo la descrizione dell'hardware dalla scheda base, la quale è stata suddivisa in due

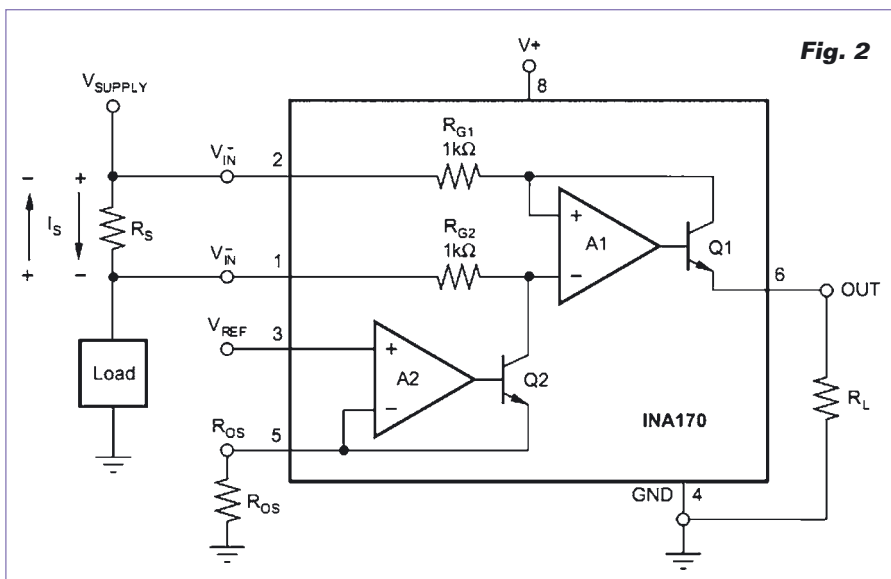


Fig. 2

sezioni; la prima è quella analogica comprendente gli stadi di misura di tensione e corrente più le alimentazioni, mentre la seconda contiene tutta la parte digitale e quindi il microcontrollore, la EEPROM, l'orologio, il buzzer e l'espansione VDIP.

Analizziamo lo schema elettrico della parte analogica iniziando dalla sezione di alimentazione. Come di consueto, ritroviamo una protezione elettronica contro le sovratensioni e sovracorrenti formata dalla resistenza R8, il termofusibile F1 e il TVS (*Transient Voltage Suppressor*) bidirezionale siglato D3. Il TVS serve per sopprimere le tensioni sopra i ± 33 V. Nel caso in cui si verifichi una sovratensione, il diodo D3 cortocircuiterà la linea a massa provocando l'assorbimento da parte del circuito di una grande quantità di corrente, la quale farà intervenire il termofusibile autoripristinante F1, il quale aprendo il circuito proteggerà tutta l'elettronica a valle della protezione.

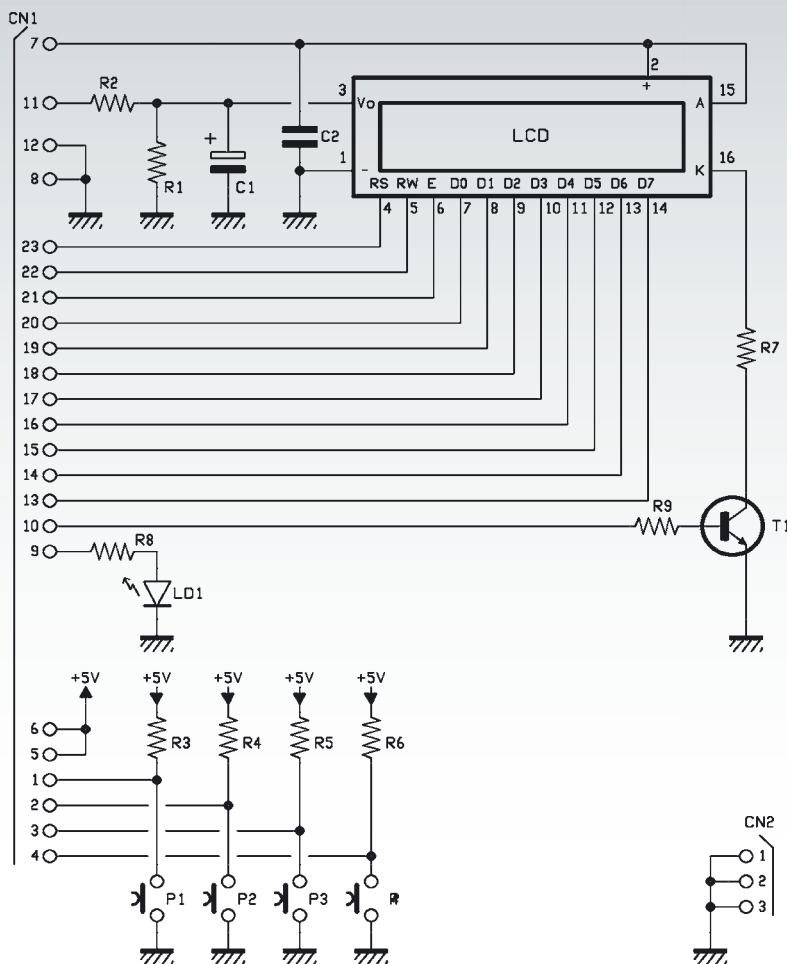
Il plug di alimentazione PWR vede collegato il polo positivo (pin 1) alla resistenza R8, mentre il polo negativo (pin 2) è connesso alla massa della scheda indirettamente, mediante i diodi D2 e D4, che determinano uno scostamento rispetto alla massa di -1,4 V. Tale tensione viene riportata al pin 11 dell'integrato U1 (alimentazione negativa).

Questo approccio si rende necessario perché, per eseguire una misura corretta della tensione prossima agli 0 V, l'integrato U1 necessita di un'alimentazione negativa oltre a quella positiva di +12 Vcc. Solitamente in questi casi si utilizza un'alimentazione duale da ± 12 Vcc per eseguire delle misure di tensione sia positive che negative, ma siccome la più bassa tensione che il nostro strumento deve misurare è 0 V, non è

necessario, per il ramo negativo, arrivare a -12 Vcc, ma bastano -1,4 Vcc, facilmente ottenibili con l'accorgimento di montare i due diodi sopra massa. Questo approccio permette un risparmio della componentistica necessaria alla realizzazione dell'alimentatore e quindi anche un risparmio economico. Se avessimo dovuto

misurare tensioni comprese tra ± 10 Vcc, allora la tensione di -1,4 Vcc non sarebbe stata sufficiente e avremmo dovuto progettare una sezione di alimentazione per generare i -12 Vcc.

La tensione di alimentazione +12 Vcc viene generata utilizzando l'integrato U2 (LM7812), il cui pin 2 è collegato alla massa del circuit-



