

Mensile di progettazione elettronica, attualità scientifica, novità tecnologiche.

€ 6,00

Elettronica In

www.elettronica.in.it

oltre l'elettronica

Anno XIX - n. 180
Ottobre 2013

■ Interfaccia Bluetooth
quattro canali I/O
Android based

■ Corso mbed

ANTEPRIMA

Arduino YÚN: la vita si fa più semplice

■ Telemetria real-time
per radiomodelli

■ GPS Logger
Shield

■ myKIT:
Sensore tilt

■ LDC: nasce la misura
con sensori induttivi

■ Riconoscimento
gestuale
Touch Free

■ Frequenzimetro
digitale a
microcontrollore

IN OMAGGIO

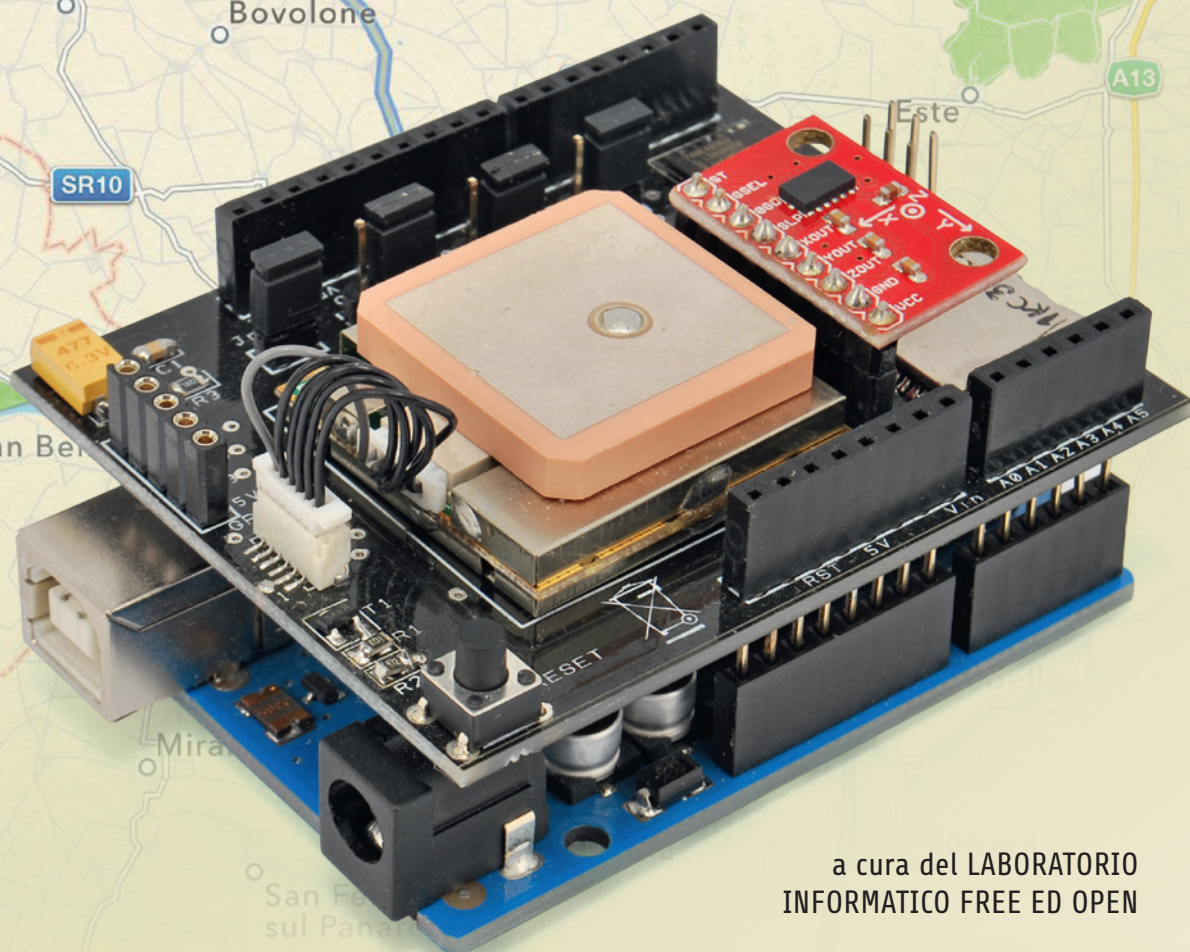
Catalogo MCU PIC® 8-bit

 **MICROCHIP**



GPS LOGGER SHIELD

Mediante un semplice sketch, registra periodicamente le posizioni desunte da un ricevitore GPS quando è in movimento e le rende disponibili su microSD formattata per essere letta in ambiente Windows.



a cura del LABORATORIO
INFORMATICO FREE ED OPEN

Il GPS (Global Positioning System) è il più noto e utilizzato tra i sistemi di navigazione satellitare e, anche grazie alle integrazioni con i sistemi di perfezionamento della precisione (WAAS/EGNOS, DGPS ecc.) che gli si sono affiancati negli anni, permette di conoscere in ogni momento posizione ed ora precisa (orario UTC) mediante l'uso di moduli ricevitori facilmente reperibili sul mercato, oramai a prezzi alla portata di tutti. Molti di questi ricevitori comunicano con l'esterno inviando i dati desunti dal sistema

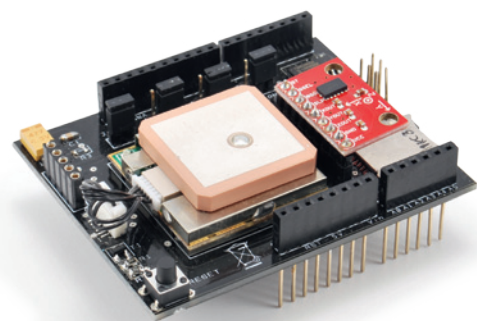
GPS tramite il protocollo NMEA0183, che prevede una comunicazione seriale e l'emissione di stringhe di dati standard facilmente gestibili da un microcontrollore, quindi anche dalle risorse computazionali tutto sommato limitate di Arduino. Non a caso, proprio abbinando un ricevitore GPS standard NMEA con Arduino abbiamo realizzato il sistema proposto in questo articolo: si tratta di un semplice logger GPS, vale a dire un dispositivo che permette di tracciare il percorso compiuto da una persona, un veicolo o una mer-

ce, semplicemente attraverso la memorizzazione periodica di punti di localizzazione desunti dal GPS stesso.

Per consentirci di vedere questo percorso, il logger salva il tracciato sotto forma di lista di record (contenenti i dati sulle posizioni registrate) su una scheda di memoria microSD, dalla quale si può passare su un Personal Computer per tenere traccia dei propri viaggi, ma

anche, magari, per contribuire a progetti open source di tracciamento di mappe alternativi a Google Maps, come ad esempio Openstreetmap (www.openstreetmap.org).

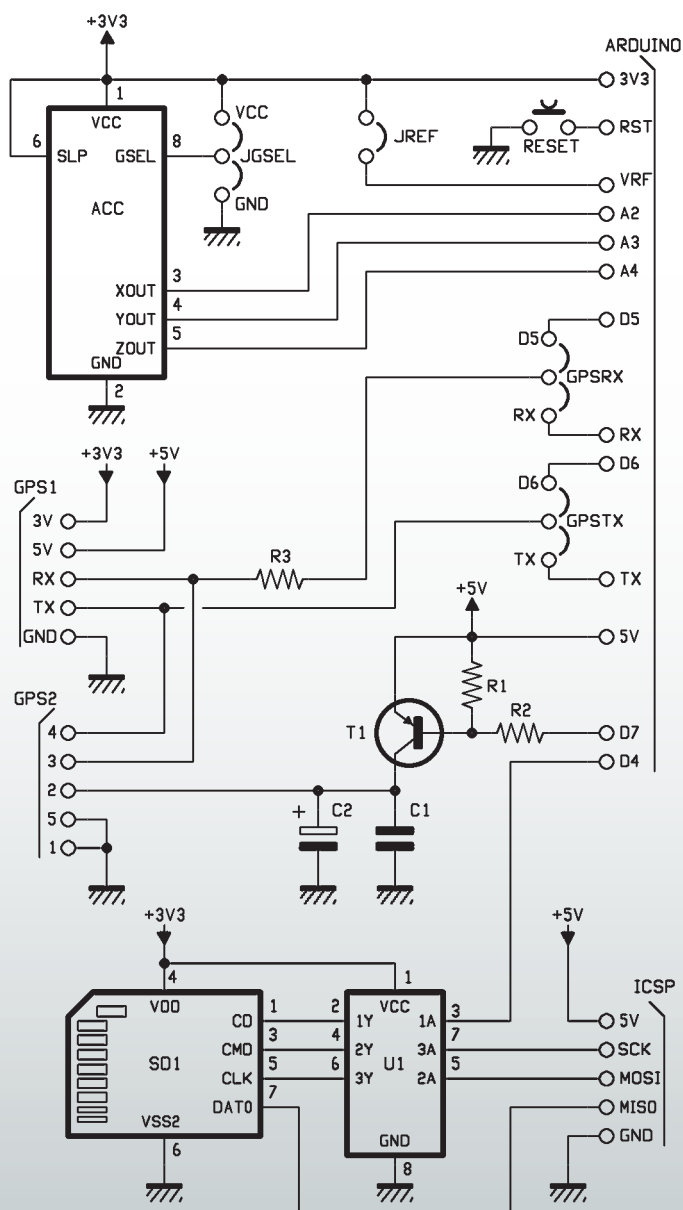
Avendo una discreta esperienza con i logger GPS disponibili sul mercato, ci siamo accorti che una delle cose più fastidiose è dimenticarsi di spegnerli una volta arrivati a destinazione; infatti ciò consuma memoria



perché il dispositivo salva continuamente la stessa posizione, o meglio, tanti punti vicinissimi tra loro la cui distanza non è dovuta al movimento del ricevitore, ma al diverso errore di localizzazione da cui il ricevitore è affetto e che non è ripetibile in più letture riguardanti le stesse coordinate. L'errore nel posizionamento dipende da fattori atmosferici e dai ritardi di propagazione tipici della logica, nonché dalla tolleranza dell'orologio locale (che non è certo preciso come quello atomico dei satelliti GPS); proprio la variabilità dei fattori in gioco fa sì che pur mantenendo fermo il ricevitore, esso fornisca nel tempo posizioni che si discostano di alcuni metri l'una dall'altra, dando l'indicazione che il ricevitore stesso si muova anche se invece è fermo.

Comunque, per evitare di inviare inutilmente dati sulla localizzazione alla microSD anche col ricevitore fermo, abbiamo deciso di implementare nel nostro logger un accorgimento firmware in grado di fermare la registrazione, ovvero riprenderla quando il sistema si rimette in movimento; ciò è stato ottenuto anche grazie ad un accelerometro, le cui informazioni elaborate da Arduino, oltre a permetterci di interrompere il salvataggio dei dati quando il circuito è fermo, ci consentono di spegnere il modulo ricevitore GPS per ridurre i consumi, cosa molto utile, visto che questo

[schema ELETTRICO]



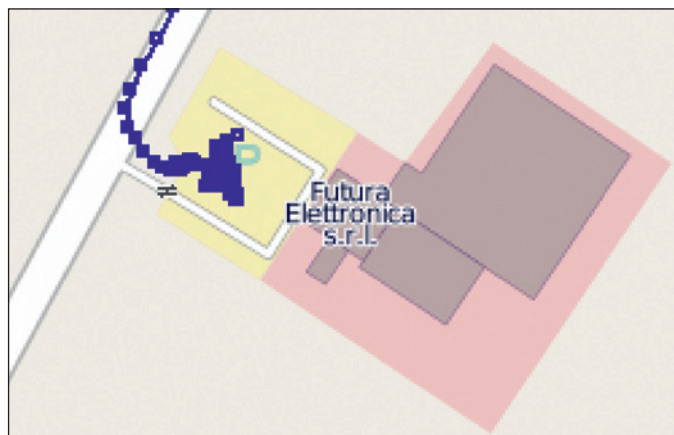
genere di sistemi di norma si alimenta a batterie.

L'HARDWARE

Il logger che vi proponiamo consta di uno shield, sul quale trovano posto principalmente il ricevitore GPS, il lettore di SD-Card e l'accelerometro, oltre che di una scheda Arduino UNO; notate che il ricevitore si collega mediante due connettori (alternativi l'uno all'altro), quindi nulla vieta di situarlo fuori dallo shield. Ciò può servire se sopra il circuito pensate di montare altri shield, i quali inevitabilmente disturberebbero la ricezione dei segnali provenienti dai satelliti.

Vediamo dunque in cosa consiste il circuito dello shield, che, in virtù del ridotto assorbimento di corrente, è alimentato direttamente da Arduino. Le nostre misurazioni, alimen-

La tipica "nuvola" di punti vicini che si ottiene con pochi minuti di sosta. Al chiuso, l'area coperta è ancora più ampia perché aumenta l'errore.

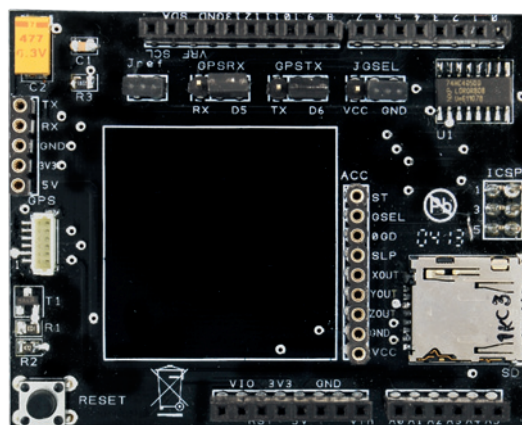
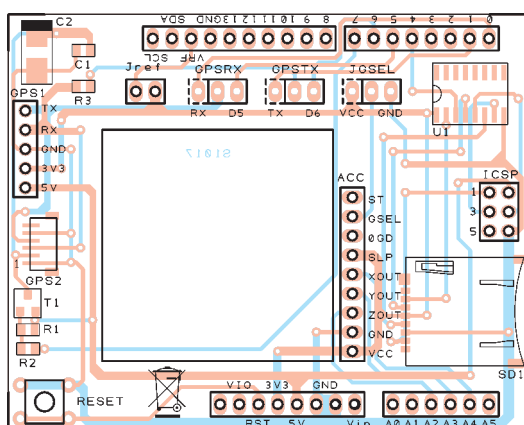


tando Arduino a 5 V, indicano consumi attorno ai 35 mA con il GPS attivo, ridotti a circa 5 mA disattivando il GPS tramite il pin 7. A questo si aggiungono i circa 40 mA dell'Arduino UNO, per un totale di 75 mA in modalità LOG (ossia durante l'acquisizione e registrazione delle coordinate) e 45 mA in modalità standby.

La sezione principale dello shield è ovviamente composta dal modulo GPS EM406 della

GlobalSat, dotato di antenna integrata (ce l'ha sopra...) il quale comunica con Arduino tramite il protocollo NMEA 0183: la comunicazione è seriale a 4.800 bps mediante linee TX ed RX a livello TTL. Questo segnale seriale è disponibile sul connettore GPS e tramite i jumper GPSRX e GPSTX può essere indirizzato ai pin dell'Arduino 0 ed 1 (RX e TX della porta seriale integrata) oppure 5 e 6; a tale riguardo va ricordato che le linee

[piano di MONTAGGIO]



Elenco Componenti:

R1: 10 kohm (0805)
R2: 4,7 kohm (0805)
R3: 2,2 kohm (0805)
C1: 100 nF multistrato (0805)
C2: 470 µF 6,3 VL tantalio (D)
T1: BC807
U1: 74HC4050D

GPS: Ricevitore EM406
SD1: Connettore micro SD-Card
RESET: Microswitch
ACC: MMA7361 accelerometro 3 assi

Varie:

- Connettore 6 poli passo 1mm
(cod. BM06B-SRSS-TB / CONNEM406A)
- Strip Maschio/Femmina 3 poli (2 pz.)

- Strip Maschio/Femmina 6 poli (1 pz.)
- Strip Maschio/Femmina 8 poli (2 pz.)
- Strip Maschio/Femmina 10 poli (1 pz.)
- Strip Maschio 2 poli (1 pz.)
- Strip Maschio 3 poli (3 pz.)
- Strip femmina 5 poli (1 pz.)
- Strip femmina 9 poli (1 pz.)
- Jumper (4 pz.)
- Circuito stampato

Lo standard NMEA0183

Il protocollo NMEA è stato sviluppato per le comunicazioni tra i dispositivi elettronici utilizzati in ambito navale; le sue specifiche sono disponibili a pagamento sul sito della National Marine Electronics Association, tuttavia il protocollo è stato ricostruito da vari appassionati usando fonti pubbliche, quindi si trova liberamente sul web, ad esempio agli indirizzi https://en.wikipedia.org/wiki/NMEA_0183 e www.gpsinformation.org/dale/nmea.htm. Il protocollo NMEA prevede una comunicazione seriale a 4.800 bps, 8 bit di dati senza bit di parità e con un bit di stop.

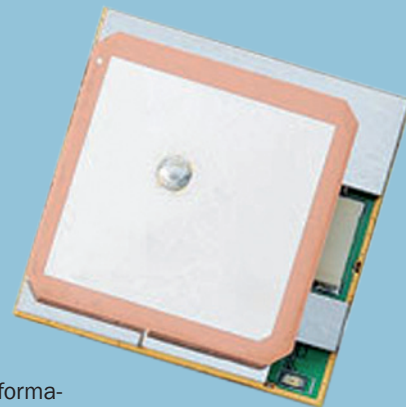
Ogni stringa trasmette delle "frasi" che iniziano con il carattere \$, terminano con la sequenza <CR><LF> (anche scritta \r\n) e contengono campi separati da virgole. L'ultimo campo, facoltativo, è separato da un * e contiene due cifre esadecimali di checksum dei caratteri della frase compresi tra \$ e *, estremi esclusi.

Il primo campo della frase ne identifica il tipo; generalmente inizia con due caratteri che identificano il dispositivo (GP per un ricevitore GPS), oppure con P per le frasi proprietarie definite dai produttori dei vari dispositivi. Il modulo GPS del nostro shield invia prevalentemente le tre frasi più importanti: GGA (Global Positioning System Fix Data, dati di posizione ed accuratezza), RMC (Recommended Minimum, i dati di

posizione più importanti) e GSA (dati sui satelliti).

I campi della frase GGA sono:

- GP GGA (Global Positioning System Fix Data);
- ora (UTC) del fix, nel formato hhmmss;
- latitudine ddmm.mmm per dd° mm.mmm';
- N o S;
- longitudine ddmm.mmm per dd° mm.mmm';
- E o W;
- qualità del fix, i cui valori più importanti sono 0 per fix non valido ed 1 per fix GPS;
- numero di satelliti;
- "HDOP", una misura dell'errore;
- altezza sul livello medio del mare;
- altezza del geoide rispetto all'ellissoide WGS84;
- tempo in secondi dall'ultimo aggiornamento DGPS;
- ID della stazione DGPS;
- checksum.



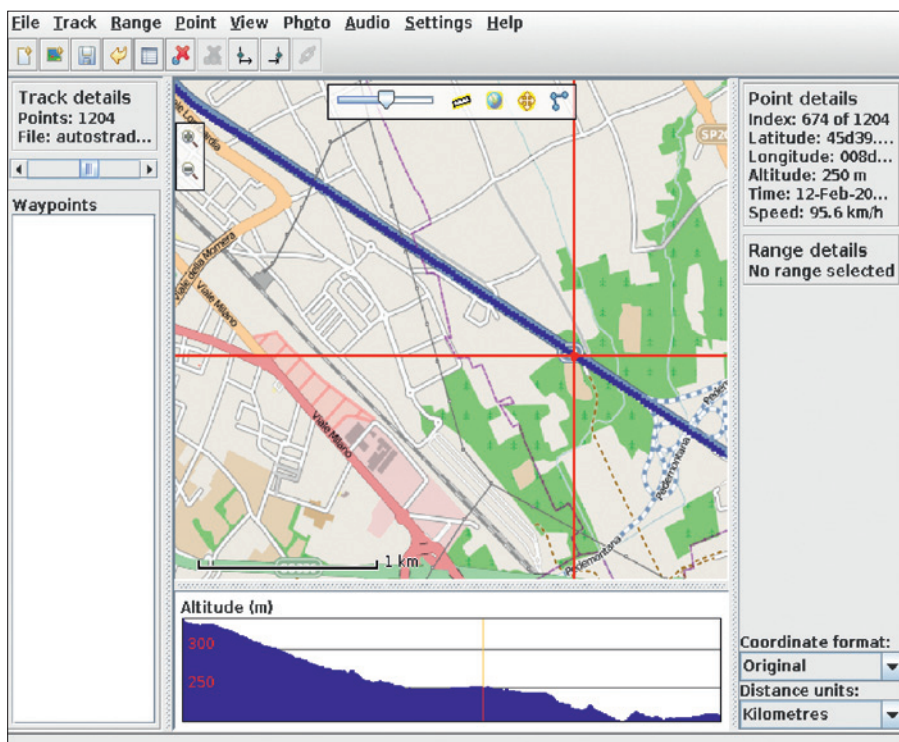
0 e 1 costituiscono l'UART fisico di Arduino, tuttavia se questo non è disponibile perché già usato da altre applicazioni (per

esempio perché avete montato altri shield) potete, tramite la libreria SoftwareSerial, impostare la vostra scheda affinché

gestisca le linee 5 e 6 come un UART virtuale.