Riquadro 1

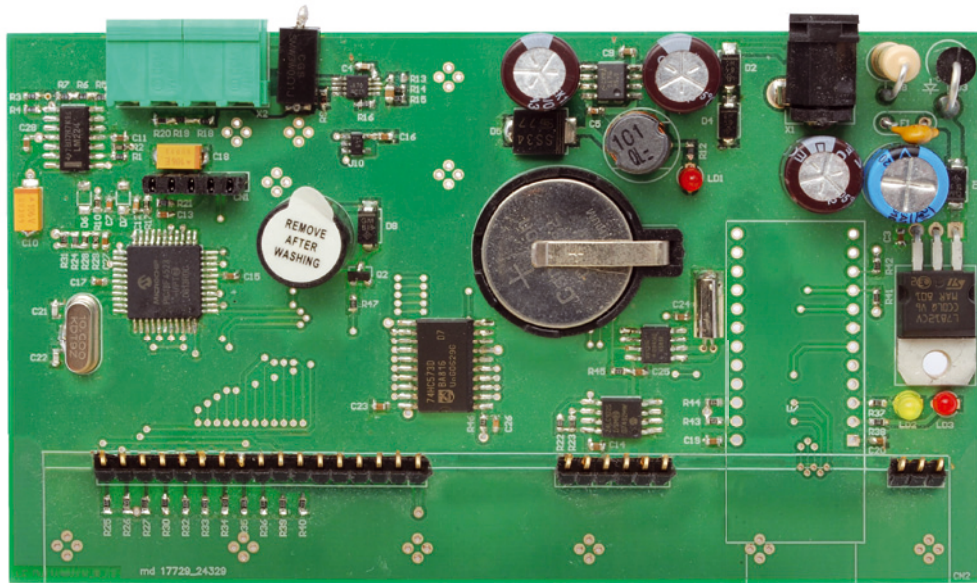
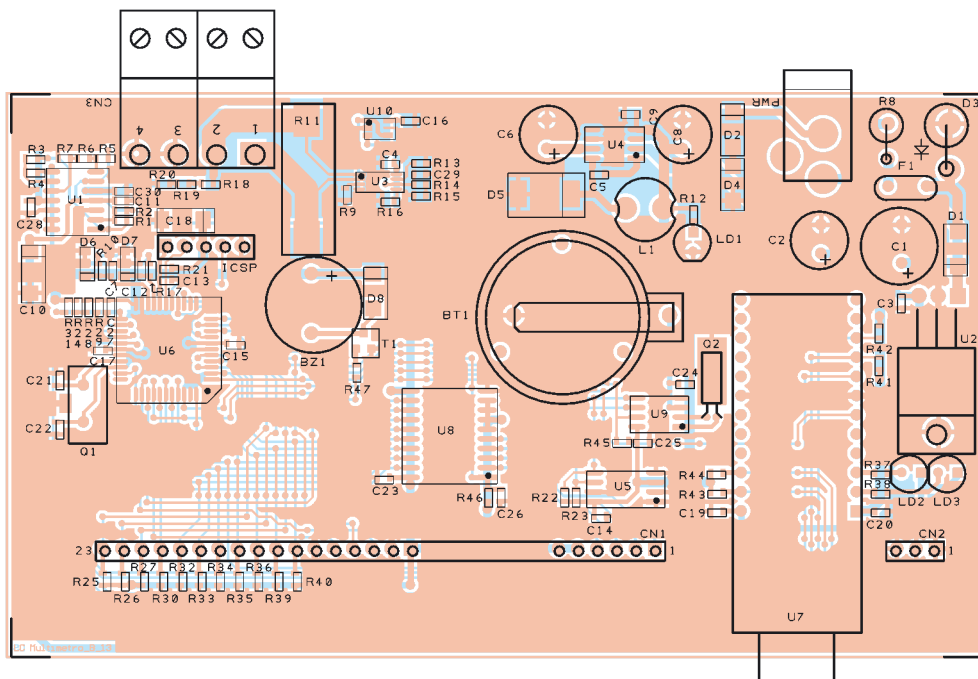
D	M	Y	HH	MM	SS	TH	TYPE	BZ	VALUE
1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	7 Byte

- ◆ D: Giorno. Indica il giorno di registrazione dell'allarme
- ◆ M: Mese. Indica il mese di registrazione dell'allarme
- ◆ Y: Anno. Indica l'anno di registrazione dell'allarme
- ◆ HH: Ora. Indica l'ora rispetto alla quale è stato registrato l'allarme
- ◆ MM: Minuti. Indica i minuti rispetto ai quali è stato registrato l'allarme
- ◆ SS: Secondi. Indica i secondi rispetto ai quali è stato registrato l'allarme
- ◆ TH: Soglia. Indica se è stata superata la soglia superiore (U), inferiore (D) o se il parametro è rientrato nella norma (O)
- ◆ TYPE: Indica il tipo di allarme. In tensione (V), in corrente (I) oppure di potenza (W)
- ◆ BZ: Buzzer. Se il valore è 1 buzzer attivo quando rilevato allarme, se 0 buzzer non attivo quando rilevato allarme
- ◆ VALUE: Valore di tensione, corrente o potenza memorizzato.

[piano di montaggio **BASE**]

Elenco Componenti:

- R1: 100 kohm (0603)
- R2: 200 kohm (0603)
- R3, R4, R5, R6, R7: 100 kohm (0603)
- R8: 10 ohm 2 W
- R9: 0 ohm (0603)
- R10: 10 ohm (0603)
- R11, R11b: 0,1 ohm 5 W
- R12: 680 ohm (0603)
- R13: 100 kohm (0603)
- R14: 4,7 kohm (0603)
- R15: 1 kohm (0603)
- R16: 100 kohm (0603)
- R17: 10 ohm (0603)
- R18: 1 Mohm (0603)
- R19: 47 kohm (0603)
- R20: 12 kohm (0603)
- R21: 22 kohm (0603)
- R22, R23, R24: 2,2 kohm (0603)
- R25, R26, R27: 4,7 kohm (0603)
- R28: 330 ohm (0603)
- R29: 10 kohm (0603)
- R30, R31, R32, R33, R34, R35, R36: 4,7 kohm (0603)
- R37, R38: 330 ohm (0603)
- R39, R40, R41, R42, R43, R44, R45, R46: 4,7 kohm (0603)
- R47: 1 kohm (0603)
- C1: 220 μ F 50 VL elettrolitico
- C2: 100 μ F 50 VL elettrolitico
- C3, C4: 100 nF ceramico (0603)
- C5: 10 nF ceramico (0603)
- C6: 100 μ F 50 VL elettrolitico
- C7: 220 pF ceramico (0603)
- C8: 100 μ F 50 VL elettrolitico
- C9: 100 nF ceramico (0603)
- C10: 10 μ F 25 VL elettrolitico
- C11: 100 nF ceramico (0603)



to ma, come evidenziato prima, la massa del circuito viene connessa al polo negativo del plug di alimentazione (pin 2) passando per i diodi D2 e D4. La tensione di alimentazione +5 Vcc viene generata dall'integrato U4, che è un regolatore di tensione switching di tipo step-down, la cui efficienza può arrivare fino al 96 %; il componente è in grado di erogare

una corrente massima di 500 mA, più che sufficiente per la nostra applicazione. Come per tutti i regolatori di tensione step-down, bisogna prestare attenzione al dimensionamento dei componenti ed alla loro posizione sul PCB. I condensatori elettrolitici devono avere una bassa ESR (*Equivalent Serie Resistance*) e vanno posizionati il più vicino possibile al pin

7 (condensatore C6), al pin 4 e all'induttanza da 100 μ H (condensatore C8). Molto importante è anche il diodo di ricircolo D5, il quale deve essere di tipo Schottky e va dimensionato affinché riesca a sopportare fino al doppio della corrente assorbita dal carico. L'alimentazione stabilizzata +5 Vcc viene portata a tutta l'elettronica presente sulla scheda ad eccezio-

C12, C13: 220 pF ceramico (0603)
 C14, C15, C16, C17:
 100 nF ceramico (0603)
 C18: 10 μ F 25 VL elettrolitico
 C19, C20: 100 nF ceramico (0603)
 C21, C22: 22 pF ceramico (0603)
 C23, C24, C25: 100 nF ceramico (0603)
 C26: 220 pF ceramico (0603)
 C27, C28: 100 nF ceramico (0603)
 C29, C30: 330 nF ceramico (0603)
 LD1: LED 3 mm rosso
 LD2: LED 3 mm giallo
 LD3: LED 3 mm rosso
 D1, D2: S1M
 D3: 1.5KE33CA
 D4: S1M
 D5: SS34
 D6, D7: MMSZ5229BT1G
 D8: S1M
 T1: BC846
 F1: Termistore 100 m Ω
 L1: Induttanza 100 μ H
 U1: LM224D
 U2: 7812
 U3: INA170EA
 U4: LM2674M_5.0
 U5: 24FC1025
 U6: PIC18F4523-I/PT (MF904)
 U7: VDIP1
 U8: 74HC573
 U9: DS1307Z
 U10: OPA340
 Q1: Quarzo 10 MHz
 Q2: Quarzo 32,768 kHz

Varie:

- Plug alimentazione
- Morsettiera componibile 2 poli (2 pz.)
- Strip maschio 5 poli
- Strip femmina 3 poli
- Strip femmina 6 poli
- Strip femmina 17 poli
- Circuito stampato

ne dell'integrato U1.

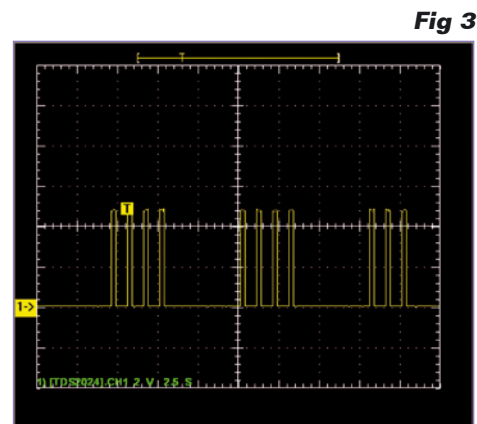
Prima di iniziare la descrizione dei due circuiti per la misura in corrente e tensione, si osservi che il carico in esame (R_L) deve essere collegato ai pin 2 e 3 della morsettiera CN3, mentre la tensione da applicare al carico deve essere sempre collegata ai pin 1 e 3 della medesima morsettiera. Il contatto 3 è sempre il polo negativo, cioè la

massa, mentre l'1 è sempre il polo positivo e quindi quello a potenziale maggiore. Collegando il polo positivo della tensione al polo 1 si assicura che la corrente circolante nel carico passa anche per la resistenza di shunt R_{11} da 5 W.