Il modulo GPS viene attivato impostando il contatto 7 a livello logico basso: da quel momento trasmetterà ciclicamente stringhe NMEA sulla seriale fino a quando non sarà disattivato impostando lo stesso piedino 7 a livello alto.

Per quello che riguarda lo slot microSD, lo shield comprende il convertitore di livelli 74HC4050D per permettere la comunicazione via SPI (infatti le SD-Card e le micro SD impiegano per la comunicazione con l'esterno il bus Serial Peripheral Interface) a livelli 0/3,3V con la scheda di memoria. I segnali corrispondenti sono collegati al solo connettore che su Arduino UNO corrisponde all'ICSP, in modo da permettere la compatibilità anche con le schede Arduino MEGA. Quindi i contatti laterali dello shield relativi alle linee 10, 11, 12, 13 di Arduino UNO (che ripetono il connettore ICSP) sono scollegati dal resto del circuito; tenete



Mapa di riferimento proveniente da OpenStreetMap, licenza CC-BY-SA
<http://www.openstreetmap.org/copyright>.

I campi della frase RMC sono:

- RMC (Recommended Minimum sentence C);
- ora (UTC) del fix, nel formato hhmmss;
- status del fix: A per "Active" (valido) V per "void" (invalido);
- latitudine ddmm.mmm per dd° mm.mmm';
- N o S;
- longitudine ddmm.mmm per dd° mm.mmm';
- E o W;
- velocità al suolo in nodi;
- direzione di viaggio in gradi;
- data in formato ddmmyy;
- variazione magnetica;
- tipo di fix: A per autonomo, D per differenziale o altri valori per fix non affidabili;
- checksum.

I campi della frase GSA sono:

- GPGGA;
- selezione del fix 3D o 2D (A per auto, M per manuale);
- tipo del fix: 1, 2 o 3 per nessun fix, fix 2D e fix 3D rispettivamente;
- 12 campi per i PRN dei satelliti usati per il fix;
- PDOP (una misura dell'errore);
- HDOP (una misura dell'errore orizzontale);
- VDOP (una misura dell'errore verticale);
- checksum.

Ad esempio, di seguito riportiamo il codice stampato all'accensione del nostro modulo, con frasi proprietarie del chipset seguite da frasi vuote prima di trovare il fix:

```
$PSRFTXT,Version:GSW3.5.0_3.5.00.00-SDK-3EP2.01 *46
$PSRFTXT,Version2:F-GPS-03-1006232*29
$PSRFTXT,WAAS Disable*13
$PSRFTXT,TOW: 237406*11
$PSRFTXT,WK: 1727*66
$PSRFTXT,POS: 6378137 0 0*2A
$PSRFTXT,CLK: 96250*25
$PSRFTXT,CHNL: 12*73
$PSRFTXT,Baud rate: 4800*65
$GPGGA,175631.867,,,,,0,00,,,M,0.0,M,,0000*58
$GPGSA,A,1,,,,,,,,,,,,,*1E
$GPRMC,175631.867,V,,,,,,120213,,,N*40
```

Di seguito, un estratto di dati NMEA una volta conosciuta la posizione:

```
$GPGGA,175802.000,4546.1373,N,00848.7133,E,1,06,1.7,335.7,M
,48.0,M,,0000*5C
$GPGSA,A,3,15,09,05,08,26,28,,,,,,2.8,1.7,2.3*31
$GPRMC,175802.000,A,4546.1373,N,00848.7133,E,37.18,190.20,1
20213,,,A*54
```

tuttavia presente che se ad Arduino applicate in cascata altri shield, non potete comunque usare le predette 10, 11, 12, 13, perché già impegnate. Queste linee possono essere usate come GPIO, oppure per gestire una comunicazione su bus SPI, non entrambe le cose insieme.

Da Arduino si può accedere alla microSD, formattata con filesystem FAT16 o FAT32, usando la libreria SD presente nell'ambiente Arduino, avendo cura di usare *SD.begin(4)* per configurare il Chip Select diverso dallo standard.

E passiamo all'accelerometro, siglato ACC nello schema elettrico: è del tipo a tre assi e si tratta dell'MMA7361 prodotto da Freescale; di questo componente usiamo la versione prodotta da Sparkfun, già montata su una basetta che riporta 9 piedini single-in-line a passo 2,54 mm. Questo modulo è alimentato a 3,3 V da Arduino tramite la linea 3.3 V e GND, che sull'MMA7361 Sparkfun raggiungono i piedini, rispetti-

vamente, VCC e GND. L'accelerometro fornisce in uscita un triplo segnale analogico riguardante l'accelerazione rilevata su ciascuno degli assi; i segnali sono disponibili al piedino 4 per l'asse X, al 3 per l'asse Y e al 2 per l'asse Z.

Tramite il jumper Jref è possibile collegare il VREF dell'Arduino ai 3,3 V, in modo che il valore di 1,65 V in uscita dall'accelerometro, corrispondente ad accelerazione nulla (0g), coincida all'incirca con il valore 512 all'uscita del convertitore A/D a 10 bit di Arduino, ossia a metà dell'escursione (l'A/D spazia da 0 a 1024).

Il jumper JGSEL permette di selezionare una delle due modalità di funzionamento dell'accelerometro: collegato a GND avrà fondo scala di 1.5 g e risoluzione di 800 mV/g, corrispondenti ad un valore AnalogRead di circa 248 per g; collegato a VCC il fondo scala passa a 6 g e la risoluzione a 206 mV/g, corrispondenti ad una lettura di circa 64 per g.

La configurazione predefinita dal nostro logger prevede che i jumper centrali dello shield, ossia GPSRX e GPSTX, siano sempre chiusi sui pin di destra (ossia, rispettivamente D5 e D6) mentre il jumper JGSEL, relativo all'accelerometro, sia chiuso a GND; inoltre il jumper Jref deve essere chiuso.

IL FIRMWARE

Per semplicità, abbiamo deciso di salvare sulla memoria le stringhe NMEA ricevute dal modulo GPS, anche per risparmiare spazio rispetto ai formati più diffusi come GPX e KML, che comunque si possono ottenere a partire dai nostri dati, tramite appositi programmi. Per gestire la comunicazione con il modulo GPS useremo SoftwareSerial; se nel circuito non è presente la scheda di memoria microSD, i dati del GPS verranno riportati sulla seriale dell'Arduino, in modo da poterli passare direttamente ad un computer, sia per il debug che per eventuali applicazioni

Listato 1

```
#include <SD.h>

#include <SoftwareSerial.h>

/* ***** Settings ***** */
char log_filename[13] = "data.log"; // keep this FAT friendly
// how long should we wait when not moving before we stop logging
#define STOP 60000
/* ***** End of user settings ***** */

#define PIN_GPS_ENABLE 7
SoftwareSerial gpsSerial(5,6);

#define PIN_SD_SS 10
File log_file;
boolean sd_available = false;

boolean moving = true;
unsigned long last_move = 0;
#define ZERO_X 512
#define ZERO_Y 512
#define ZERO_Z 512
#define M_THRESH 60000
#define PIN_X 4
#define PIN_Y 3
#define PIN_Z 2

void setup() {
    Serial.begin(9600);

    //GPS setup
    gpsSerial.begin(4800);
    pinMode(PIN_GPS_ENABLE,OUTPUT);
    digitalWrite(PIN_GPS_ENABLE,HIGH);

    //SD setup
    pinMode(PIN_SD_SS,OUTPUT);
    pinMode(10,OUTPUT);
    start_sd();

    start_gps();
}

void loop() {
    if(moving) {
        if(sd_available) {
            log_file = SD.open(log_filename,FILE_WRITE);
            if(!log_file) {
                sd_available = false;
            }
        }
        if(gpsSerial.overflow()) {
            if(sd_available) {
                log_file.write("\r\n$PXXXX,SoftwareSerial overflow!!!");
            } else {
                Serial.println("\r\n$PXXXX,SoftwareSerial overflow!!!");
            }
        }
        while(gpsSerial.available()) {
            int r = gpsSerial.read();
            if(sd_available) {
                log_file.write(r);
            } else {
                Serial.write(r);
            }
        }
        check_movement();
        log_file.close();
    } else {
        delay(1000);
        check_movement();
    }
}

void start_gps() {
    gpsSerial.listen();
    digitalWrite(PIN_GPS_ENABLE,LOW);
}

void stop_gps() {
    digitalWrite(PIN_GPS_ENABLE,HIGH);
}
```

(Continua)

avanzate. Il controllo del movimento tramite accelerometro avviene all'incirca ogni secondo; nel caso venga percepito un movimento (in qualsiasi direzione), il logger rimane attivo o viene immediatamente attivato, mentre se i movimenti sono al di sotto di una certa soglia (M_THRESH) per tutti i controlli effettuati durante un minuto (configurabile tramite il valore STOP, in millisecondi) il GPS viene spento.

I valori presenti nello sketch sono stati controllati empiricamente per fornire risultati accettabili nel tipico utilizzo. Per comodità, in fase di debug, quando smettiamo di salvare i dati, lo scriviamo nel file NMEA, usando una frase proprietaria identificata dalla stringa PXXXX, che verrà correttamente ignorata dai programmi. La stessa stringa viene usata per segnalare eventuali overflow della SoftwareSerial, che potrebbero essere segno di problemi.

Il file su SD (nel nostro esempio DATA.LOG) viene chiuso frequentemente, in modo tale che in qualunque momento sia possibile togliere l'alimentazione, spostare la microSD in un lettore collegato ad un computer ed ottenere la registrazione del percorso fatto.

Il firmware da caricare in Arduino per far funzionare il logger GPS si trova nel **Listato 1**.

FUTURI SVILUPPI

Una personalizzazione semplice del logger consiste nell'integrare alcuni sensori ambientali per aggiungere informazioni al nostro log di viaggio: ad esempio per correlare con la posizione certe condizioni di umidità, temperatura, vento, così da ottenere una georeferenziazione dei dati stessi.

In futuro spiegheremo, invece, come far interpretare le frasi NMEA da Arduino, in modo da poter realizzare progetti più complessi del semplice salvataggio.

GESTIRE I FILE NMEA

I file NMEA salvati dal nostro logger possono essere letti direttamente solo da un numero limitato di programmi. Per fortuna esiste GPSBabel, un programma libero e multiplatforma che permette di convertire da e verso la maggior parte dei formati usati dai sistemi GPS. GPSBabel può essere scaricato dal sito ufficiale, <https://www.gpsbabel.org/> per Windows e OS X (gli utenti Linux possono trovarlo nel repository della loro distribuzione): usarlo è semplice, basta selezionare il formato di ingresso (nel nostro caso NMEA), il nome del file da leggere (DATA.LOG, sulla microSD), il formato da generare (ad esempio GPX per la maggior parte dei programmi o KML per Google Earth) e il nome del file da scrivere. GPSBabel offre innumerevoli opzioni per rifinire la conversione, ma l'impostazione predefinita è generalmente adatta alla maggior parte degli usi.

Un altro programma che può tornare utile è GpsPrune, anche questo libero e multiplatforma (scritto in Java), che permette di visualizzare, convertire e manipolare tracce GPS usando come sfondo le mappe libere del progetto OpenStreetMap, e che supporta direttamente il formato NMEA. GpsPrune è scaricabile dal sito ufficiale <http://activityworkshop.net/software/gpsprune/> ed è anche disponibile nei repository di molte distribuzioni Linux. Per aprire il nostro file sarà necessario rinominarlo con estensione **.nmea**, in modo

Listato 1 - Segue

```
void start_sd() {
  if (!SD.begin(PIN_SD_SS)) {
    Serial.println("Card failed, or not present");
  } else {
    log_file = SD.open(log_filename, FILE_WRITE);
    if (log_file) {
      log_file.close();
      sd_available = true;
    } else {
      Serial.print("Can't open file <");
      Serial.print(log_filename);
      Serial.println("> for writing.");
    }
  }
}

void check_movement() {
  int x = analogRead(PIN_X) - ZERO_X;
  int y = analogRead(PIN_Y) - ZERO_Y;
  int z = analogRead(PIN_Z) - ZERO_Z;
  long acc = (long)x*x + (long)y*y + (long)z*z;
  if (acc > M_THRESH) {
    last_move = millis();
    if (!moving) {
      start_gps();
      start_sd();
      // Ignore data that has overflowed
      gpsSerial.overflow();
    }
    moving = true;
  } else {
    if (moving) {
      unsigned long now = millis();
      // TODO: check for the overflow condition of millis
      // (or reboot the arduino every 49 days :) )
      if (now - last_move > STOP) {
        moving = false;
        stop_gps();
        if (sd_available) {
          log_file.write("\r\n$PXXXX,Device is idle, stop logging\r\n");
        } else {
          Serial.println("\r\n$PXXXX,Device is idle, stop logging\r\n");
        }
      }
    }
  }
}
```

da permettere il riconoscimento automatico del formato, quindi lo si potrà aprire dal menu *File>Open File*.

REALIZZAZIONE PRATICA

La costruzione dello shield è piuttosto semplice: una volta ottenuta la basetta (allo scopo

Le linee di Arduino

Ecco le linee di Arduino che interessano il logger GPS.

- 0 = Serial RX
- 1 = Serial TX
- 4 = SD Chip Select
- 5 = SoftwareSerial RX
- 6 = SoftwareSerial TX
- 7 = GPS enable
- 10 = SPI SS (non usato)
- 11 = SPI MOSI 1
- 12 = SPI MISO 1
- 13 = SPI SCK 1
- A2 = Acc Z
- A3 = Acc Y
- A4 = Acc X

Sullo shield, le 10, 11, 12, 13 non sono collegate perché per la comunicazione SPI usiamo le corrispondenti del connettore ICSP; questo accorgimento ci consente di usare lo shield anche su Arduino MEGA, che ha il connettore ICSP nella stessa posizione e le cui linee SPI sono ripetute sulle 50, 51, 52, 53. Siccome le linee ICSP sono in parallelo con esse, non è possibile impiegarle come GPIO; su Arduino MEGA è invece possibile utilizzarle, ma solo per comunicazioni con altri dispositivi SPI.



dell'informatica elettronica e radiantismo

Rovigo

Cen.Ser. - Viale Porta Adige 45

26 e 27 ottobre 2013

apertura dalle ore 9,00 alle ore 19,00

per tutta la durata della manifestazione saranno ospitate le iniziative

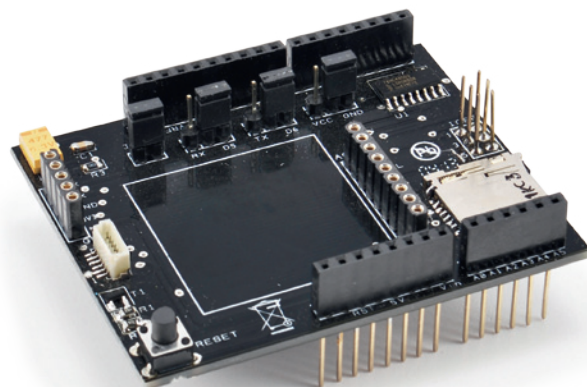
MERCANTE IN FIERA

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

TRASHWARE

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

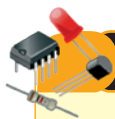
info: 0425 27401 - www.arearebus.com - fiera@arearebus.com



seguite le tracce lato rame scaricabili dal nostro sito Internet) montatevi per primo il lettore di microSD, di cui vanno stagnati sia i piedini, sia le linguette di fissaggio che serviranno a mantenerlo ben fermo durante l'inserzione e l'estrazione della card di memoria. Saldate poi l'integrato 74HC4050D, posizionandolo al centro delle rispettive piazzole, quindi fissandolo con una goccia di stagno su un paio di piedini agli angoli e poi procedendo con i pin rimanenti. Per le connessioni con il ricevitore GPS adottate, in base al modello che avete, il connettore femmina SIL a passo 2,54 mm o il connettore miniatura 6 poli SIL a passo 1,25 mm.

Per il modulo accelerometro usate una strip di 9 contatti femmina a passo 2,54 mm, mentre per i jumper disponete nelle rispettive piazzole dei pezzi di strip maschio a passo 2,54 mm.

I connettori per l'inserzione in Arduino sono i soliti strip femmina (uno da 6, due da 8 ed uno da 10 contatti, almeno se volete la compatibilità con Arduino UNO) con piedini lunghi 20 mm. Per la connessione su ICSP serve inoltre un connettore 6x2 poli a passo 2,54 mm con pin lunghi 20 mm. ■



per il MATERIALE

Lo shield GPS già montato e collaudato è disponibile al prezzo di 22,00 Euro (FT1017M); lo shield non comprende il ricevitore GPS con antenna integrata (cod. EM406A, Euro 45,00), né l'accelerometro a tre assi (cod. MMA7361, Euro 16,00). Tutti i prezzi si intendono IVA compresa.

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)
Tel: 0331-799775 • Fax: 0331-792287 - www.futurashop.it

FREQUENZIMETRO DIGITALE A MICROCONTROLLORE



Descriviamo il montaggio nel contenitore, il firmware e il collaudo finale del nostro strumento di misura della frequenza di segnali analogici BF fino a 10 MHz, TTL e CMOS fino a 50 MHz, e di radiofrequenza fino a 1,1 GHz. Ultima puntata.

del Prof. MICHELE MENNITI

Ci siamo lasciati dopo aver spiegato lo schema elettrico ed il montaggio dei tre circuiti stampati costituenti le altrettante unità del nostro frequenzimetro (Sezione di alimentazione, Sezione logica e display LCD, Sezione ingressi). Abbiamo anche studiato il comportamento dei vari segnali applicati ai tre ingressi, mediante immagini prese dal DSO, in modo da

poter valutare appieno le notevoli qualità del nostro frequenzimetro digitale. In quest'ultima puntata vedremo come montare tutto il frequenzimetro in un contenitore idoneo, in che modo programmare il microcontrollore che governa lo strumento e come utilizzare lo strumento finito. Il contenitore che abbiamo previsto per ospitare

il frequenzimetro è il modello 5100-AUS12 della Teko: è realizzato con i fianchi in plastica e i frontalini anteriore e posteriore in alluminio e dispone di fessure di aerazione sui coperchi superiore e inferiore; dispone inoltre di diversi punti di aggancio sul fondo, utili per il fissaggio degli stampati. Ma prima di parlare del contenitore è necessario

riprendere il discorso, lasciato in sospeso, sull'ingresso di alimentazione 220 Vac: come già detto, la radiofrequenza tende ad infiltrarsi dappertutto, creando disturbi di ogni genere, allo strumento ma anche ad altre apparecchiature localizzate nello stesso locale. Ciò perché spesso l'RF riesce a raggiungere



Fig. 1 - Il pannello posteriore completo.

perfino la rete elettrica e, tramite l'impianto, tutte le apparecchiature ad essa collegate.

Abbiamo lavorato molto per bloccare ogni tentativo dell'RF di infiltrarsi nei circuiti componenti il frequenzimetro e non potevamo quindi trascurare la protezione della rete elettrica. La soluzione che vi proponiamo è estremamente semplice: si tratta di smontare da un vecchio alimentatore da computer la presa filtrata di cui esso dispone e tutto ciò che è ad essa collegato (bobine, condensatori, resistenze, anelli di ferrite) compresi i fili che vanno sullo stampato, che di norma sono di buona qualità e sezione adeguata. Per l'innesto della presa sul pannello posteriore occorrerà lavorare con molta pazienza, ma una volta montata essa darà allo strumento l'aspetto professionale che merita.

I due fili provenienti da tale presa andranno collegati ai capi di un interruttore bipolare, idoneo per commutare 220 Vac, che può essere preso dallo stesso alimentatore da PC, purché presente e di tipo bipolare, in modo da interrompere entrambi i cavi della corrente elettrica. Nei nostri impianti, infatti, non c'è un riferimento fisso per la fase ed il neutro, e siccome quella che va scollegata è sempre almeno la fase, nel dubbio è decisamente preferibile scollegare entrambi i fili. Anche l'interruttore andrà fissato sul pannello posteriore del contenitore. Gli altri due capi dell'interruttore bipolare ovviamente andranno collegati al connettore J1 del PCB dello stadio di alimentazione. Le prese 220V sono sempre a tre poli, il terzo dei quali è la terra e ad esso è nel 99% dei casi collegato un cavetto giallo/verde che termina con un occhiello.

Se, come auspicabile e come prescrive la legge vigente, il vostro impianto elettrico è dotato di messa a terra, dovete collegare l'occhiello della presa al pannello di alluminio posteriore, fissandolo con un bullone più dado. In caso contrario, copritelo completamente con del nastro isolante di buona qualità e lasciatelo libero.

Se non disponete di un alimentatore da PC da smontare, usate un filtro di rete di piccola portata (bastano 0,5 ampere), ad esempio di quelli prodotti dalla Schaffner (www.schaffner.com). I filtri di rete

hanno dei contatti di ingresso (due lamine più quella di massa, che va sull'involucro metallico), cui salderete una presa tripolare da pannello, ovvero direttamente il cordone di alimentazione, e due di uscita, che conatterete all'interruttore bipolare. Ce ne sono anche dotati di presa tripolare incorporata, sicuramente da preferire.

In ogni caso, i collegamenti della sezione d'ingresso 220Vac vanno fatti come mostra la **Fig. 3** della prima puntata (ma anche la **Fig. 3** di questo stesso numero dovrebbe aiutare) dove vedete che tutti i fili che portano corrente alternata sono stati perfettamente isolati con della guaina termorestringente, operazione che vi consigliamo vivamente di effettuare, al fine di evitare di prendere la scossa mentre maneggiate il frequenzimetro col coperchio aperto. In **Fig. 1** il pannello posteriore con il bullone della messa a terra, la presa e l'interruttore.

Il lavoro più delicato e lungo da fare è certamente quello della preparazione del pannello anteriore, in quanto esso ospiterà il display LCD, i due pulsanti di selezione ingresso e portata, le tre prese BNC, la presa SMA, i tre LED di indicazione dell'ingresso attivo. Bisogna prima collocare fisicamente i PCB Logica e Display e quello degli Ingressi nel contenitore e poi iniziare col rilevare i contorni del display, la posizione precisa dei due pulsanti e della presa SMA (saldatura direttamente sul PCB); conviene segnare questi punti nella parte interna

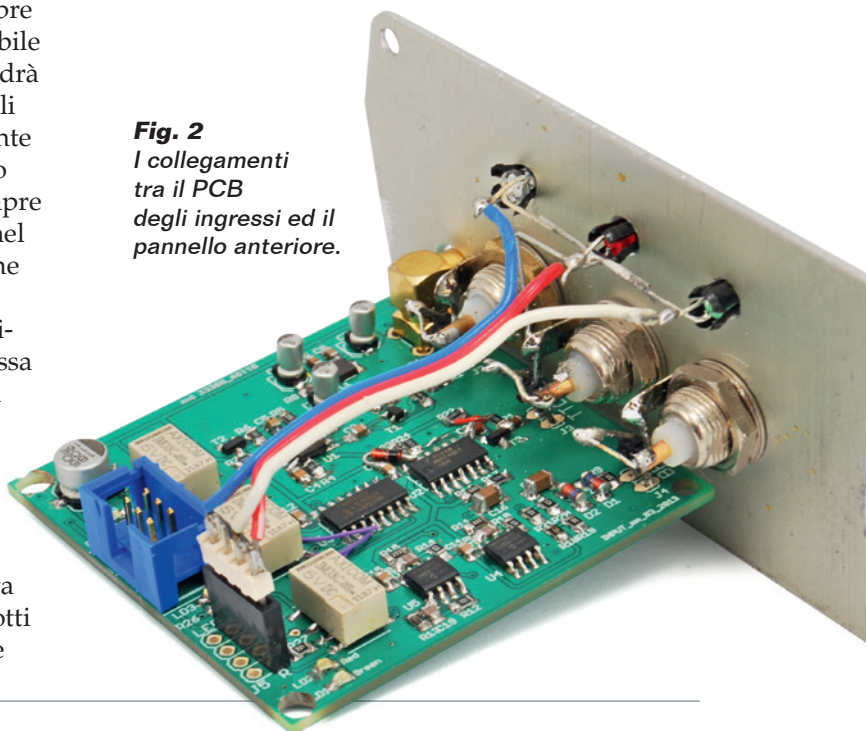


Fig. 2
I collegamenti
tra il PCB
degli ingressi ed il
pannello anteriore.

del frontalino ed iniziare con delle forature molto piccole (consigliamo una punta da 1 mm), in modo da verificare la precisione dei posizionamenti. Solo dopo si potranno usare punte via-via più larghe, così da effettuare dei fori precisi. Anche per il display, conviene realizzare un rettangolo più piccolo rispetto alle dimensioni reali del display, così sarà più semplice lavorarlo con una lima piatta in modo da realizzare contorni precisi.

Più semplice sarà la foratura relativa alle tre prese BNC ed ai tre LED, visto che questi sei elementi saranno collegati al PCB della Sezione ingressi mediante spezzoni di filo rigido; unica accortezza, sarà l'equidistanza tra i tre BNC e tra i tre LED. Inoltre bisognerà curare la perfetta perpendicolarità tra ogni LED ed il rispettivo BNC.

I connettori BNC devono essere dotati dell'occhiello per il collegamento delle masse; il polo centrale di ognuno va collegato alla piazzola del relativo ingresso mediante uno spezzone di filo rigido, mentre l'occhiello andrà collegato alla piazzola di massa (GND). Per semplificare questa operazione ed anche eventuali successivi smontaggi del PCB degli ingressi, noi abbiamo saldato sulle tre coppie di piazzole altrettante coppie di strip maschi, come mostrato nella Fig. 2.

Riguardo ai tre LED da 3 mm, i loro catodi sono collegati insieme e poi all'occhiello della presa BNC "TTL", quindi ad un qualsiasi punto GND, mentre i tre anodi sono stati collegati al relativo connettore mediante una piattina colorata. Il connettore prevede, come detto, anche la connessione per il LED giallo, tuttavia nel nostro prototipo non abbiamo portato questo LED al pannello ma ci siamo accontentati di lasciare solo quello sul PCB per le necessarie verifiche funzionali. Il connettore è anche predisposto per portare due fili per ogni LED, quindi se non volete adottare la nostra soluzione di collegamento, potete usare una piattina da sei fili (otto, se volete portare anche il LED giallo) per collegare anodi e catodi direttamente al PCB; per il collegamento seguite attentamente l'immagine del piano di montaggio del PCB e la serigrafia: l'anodo di ogni LED è indicato con l'iniziale del colore (B=blue, Y=yellow, R=red, G=green).

Comunque sia, conviene adottare un connettore header femmina sul PCB ed uno strip maschio per la piattina; nel caso in cui dobbiate smontare il PCB, invece di dissaldare tutto basterà scollegare lo strip. Per fissare in modo stabile i LED da 3 mm al pannello anteriore, consigliamo di adottare i portaled in metallo o in plastica (questi ultimi sono disponibili presso Futura Elettronica -www.futurashop.it- con il

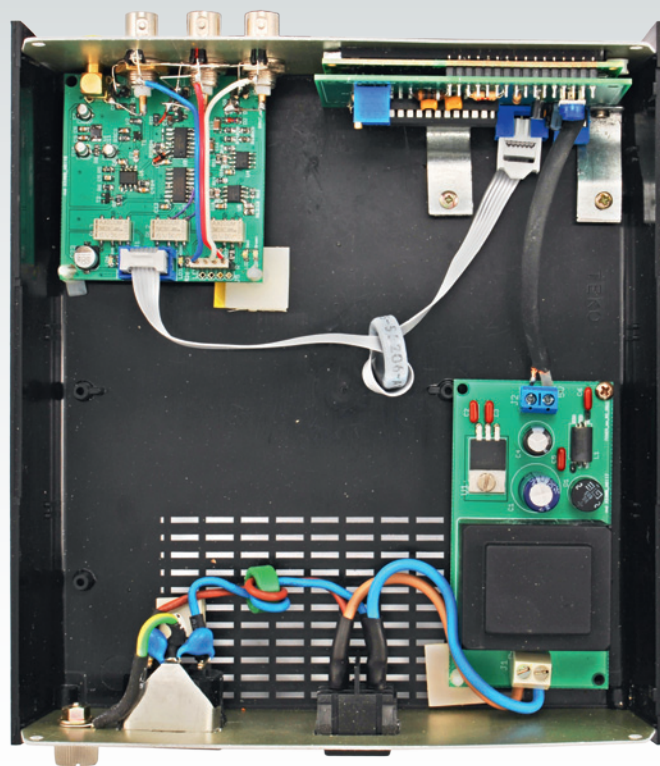


Fig. 3 - Il fissaggio dei tre PCB nel contenitore.



codice 8220-CLIP3); in caso contrario, probabilmente dovrete ricorrere ad un filo di colla a caldo o silicone. Osservando la Fig. 3 potrete prendere spunto per il fissaggio dei tre PCB; noi abbiamo usato viti, distanziatori in plastica (con o senza base adesiva) e linguette di alluminio; insomma, l'importante è collocare correttamente i PCB e fare in modo che non si muovano nemmeno girando il contenitore a testa in giù o scuotendolo.

IL FIRMWARE

La realizzazione di questo progetto è stata possibile fondamentalmente grazie alla libreria "Frequency Counter", a firma di Martin Nawrath della Academy of Media Arts-Cologne, la quale si occupa di tutta la necessaria gestione dei timer del micro, ai fini del calcolo della frequenza applicata al pin

Come il **watchdog** gestisce i pulsanti

Ogni volta che si preme P1, il firmware verifica se questo è stato premuto in precedenza e, in caso affermativo, quanto è durata tale pressione e da quanto tempo è avvenuta; a ciò provvede il debounce software basato sul conteggio del numero degli overflow del contatore del watchdog. Nel caso in cui non sia passato il tempo di debounce previsto, il firmware riconosce la pressione come un rimbalzo del contatto e la ignora. Invece se è trascorso almeno tale tempo, entra nella routine di gestione del pulsante, quindi nella routine d'interrupt "ingresso" che incrementa la variabile SW_COUNT (da 1 a 3, poi si resetta); in base al valore di quest'ultima, viene selezionato il relativo ingresso (BF, TTL, RF),

mediante l'opportuna combinazione dei tre relé della Sezione ingressi. Contestualmente si accenderà il LED che indica quale dei tre BNC a pannello è attivo. Allo stesso modo, ogni volta che si preme P2 viene attivata la routine d'interrupt "portata" che incrementa la variabile HZ_COUNT (da 1 a 4, poi si resetta); in base al valore di quest'ultima, viene selezionata la relativa portata (Hz, kHz, MHz, GHz) che dà origine ad una conversione software della frequenza misurata in base alla portata. Ad esempio, se si misura una frequenza di 1 MHz, cambiando le varie portate si otterranno le seguenti visualizzazioni: 1000000 Hz, 1000.000 kHz, 1.000000 MHz, 0.001000000 GHz. E vediamo ora cosa suc-

cede una volta impostati l'ingresso e la portata desiderati. Siccome il programma opera a ciclo infinito e ogni secondo effettua una lettura dell'ingresso del micro (pin 11) e ne fornisce i dati in base alle impostazioni di portata, se durante un ciclo si preme P1 e/o P2, la lettura viene falsata; perciò bisogna sempre aspettare un paio di nuove letture prima di poter essere sicuri di quanto si legge sul display. Questo vale anche quando si cambia il punto di misura o il valore da leggere: infatti è praticamente impossibile che l'arrivo della nuova frequenza coincida con l'inizio di un ciclo di lettura. Osservando le poche righe presenti nel loop del codice si vede che, una volta effettuata la lettura

della frequenza, vengano eseguite queste routine:

- *Checkingr* legge lo stato dei tre relé e determina la provenienza del segnale d'ingresso;
- *Calcport* converte il valore numerico della frequenza in una stringa, in modo da semplificare i calcoli in base alla portata selezionata;
- *Backlight* gestisce la retroilluminazione, mantenendola a bassa luminosità in assenza di segnale in ingresso e portandola ad alta luminosità quando viene rilevata una qualsiasi frequenza;
- *Lcdprint* prepara la stringa definitiva e la visualizza sul display LCD, mostrando contemporaneamente la portata e l'ingresso selezionato (B, T o R); quest'ultimo

11. La libreria è scaricabile dal link originale <http://interface.khm.de/index.php/lab/experiments/arduino-frequency-counter-library/> e va inserita nella propria cartella personale o in quella originale delle librerie di Arduino, quindi inclusa nello sketch assieme alle altre librerie necessarie ("LiquidCrystal" per la gestione dell'LCD, "avr/wdt" per la gestione del watch dog, entrambe in dotazione all'IDE versione 1.0.1 o successive).

Osservando il firmware (scaricabile dalla sezione download del sito www.elettronica.in), dopo l'inclusione delle librerie troverete l'elenco delle costanti e variabili usate nello sketch; al solito, lo sketch è molto ben commentato, quindi non avrete alcuna difficoltà a seguirlo e comprenderlo: per questa ragione non lo spiegheremo passo-passo, ma solo per sezioni logiche. Il firmware si occupa di gestire le funzionalità dei due pulsanti P1 e P2 che servono, rispettivamente, per scegliere l'ingresso attivo tra BF, TTL e RF e per cambiare la portata tra Hz, kHz, MHz e GHz. Poiché i timer del micro sono impegnati dalla libreria del frequenzimetro, si è fatto ricorso all'uso del watchdog (vedi l'apposito riquadro) per gestire correttamente il debounce (anti-rimbalzo) e garantire il perfetto funzionamento dei due pulsanti. Il bounce è quel fenomeno per il quale, alla semplice ed unica pressione di un pulsante, in

realtà corrispondono diversi contatti; quando si opera in ambiti molto lenti tale problematica può essere ininfluenza, ma se la gestione di un pulsante serve, come in questo caso, ad attivare una routine di interrupt, un qualsiasi microcontrollore riesce a contare tutti i contatti con conseguenze inimmaginabili. Il debounce consiste nel considerare come un solo contatto tutti quelli che si verificano in un intervallo di tempo: tipicamente tra 100 e 200 ms. Il firmware va caricato nel micro mediante la tecnica ISP, già ampiamente illustrata su diversi numeri di Elettronica In o nella Guida alla Programmazione dei microcontrollori ATMEL, scaricabile gratuitamente dal link: www.michelemenniti.it/Arduino_burn_bootloader.php. È di fondamentale importanza settare i fuse del micro (ATmega328P) con i seguenti valori: LOW=e0, HIGH=df, EXTENDED=07, in quanto essi sono calcolati per il funzionamento con l'oscillatore quarzato esterno di precisione previsto nel nostro progetto.

TARATURA E USO DEL FREQUENZIMETRO

Bene, a questo punto siamo pronti per la fase finale: la taratura; si tratta di un'operazione semplice ma anche estremamente delicata, in quanto da essa dipende la precisione che avrà il vostro frequenzimetro. L'ideale sarebbe poter disporre di un genera-

sarà mostrato alternato col carattere "*", creando una sorta di lampeggio che corrisponderà alla frequenza di gate, cioè al ciclo di lettura (1 secondo).

Come abbiamo visto studiando la sezione hardware, il segnale in ingresso può arrivare al micro direttamente (dopo eventuali amplificazioni e/o squadrature), oppure diviso x10 o ancora diviso x400. Inoltre c'è da considerare il comportamento del micro che, quando riceve in ingresso segnali con frequenza oltre i 6 MHz, tende a dare risultati completamente sballati ed incontrollabili; per questa ragione si è impostato un limite di 5,5 MHz, per gli ingressi BF e TTL, oltre il quale viene attivato il

divisore x10. Quindi ogni segnale, dopo la verifica della routine Checkinrg, viene trattato da una successiva routine, selezionata tra chkBF, chkTTL e chkRF, per essere riportato al valore originale, se necessario. Qualche esempio chiarirà il meccanismo. All'accensione dello strumento il firmware, per impostazione predefinita, attiva l'ingresso BF, mantenendo il divisore x10 disattivo ed inizia immediatamente a leggere eventuali segnali applicati a tale ingresso. Immaginando ora di applicare un segnale di 1 MHz all'ingresso BF (1.000.000), essendo un valore inferiore a 5,5 MHz al micro arriverà la frequenza originale, la routine chkBF non eseguirà alcuna operazione, visto che non c'è ne bisogno.

Immaginiamo ora, con il divisore x10 disattivato, di applicare un segnale di 8 MHz (8.000.000); essendo superiore a 5,5 MHz la routine chkBF attiva immediatamente il divisore x10 e "ordina" una nuova lettura che, ovviamente porterà al micro solo 800kHz (800.000); a questo punto la routine moltiplicherà tale valore x10, riportandolo a 8MHz ma non disattiverà il divisore x10, in previsione di ulteriori letture di tale segnale. Se ora andiamo ad applicare allo stesso ingresso BF un segnale di 3 MHz (3.000.000) esso, in funzione del divisore attivo arriverà al micro con valore 300 kHz (300.000). In questo caso la funzione chkBF moltiplica la lettura x10, ripristinando il valore

originale, quindi disattiva immediatamente il divisore x10 e ordina una nuova lettura che questa volta, naturalmente, porterà al micro la frequenza diretta (3.000.000). Il comportamento della procedura per i segnali applicati all'ingresso TTL è identico a quello appena visto per l'ingresso BF; solo che essendo il range di lettura molto più ampio (ricordiamo che è possibile leggere segnali fino a 50 MHz), abbiamo predisposto che, quando si seleziona tale ingresso, si attivi automaticamente il divisore x10. Nel caso di segnali provenienti dall'ingresso RF, poiché essi vengono sempre divisi x400, la relativa routine si limita, ad ogni ciclo di lettura, di moltiplicare x400 il valore in arrivo al micro.

tore di segnali professionale, ma in realtà è più che sufficiente un qualsiasi segnale a frequenza nota, tra 1 MHz e 5 MHz, possibilmente ad onda quadra, purché stabile e preciso. Prima di tutto, però, bisogna regolare il contrasto del display per ottenere la migliore visibilità: l'operazione si effettua ruotando il trimmer R6.

Ricordate che, come spiegato nella sezione relativa al firmware, la retroilluminazione viene gestita in automatico dal software, in base alla rilevazione o meno di un segnale sull'ingresso del micro, quindi non fate caso al suo comportamento, almeno per ora.

Prima di iniziare la taratura bisogna impostare correttamente alcune righe del firmware, come indicato di seguito.

- Nella sezione "VARIABILI" impostate:

```
volatile unsigned val = 0;
```

- Nella routine *lcdprint* occorre riattivare le righe:

```
lcd.print(val); // visualizza la calibrazione.
lcd.print(" ");
```

- Nella routine *loop* occorre riattivare le righe:

```
for (int i=0; i<media; i++)
{
    val += analogRead(f_comp_reg);
}
val = (val/media); // calcolo media e moltiplicatore ADC
```

Come ogni strumento di misura che si rispetti, anche il nostro frequenzimetro necessita di raggiungere una temperatura operativa stabile, quindi consigliamo di eseguire le operazioni di taratura dopo che è acceso da almeno 30 minuti.

Vi ricordiamo che la taratura reale consiste nel trovare il miglior valore da attribuire alla costante di calibrazione *f_comp*, mediante la variabile *val*. Nelle nostre prove iniziali ci siamo scontrati col fatto che ogni tentativo ci costringeva a riprogrammare il micro, allora abbiamo pensato di trasformare, ma solo temporaneamente, la taratura software in una taratura hardware.

Come già sapete, il trimmer di precisione R2 (un multigiri verticale) serve per applicare al pin 28 del micro (pin 5 dell'ADC interno) una tensione variabile tra 0V e quella generata da Aref (1,1 V circa, per impostazione software); tale tensione, ad opera dell'ADC, sarà convertita in un valore intero tra 0 e 1.023, che verrà assunto dalla variabile *val* e quindi dalla costante di calibrazione *f_comp*. In questo modo, una volta applicata la frequenza campione all'ingresso più idoneo per misurarla, e selezionata la portata in Hz (per avere la massima risoluzione possibile), bisognerà ruotare il cursore del trimmer R2 fino a leggere sul display il valore della frequenza campione. Anche se il trimmer è un multigiri, la

regolazione è finissima, per cui quando iniziate a leggere un valore abbastanza preciso dovete ruotare molto delicatamente il cursore; inoltre, e questo è molto importante, considerando che abbiamo applicato il metodo della media, per stabilizzare la lettura dell'ADC, dopo ogni ritocco del cursore conviene sempre attendere un paio di letture per vederne l'effetto corretto. Quando raggiungerete il valore più preciso possibile, non toccate più il trimmer e prendete nota del valore visualizzato sul display (per esempio 992).

A questo punto spegnete lo strumento e modificate nuovamente il firmware, come indicato di seguito.

- Nella sezione "VARIABILI" scrivete:

```
volatile unsigned val = xxx;
// al posto di xxx inserite il valore visualizzato
// sul display a taratura completata (per esempio 992)
```

- Nella routine *lcdprint* occorre disattivare le righe:

```
/*
lcd.print(val);    // visualizza la calibrazione.
lcd.print(" ");
*/
```

- Nella routine *loop* occorre disattivare le righe:

```
/*
for (int i=0; i<media; i++)
{
    val += analogRead(f_comp_reg);
}
val = (val/media); // calcolo media e moltiplicatore ADC
*/
```

Quindi riprogrammate il micro. Da questo momento in poi il trimmer R2 sarà disattivato.

Avevamo pensato di lasciare il metodo di taratura hardware come definitivo, ma nelle varie prove ci siamo accorti che l'ADC subisce sempre una qualche influenza da parte del circuito, soprattutto nelle misure RF, ma anche nelle diverse condizioni operative, per cui il valore tende a variare di ± 1 e questo comporta variazioni di lettura anche importanti, soprattutto alle frequenze elevate. Invece il valore fissato come costante è invariabile e non subisce alcuna alterazione. Naturalmente, nel tempo i componenti tenderanno a mutare le loro caratteristiche, quindi potrebbe rendersi necessaria una nuova taratura; in questo caso, nessun problema: vi basterà ripristinare il firmware e ripetere l'operazione hardware.