Passiamo quindi alla descrizione del circuito per la misura della corrente. Il circuito in esame è composto dall'integrato U3, dall'integrato U10 e dalla resistenza di shunt R_{11} . L'integrato U3 (INA170EA) si occupa di convertire la tensione differenziale presente sui pin di ingresso 2 e 1 in una corrente di uscita la quale, fluendo in una resistenza appositamente dimensionata, determinerà una tensione proporzionale alla corrente misurata ai capi della resistenza di shunt. La struttura interna dell'integrato INA170EA è illustrata nella Fig. 2; come si può osservare, la corrente I_s fluisce dalla V_{SUPPLY} nella resistenza di shunt R_s verso il carico (Load) da monitorare. La tensione V_s sulla resistenza di shunt viene forzata sulla resistenza R_{G1} causando il passaggio di corrente nel collettore del transistor T1. La resistenza di uscita R_L convertirà la corrente in uscita in una tensione V_{OUT} . Senza offset, la funzione di trasferimento dell'integrato INA170 è:

$$I_O = g_m \cdot (V_a - V_b)$$

Dove V_a corrisponde alla tensione V_{IN+} e V_b alla tensione V_{IN-} mentre g_m è uguale a 1.000 μ A/V. Osservando la Fig. 2 si nota che la differenza di potenziale $V_a - V_b$ si ricava applicando la legge di ohm tra la resistenza R_s e la corrente I_s . Un discorso analogo vale per la tensione di uscita V_{OUT} , la quale è data dall'applicazione della legge



di ohm considerando la resistenza R_L e la corrente I_O . Quindi la funzione di trasferimento può essere riscritta come segue:

$$V_{OUT} = I_s \cdot R_s \cdot 1000 / V \cdot R_L$$

Considerando anche l'offset, la formula diventa:

$$V_{OUT} = (V_{REF} \cdot R_L) / R_{OS} \pm (I_s \cdot R_s \cdot R_L) / 1$$

La tensione di offset viene determinata dalla V_{REF} applicata al pin 3 ed è necessaria nel caso in cui si debbano eseguire delle misure di correnti sia positive che negative. In altre parole l'offset impedisce che la tensione in uscita sia negativa e va dimensionato a seconda dell'applicazione. Nel caso in esame, le correnti misurate sono solo positive e quindi non è necessario applicare una tensione di offset. Guardando lo schema elettrico, si può notare che il pin 3 dell'integrato U3 è collegato a massa tramite la resistenza R_9 (da 0 ohm) quindi direttamente. La massima differenza di potenziale applicabile ai pin V_{IN+} e V_{IN-} senza danneggiare l'integrato

00H	SECONDS
	MINUTES
	HOURS
	DAY
	DATE
	MONTH
	YEAR
07H	CONTROL
08H	RAM
3FH	56 x 8

Fig 4

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
OUT	0	0	SQWE	0	0	RS1	RS0

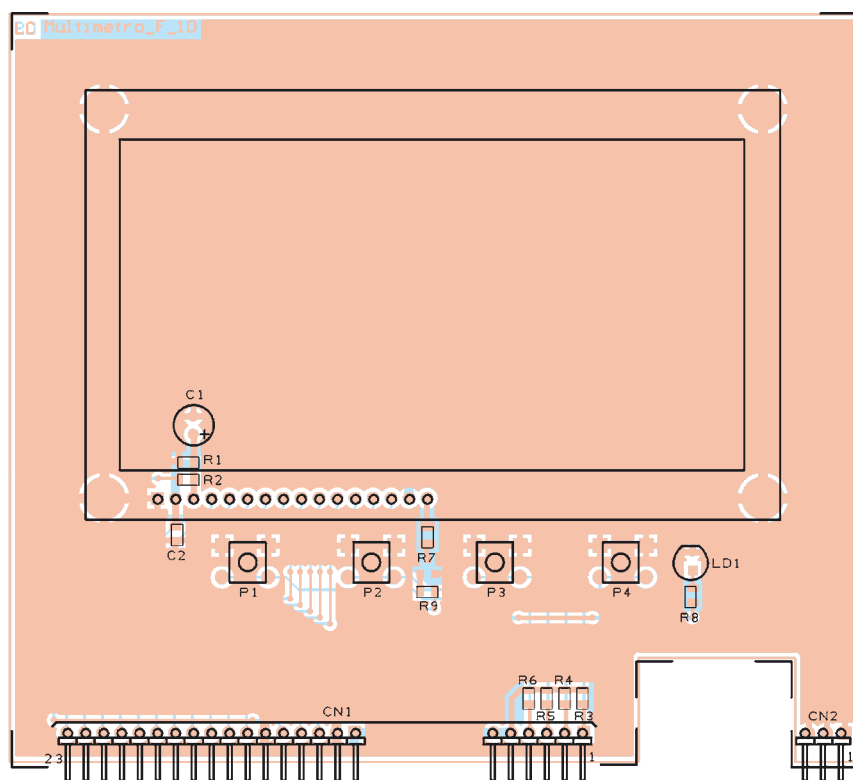
Fig 5

Elenco Componenti:

R1: 220 ohm (0805)
 R2: 680 ohm (0805)
 R3, R4, R5, R6:
 4,7 kohm (0805)
 R7: 10 ohm (1206)
 R8: 680 ohm (0805)
 R9: 1,2 kohm (0805)
 C1: 10 μ F 25 VL elettrolitico
 C2: 100 nF ceramico (0805)
 LD1: Led 3 mm verde
 P1, P2, P3, P4: Microswitch
 T1: BC846
 LCD: Display LCD
 20x4 (CM200400)

Varie:

- Strip maschio 3 poli 90°
- Strip maschio 6 poli 90°
- Strip maschio 17 poli 90°
- Circuito stampato



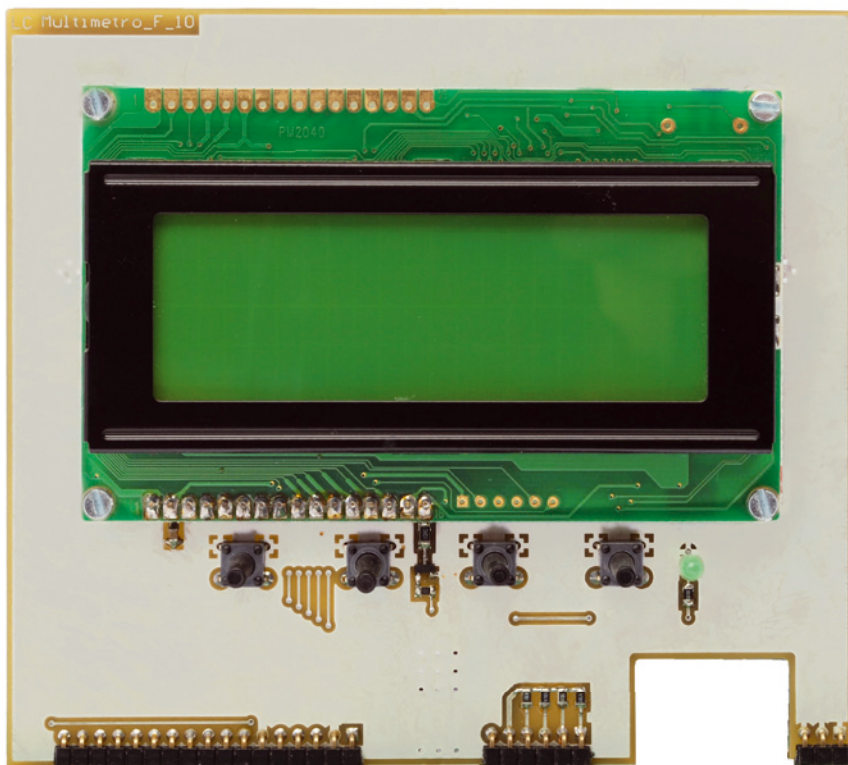
è di 2 Vcc, però per avere misure affidabili e precise all'ingresso non si devono superare i 500 mV. Con questa tensione viene generata una corrente di uscita di 500 μ A. Nell'applicazione in esame è stata selezionata una resistenza R_s da 100 m Ω - 5W (R11) che abbiamo dimensionato in base alla massima corrente misurabile dallo strumento (5 A), quindi la massima differenza di potenziale sulla resistenza sarà di 500 mV, un valore più che accettabile. La resistenza di uscita R_L è stata imposta da 8 k Ω (R14 + R15) assicurando così, in uscita, una tensione massima di +4 Vcc quando la corrente assorbita dal carico è uguale a 5 A. Questo valore è compatibile con il convertitore ADC presente sul PIC18F4523, in quanto la sua tensione di riferimento è impostata esattamente a +4 Vcc. Esplichiamo il concetto con alcuni esempi:

- se I_s è 5A la tensione di uscita

- V_{OUT} sarà +4 Vcc;
- se I_s è 1A la tensione di uscita V_{OUT} sarà di +800 mV;
- se I_s è 100 mA la tensione di uscita V_{OUT} sarà di +80 mV.