Non dimenticate mai che la taratura l'avete effet-

tuata dopo 30 minuti di accensione dello strumento, perciò potrete ottenere la massima precisione solo dopo aver acceso lo strumento per lo stesso periodo di tempo, mentre durante la fase di riscaldamento, facilmente noterete valori non precisi.

Parliamo ora di un componente fondamentale, ma spesso trascurato nella rilevazione delle misure in frequenza: la sonda. In commercio si trovano i classici cavi con un capo BNC maschio e dall'altro lato i due coccodrilli rosso e nero; esistono poi i cavi BNC maschio-BNC maschio, quindi le sonde, ad alta o bassa impedenza, per misure di segnali RF o BF. Insomma, la scelta è davvero vasta, ma se viene fatta male si vanificano le prestazioni dello strumento, sia in termini di sensibilità che in termini di massima frequenza.

Una sonda costruita con del comune cavo schermato per bassa frequenza fornirà buone letture sull'ingresso BF, ma fino a 40÷50 kHz, al massimo 100 kHz, poi le prestazioni diminuiranno a vista d'occhio.

L'ingresso che causa meno problemi è quello TTL, in quanto i segnali hanno ampiezze notevoli, da 3 a 12 Vpp, quindi qualsiasi cavo schermato di decorosa qualità consentirà di leggere anche i 50 MHz.

Invece la questione diventa estremamente problematica con la misura di segnali di radiofrequenza, laddove è indispensabile ricorrere a sonde specifiche, costruite con cavi che devono essere almeno del tipo RG58; ma questo tipo di cavo, oltre i 500÷600 MHz provoca un'attenuazione del segnale che peggiora la sensibilità di lettura dello strumento, tuttavia se non superate il metro di lunghezza l'RG58 fornisce prestazioni di ottimo livello. A



riprova di ciò, considerate che le sensibilità indicate nelle caratteristiche dello strumento sono state effettuate proprio con questo tipo di cavo, nella configurazione a doppio BNC maschio, che è la condizione ideale, in quanto non ci sono dispersioni.

Ma il sistema del doppio BNC è utilizzabile solo nelle misure tra strumenti dotati di tale uscita; l'ideale per le misure RF sono le sonde specifiche, che terminano con un puntale rigido ed un cocco-drillo da collegare alla massa del circuito sotto test; esse sono protette contro eventuali picchi di elevata tensione e permettono di rilevare misure in punti anche poco accessibili. Le sonde che terminano a doppio cocco-drillo, in genere vanno bene per rilevazioni su prototipi, breadboard, millefori, o circuiti con componenti spazati e sui quali sia possibile agganciare il morsetto; in caso contrario possono anche provocare cortocircuiti, quindi nell'usarle bisogna fare molta attenzione.

Per quanto riguarda la protezione degli ingressi, il nostro frequenzimetro è in grado di sopportare ampiezze e potenze abbastanza elevate, ma naturalmente lo strumento va usato in maniera adeguata, soprattutto sull'ingresso RF.

Per segnali particolarmente potenti, quali quelli in uscita da trasmettitori RF, è necessario adottare un sistema di protezione, come un attenuatore di segnale o una sonda adeguatamente protetta, oppure si può evitare il contatto diretto tra la sonda ed il segnale, mediante due tecniche che, naturalmente sono efficaci solo con segnali di elevata potenza.

Per la prima tecnica abbiamo previsto un ingresso di tipo SMA, in quanto con questo tipo di attacco esistono in commercio diversi tipi di antenne accordate sulle frequenze standard oggi in uso: 434 MHz, 868 MHz, 915 MHz, 2,4 GHz le più note.

Collegando una di queste antenne (in base alla frequenza da misurare) alla presa SMA ed attivando la trasmissione dell'apparecchio RF da misurare, se il segnale è adeguato sarà sufficiente avvicinare le due antenne tra loro per leggere la relativa misura sul display, evitando il contatto elettrico e quindi il sovraccarico dell'ingresso del frequenzimetro.

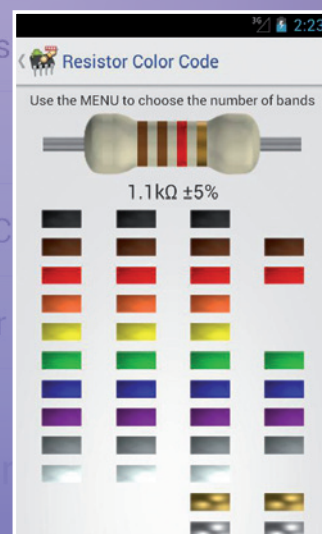
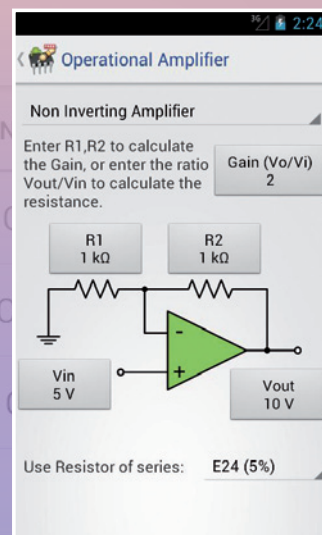
Un'altra tecnica, meno efficace ma estremamente economica consiste nel realizzare un'antenna a cavetto, il fattore fondamentale è la lunghezza dello spezzone; essa si calcola mediante una formula molto semplice:

$$l = 299/F$$

in cui l (*lambda*) è la lunghezza del cavo espressa in metri e F la frequenza da misurare, in MHz. Se

L'elettronica secondo Android

Tra le innumerevoli App per il mondo Android non ne poteva mancare una dedicata all'elettronica: **ElectroDroid** è un insieme di tool (disponibile in versione gratuita che comporta la comparsa di banner pubblicitari, o a pagamento, priva di banner) che contengono il codice colori e numerico per resistori normali ed SMD, il codice colori degli induttori e le formule per il calcolo di reattanza e risonanza capacitiva/induttiva, del partitore di tensione e delle serie e parallelo di resistenze e condensatori, nonché il dimensionamento degli operazionali, dei regolatori di tensione, dei circuiti contenenti LED, di semplici filtri passa basso/alto, dei circuiti con l'NE555 e la piedinatura dei più comuni connettori utilizzati in elettronica (USB, seriale, parallela, Ethernet, RJ, SCART, DVI, HDMI). L'App supporta l'aggiunta di plugin per espanderne le già numerose funzioni.



il valore l è eccessivo, si può ricorrere al fattore λ , dividendo ulteriormente il risultato per 2 o 4 o 8, ed ottenendo quindi un'antenna da $1/2$, $1/4$ o $1/8 \lambda$.

Vediamo un esempio, supponendo di voler verificare la corretta trasmissione di un apparato RF per la CB (Citizen Band, sui 27-30 MHz): in genere questi apparati emettono potenze di 2 o anche 5 watt, che, se si utilizza il collegamento diretto, possono rappresentare un problema per l'ingresso RF. Ricorrendo alla formula sopra citata, vediamo che ci servirebbe un cavo lungo $299/27=11,07$ metri: decisamente un po' troppo da tenere su un banco! Questo risultato si può allora dividere per 2 (5,5 m), per 4 (2,77 m) o per 8 (1,38 m). Una qualsiasi di queste lunghezze, ovviamente realizzata con



Fig. 4 - Visualizzazione della misura di 1GHz.

cavo di buona qualità, è valida per rilevare segnali di frequenza tra 24 e 30 MHz, tenendo conto che le tolleranze (sia in lunghezza che in frequenza) sono abbastanza elevate: infatti il calcolo si effettua tenendo conto della frequenza di centro banda del range che si desidera misurare. Ma se un'estremità del cavo andrà crimpata con uno spinotto BNC maschio, cosa si metterà all'estremità opposta? Ebbene, si può lasciare libera, semplicemente sguainandone qualche centimetro, oppure tra centrale e calza si può saldare un condensatore ceramico, in modo da creare una specie di arco che capterà l'RF permettendo all'ingresso specifico di amplificarne l'ampiezza e misurarne la frequenza.

Molti realizzano queste antenne a filo con del comune cavetto in rame morbido guainato, specialmente se sono molto lunghe, per poterle così raccogliere in qualche modo, ottenendo buoni risultati. Una volta che avrete tra le mani il frequenzimetro pronto e funzionante potrete sbizzarrirvi con tutte le prove che volete, fino a trovare quelle più confacenti alle vostre esigenze.

Ora, un ultimo veloce riepilogo sul semplice uso dello strumento: all'accensione verranno selezionati automaticamente l'ingresso BF (LED verde) e la portata "Hz"; il display LCD, in assenza di segnale avrà una retroilluminazione molto bassa. In queste condizioni bisogna attendere 20-30 minuti di preriscaldamento (oppure dovrete mettere in conto una minima imprecisione iniziale nelle letture), quindi si può selezionare l'ingresso desiderato (se è proprio il BF non bisogna fare alcunché), mentre per quanto riguarda la portata, il problema non si pone, potete ciclare a piacimento tra Hz, kHz, MHz e GHz, senza timore di over-range (fuori portata), in quanto il numero di cifre attive è abbondante anche sulla portata più bassa (Hz).

Quando cambia la portata, il display mostra automaticamente non solo l'unità di misura, ma anche un punto separatore (eccetto la portata in Hz) per facilitare la lettura delle cifre.

In Fig. 4 il display visualizza ben 1.000 MHz, ovvia-

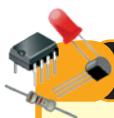
mente rilevati sull'ingresso RF; da notare, appunto, l'indicazione della portata (MHz), il punto separatore tra la parte dei MHz e la restante parte del valore visualizzato, nonché la lettera "R" che indica l'ingresso attivo in questo momento.

Nota conclusiva: considerate che, come ogni strumento di misura, anche il nostro frequenzimetro garantisce la lettura ± 1 digit, cioè è normale che l'ultima cifra tenda a cambiare di un'unità; per esempio, la lettura di una frequenza di 1.258 Hz potrebbe variare tra 1.257 e 1.259 Hz. Ma nel nostro caso bisogna considerare il fattore divisore automatico; per poter sopperire alla problematica della frequenza massima che riesce a leggere il micro abbiamo fatto ricorso a due divisori: uno, fisso, che divide ogni segnale RF $\times 400$ e l'altro, automatico, che all'occorrenza può dividere il segnale BF o TTL $\times 10$. In questi casi il "digit" è rappresentato proprio da questi fattori di divisione.

Cioè mentre una frequenza BF o TTL letta in assenza di divisore (quindi massimo 5,5 MHz) ha una risoluzione di ± 1 Hz, una frequenza maggiore di 5,5 MHz (sempre su BF o TTL) che richiede l'innescio automatico del divisore $\times 10$, avrà una risoluzione di ± 10 Hz; ad esempio 8,5 MHz letti su scala Hz diventano 8500000, ma in questo caso il "digit" diventa la seconda cifra da destra, e non più la prima, quindi potreste leggere 8499990 o 8500000 o 8500010.

Allo stesso modo, nelle misure RF, in cui abbiamo un divisore fisso $\times 400$, la risoluzione diventa appunto di ± 400 Hz; ad esempio leggendo 550 MHz, il "digit" diventa la terza cifra, quindi potreste vedere 549999600 o 550000000 o 550000400.

Comunque sia, un "errore" sulla 6ª cifra nell'ordine dei MHz o sulla 7ª cifra nell'ordine delle centinaia di MHz è abbastanza trascurabile da poter essere ignorato; bisogna infatti considerare che molti generatori di segnale hanno addirittura una risoluzione più bassa, non andando oltre le sei cifre complessive.



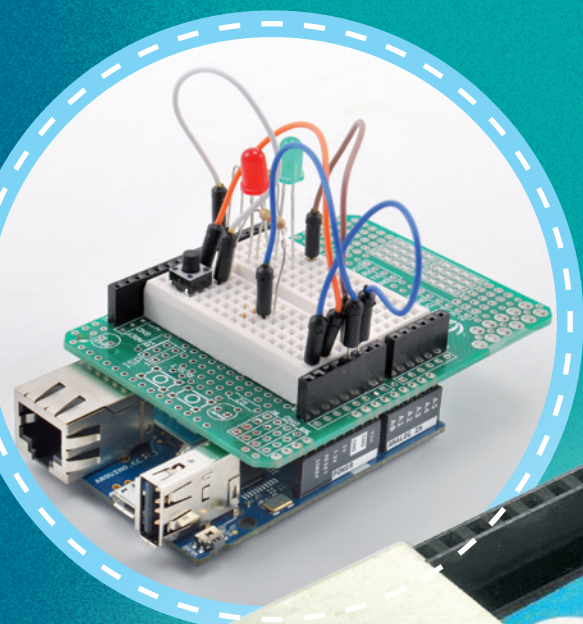
per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono di facile reperibilità. I master dei circuiti stampati possono essere scaricati dal sito della rivista così come il firmware utilizzato per programmare il microcontrollore Atmel.

Collegati subito a www.elettronica.in.it !

ARDUINO YÚN: LA VITA SI FA PIÙ SEMPLICE

di MARCO MAGAGNIN



Ecco l'ultima board di casa Arduino. Il DNA è quello di famiglia, ma l'evoluzione l'ha resa aperta alle nuove esigenze: Linux embedded, connettività ethernet e WiFi. Per provarla, sviluppiamo un'applicazione che invia dei Tweet al verificarsi di eventi locali.

Appena giunta in redazione la Arduino Yún, non abbiamo resistito alla tentazione di provarla, per verificarne le caratteristiche innovative anticipate nelle presentazioni ufficiali. Secondo gli annunci, la scheda integra le caratteristiche di un'Arduino Leonardo standard, con una sezione hardware aggiuntiva dotata di un processore in grado di ospitare una distribuzione GNU/Linux completa, nella fattispe-

cie OpenWRT, specificatamente destinata alla connettività Internet (è la distribuzione open source utilizzata nei router ed access point WiFi). Presa in mano la confezione, nasce la prima domanda: sicuri che mi avete dato una Yún? La scatola è delle stesse dimensioni di quella di Arduino Uno. Aperta

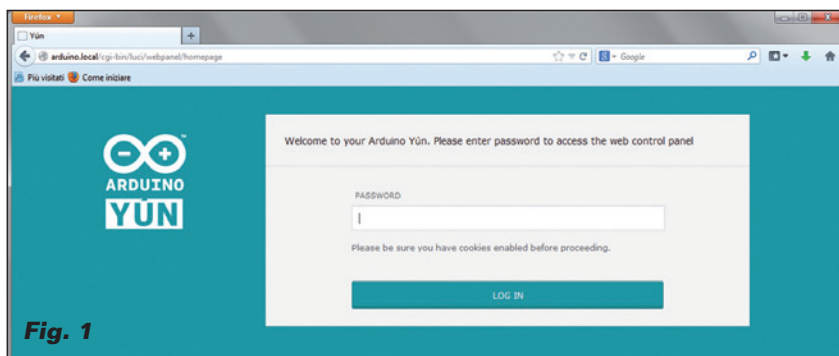


Fig. 1

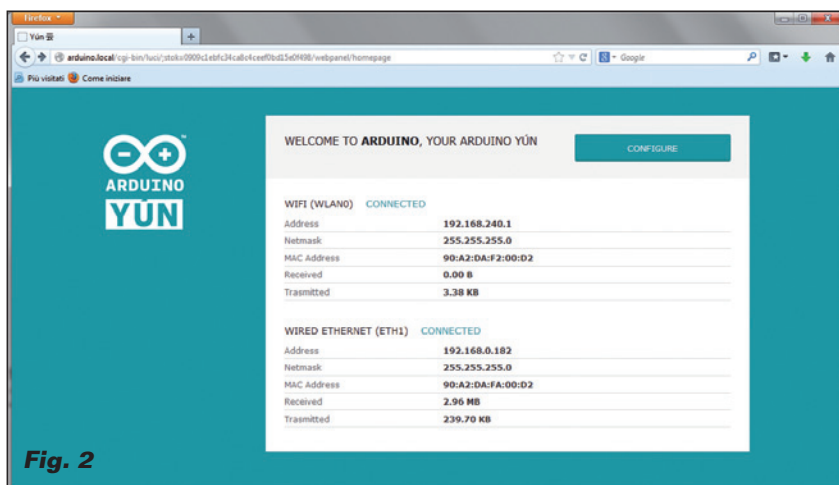


Fig. 2

la scatola ne siamo stati certi: sì, è una Yún, anche se ha le stesse dimensioni di un Arduino Uno (ovvio, perché deve essere compatibile con gli stessi shield). Rigi-randola tra le mani si notano due prese USB, una presa Ethernet, un alloggiamento per una micro SD-Card, i connettori standard di un Arduino. Sono inoltre presenti tre pulsanti e un connettore per collegare un'antenna esterna alla sezione WiFi. Il tutto circondato da una buona quantità di elettronica SMD posizionata su entrambe le facce del circuito stampato. Volendo avere un approccio tipico da sperimentatori provenienti dal mondo GNU/Linux, prima di approfondire ulteriormente le caratteristiche della scheda proviamo a vedere cosa succede, caso mai, dopo leggiamo il manuale! Dunque, colleghiamo un cavo di rete alla presa Ethernet di Arduino Yún e diamo alimentazione mediante un alimentatore a 5V

con presa micro USB (quella per la ricarica dei telefonini, per intenderci) al connettore micro USB al centro della scheda. Il LED ON si accende di un bel verde e poco dopo il LED WAN diventa giallo. Vuoi dire?

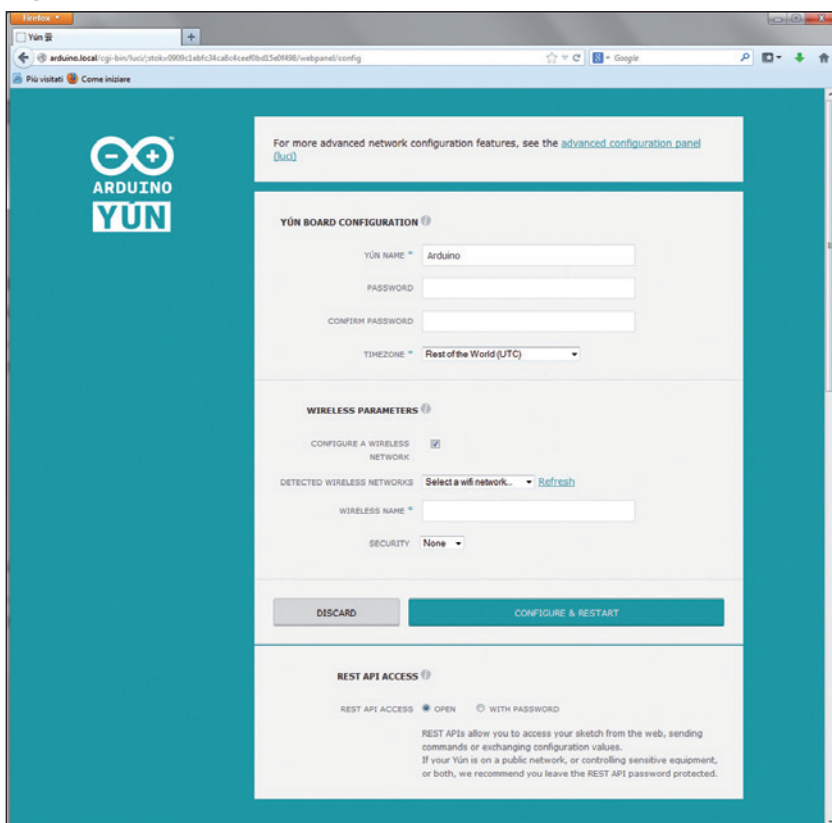
Apriamo un browser e digitiamo l'indirizzo:

http://arduino.local

(in realtà qualcosa bisogna pur leggerlo, nel "getting started" sul sito *www.arduino.cc* dedicato a Yún). Sorpresa: otteniamo la pagina visibile in Fig. 1.

Diamo la password "arduino" (è la password dell'utente ("root") predefinito) e vediamo cosa succede. Otteniamo la pagina visibile in Fig. 2, che ci mostra la configurazione delle interfacce di rete attive. Già questa pagina ci dice molte cose. Yún nasce con un sistema operativo GNU/Linux

Fig. 3



installato, con le interfacce di rete preconfigurate ed un web server attivo con l'applicazione LuCI già installata per configurarne il funzionamento lato GNU/Linux. A proposito: avete notato che per il primo accesso abbiamo usato un nome (*arduino.local*) e non un indirizzo IP? Questo significa che non è necessario dare la caccia all'indirizzo IP assegnato a Yún dal server DHCP della nostra rete, ma possiamo accedere direttamente con il nome host della scheda, in quanto la distribuzione GNU/Linux di Yún, orientata alla connettività, è dotata di un server DNS interno.