L'ultima considerazione sull'integrato U3 riguarda la sua alimentazione; essa è indipendente dalla tensione applicata al carico e quindi ci permette di alimentare lo stesso a +5 Vcc come il resto dell'elettronica, mentre la tensione del carico può arrivare fino a +60 Vcc senza creare alcun problema di sorta. La tensione in uscita sulla resistenza R_L (R14 + R15) non viene collegata direttamente all'ingresso del convertitore analogico digitale del PIC18F4523, ma è prima bufferizzata utilizzando l'operazionale OPA340 (U10) in configurazione inseguitore di tensione (Buffer). Questa scelta viene dettata dal fatto che l'impedenza di uscita dell'integrato INA170 è molto alta,

il che permette di utilizzare delle resistenze di uscita (R_L) dell'ordine dei 100 k Ω senza degradare l'accuratezza delle misure (nel nostro caso la resistenza R_L è di 8 k Ω). Quindi è necessario assicurare che qualsiasi circuito collegato in cascata all'uscita dell'integrato INA170 abbia un'impedenza di ingresso molto maggiore di quella di uscita di quest'ultimo. Così facendo, non si degradano le misure compiute dall'integrato. Per soddisfare in qualsiasi situazione questa condizione, si preferisce inserire un buffer, che nel nostro caso è un OPA340. La tensione in uscita dall'operazionale viene riportata direttamente all'ingresso del convertitore ADC, attraverso la resistenza R10, la quale ha un valore ohmico pari a 10 Ω . Il condensatore C7 serve per filtrare eventuali disturbi mentre il diodo Zener D6 protegge l'ingresso del convertitore nel caso in cui la tensione in



ingresso superiori a +4,3 V. Passiamo alla descrizione della sezione che si occupa della misura in tensione, la quale si avvale di un amplificatore per strumentazione realizzato con tre operazionali opportunamente collegati tra di loro. Il circuito può essere suddiviso in due stadi distinti, il primo formato dagli operazionali U1A e U1B e relative resistenze e il secondo dall'operazionale U1C. Il secondo stadio altro non è che un amplificatore differenziale. Le tensioni in ingresso agli operazionali U1A e U1B sono state siglate rispettivamente $e1$ ed $e2$; $e1$ è portata direttamente a massa. La tensione di uscita è stata, invece, chiamata $e0$. La funzione di trasferimento generalizzata per l'amplificatore per strumentazione è la seguente:

$$e0 = -(R4/R3) * (1 + (2 * R1)/R2) * (e1 - e2)$$

Dato che la tensione $e1$ è portata

direttamente a massa, la funzione può essere riscritta come:

$$e0 = (R4/R3) * (1 + (2 * R1)/R2) * e2$$

Le resistenze R1, R3 e R4 sono tutte da 100 kΩ, mentre la resistenza R2 è da 200 kΩ. In questo modo il guadagno G dell'amplificatore per strumentazione è pari a 2. Si osservi che la tensione in ingresso all'operazionale U1B ($e2$) viene ricavata tramite partitore di tensione dalla tensione collegata direttamente al carico sotto osservazione. Quindi, dato che la massima tensione applicabile al carico è pari a +36 V, ne consegue che la tensione $e2$ al massimo ammonterà a +2 Vcc. Quindi la massima tensione in uscita dall'amplificatore per strumentazione sarà di +4 V, valore totalmente compatibile con il convertitore ADC e la sua tensione di riferimento. L'operazionale U1D non è utilizzato e quindi è buona norma non

lasciare i suoi pin flottanti; perciò il suo pin 12 (ingresso positivo) è collegato a massa e il pin 12 con il pin 14 (configurazione ad inseguitore). La tensione $e0$ viene portata direttamente all'ingresso del convertitore ADC tramite la resistenza R17 da 10 Ω seguita dal condensatore C12 e il diodo D7. Il condensatore C12 serve per filtrare eventuali disturbi, mentre il diodo Zener D7 fa da protezione alle sovratensioni come descritto per il diodo D6.

Le due tensioni ricavate dalla misura in corrente e da quella in tensione vengono rispettivamente collegate ai pin RA1 ed RA0. La tensione positiva di riferimento per il convertitore ADC viene ricavata dal partitore di tensione formato dalle resistenze R24, R28 e R29, il quale assicura una tensione pari a +4 Vcc (pin RA3). La tensione di riferimento negativa è bloccata a massa (pin RA2).

Il convertitore ADC interno al PIC18F4523 ha una risoluzione di 12 bit, la quale permette una quantizzazione del segnale analogico in ingresso di 4096 punti ($2^{12} = 4096$). Dato che la tensione di riferimento positiva è stata impostata a +4 Vcc, ne consegue che la risoluzione del convertitore è di circa 976 μV. Quindi lo strumento è in grado di apprezzare le variazioni minori di 1 mV sui segnali in ingresso. Al microcontrollore, tramite bus PC, viene collegata una memoria EEPROM da 1 Mbit (U5) necessaria per la memorizzazione degli allarmi dovuti al superamento delle soglie impostate; tramite lo stesso bus viene collegato il clock calendar della Dallas Semiconductor (U9) per avere il riferimento di data e ora da abbinare agli allarmi registrati.

Come detto prima, la EEPROM utilizzata è da 1 Mbit; ciò significa che si hanno a disposizione 128



Patate elettriche

Quasi tutti conosciamo le patate per averle gustate fritte, al forno o in insalata, ma probabilmente nessuno ancora sa che presto saranno utilizzate anche per produrre energia elettrica. A sostenerlo sono

stati alcuni ricercatori della Yissum Research Development Company dell'Università

di Gerusalemme, che sono riusciti a costruire una batteria utilizzando delle patate, sostituendo così le classiche pile da 1,5 Volt con un costo 50 volte inferiore. Gli studiosi durante la realizzazione della batteria hanno anche notato che bollendo la patata prima di usarla in elettrolisi, l'energia elettrica prodotta è fino a 10 volte superiore rispetto a quella derivante da una patata cruda; cresce anche la durata, dato che minore è la resistenza del ponte salino in essa e più la batteria risulta longeva ed efficiente. Un'ottima notizia che può aprire nuove strade nel settore elettrico soprattutto nei Paesi in via di sviluppo.

kbyte per la memorizzazione degli allarmi. Per la memorizzazione di un singolo allarme sono necessari 16 byte e questo significa che nella EEPROM si possono memorizzare fino a un massimo di 8.192 allarmi. I 16 byte di ogni allarme sono suddivisi come mostra il **Riquadro 1**. Parleremo nella prossima puntata di come gestire gli allarmi memorizzati nella EEPROM con l'apposita pagina del menu. Passiamo all'integrato DS1307 (U9), al quale abbiamo accennato nelle pagine precedenti, necessario per gestire l'ora e la data per la registrazione degli allarmi rilevati durante il funzionamento dello strumento. Perché funzioni correttamente, servono pochi componenti esterni, tra cui una batteria tampone ed un quarzo da 32.768 kHz. I circuiti interni al DS1307 monitorano sia la tensione di alimentazione "standard" +5 Vcc, sia la tensione della batteria tampone. Quando

l'alimentazione principale si mantiene stabile a +5 V, tutte le operazioni di lettura e scrittura verso i registri interni saranno accessibili, mentre quando la tensione di alimentazione scende sotto la soglia di 1,25 volte la tensione di batteria, le funzioni di lettura e scrittura verranno disabilitate. In queste condizioni il circuito interno di gestione della data e dell'ora (cronometro) continuerà a funzionare correttamente senza perdita di precisione. Se la tensione di alimentazione scende ad un valore minore di quella della batteria, sia il cronometro che la RAM interna vengono commutate sull'alimentazione fornita dalla batteria tampone. La memoria interna e i vari registri sono mappati come evidenziato dalla **Fig. 4**, dove si nota che a partire dall'indirizzo 0x00 fino all'indirizzo 0x06 sono memorizzati i valori inerenti ai secondi, ai minuti, alle ore, al giorno della settimana, oltre che a giorno,

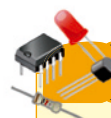
mese ed anno. Il bit 7 (CH) del registro di indirizzo 0x00 abilita/disabilita l'oscillatore e quindi il cronometro interno all'integrato. Alla locazione 0x07 trova posto un byte di controllo. Infine, dall'indirizzo 0x08 a 0x3F c'è la memoria RAM interna all'integrato. I valori di data e ora sono salvati in formato BCD. Il byte di controllo (indirizzo 0x07) è visibile in **Fig. 5**; notate il bit SQWE, che serve per abilitare/disabilitare l'uscita SQW/OUT sulla quale viene riportata un'onda quadra che può avere una frequenza pari ad 1, 4.096, 8.192 e 32.768 Hz. La frequenza di uscita viene selezionata dai bit RS1 e RS0. Il bit 7 (OUT) serve per impostare il valore logico di uscita sul pin SQW/OUT nel caso in cui il bit SQWE sia impostato a 0 logico. Tutti i rimanenti bit del registro sono forzati a livello basso.

Per il corretto funzionamento del circuito, la batteria tampone deve avere un valore di tensione compreso tra 2 e 3,5 V. La batteria utilizzata nel progetto è una pila a bottone CR2032, la quale ha una tensione nominale a piena carica di +3 V ed una capacità minima di 240 mAh. Con questo tipo di batteria viene assicurato un funzionamento dell'integrato per oltre 10 anni in mancanza di alimentazione primaria. Ovviamente, come da circuito, la batteria tampone deve alimentare solo l'integrato DS1307. L'integrato U9 è collegato al PIC18F4523 tramite bus I²C, assieme alla EEPROM. Le due linee SDA e SCL prevedono entrambe una resistenza di pull-up da 2,2 kohm; con questo valore resistivo è possibile impostare il clock della linea SCL fino a 400 kHz. Tuttavia l'integrato DS1307 non accetta, sulla linea SCL, una frequenza di clock maggiore di 100 kHz, di conseguenza anche la memoria EEPROM lavorerà ad

un massimo di 100 kHz. La porta D del PIC18F4523 viene utilizzata per il bus dati (DB0÷DB7) al quale vengono collegati il display alfanumerico da 4 righe per 20 caratteri ed i quattro pulsanti di gestione del menu. Sia il display che i pulsanti si trovano sulla scheda frontale (*pannello*). Per gestire sia il display che i pulsanti con un unico bus dati, è necessario introdurre l'integrato 74HC573 (U8). Questo è un latch di tipo D che mantiene lo stato degli ingressi, ad esso collegati, e riporta tale stato sul BUS solo quando il livello logico sul pin OE diventa basso. Lo stesso discorso si può fare per il display, il quale ha un pin di Enable (E) che, coadiuvato dai segnali RW e RS, permette la lettura o scrittura della memoria del display e relativi registri. Il segnale RS serve per decidere se si vuole leggere o scrivere la memoria RAM (RS=1) oppure se si vogliono leggere o scrivere i registri interni di configurazione (RS=0). Il segnale RW determina se si vuole eseguire una operazione di lettura o scrittura. Per leggere si imposta la linea ad 1 logico; per scrivere, la si porta a 0. Il segnale BCK serve per accendere e spegnere la retroilluminazione del display LCD, che normalmente teniamo spenta; premendo uno dei pulsanti del circuito, si accende consentendo una visualizzazione migliore del menu di gestione dello strumento. Il segnale CT_DISP serve per regolare il contrasto del display; è di tipo PWM e non è collegato direttamente al pin Vo del display, ma alla rete RC formata dalle resistenze R1, R2 e dal condensatore C1. Il segnale PWM generato ha una frequenza pari a 5 kHz. A seconda del duty-cycle impostato, il contrasto del display diminuirà o aumenterà di conseguenza. Tutti gli otto bit del BUS dati sono

collegati a massa tramite delle resistenze di pull-down, le quali assicurano un valore stabile sul BUS nel caso in cui la porta D del microcontrollore vada in alta impedenza. Tutti i segnali di gestione del display, le alimentazioni necessarie ed i segnali per la gestione dei pulsanti, fanno capo al connettore CN2, al quale è connessa la scheda frontale. Sulla scheda frontale trova posto anche il LED di diagnostica LD1. Quando lo strumento rileva un'anomalia dovuta al superamento di una soglia preimpostata, oltre a registrare l'evento sulla EEPROM, se impostato, può dare anche un allarme di tipo acustico mediante il cicalino buzzer BZ1. Per pilotare quest'ultimo si utilizza un semplice transistor BC846B (T2). La forma d'onda che pilota il buzzer è visibile nella **Fig. 3**; come si può osservare, vengono generate quattro onde quadre consecutive con periodo di 1 secondo ciascuna e duty-cycle pari al 30 % ON e 70 % OFF, più una pausa di quattro secondi dove il segnale è sempre OFF. Il periodo totale del treno di impulsi è quindi 8 secondi. Per tacitare il buzzer è sufficiente premere contemporaneamente i pulsanti P1 e P2. Infine, è stata prevista la possibilità di gestire la memorizzazione degli allarmi su chiavetta USB. Per fare ciò si utilizza un prodotto della FTDI: il VDIP1, che fa parte della famiglia Vinculum, il quale altro non è che una MCU con un controller USB Host integrato. Il collegamento tra il modulo VDIP1 e il PIC18F4523 viene realizzato tramite interfaccia UART (RX e TX) più quattro pin di controllo RTS (Request To Send Control Output), CTS (Clear to Send Control Input), RST (Reset) e PRG (Program). Il tipo di interfaccia

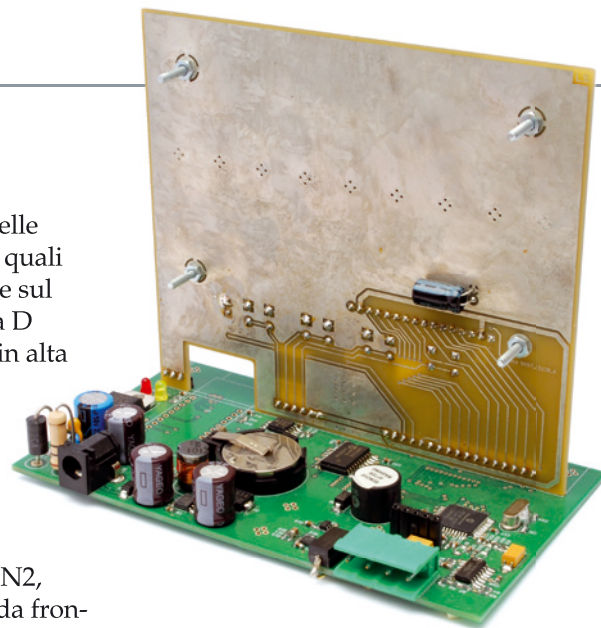
da utilizzare è selezionabile tramite i jumper presenti sul modulo VDIP1; in particolare si può scegliere l'interfacciamento UART, SPI o FIFO parallelo. Il pin RST serve per resettare il modulo VDIP1 e viene usato unitamente a PRG e all'interfaccia UART. Il pin PRG viene usato per programmare il modulo VDIP1. Tutti e quattro i pin di controllo menzionati vengono collegati al PIC18F4523 e sono dotati della loro resistenza di pull-up. Per completare la configurazione base del modulo VDIP1 vengono collegati due diodi luminosi (LD2 e LD3) necessari a dare un'indicazione visiva sullo stato della scrittura/lettura della periferica USB collegata. ■



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 firmware per programmare il microcontrollore. Quest'ultimo, un PIC18F4523-I/PT, costa 8,00 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>





dell'ing.
MIRCO
SEGATELLO

Conoscere e usare Arduino

Analizziamo la neonata Arduino UNO, la scheda di sviluppo e prototipazione creata per sostituire la popolare Duemilanove.