Nella Fig. 2 vediamo poi qual è l'indirizzo IP assegnato alla scheda lato WAN e la configurazione della sezione WiFi, già attiva e preconfigurata come access point con un IP statico. Questo significa che possiamo collegarci alla Yún anche in WiFi, semplicemente

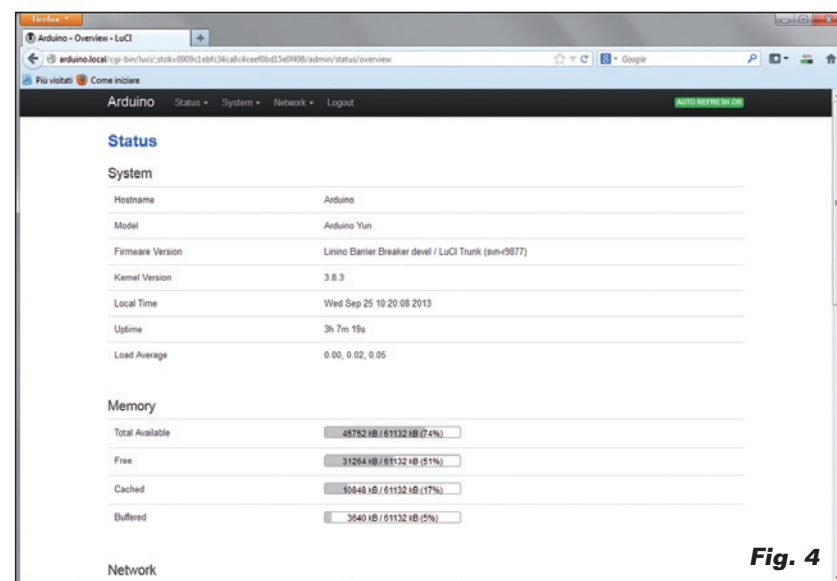


Fig. 4

facciamo clic sul grosso pulsante "Configure" in alto a destra della pagina: otteniamo un'ulteriore pagina di configurazione (Fig. 3) riportante, in alto a destra, un link che attira la nostra attenzione. Prima, però, ne approfittiamo per configurare la giusta Time Zone (Europe/Rome) e selezionare il check box "Open" nella sezione

strazione del sistema GNU/Linux (Fig. 5).

A proposito, la distribuzione personalizzata per Arduino Yún si chiama "Linino" e, come già detto e come possiamo vedere dal menu di configurazione, è derivata da Open WRT. Se per curiosità andiamo alla sezione "Network" e "Wifi", all'interno della pagina,

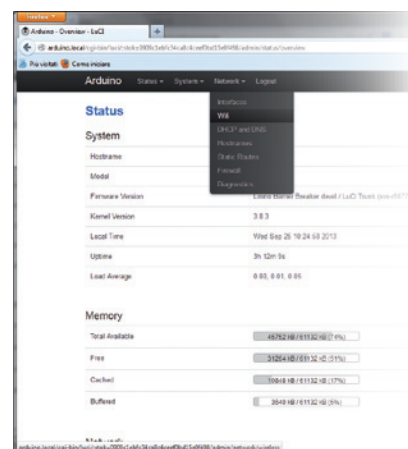
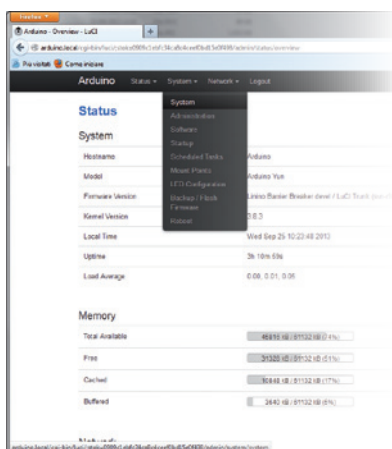
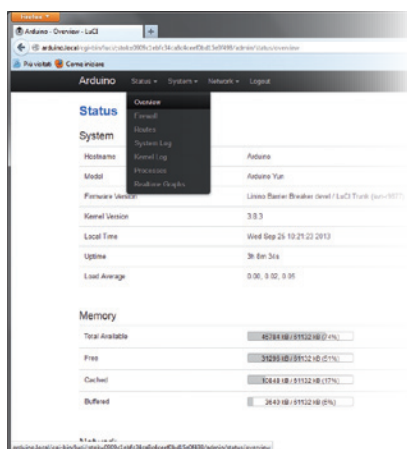


Fig. 5

connettendosi alla rete con SSID "Arduino Yun-<mac-address-della-scheda>". Niente monitor, niente tastiera né mouse, niente collegamento USB col PC e, volendo, niente cavo di rete. Il tutto è finalizzato a facilitare le cose a chi è alle prime armi. Continuiamo a curiosare e

"Rest API Access"; di questo parleremo in seguito. Bene, clicchiamo sul link "advanced configuration panel" ed entriamo in un vero e proprio ambiente di amministrazione, visibile in Fig. 4. Dal menu in alto, in stile Google, possiamo accedere alle diverse sezioni di ammini-

clicchiamo sul pulsante "Edit" e scorriamo la pagina un po' verso il basso, troviamo la sezione che ci permette di configurare il modulo WiFi della scheda nelle diverse modalità (Access Point, Client, Ad Hoc, ecc). Le altre sezioni ci consentono di impostare regole di sicurezza accessi

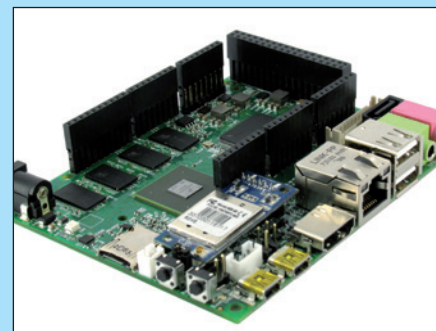
(Firewall), regole di Routing, DHCP e DNS. In definitiva, ci troviamo di fronte ad un vero e proprio dispositivo di connettività che ci permetterà di risolvere una vasta gamma di applicazioni, di esigenze di elaborazione e di configurazione di rete.

Un'ultima curiosità: proviamo a connetterci alla scheda utilizzando i client SSH Putty e WinSCP (per chi volesse saperne di più su tutto quanto detto finora, rimaniamo agli articoli introduttivi su Raspberry Pi ed al libro "Il mio primo Linux embedded"). Ci connettiamo con l'indirizzo assegnato dal DHCP della nostra rete (lo vediamo nella pagina in Fig. 2), con utente "root" e password "arduino". Otteniamo i risultati visibili in Fig. 6 e Fig. 7, che ci dimostrano che anche il server SSH è attivo e funzionante "out of the box".

Già con ciò che abbiamo intravisto ci sarebbe materiale sufficiente a scrivere un libro sulle possibilità di questa scheda, la quale contiene una vera e propria architettura integrata che riunisce in pochi centimetri quadrati un access point WiFi, un sistema GNU/Linux con applicazioni web già disponibili ed un microcontrollore

Arduino in grado di interfacciarsi al mondo esterno nei modi che già conosciamo. Quasi ci stavamo dimenticando che si tratta di una scheda Arduino...

A questo punto non possiamo non sottolineare la valenza didattica di questo dispositivo, che, all'approccio semplificato alla programmazione dei microcontrollori, affianca la possibilità di avvicinarsi in modo altrettanto semplificato al mondo GNU/Linux nella sua configurazione embedded ed al mondo della connettività wireless e di rete in generale, nelle sue diverse configurazioni. Approfondiremo questi argomenti, anche se non sono di taglio specificamente elettronico, in articoli dedicati, ben consapevoli che i mondi dell'elettronica, dell'informatica e delle architetture di rete si stanno fondendo in un tutt'uno fortemente integrato. Da questo punto di vista, Arduino Yún rappresenta il punto di partenza ottimale per iniziare a sperimentare l'integrazione delle diverse discipline elettroniche, informatiche e di rete, oltre che per realizzare applicazioni che uniscono l'acquisizione ed elaborazione di dati provenienti da sensori con la pubblicazione



La necessità di rendere disponibili alle schede embedded le connettività di rete cablata e wireless è sentita non solo dal team Arduino, ma anche da chiunque sviluppi board di prototipazione. Vediamo cosa si muove in questo settore.

UDOO

Lo stesso giorno dell'annuncio da parte del team di Arduino della disponibilità della Yún, sul blog di Udo veniva annunciato l'inizio della produzione della tanto attesa board Udo, un progetto finanziato grazie a Kickstarter, con l'intento di superare

e gestione degli stessi dal mondo esterno. Al momento, la massima espressione di questa tendenza è il dispositivo myRIO di National Instruments, che integra in un unico chip la potenza real time di un tessuto FPGA ed un processore di elaborazione che ospita GNU/Linux con la connettività tra i due, basata su sedici canali DMA ultraveloci.

Fig. 6

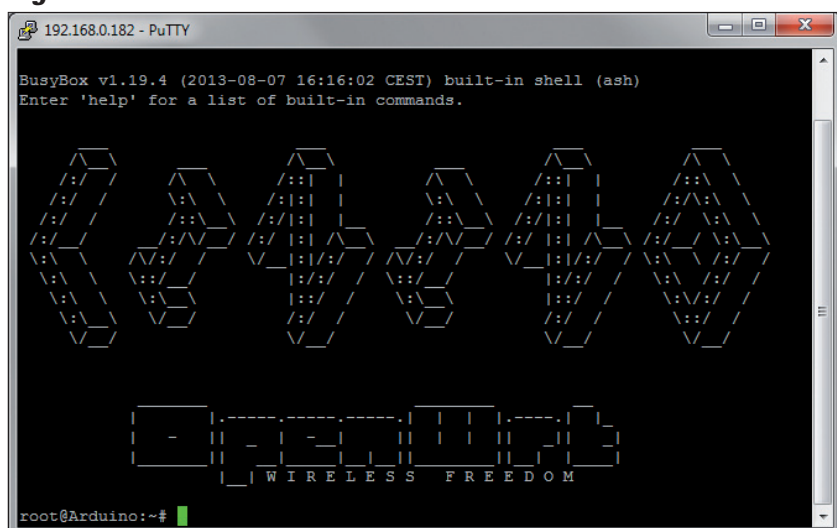
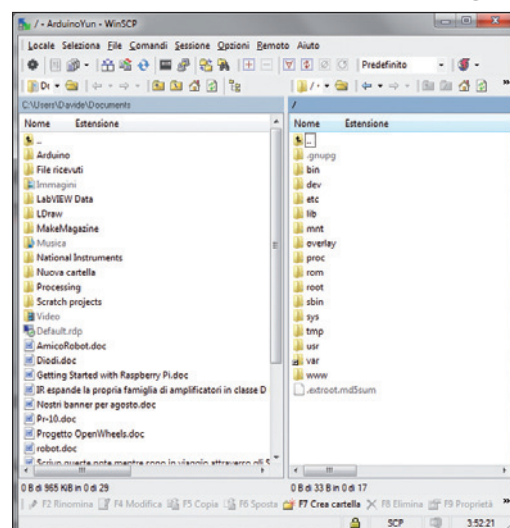


Fig. 7



la scarsa propensione di un sistema a microcontrollore quale Arduino alla connettività (cosa naturale per sistemi Linux-based, che possono fornire l'accesso a Internet in maniera naturale). Infatti la scheda è corredata da due processori: un Freescale i.MX6 ARM quad o dual core (in base alla versione che sceglierete) con 1 GB di RAM, per la sezione GNU/Linux, ed un Atmel SAM3X8E ARM Cortex-M3 per la sezione Arduino (lo stesso di Arduino Due). La comunicazione tra le due sezioni dovrebbe avvenire mediante un collegamento seriale dedicato. La scheda è corredata da una versione personalizzata di Ubuntu, ma grazie all'apposito slot per la scheda

SD è possibile installare un'altra distribuzione Linux o Android. La scheda è dotata di modulo WiFi ed espone anche un connettore Ethernet, un'uscita HDMI, una porta SATA, un controller USB, un jack per gli altoparlanti ed uno per il microfono.

NI myRIO

Un sistema embedded compatto e dotato di grande connettività è myRIO di National Instruments, nato per offrire agli studenti la possibilità di sviluppare rapidamente sistemi ingegneristici reali e completi. Basato sulla stessa tecnologia della piattaforma NI CompactRIO, NI myRIO utilizza un SoC Xilinx Zynq-7020 che combina un processore dual-core ARM

@Cortex™-A9 e FPGA 7-Series di Xilinx con 28000 celle logiche programmabili. myRIO si programma in LabVIEW, quindi rapidamente e intuitivamente. NI myRIO dispone di 10 ingressi analogici, sei uscite analogiche, canali I/O audio e 40 linee di I/O digitale. Inoltre è dotato di WiFi su scheda, di accelerometro a tre assi e diversi LED programmabili. L'introduzione di NI myRIO nell'architettura LabVIEW RIO (Reconfigurable I/O) permette a National Instruments di fornire strumenti a tutti i livelli di competenza, dall'apprendimento di base degli studenti, fino alle soluzioni più avanzate per i progettisti. Ideale per lezioni in classe e in laboratorio, NI myRIO include

risorse didattiche gratuite, è compatibile con tutti gli NI miniSystems e con una vasta gamma di attuatori e sensori di terze parti. NI myRIO è programmabile in LabVIEW, C, C++ e altri ambienti di sviluppo, in modo da poterlo integrare facilmente nei corsi didattici sui controlli, la robotica, la mecatronica e i sistemi embedded.



È ora il momento di fermarsi un attimo e approfondire le caratteristiche generali della Yún. Riprendendola in mano notiamo un'ulteriore caratteristica dall'indubbia utilità: sui lati esterni dei connettori per gli shield sono serigrafati i riferimenti principali dei singoli pin, che permettono di utilizzare la scheda senza bisogno di schema dei collegamenti. In **Fig. 8** è visibile l'immagine della scheda con i collegamenti esterni e i pulsanti di reset.

Ma procediamo con ordine: dal lato "microcontrollore" Yún utilizza il chip Atmega32U4, lo stesso di Arduino Leonardo, del quale riproduce le caratteristiche (16 MHz di clock e 32 kB di RAM, dei quali quattro occupati dal bootloader). Per quanto riguarda l'alimentazione, non c'è il regolatore a 5V, quindi bisogna fare attenzione a non alimentare la scheda con una tensione superiore (pena danneggiamenti anche irreparabili) se si intende fornire l'alimentazione tramite il contatto

Vin. A questo riguardo, viene in aiuto la presa di alimentazione in formato micro-USB: per loro natura, gli alimentatori con questo tipo di uscita erogano 5V. Questa micro-USB serve anche a collegare la scheda al PC per la programmazione tradizionale lato Arduino. Ma è anche possibile eseguire l'upload degli sketch via rete, fisica o WiFi direttamente

dall'IDE. A proposito: l'IDE da utilizzare, scaricabile liberamente (come sempre) dall'indirizzo <http://arduino.cc/en/Main/Software>, è la versione 1.5.4 r2 o superiore, che risolve un bug di funzionamento sotto Windows 8. Esistono anche release per GNU/Linux e Mac OS X. Un'altra differenza di comportamento da Leonardo, sta nel fatto

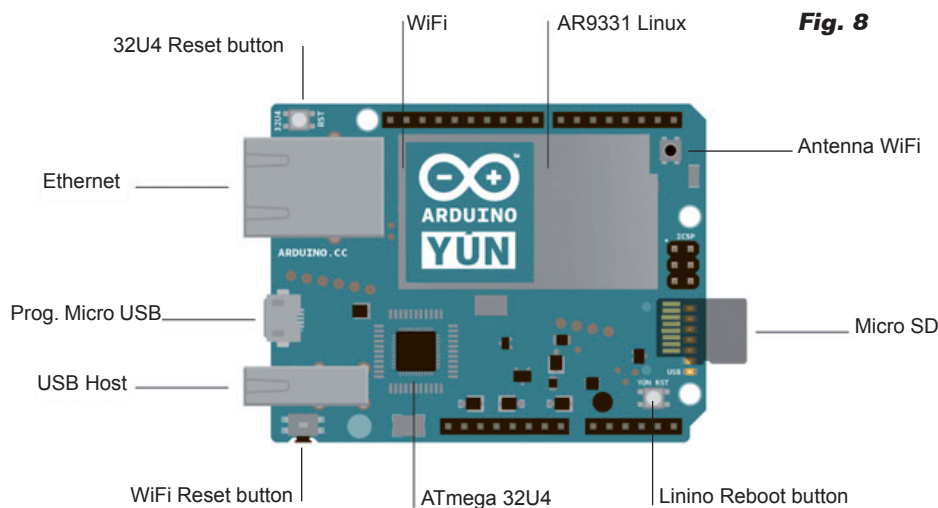


Fig. 8

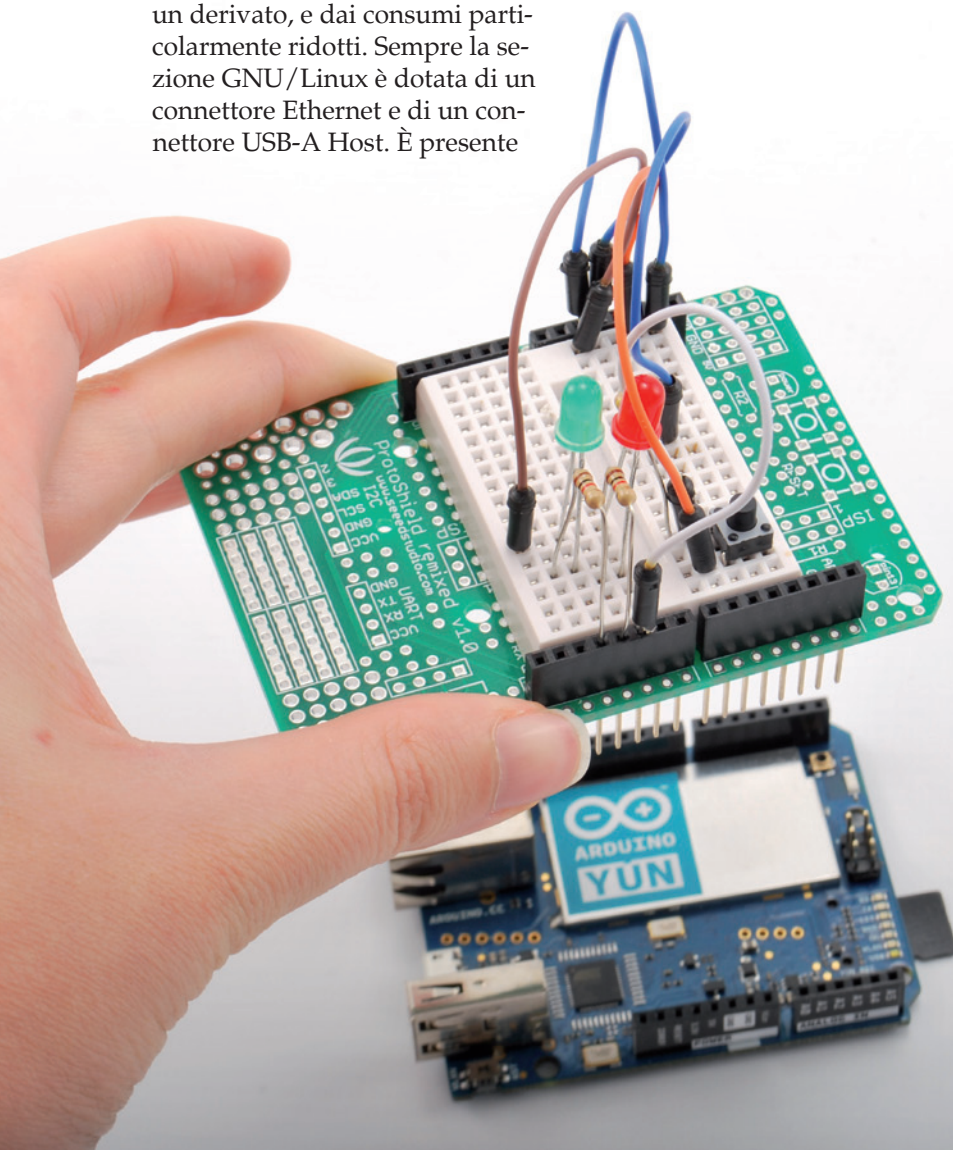
che il microcontrollore non si resetta quando viene aperto il monitor seriale dell'IDE. Inoltre non è possibile utilizzare l'interfaccia Serial1, riservata alla comunicazione con la sezione GNU/Linux. Per resettare il microcontrollore è presente l'apposito pulsante "32U4 Reset Button" a fianco del connettore Ethernet (vedi Fig. 8). La sezione GNU/Linux è gestita dal secondo processore, un Atheros AR9331 a 400 MHz. Probabilmente la scelta è stata determinata dalla particolare predisposizione del SoC ad ospitare la distribuzione GNU/Linux OpenWRT, della quale Linino è un derivato, e dai consumi particolarmente ridotti. Sempre la sezione GNU/Linux è dotata di un connettore Ethernet e di un connettore USB-A Host. È presente

uno slot per accogliere una micro SD-Card, sulla quale indirizzare la parte "personale" del file system della sezione GNU/Linux, in modo da risparmiare la poca RAM NAND assegnata al processore Atheros (peraltro caratterizzata da un numero di possibili riscritture abbastanza limitato: circa 100.000). Per fare questo è sufficiente inserire una micro SD nello slot e creare due cartelle che sono automaticamente riconosciute e utilizzate dal sistema operativo:

*/arduino e
/data*

In realtà, al momento del boot il sistema operativo monta automaticamente tutti i device di memoria di massa esterni denominati "sda", più qualche cosa d'altro. Vedremo in dettaglio l'utilizzo di queste cartelle.