Alla fine di settembre dell'anno scorso il team di Arduino ha annunciato alla stampa l'imminente disponibilità di due nuove schede, una delle quali è Arduino UNO; in questa puntata del corso riguardante il mondo Arduino vogliamo introdurvi ed insegnarvi ad utilizzare proprio questa novità. L'origine Italiana del prodotto è (come non mai) esaltata dal nome, il quale, non a caso, è UNO e non ONE (come si direbbe in inglese) e dalla presenza di un ben evidente marchio "Made in Italy" sulla confezione (e sul prodotto), con tanto di simbolo geografico dell'Italia.

Arduino UNO è la prima su cui appare il nuovo logo scelto dal progetto Arduino, che è il simbolo matematico di infinito, scelto probabilmente per simboleggiare le infinite possibilità d'uso della scheda o la sconfinata espandibilità del progetto open-source da cui Arduino trae origine. Acquistando Arduino UNO, all'interno della confezione trovate alcuni adesivi ed un piccolo manuale -riportante le indicazioni principali sulla licenza di utilizzo- in lingua inglese, come è logico aspettarsi da un prodotto internazionale open-source. L'impressione che si ha, prendendo in



Fig 1
La confezione del nuovo Arduino.

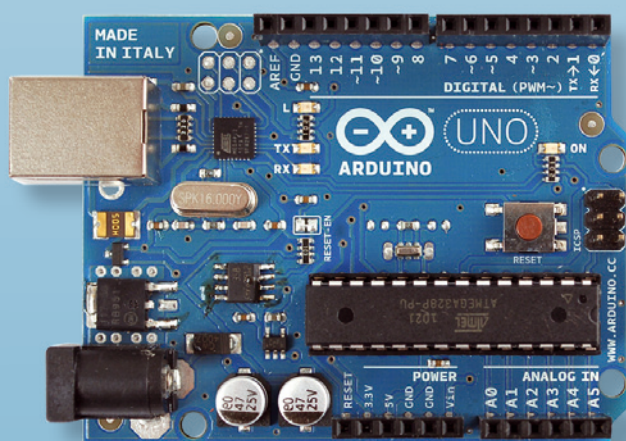


Fig 2
Aspetto dell'Arduino UNO vista dall'alto.

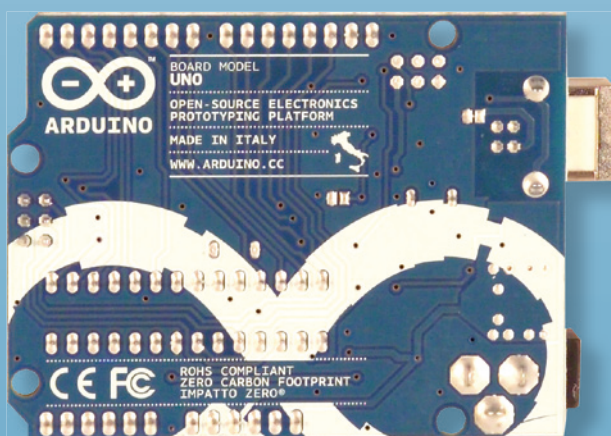


Fig 3
La scheda Arduino UNO vista da sotto.

mano questo nuovo hardware, è eccellente: la scheda è di buona qualità e lo stampato è di ottima fattura, con una serigrafia molto precisa e completa.

SCHEMA ELETTRICO

La particolarità dell'Arduino UNO e quindi ciò che la distingue nettamente da tutte le altre versioni proposte in questi anni, è che in essa per la connessione USB non viene più utilizzato il convertitore USB-Seriale della FDTI, bensì un microcontrollore ATmega8U2 programmato per funzionare come convertitore USB-Seriale. Questo nuovo prodotto della ATMEL è infatti un microcontrollore con a bordo un modulo Transceiver USB liberamente programmabile, come lo hanno ad esempio alcuni PIC della serie 18F della Microchip. Il nuovo Arduino ha anche ricevuto la certificazione FCC sulle emissioni elettromagnetiche; le diciture "ROHS Compliant" e "Zero carbon footprint" fanno emergere l'interesse del team Arduino per la tutela dell'ambiente. Arduino UNO è ancora basato sul microcontrollore ATmega328 (in formato DIP) e dispone di 14 pin di I/O (di cui 6 utilizzabili come uscite PWM), 6 ingressi analogici, un oscillatore a 16 MHz, un connettore per la programmazione In-Circuit ed un Plug per l'alimentazione. Come nelle ultime versioni di Arduino, è presente un connettore USB che, semplicemente connesso ad un Personal Computer, permette sia di alimentare la scheda, sia di programmarla.

Questa versione di Arduino è da intendersi come la 1.0 basata sulla nuova tecnologia per la connessione alla USB; la pagina di riferimento per comparare tutte le versioni hardware è la seguente: <http://arduino.cc/en/Main/Boards>. Al solito, il sito Internet di riferimento dove vedere le caratteristiche e trovare tutto quello che riguarda Arduino UNO è www.arduino.cc (in lingua inglese).

La **Tabella 1** riepiloga le caratteristiche più importanti della scheda Arduino UNO. L'alimentazione della scheda può avvenire tramite la porta USB, ma è disponibile il solito connettore plug che accetta, in ingresso, una tensione non regolata con valore compreso tra 7 e 12 volt; in questo caso un semplice alimentatore non stabilizzato universale impostato sul valore di 9 volt è l'ideale, ma nulla vieta di

Tabella 1 - Caratteristiche di Arduino UNO

Microcontrollore	ATmega328
Tensione di lavoro	5 V
Alimentazione esterna (raccomandata)	7÷12 V
Alimentazione esterna (limiti)	6÷20 V
I/O digitali	14 (di cui 6 usabili come PWM output)
Ingressi analogici	6
Corrente per ogni I/O Pin	40 mA
Corrente prelevabile dal pin 3,3 V	50 mA
Flash memory	32 kB (ATmega328) di cui 0,5 kB usati per il bootloader
SRAM	2 kB (ATmega328)
EEPROM	1 kB (ATmega328)
Frequenza di clock	16 MHz

alimentare la scheda tramite una batteria a 9 o 12 volt. La sorgente di alimentazione viene riconosciuta in automatico e non serve impostare alcun deviatore. La porta USB è comunque protetta da accidentali cortocircuiti nella scheda e comunque da essa non vengono prelevati più dei 500 mA massimi ammessi.

I pin di alimentazione nella scheda Arduino sono i seguenti:

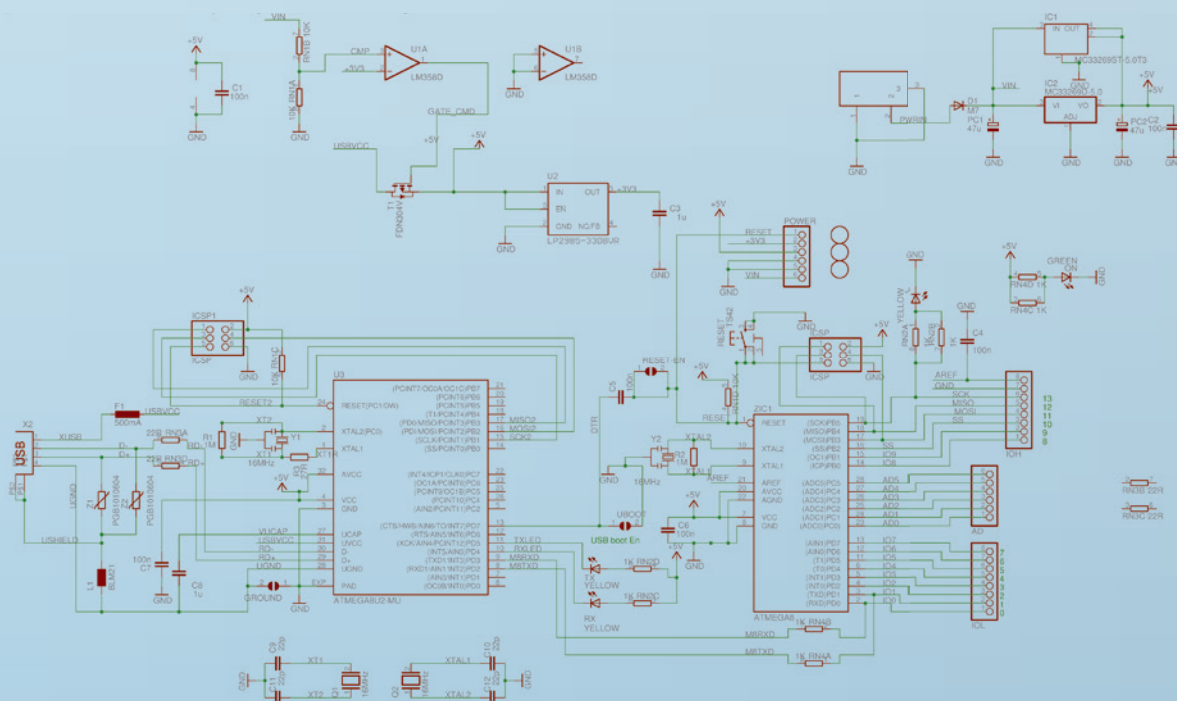
- VIN; questo, semplicemente replica la tensione fornita in ingresso sul connettore plug e può essere usato per alimentare altri circuiti che dispongano già di un regolatore di tensione (ad esempio gli shield applicati al modulo);
- 5 V; fornisce i 5 volt dello stabilizzatore di tensione interno alla scheda ed è utile per