Nella sezione GNU/Linux è presente anche un modulo WiFi, collegato anch'esso al processore Atheros AR9331, che fornisce l'accessibilità wireless all'intero sistema, con l'ulteriore possibilità di collegare un'antenna esterna. Veniamo ora ai pulsanti di reset: il primo lo abbiamo già visto e permette il classico reset del microcontrollore Atmega32U4. Gli altri due consentono, rispettivamente, il reset del processore Atheros AR9331 e del modulo WiFi. La pressione del pulsante "Linino reset button" permette di eseguire il reboot del sistema operativo Linino. Chiaramente tutti i dati in memoria saranno persi e tutti i processi terminati. Il pulsante WiFi reset button ha una doppia funzione: premendolo brevemente fa riportare il modulo WiFi alla configurazione di fabbrica, cioè al funzionamento come Access Point con IP statico pari a 192.168.240.1. Invece, premendo a lungo (per 5 secondi) il pulsante, si resetta il modulo WiFi. Il reset del modulo WiFi comporta anche il reboot di Linino. È importante notare che il tasto non va premuto per più di 5 secondi, in quanto tenuto premuto per più di 30 secondi, provoca il ripristino del sistema operativo Linino con la versione originale, ovvero quella caricata in fabbrica. Questo è stato voluto per aiutarci nel caso vogliamo recuperare il sistema dopo eventuali "operazioni maldestre". La funzione di ripristino ci lascia la libertà di sperimentare senza il timore di trovarci in situazioni senza via d'uscita: più semplice di così...



Chiaramente ripristinando il sistema si perdono tutte le configurazioni fatte dopo la prima accensione e gli eventuali aggiornamenti eseguiti su Linino.

Adesso spendiamo qualche parola sui LED della scheda. Arduino Yún dispone dei seguenti LED:

- USB - blu, indica attività sul connettore USB;
- WLAN (WiFi); WiFi attivo;
- POWER; indica la presenza di alimentazione;
- WAN; indica la presenza di una connessione di rete Ethernet attiva;
- Pin 13; riporta lo stato del pin 13 (a scheda nuova lampeggia per il programma Blink caricato all'origine);
- Serial TX; segnala attività sulla porta seriale in trasmissione
- Serial RX segnala attività sulla porta seriale in trasmissione

In Fig. 9 è visibile lo schema dell'architettura generale della scheda. Sono chiaramente visibili le due sezioni "Arduino" e "Linino", con le periferiche assegnate a ciascuna ed il "ponte" di collegamento indicato come "Bridge". Questa caratteristica è quella che permette ai due ambienti di operare congiuntamente e costituire un'innovativa soluzione Linux embedded in grado di portare nel mondo esterno tutte le funzionalità del microcontrollore Arduino, finora confinate al solo ambito locale. A dire il vero, nel passato sono state presentate soluzioni di connettività basate su shield Ethernet, WiFi, Bluetooth, ecc., ma tutte dovevano fare i conti con la vocazione "microcontrollore" dei dispositivi e con le limitate risorse di memoria a disposizione. Era pressoché impossibile gestire diversi utenti e pagine HTML complesse, per non parlare della gestione delle concorrenze tra accessi web e gestione di sensori

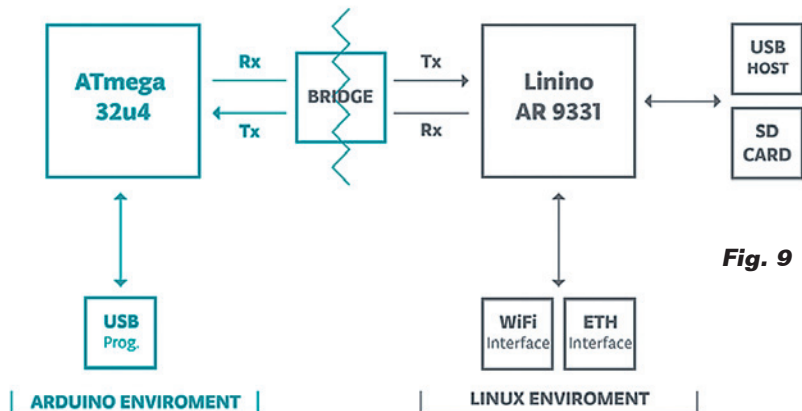


Fig. 9

e periferiche di I/O. Con questa architettura tutto è superato e ciascuna sezione si occupa al meglio della rispettiva funzionalità. La disponibilità di librerie e ambienti di sviluppo adatti a gestire il "ponte" di comunicazione integrati nell'IDE e sviluppati nella logica di semplificazione propria della "filosofia" Arduino, fanno il resto, dando la possibilità di sviluppare soluzioni, anche complesse, con la consueta semplicità e linearità.

Soffermiamoci un momento sulla soluzione Bridge: i processori ATmega 32u4 e Atheros AR9331 sono collegati mediante porta seriale; dal lato Arduino, la libreria "Bridge" permette di realizzare il "ponte" verso l'ambiente Linino, utilizzando la porta seriale Serial1, che non può essere utilizzata per altri scopi. Semplificando molto, la libreria "Bridge" invia sulla porta seriale dei "comandi" di "console" a Linino. I "comandi" permettono di attivare dal lato Linino i processi in grado di fare da ponte con i servizi e le applicazioni "GNU/Linux". Dal lato Linino, la seriale è assegnata, al momento del boot, all'interfaccia CLI (Command Line Interface). In questo modo tutte le stringhe di comandi inviati dalla libreria "Bridge" sono interpretati da Linino come comandi di shell, che attivano programmi in Python che si

incaricano di realizzare le diverse modalità di interfaccia disponibili. In questo modo è possibile instaurare la comunicazione tra la sezione Arduino e Linino, simulando un'emulazione di terminale remoto (Console), come shell per la gestione di comandi e processi remoti, come ponte per leggere e scrivere file sulla SD-Card di Linino dagli sketch Arduino, come ponte di comunicazione di URL HTTP per gestire dallo sketch le richieste passate del web server uHTTPd di Linino (micro Hypertext Transfer Protocol Daemon) e in altre modalità.

Sempre dal lato Linino, sono disponibili diversi strumenti per utilizzare applicativamente questa tipologia di comunicazione. Citiamo il framework REST (REpresentational State Transfer) che permette di interfacciare i pin di I/O lato Arduino mediante URL HTTP del tipo:

<http://arduino.local/arduino/digital/13/1>

che significa: porta a livello alto il pin di I/O digitale 13. Nei prossimi articoli vi presenteremo applicazioni di esempio basate su questo framework. Un altro framework gestibile con la libreria "Bridge" è *Spacebrew*, che permette a diversi dispositivi Client di comunicare tra loro in

Fig. 10

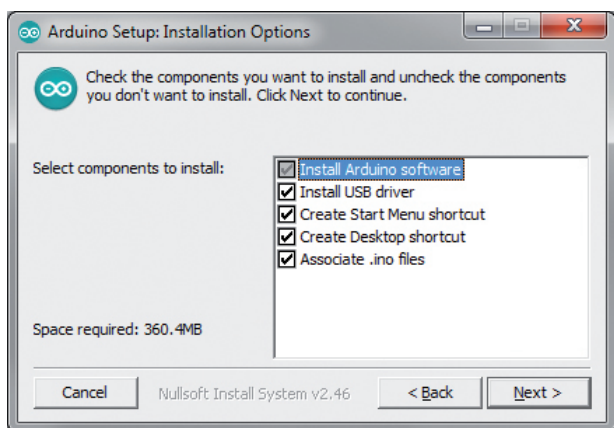


Fig. 11

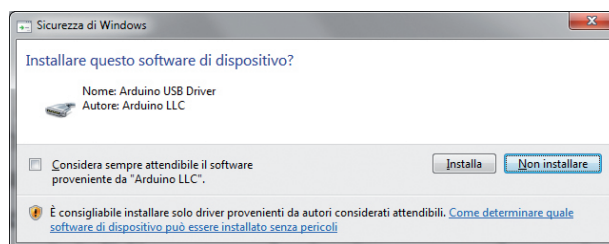
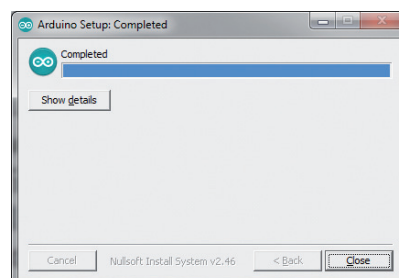


Fig. 12



modalità Publisher o Subscriber per mezzo di un server “coordinatore” pubblico o privato. Infine, c'è la possibilità di interfacciare le API di Temboo che permettono di comunicare con una grande quantità di servizi “web” come i social network, gli educational, il Khan Academy, i servizi di trasporto, quelli medicali ed infiniti altri. In questo articolo, come primo assaggio dell'utilizzo di Arduino Yún, vi proponiamo un esempio di uso delle API Temboo con la libreria “Bridge” finalizzato a spedire dei Tweet a fronte di eventi provenienti dall'I/O di Arduino. Per far questo, dobbiamo preparare uno sketch da compilare e trasferire sulla memoria flash della sezione Arduino di Yún. Ne approfittiamo per introdurre l'utilizzo dell'IDE con la Yún collegata via rete Ethernet.

L'IDE DI ARDUINO

Abbiamo già anticipato che possiamo scaricare l'IDE di sviluppo per Arduino Yún dal sito di Arduino. È necessario scaricare i due file *arduino-1.5.4-r2-windows.exe* e *arduino-1.5.4-r2-windows.zip*, o, se esistono, le successive release. Il primo file permette il caricamento dei driver USB necessari a riconoscere la scheda Yún e ad installare il secondo pacchetto. Quindi eseguiamo il primo file, accettiamo i termini di licenza, selezioniamo tutte le opzioni come in Fig. 10. Alle richieste delle

schermate successive rispondiamo con “Install” (Fig. 11) ed alla fine confermiamo con “Close” (Fig. 12). Terminata l'installazione lanciamo l'IDE dall'icona sul desktop o dal menu “Start”, allorché si aprirà la consueta interfaccia dell'IDE, contenente alcune novità. In Fig. 13 vediamo la configurazione della porta di comunicazione tra IDE e scheda Yún; in realtà ci viene presentato l'indirizzo IP della scheda: lo selezioniamo ed attiviamo il canale di comunicazione, senza necessità di collegare cavi USB e installare driver. Teniamo presente che questo metodo funziona, ovviamente, anche se la scheda è collegata in modalità wireless, il che costituisce una grossa innovazione rispetto al passato, permettendo di programmare e configurare

la Yún anche se questa si trova in un luogo recondito o montata sopra un dispositivo mobile, come ad esempio un robot. Fatta la scelta della porta di comunicazione, non ci resta che configurare il tipo di dispositivo -ovviamente Arduino Yún-, tramite il percorso “Strumenti”>>“Board”. Il resto corrisponde a ciò che abbiamo sempre fatto con le altre schede Arduino.

L'INTERFACCIA TWITTER

L'applicazione che vi proponiamo è tratta da uno degli esempi forniti con l'IDE e permette, utilizzando le librerie ed il servizio Temboo, di inviare dei messaggi

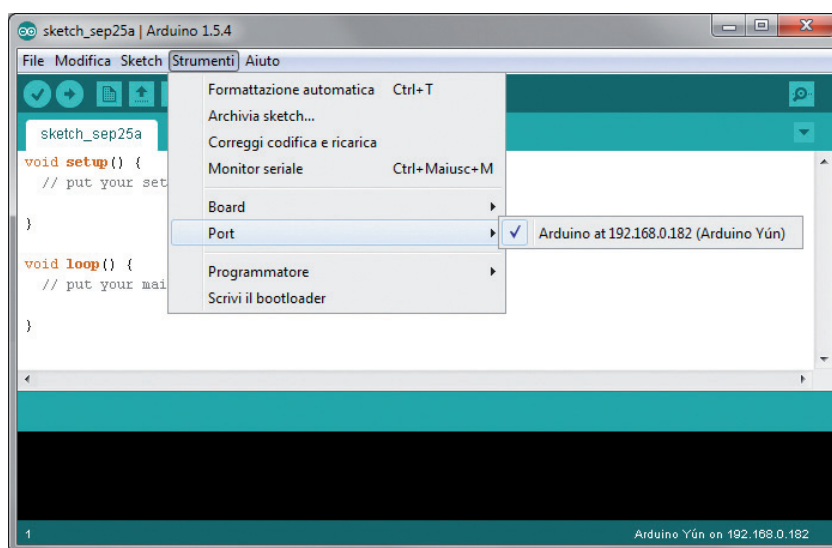


Fig. 13

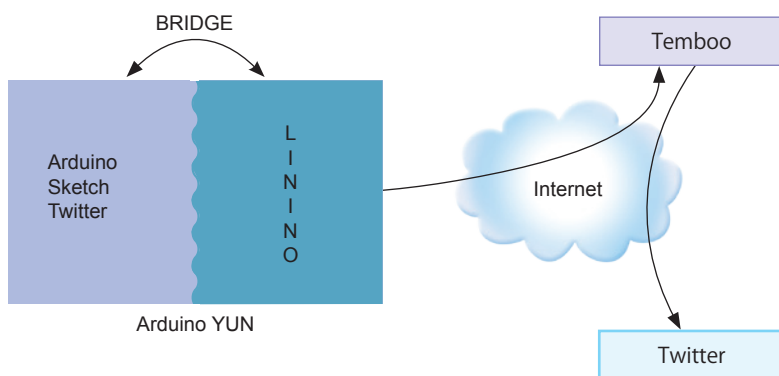


Fig. 14

Twitter al verificarsi di un evento locale, che nel nostro caso è la pressione di un pulsante collegato al pin 5 di Arduino Yún. Per ottenere questo risultato dobbiamo realizzare una catena di comunicazione come quella visibile in **Fig. 14**. Il servizio Temboo offre la possibilità di collegarsi per inviare e ricevere messaggi da e per diverse tipologie di servizi disponibili in rete tra i quali i social network, servizi di pagamento come PayPal, servizi meteo e medici, di formazioni e molti altri, per un totale di oltre cento classi di API web disponibili. Semplificando molto ci si collega a Temboo, si istanziano oggetti utilizzando le classi messe a disposizione da Temboo stesso e, mediante questo meccanismo, si chiede a Temboo di inviare o di reperire per noi informazioni da o verso un certo servizio.

La registrazione gratuita permette di inviare/ricevere 1.000 messaggi al mese; per gestirne di più è necessario sottoscrivere abbonamenti a pagamento.

Il programma che “gira” sulla sezione Arduino predispone il collegamento verso il servizio Temboo, che include sia le credenziali di accesso al servizio stesso, che all’applicazione Twitter cui si vuole inviare il messaggio. Per quanto riguarda Temboo, nella sezione “setup” viene istanziato l’oggetto necessario al collegamento con Twitter. Al

termine del setup, il programma entra nel loop principale, dove, ad ogni ciclo, viene verificato se il pulsante è stato premuto o meno. Se il pulsante non viene premuto, il programma si predispone per un nuovo ciclo; in caso contrario, viene iniziato il processo di invio del messaggio Twitter, che comporta la compilazione dei dati dell’oggetto con l’impostazione del messaggio stesso, delle credenziali di accesso a Temboo e a Twitter, ed infine con l’invocazione del metodo “run” dell’oggetto che procede all’esecuzione del processo vero e proprio di invio del messaggio. A questo punto il programma si sospende fino al ricevimento dello stato di ritorno dell’elaborazione, con eventuale segnalazione delle anomalie. Per tenere sotto controllo il comportamento del programma abbiamo aggiunto due LED: uno verde che resta acceso, dopo la pressione del pulsante, per tutto il tempo necessario all’invio del Tweet, ed uno rosso che si accende a lenta intermittenza, in caso si verifichi qualche errore, continuando in questo stato fino a quando l’errore si risolve. Per realizzare il semplice circuito della nostra applicazione, abbiamo utilizzato lo shield sperimentale per Arduino con la breadboard

millefori a inserimento facilitato. Entrambe le board sono reperibili presso Futura Elettronica, con i codici 7300-PCBPROTOV10 (più 1 7300-STRIP6 e 3 7300-STRIP8) e 7300-BBMINIW.

Nella **Fig. 15** vedete il prototipo, il cui schema di cablaggio è illustrato nella **Fig. 16**.

CONFIGURAZIONE TWITTER E TEMBOO

Per realizzare questo esempio bisogna disporre di un account Twitter, al quale inviare i Tweet, e configurare un’apposita applicazione della quale utilizzeremo le credenziali di accesso all’interno del nostro sketch. È inoltre necessario registrarsi a Temboo per ottenere le credenziali necessarie a utilizzare il servizio e configurare il nostro sketch anche con queste. Iniziamo con Twitter: se non abbiamo un account Twitter creiamone uno, seguendo le istruzioni indicate dalla procedura di registrazione. Una volta ottenuta la password (via e-mail) andiamo all’indirizzo <https://dev.twitter.com/apps/new>, dove, se già non siamo loggati, ci verrà richiesto di inserire userid e password di Twitter (**Fig. 17**).

Nella pagina che ci appare (**Fig. 18**) creiamo un’applicazione ri-

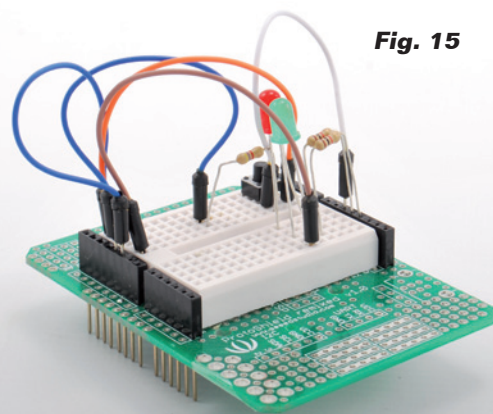


Fig. 15

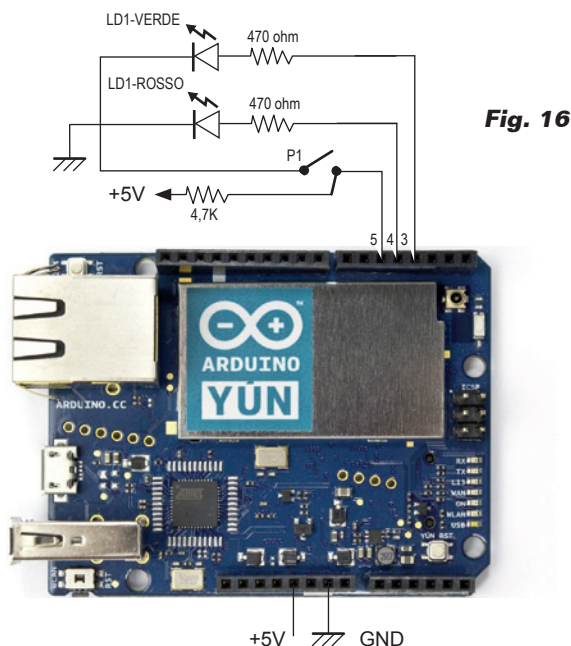


Fig. 16

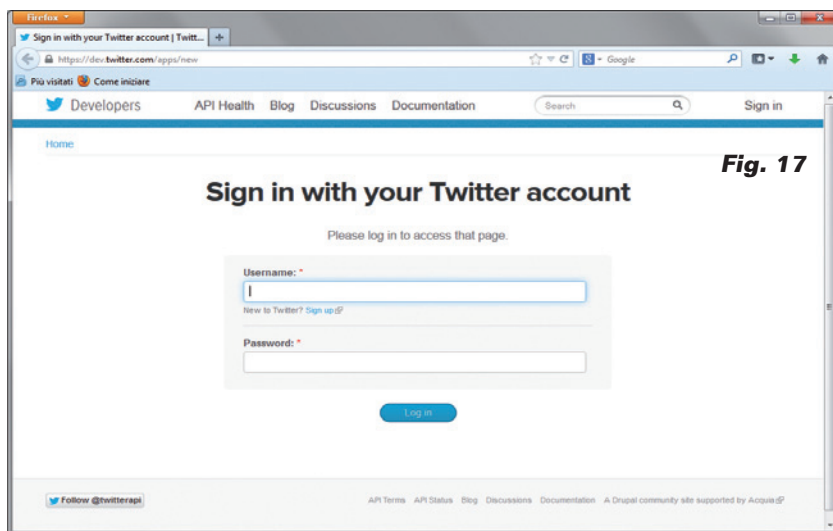


Fig. 17

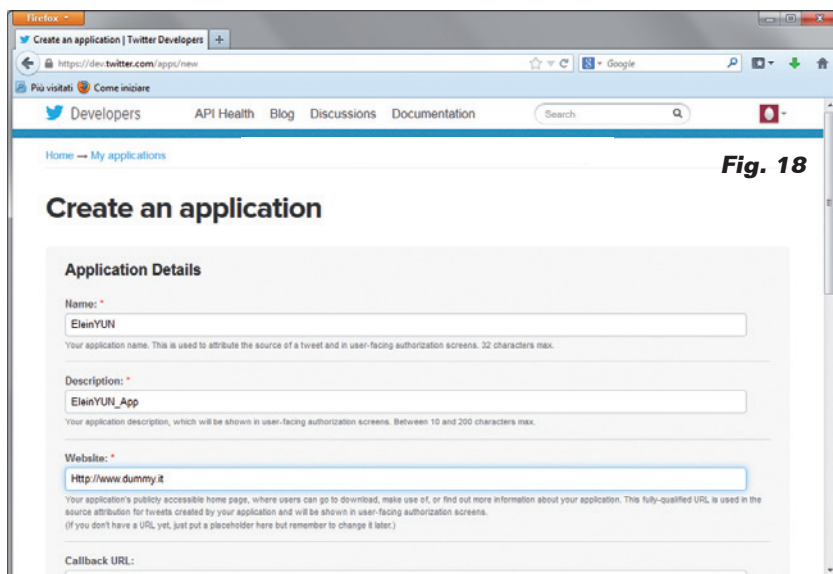


Fig. 18

empiendo i campi obbligatori. Per quanto riguarda il campo "website" possiamo fornire un sito di fantasia, purché sintatticamente corretto: tipo www.nessunsito.it. Compiliamo il campo captcha e confermiamo facendo clic sul pulsante blu in fondo alla pagina. Nella pagina "settings" di Fig. 19, nella sezione "Application Type" selezioniamo l'opzione "Read and Write". Confermiamo anche questa pagina e passiamo alla successiva, visibile in Fig. 20, dove ci viene fornita la prima serie di codici che dovremo inserire nel nostro sketch. Si tratta, in particolare, dei campi "Consumer key" e "Consumer secret". Nella sezione "Your access token" premiamo il pulsante blu per farci creare la coppia di token di accesso. Di questa sezione ci serviranno i campi "Access token" e "Access token secret". Teniamo aperta questa pagina, apriamo una nuova scheda nel browser e andiamo all'indirizzo di registrazione del servizio Temboo (Fig. 21) <https://temboo.com/signup>; inseriamo la nostra e-mail nell'unico campo disponibile ed attendiamo la conferma da parte di Temboo. Una volta ricevuta la mail clicchiamo sul link fornito e, nella pagina di conferma della registrazione, scegliamo un utente ed una password che utilizzeremo per il login nella pagina (Fig. 22) all'indirizzo <https://temboo.com/login>. Inseriamo tutti i campi richiesti ed entriamo nella pagina del nostro account, visibile in Fig. 23; qui selezioniamo la voce di menu "APPLICATION KEYS" e, una volta entrati, clicchiamo sulla voce "(show)" accanto alla fila di asterischi sotto la dicitura "KEY". In questo modo otteniamo gli ultimi dati necessari a personalizzare il nostro sketch: in pratica, l'equivalente di userid e password necessari a usare le

librerie delle classi e le chiamate API messe a disposizione di Temboo stesso. Di fatto, una sorta di servizio a “consumo” dove un certo numero di utilizzi al mese è gratuito, mentre oltre bisogna sottoscrivere un abbonamento a pagamento.