alimentare altri circuiti compatibili con i 5 volt;

- 3V3; questo pin fornisce i 3,3 volt ricavati dallo stabilizzatore interno alla scheda e consente di alimentare circuiti compatibili con tensioni di 3,3 volt (la massima corrente prelevabile è di 50 mA);
- GND: è il contatto di massa (GND).

CARATTERISTICHE DI ARDUINO UNO

Vediamo adesso le prerogative e la dotazione della nuova scheda Arduino, a partire dalla memoria: il microcontrollore ATmega328 dispone di 32 kB di memoria di programma, della quale 0,5 kB sono usati per il bootloader. Dispone inoltre di 2 kB di SRAM ed 1 kB di EEPROM utilizzabile, quest'ultima, per il sal-



Schema elettrico di Arduino UNO

vataggio di dati permanenti (mantiene i dati anche in assenza di alimentazione).

Quanto agli ingressi e alle uscite, ciascuno dei 14 pin di I/O può essere usato come pin di input o output e può erogare/assorbire una corrente massima di 40 mA; inoltre dispone di una resistenza di pull-UP del valore di $20 \div 50$ kohm (attivabile tramite programmazione). Arduino UNO dispone di sei ingressi analogici (A0,A1,A2,A3,A4 e A5) che per impostazione predefinita hanno risoluzione di 10 bit ed accettano una tensione compresa fra 0 e 5 volt; tuttavia è possibile usare l'ingresso Aref per modificare il range di misura. Arduino UNO prevede degli interrupt esterni localizzati ai pin 2 e 3; questi possono essere configurati come trigger per eventi esterni, come ad esempio il rilevamento di un fronte di salita o di discesa di un segnale in ingresso.

Il modulo PWM del microcontrollore ATmega si può assegnare ai pin 3, 5, 6, 9, 10, e 11. Questi ultimi possono essere configurati via software per generare segnali PWM con risoluzione di 8 bit. Tramite un semplice filtro RC è possibile ottenere tensioni continue di valore variabile.

E veniamo alle porte di comunicazione: la seriale fa capo ai pin TX(1) e RX(0) che sono i corrispondenti dell'USART interno al microcontrollore e sono connessi al convertitore USB-Seriale della scheda.

Sempre in tema di comunicazione, va detto che il microcontrollore ATmega328 utilizza il modulo UART interno per comunicare, con livelli logici 0/5 volt, via seriale con altri dispositivi o con un PC. I segnali corrispondenti sono disponibili sui pin esterni (TX e RX) e risultano connessi anche al convertitore

Installare Arduino UNO su Seven

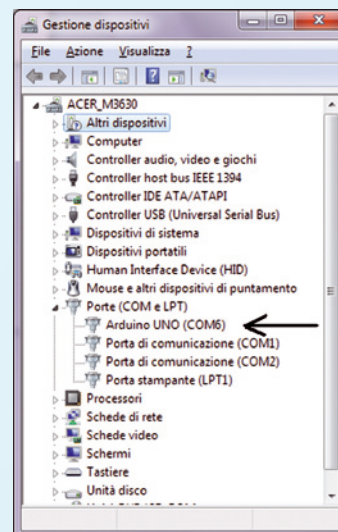
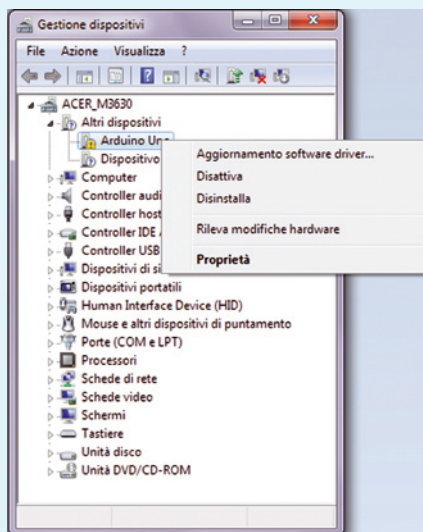
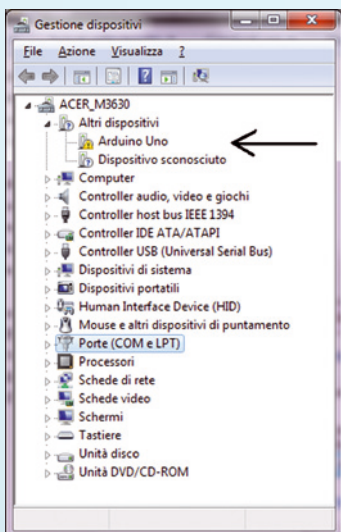
Seven è strutturato in modo da installare in automatico ogni periferica eventualmente cercando in rete i driver più opportuni. Arduino è però una periferica molto particolare, ragion per cui la procedura automatica potrebbe non andare a buon fine.

Quando inserite Arduino il sistema operativo identifica la presenza di una nuova periferica senza però riuscire ad installare i driver, e ovviamente non disponete del CD di installazione, in questo caso dovete procedere manualmente all'installazione dei driver. Dopo aver inserito la scheda Arduino Cliccate su Avvio-Pannello di controllo-

Gestione dispositivi, troverete evidenziata la periferica Arduino con un punto esclamativo in quanto presente ma non correttamente installata.

Cliccate con il pulsante destro del mouse sopra la periferica e selezionate Aggiornamento software Driver... Quindi selezionate a mano la cartella "driver" del software Arduino, la procedura proseguirà adesso in automatico sino alla completa installazione.

A questo punto sarà presente la periferica Arduino correttamente installata e verrà indicato quale COM le è stata assegnata.



USB-Seriale della scheda, il che permette la comunicazione tramite la porta USB del computer. A differenza del chip della FDTI, per il quale era necessario installare appositi driver, con l'utilizzo dell'integrato ATmega8U2 ciò non è più necessario, in quanto vengono usati i driver comuni della periferica USB già disponibili con il sistema operativo. Tuttavia, con sistemi operativi Windows, per la corretta creazione di una porta COM virtuale è necessario installare un driver aggiuntivo. Allo scopo ricordiamo che Arduino UNO è compatibile con i sistemi operativi Windows, Mac OS X e Linux, per i quali sono previsti i driver. Il bus SPI fa capo ai pin 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK) i quali possono essere programmati per realizzare una comunicazione SPI, appunto.

Arduino UNO prevede anche un bus I²C, localizzato ai piedini 4 (SDA) e 5 (SCL) che permettono di realizzare una comunicazione nello standard I²C a due fili, in abbinamento alla libreria Wire.

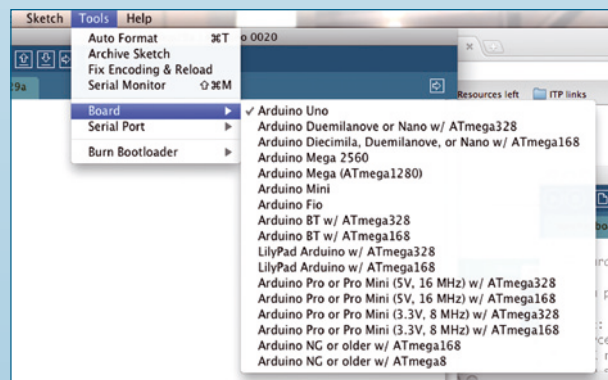
Concludiamo la carrellata sulle funzionalità di Arduino UNO con il piedino LED (pin 13) che è connesso a un LED interno alla scheda, utile per segnalazioni di diagnostica.