LO SKETCH ARDUINO

Passiamo ora a descrivere il funzionamento dello sketch per Arduino (**Listato 1**) la cui dovezia di commenti non dovrebbe lasciare dubbi sul funzionamento. Copiamo il codice nell’IDE, tenendo conto che è sempre possibile scaricarlo dal sito *www.elettronica.in.it*. Ora copiamo e incolliamo i campi delle credenziali dalle pagine web di Twitter, che abbiamo lasciato aperte, e Temboo, nei relativi campi dello sketch. Suggeriamo il metodo del copia e incolla perché la complessità delle chiavi, in caso di copia manuale, porta pressoché con certezza a commettere errori difficilmente individuabili. Noi abbiamo scelto la modalità di upload del codice su Arduino mediante ethernet, rinunciando a usare l’USB. Configuriamo correttamente “Board” e “Port” e facciamo clic sul pulsante “Carica”: prima verrà eseguita la compilazione del codice e successivamente avrà inizio il caricamento del codice compilato sulle scheda Yún. Tutto qui. Al termine premiamo il pulsante sulla breadboard: il LED verde resterà illuminato per tutto il tempo necessario alla spedizione del Tweet al nostro account; al termine si spegnerà e sarà possibile premere nuovamente il pulsante. Se si verifica un errore si accende anche il LED rosso e viene ritentato l’invio. Per vedere cosa accade, possiamo provocare l’errore staccando il cavo di rete dopo aver premuto il pulsante: dopo un po’ riattacchiamo il cavo

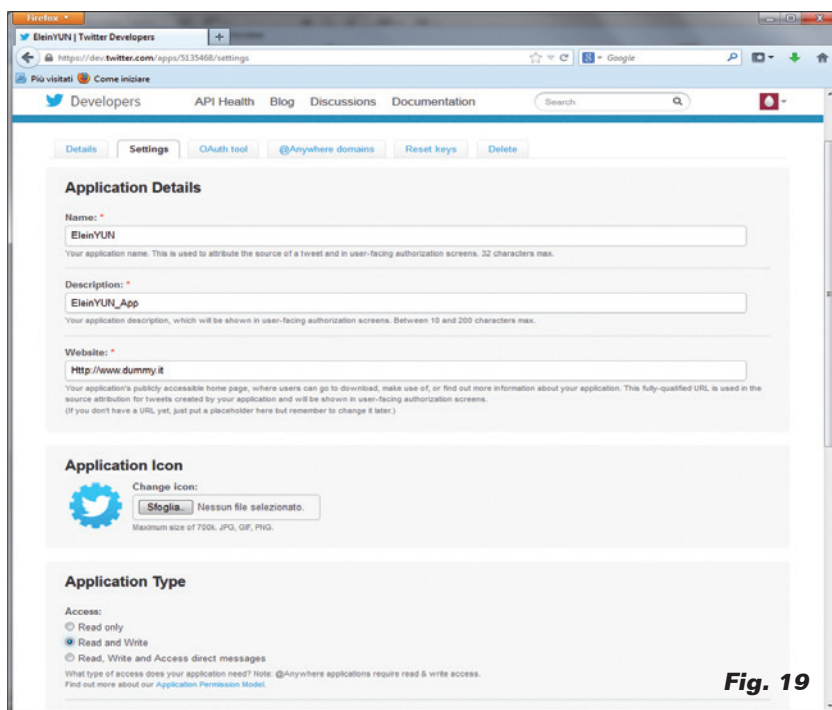


Fig. 19

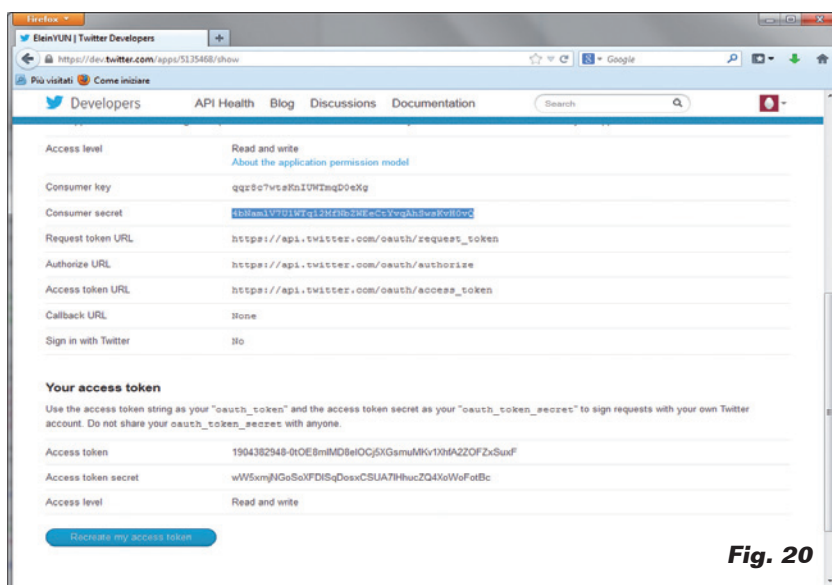


Fig. 20

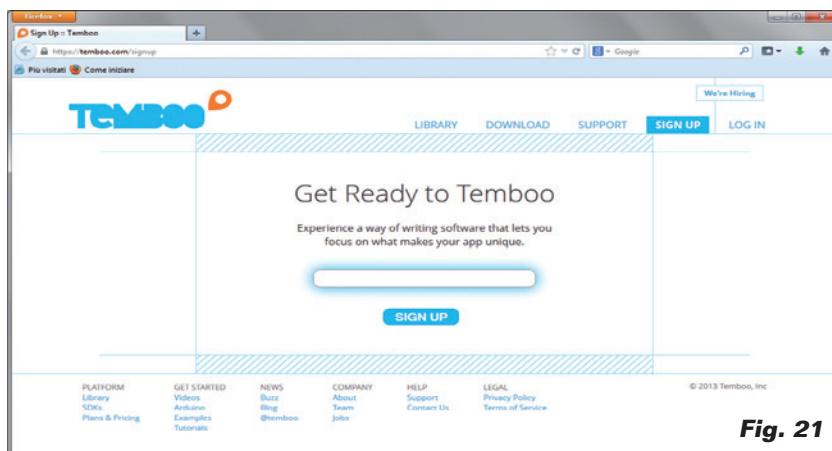


Fig. 21

Listato 1

```

/*
  Invia un Tweet
*/

#include <Bridge.h>
#include <Temboo.h>

/** SOSTITUIRE LE CREDENZIALI */

const String TWITTER_ACCESS_TOKEN = "1904382948-0...v1XhfA2ZOFZxSuxF";
const String TWITTER_ACCESS_TOKEN_SECRET = "wW5xmjNGoSoX.....7IHhucZQ4XoWoFotBc";
const String TWITTER_CONSUMER_KEY = "qqr8c7wt.....qD0eXg";
const String TWITTER_CONSUMER_SECRET = "4bNamlV7U1.....eCtYvqAhSwsKvH0vQ";

const String TEMBOO_ACCOUNT = "dev....cain"; // your Temboo account name
const String TEMBOO_APP_KEY_NAME = "myF....pp"; // your Temboo app key name
const String TEMBOO_APP_KEY = "1b.....32-9"; // your Temboo app key

int numRuns = 0; // Contatori per gestire l'invio in caso di errori
int maxRuns = 1;

int led_rosso = 4; // LED errore in corso
int led_verde = 3; // LED invio TWEET in corso
int buttonPin = 5; // Pulsante invio TWEET
int buttonState = 1;

void setup() {
  pinMode(led_rosso, OUTPUT);
  pinMode(led_verde, OUTPUT);
  pinMode(buttonPin, INPUT);
  Bridge.begin(); // Inizializza la comunicazione con LINUX via seriale condivisa
}

void loop()
{
  buttonState = digitalRead(buttonPin);
  if (buttonState == LOW) { // Se viene premuto il pulsante

    while (numRuns < maxRuns) { // tenta gli invii fino a buon fine

      // Testo del messaggio TWEET
      String tweetText("Qualcuno ha premuto il pulsante - " + String(millis()) );

      digitalWrite(led_verde, HIGH);

      TembooChoreo StatusesUpdateChoreo; // Istanza oggetto per invio TWEET

      // Invoca il client Temboo
      StatusesUpdateChoreo.begin();

      // Imposta le credenziali per accedere a Temboo
      StatusesUpdateChoreo.setAccountName(TEMBOO_ACCOUNT);
      StatusesUpdateChoreo.setAppKeyName(TEMBOO_APP_KEY_NAME);
      StatusesUpdateChoreo.setAppKey(TEMBOO_APP_KEY);

      // Imposta la libreria di API da utilizzare (Twitter > Tweets > StatusesUpdate)
      StatusesUpdateChoreo.setChoreo("/Library/Twitter/Tweets/StatusesUpdate");

      // Imposta le credenziali di accesso a Twitter
      StatusesUpdateChoreo.addInput("AccessToken", TWITTER_ACCESS_TOKEN);
      StatusesUpdateChoreo.addInput("AccessTokenSecret", TWITTER_ACCESS_TOKEN_SECRET);
      StatusesUpdateChoreo.addInput("ConsumerKey", TWITTER_CONSUMER_KEY);
      StatusesUpdateChoreo.addInput("ConsumerSecret", TWITTER_CONSUMER_SECRET);

      // riferimento al messaggio da inviare
      StatusesUpdateChoreo.addInput("StatusUpdate", tweetText);

      // Richiede esecuzione del processo di invio
      unsigned int returnCode = StatusesUpdateChoreo.run();

      // Il codice di ritorno (0) significa processo andato a buon fine
      if (returnCode == 0) {
        numRuns = numRuns + 1;
        digitalWrite(led_verde, LOW);
      } else {
        // In caso di errore non si esce dal loop while e si ritenta invio
        while (!StatusesUpdateChoreo.available()) {
          char c = StatusesUpdateChoreo.read();
        }
        digitalWrite(led_rosso, HIGH);
        delay(5000);
        digitalWrite(led_rosso, LOW);
        delay(5000);
      }
      StatusesUpdateChoreo.close();
    }
    numRuns = 0; // Tutto ok in attesa di una nuova pressione del pulsante
  }
}

```

e vedremo che, alla fine, il nostro Tweet giungerà a destinazione (Fig. 24).

CONCLUSIONI

In questa prima presa visione della scheda Arduino Yún abbiamo toccato solo superficialmente le sue potenzialità, soprattutto sul piano didattico. Yún si presta infatti come banco di prova, semplice e poco costoso per approfondire le tematiche relative al mondo GNU/Linux ed alle reti, oltre a quelle proprie del mondo Arduino, come già possibile con i suoi predecessori. Il sistema operativo Linino, con il suo cuore OpenWRT, si presta a sperimentare a poco costo la configurazione e gestione di reti dal punto di vista dell'amministrazione e della sicurezza, in ambiente sia cablato che wireless. Con una manciata di schede Yún si possono realizzare configurazioni di reti anche complesse; e ciò, soprattutto nei tempi di ristrettezze economiche in cui viviamo, dovrebbe interessare decisamente le Scuole Superiori e le Università. Stessa cosa dicasi per la sperimentazione di applicazioni embedded, tipicamente composte da sezioni finalizzate al campionamento ed elaborazione dati da e verso sensori ed I/O analogici e digitali e da sezioni di coordinamento e interfacciamento umano che si avvalgono di soluzioni applicative web based. Il tutto portato al livello di semplificazione, proprio della filosofia "Arduino", che permette anche e soprattutto a chi è alle prime armi di trovare soddisfazione immediata e spinta ad approfondire tecnologie e discipline che costelleranno sempre di più il nostro futuro. Non mancheremo di approfondire le possibilità di Yún e di proporvele nei prossimi articoli, insieme alle altre novità che si stanno profilando. ■

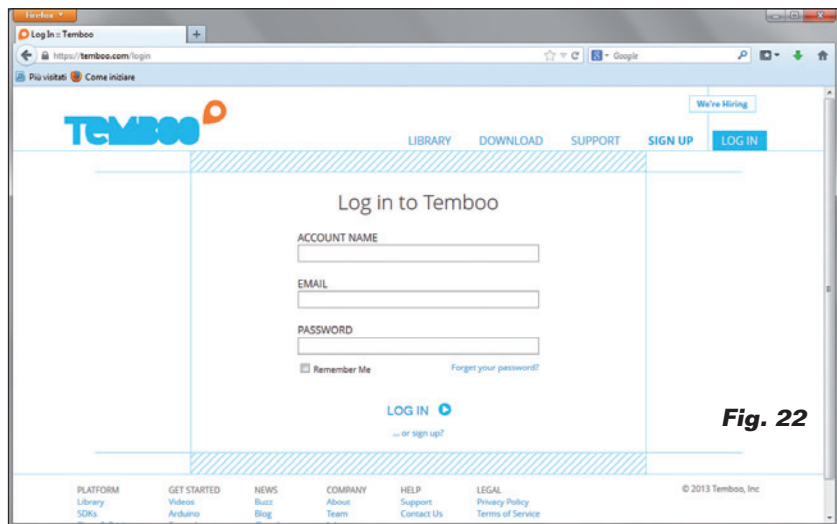


Fig. 22

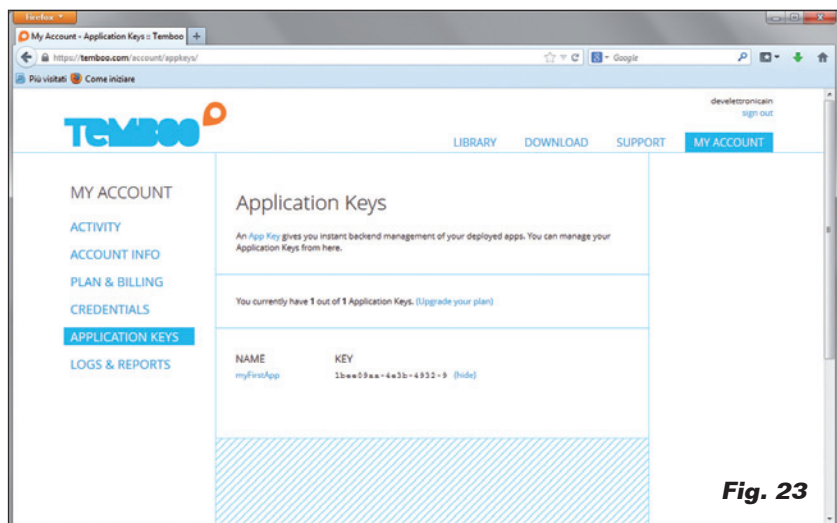


Fig. 23

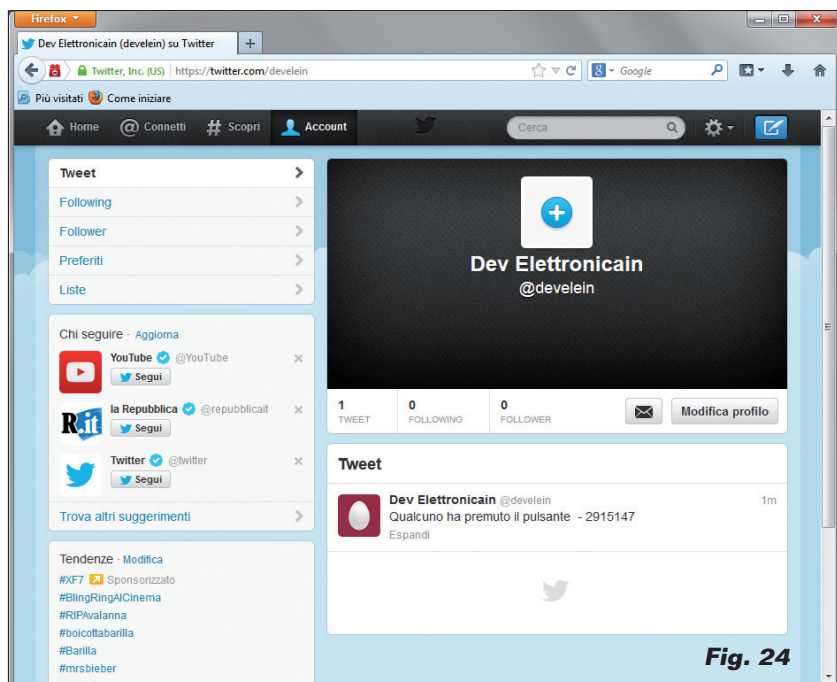
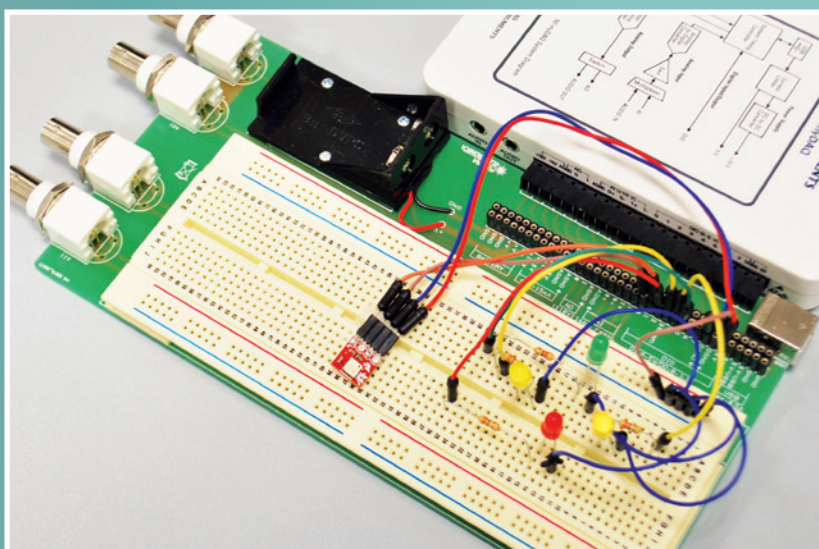


Fig. 24

myKIT

UTILIZZO DI UN SENSORE TILT PER LA DETERMINAZIONE DELLA DIREZIONE DEL MOVIMENTO DI UN OGGETTO

di DAVIDE SCULLINO



Per realizzare un programma in grado di indicare la direzione di un oggetto o di un veicolo, è possibile utilizzare sensori di vario genere; in questo caso utilizziamo un semplice ed economico sensore di tilt. Il circuito realizzato con questo componente

è connesso ad una scheda di acquisizione myDAQ ed il programma è realizzato col software LabVIEW di National Instruments. I dati sono acquisiti tramite DAQ Assistant, che viene installato in LabVIEW con il driver NI DAQmx. Utilizzare un sensore

Utilizzando un sensore di tilt, realizziamo un programma in LabVIEW in grado di indicarci la direzione del movimento dell'oggetto o del veicolo solidale col sensore.