Infine abbiamo il Reset: questo contatto, portato a livello logico basso resetta il microcontrollore. La funzione corrispondente può essere attivata anche tramite il pulsante presente nella scheda Arduino.

INSTALLAZIONE ED UTILIZZO

Per il corretto utilizzo di questa nuova versione di Arduino è opportuno disporre dell'ultima versione del software -la 21- scaricabile gratuitamente all'indirizzo <http://arduino.cc/en/Main/Software>, la quale contiene sia i driver per Arduino UNO, sia i Driver per il chip FTDI di Arduino 2009. Con sistema operativo Windows, appena connettete Arduino al PC, esso verrà riconosciuto come nuovo hardware e vi verrà chiesta l'installazione del driver; allora non dovrete far altro che specificare come percorso la cartella "driver" del software Arduino (*Arduino UNO.inf*); il sistema operativo provvederà alla sua installazione ed alla creazione della porta COM virtuale. Fatto ciò avviate il software Arduino e specificate l'utilizzo di Arduino UNO, oltre alla COM alla quale è connesso. Il semplice esempio

Fig 4 - Impostazione del software per Arduino UNO.



applicativo chiamato *File-Esempi-1.Basics-Blink* vi permetterà di testare il funzionamento della scheda. Inoltre sono disponibili tutorial e documentazione agli indirizzi web <http://arduino.cc/en/Tutorial/HomePage> ed <http://arduino.cc/en/Reference/HomePage>.

Grazie al Bootloader preinstallato a bordo, non è necessario utilizzare alcun programmatore esterno né è necessario rimuovere il microcontrollore, la connessione USB tra PC e Arduino è sufficiente a permettere la programmazione e la gestione della comunicazione. La funzione di autoreset interna alla scheda permette la programmazione con un solo click del mouse.

Su questa nuova versione di Arduino è presente un nuovo Bootloader basato sul protocollo STK-500, che occupa un quarto di memoria rispetto alla versione precedente. Adesso sono sufficienti 512 byte di memoria al posto dei 2 kb della precedente versione; inoltre, il nuovo bootloader può gestire la velocità di comunicazione sino a 115 kbps, contro i 57,6 k della precedente versione.

Tutta la documentazione sia hardware che software, compresi i sorgenti, è disponibile per il download sul sito di Arduino.

Arduino UNO è predisposta per comunicare in seriale con il PC semplicemente avviando il tools "Serial monitor" sull'ambiente di sviluppo; in questo modo un dato acquisito dalla scheda può essere facilmente visualizzato a video.

Il microcontrollore ATmega8U2 usato come convertitore USB-Seriale può essere facilmente programmato in quanto, al suo interno, è già precaricato il bootloader. In questo caso è possibile utilizzare le apposite piazzole di programmazione dopo aver attivato la modalità di programmazione saldando il piccolo jumper disponibile sul retro della scheda. Il software necessario per lo sviluppo dei pro-

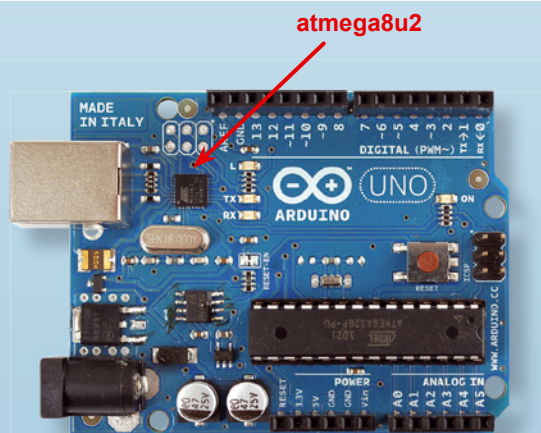


Fig 5 - Posizione del microcontrollore 8U2 sulla scheda Arduino UNO.

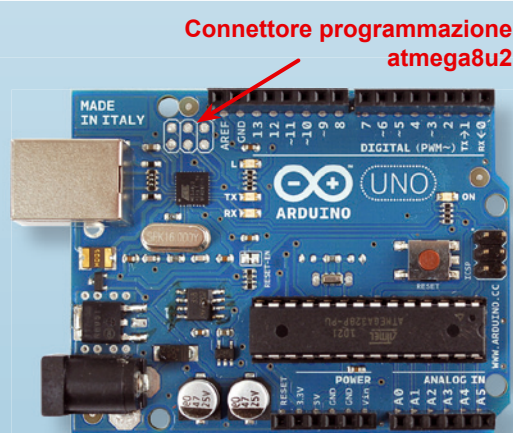


Fig 7 - Piazzole per la programmazione di 8U2.

Cortocircuitare questo ponticello per abilitare il bootloader atmega8u2

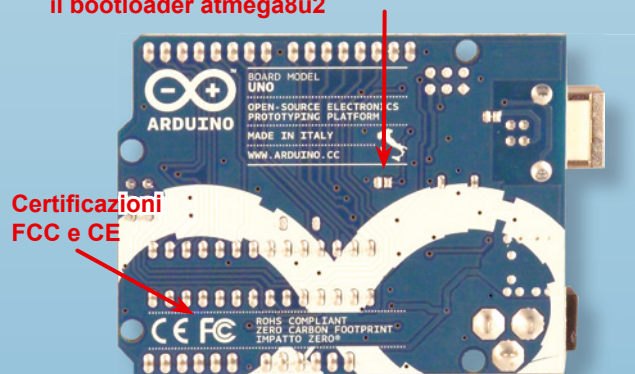


Fig 6 - Jumper per attivare modalità aggiornamento firmware 8U2.

grammi per l'ATmega8U2 si chiama Atmel's FLIP software per Windows (si scarica dalla pagina web http://www.atmel.com/dyn/products/tools_card.asp?tool_id=3886) e DFU programmer per Mac OS X e Linux (si scarica da <http://dfu-programmer.sourceforge.net/>). Ai più preparati non passerà inosservato il

grosso vantaggio di avere il convertitore USB-Seriale programmabile; infatti sino ad ora Arduino poteva essere visto dal PC solo come una periferica seriale e infatti i driver del convertitore della FDTI creavano una seriale virtuale.

Adesso, invece, potendo programmare il convertitore, Arduino può essere visto dal sistema operativo del PC come una periferica ad-hoc. Nulla vieterebbe di far rilevare Arduino come una stampante e quindi, qualora inviasse il comando di stampa da un qualsiasi software, i dati giungerebbero ad Arduino, il quale, magari, può così controllare una macchina CNC! Allo stesso modo si potrebbe inventare un nuovo sistema di puntamento che, una volta connesso al PC, verrebbe riconosciuto come periferica tipo mouse ed il cursore sullo schermo si muoverebbe guidato da questa nuova periferica. Vi lasciamo immaginare quale interessante scenario Arduino UNO apre su quanto riguarda lo sviluppo di nuove periferiche per PC. ■



cod:
ARDUINOUNOKIT

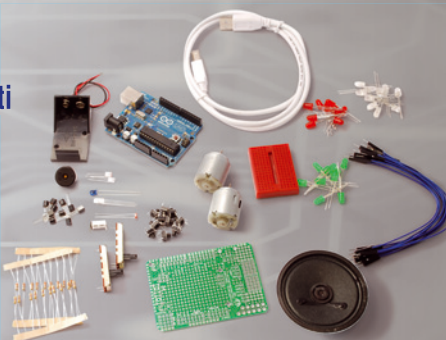
€ 55,00
IVA inclusa.

Arduino

la piattaforma open source alla portata di tutti

STARTER KIT CON ARDUINO UNO

kit composto da scheda Arduino UNO, cavo USB, mini Breadboard a 170 contatti con 10 cavetti da 15cm, piastra sperimentale (58,5 x 82,7mm), 2 motori elettrici, fotoresistenza, termistore, LED, micropulsanti, transistor e molti altri componenti necessari per cominciare ad utilizzare questa potente piattaforma hardware.



FUTURA ELETTRONICA

Via Adige, 11 • 21013 Gallarate (VA)
Tel. 0331/799775 • Fax. 0331/792287

Maggiori informazioni su questo prodotto e su tutte le altre apparecchiature sono disponibili sul sito www.futurashop.it tramite il quale è anche possibile effettuare acquisti on-line.