Fig. 1 - Sensore di Tilt RPI-1031 - 4 direzioni.

di tilt RPI-1031 - 4 direzioni per determinare la direzione del movimento dell'oggetto sul quale è installato è piuttosto semplice; nel nostro caso sfruttiamo gli ingressi digitali della scheda myDAQ e alcuni LED montati sulla breadboard per segnalare la direzione.

Il sensore impiegato è in grado di rilevare la direzione del movimento in quattro direzioni: avanti, indietro, a sinistra, a destra. Il circuito integrato RPI-1031 è già saldato su una breakout board che ne rende più facile l'utilizzo dal momento che i piedini dell'integrato sono riportati all'esterno su pad con spaziatura standard da 2,54 mm. Sono inoltre inclusi i componenti necessari per il funzionamento.

Il circuito può essere alimentato sia a 3,3 che a 5V. Questo sensore è ideale nei casi in cui bisogna soltanto rilevare la direzione del movimento e non è necessario l'utilizzo di un ben più costoso accelerometro: ad esempio, nelle cornici digitali per cambiare l'orientamento delle fotografie in base alla rotazione della cornice oppure nei telefoni cellulari di ultima generazione.

Il funzionamento è molto semplice: all'interno del sensore ci sono due fototransistor, un LED e una sferetta. Ruotando il dispositivo, la sferetta chiuderà il passaggio di luce dell'uno, dell'altro o di entrambi i fototransistor, segnalando quattro diverse combinazioni 1/0 sui pin dei contatti (**Fig. 2**).

ISTRUZIONI PER IL CABLAGGIO

Il sensore di tilt ha quattro pin di collegamento, perciò dovrete collegare il pin VCC all'ingresso +5V del dispositivo myDAQ, il pin GND al ground DGND, i pin S1 e S2 alle uscite digitali DIO0 e DIO1 rispettivamente. Inoltre, dopo aver disposto sulla breadboard i quattro LED a formare una croce (come in **Fig. 3**) dovrete collegare l'anodo (+) del LED



Fig. 4 - Pannello Frontale LabVIEW.

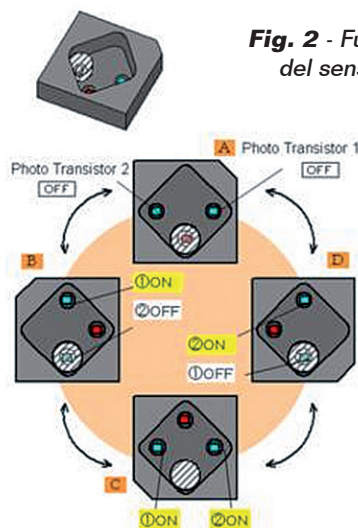
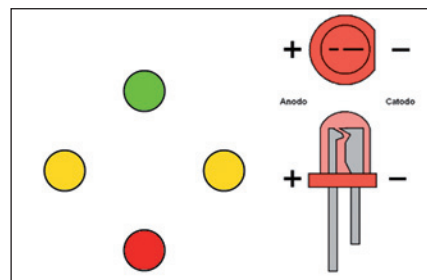


Fig. 2 - Funzionamento del sensore di tilt.

Fig. 3
Disposizione dei LED sulla breadboard.



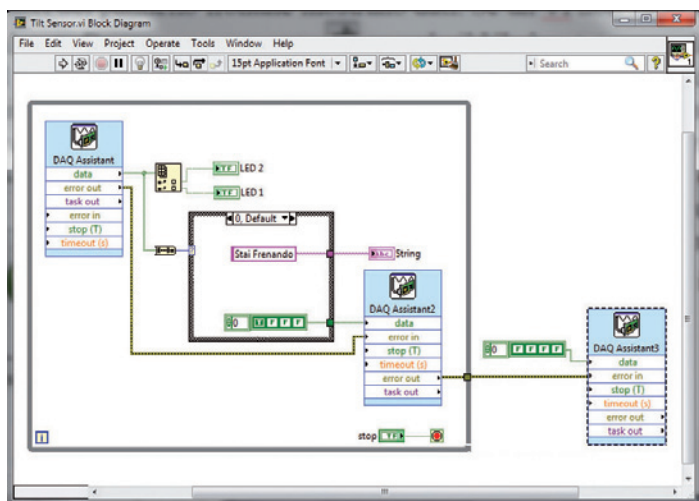
rosso all'uscita digitale DIO2, gli anodi dei due LED gialli di destra e di sinistra rispettivamente alle uscite digitali DIO3 e DIO4, e l'anodo del LED verde all'uscita DIO5 del dispositivo.

Per finire, dovete collegare il catodo(-) di ciascun LED all'ingresso DGND della scheda myDAQ (dovreste riconoscere il catodo dei LED perché ha un collegamento più corto di quello dell'anodo e perché situato dalla parte del LED in cui la base risulta piana e non circolare).

Nota: Per preservare il sensore è consigliabile mantenere scollegato il cavo di alimentazione dello stesso quando non lo si utilizza perché potrebbe scaldarsi eccessivamente.

PANNELLO FRONTALE LABVIEW

Supponiamo che il sensore di tilt sia installato su una moto. Il Pannello Frontale che dovrete implementare dovrà dirvi tramite una stringa e due LED sferici se state accelerando, frenando oppure se



state girando a destra o a sinistra.

I due LED sferici sul Pannello Frontale si illumineranno singolarmente in base alla direzione in cui state procedendo, oppure si accenderanno o si spegneranno entrambi se state accelerando o frenando rispettivamente. Inoltre i LED montati sulla breadboard vi forniranno un'ulteriore indicazione su questi movimenti (non saranno però visibili sul Pannello Frontale).

CODIFICA DELLA STRATEGIA

Come si vede dal diagramma a blocchi (Fig. 5), il DAQ Assistant acquisisce i valori digitali provenienti dal sensore. Essendo le due linee digitali, le possibili combinazioni di valori zero/uno saranno 00, 01, 10, 11. In base a quale di queste quattro combinazioni acquisirete, si accenderanno i due LED del Pannello Frontale e verrà visualizzata una stringa con un messaggio riguardante il movimento effettuato. Con lo stesso metodo usato per decidere quale stringa sarà visualizzata sul Pannello Frontale verrà inviato a un secondo DAQ Assistant il comando relativo a quale LED della breadboard accendere.

COME FUNZIONA

Come accennato, il primo DAQ Assistant situato sulla sinistra acquisisce i valori digitali provenienti dalle linee DIO0 e DIO1 della scheda myDAQ. All'uscita

“data” di questo blocco avrete quindi un array di combinazioni di valori booleani (00, 01, 10, 11). Questo array viene inviato a un primo blocco che permette di scomporlo in singoli elementi (zero o uno, ma non una combinazione di entrambi) che a loro volta vengono inviati ai due indicatori LED posti sul Pannello Frontale. Lo stesso array contenente le combinazioni booleane viene poi convertito in un numero intero decimale (00=0, 01=1, 10=2, 11=3) e inviato al *selector terminal* (il quadratino blu con il punto di domanda) della *Case Structure*.

Una struttura *Case* ha due o più sottodiagrammi, o casi (condizioni). È visibile solo un sottodiagramma alla volta e la struttura esegue un solo caso alla volta. Il valore in ingresso al *selector terminal* determina quale sottodiagramma deve essere eseguito. L'etichetta del case selector nella parte superiore della struttura *Case* contiene al centro il nome del *selector value* corrispondente al caso e ai lati le frecce di decremento o incremento. Cliccate sulle frecce di decremento e incremento per scorrere tra i casi disponibili. Potete anche cliccare sulla freccia verso il basso accanto al nome del caso e, dal menu a tendina, selezionare il caso desiderato. È possibile collegare al *selector terminal* un numero intero, un valore booleano, una stringa o un valore di tipo *enumerated*. Se il tipo di dati del *selector terminal* è booleano, la struttura ha un caso TRUE e un caso FALSE. Se il *selector terminal* è un intero, come nel nostro caso, la struttura può

avere un numero di condizioni qualsiasi. Inoltre uno dei casi deve essere dichiarato di "Default" in modo che se la struttura riceve in ingresso al *selector terminal* un valore non presente tra i possibili casi, il software esegue il caso di "Default".

Siccome le possibili combinazioni di valori booleani che acquisite sono quattro, il numero di casi della struttura è quattro. Inoltre, le etichette dei quattro casi sono 0, 1, 2, 3, che corrispondono alle combinazioni di valori booleani convertiti in numeri decimali. All'interno di ogni caso si trova la rispettiva stringa da visualizzare, sul Pannello Frontale, associata alla direzione del movimento rilevato; c'è anche un array di quattro costanti di valori booleani



Tabella 1

Tipo di collegamento	Scalare	Array 1D	Array 2D	Colore
Numerico				Arancione (virgola mobile), Blu (intero)
Booleano				Verde
Stringa				Rosa

che serve a decidere quale LED della breadboard accendere. Questo array di costanti booleane viene inviato al secondo DAQ Assistant che si occupa di controllare, tramite le linee digitali DIO2, DIO3, DIO4, DIO5, i quattro LED posti sulla breadboard. Tutto il codice spiegato finora deve poi essere inserito in un ciclo *While* per permettere l'acquisizione e la visualizzazione continua di dati. Il codice continua a funzionare fino a che non viene premuto il pulsante Stop sul Pannello Frontale. All'esterno del ciclo *While* c'è l'ultimo DAQ Assistant collegato a un array di quattro costanti booleane False, configurato allo stesso modo del secondo DAQ Assistant utilizzato; esso fa in modo che, una volta terminata l'esecuzione del ciclo *While*, tutti i LED presenti sulla breadboard si spengano. Questo perché il secondo DAQ Assistant che gestisce l'accensione dei quattro LED mantiene acceso l'ultimo LED utilizzato anche dopo aver terminato l'esecuzione. Per ovviare al problema del parallelismo (non si sa se il software esegua prima il codice all'interno del ciclo *while* o il frammento di codice al suo esterno) dovete collegare tra loro gli error out agli error in dei tre DAQ assistant. Questo garantisce che il terzo DAQ Assistant vada in esecuzione solo al termine del ciclo *While*, ovvero quando saranno disponibili tutti i dati al suo ingresso.

COSTRUZIONE DEL PANNELLO FRONTALE

Occupiamoci ora della costruzione del Pannello Frontale. Per prima cosa lanciate LabVIEW, aprite un nuovo VI e cliccate all'interno del Pannello Frontale fino a veder comparire la palette *Controls*, dove sono elencati tutti i tipi di controllo e di indicatore, ma anche decorazioni, tabelle, ecc. Per comodità, potete bloccare la palette cliccando sulla "puntina" in alto alla sua sinistra; in tal modo, quando vi servirà sarà subito disponibile. Adesso, nella palette *Controls* alla voce *Modern* selezionate *Boolean* e inserite nel Pannello Frontale un

pulsante di Stop e due Round LED. Ora, sempre alla voce *Modern*, selezionate *String & Path* e inserite una *String Indicator*. Con lo strumento *Positioning* disponete e ingrandite il tutto, come visibile in Fig. 6. Per visualizzare un altro stile o un'altra dimensione del testo all'interno della *String Indicator*, con lo strumento *Labeling* cliccate all'interno dell'indicatore come se volesse scrivervi e nella barra del menu in alto cliccate su *Text Settings* (è il riquadro che riporta la dimensione e il font impostati: 15pt Application Font) e selezionate ciò che preferite. Potete cambiare le impostazioni di scrittura anche per scrivere le etichette sul Pannello Frontale o sul diagramma a blocchi.

COSTRUZIONE DEL DIAGRAMMA A BLOCCHI

Ora implementerete il Diagramma a Blocchi. Se non lo visualizzate, potete farlo selezionando *Window»Show Block Diagram* o digitando <Ctrl+E>. Gli oggetti del Pannello Frontale appaiono come terminali sul diagramma a blocchi. I terminali sono porte d'ingresso e di uscita che scambiano informazioni tra il Pannello Frontale e il diagramma a blocchi.

Qui dovreste già visualizzare i controlli numerici e il pulsante Stop con le stesse etichette che avete messo sul Pannello Frontale. Inoltre dovreste visualizzare i terminali, ad esempio quello del pulsante Stop, con questa icona: . È tuttavia preferibile visualizzarlo in questo modo: . Ciò, per ridurre lo spazio che occupa sul diagramma a blocchi. Per fare questo selezionate *Tools»Options* e, sulla sinistra della finestra che si aprirà, selezionate *Block Diagram* e, alla voce *General*, disabilitate *Place front panel terminals as icons*.

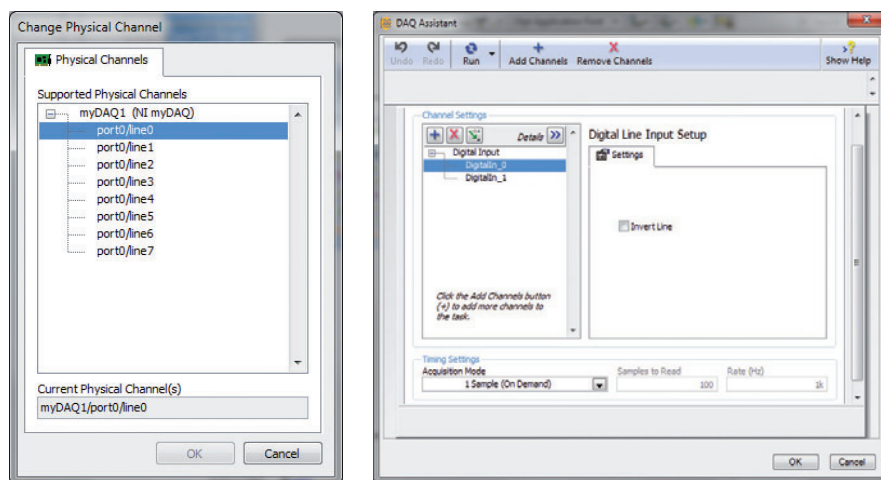


Fig. 7 - Configurazione del primo DAQ Assistant.

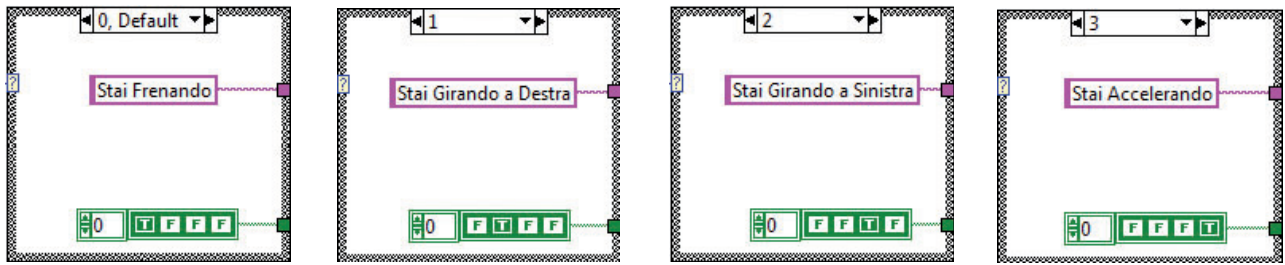


Fig. 8 - I quattro casi della "Case Structure".

Ora, prima di iniziare ad implementare il diagramma a blocchi, osservate che i rettangolini che rappresentano i controlli possono essere impostati sia come indicatori (cioè ricevono dati) sia come controlli (cioè producono dati). Nel diagramma a blocchi, il controllo si distingue dall'indicatore per due particolari: il primo è una freccia sul terminale che indica la direzione del flusso dei dati (i controlli hanno frecce che mostrano il dato partire dal terminale, mentre negli indicatori la freccia che mostra il dato entra nel terminale); il secondo è il bordo intorno al terminale (i controlli hanno un bordo più spesso di quello degli indicatori). Per passare da indicatore a controllo dovete cliccare col tasto destro del mouse sopra ogni rettangolino che rappresenta l'oggetto e impartire il comando *Change to Control*; questa operazione può essere effettuata anche dal pannello frontale. Notate che il pulsante Stop è già impostato come controllo. Prestate attenzione ai collegamenti che effettuerete sul diagramma a blocchi: essi permettono di trasferire dati tra oggetti del diagramma a blocchi; ogni collegamento parte da una sorgente di dati e la collega ad uno o più VI e funzioni che leggono i dati stessi. I collegamenti hanno differenti colori, stili e spessori, in funzione del tipo di dati. Un collegamento spezzato  appare come una linea nera tratteggiata con una X rossa in mezzo. I collegamenti possono rompersi per svariati motivi: ad esempio se si tenta di collegare due oggetti con tipi di dati tra loro incompatibili. La **Tabella 1** illustra i tipi di collegamento più comuni.

I collegamenti possono essere realizzati solamente tra ingressi e uscite che veicolino dati compatibili tra loro. Nel diagramma a blocchi, facendo clic con il pulsante destro del mouse compare la palette *Functions*, che contiene i VI, le funzioni e le costanti da usare per creare il diagramma a blocchi. Se volete bloccare la palette *Functions* sullo schermo, fate clic sulla puntina che si trova in alto a sinistra.

Ora potete iniziare la costruzione del diagramma a blocchi. Per prima cosa dovete inserire e confi-

gurare il blocco del DAQ Assistant per acquisire il segnale digitale proveniente dal sensore di tilt. Le seguenti istruzioni vi guideranno passo-passo nella configurazione del primo DAQ Assistant.

1. Assicuratevi che myDAQ sia collegato al PC;
2. Premete Ctrl-Spazio per aprire la finestra *Quick Drop* (al primo utilizzo, ciò richiederà circa un minuto);
3. Cercate la voce DAQ Assistant e fate doppio clic su di esso quando appare nella lista;
4. Trascinatelo sul diagramma a blocchi;
5. Quando appare un nuovo riquadro per le attività di configurazione rapida, selezionate:
 - a. Acquire Signals;
 - b. Digital Input;
 - c. Line Input;
 - d. Dev 1 (NI myDAQ) * Nota: Se si dispone di un altro hardware NI installato, il myDAQ non sarà Dev1;
 - e. Port0/line0 e Port0/line1 (per selezionare entrambe tenete premuto <Ctrl>);
 - f. Finish;
6. Lasciare le impostazioni di temporizzazione a:
 - a. 1 Sample (On Demand);
7. Fate clic su OK.

Sulla palette *Functions*, estendete la prima voce *Programming»Array* e selezionate il blocco *Index Array*, quindi con lo strumento *Positioning* estendete tale blocco per avere da esso due uscite disponibili. Sempre alla voce *Programming*, selezionate *Boolean* e inserite il blocco *Boolean Array to Number*. Adesso andate su *Programming»Structures* e inserite una *Case Structure*. Intanto, con gli strumenti *Positioning* e *Wiring* potete disporre e collegare ciò che avete appena inserito (**Fig. 10**). Appena avrete collegato l'uscita del blocco *Boolean Array to Number* al selector terminal della case structure, il selector terminal diventerà blu (colore associato da LabVIEW ai numeri interi). L'etichetta della case structure, invece, cambierà dai possibili valori "true" o "false" (visibili scorrendo

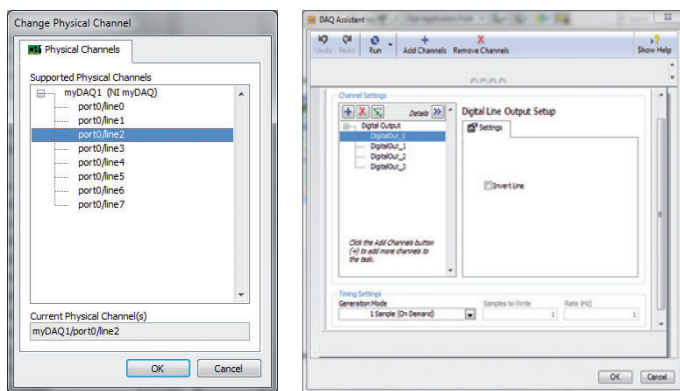


Fig. 9 - Configurazione del secondo DAQ Assistant.

con le frecce poste ai lati dell'etichetta) ai valori interi "0, Default" e "1". Ora utilizzando la freccia di destra posta sull'etichetta portatevi al caso "1" e cliccando sull'etichetta con il tasto destro del mouse selezionate *Add Case After*, facendo comparire il caso "2". Ripetete questa operazione anche sull'etichetta del caso "2" per far comparire anche il caso "3". All'interno di ciascuno di questi quattro casi dovranno esservi una *String Constant* e un array di quattro costanti booleane che servirà per selezionare quale dei quattro LED presenti sulla breadboard accendere. Inserite quindi in ognuno dei quattro casi una *String Constant* che trovate alla voce *Programming»String* e con lo strumento *Labeling* scrivete in ognuna di esse quello che è riportato in **Fig. 8** (oppure delle frasi scelte da voi). Il testo appena inserito nelle quattro stringhe è quello che comparirà nel Pannello Frontale in base alla direzione del movimento del sensore. Per quanto riguarda l'array di costanti, andate al caso "0, Default" e al suo interno inserite un *array constant* situato nella palette *Functions* alla voce *Programming»Array*. Ora trascinate all'interno dell'array vuoto una *False Constant* che trovate alla voce *Programming»Boolean* e con lo strumento *Positioning* portatevi sul bordo destro dell'array ed estendetelo perché contenga quattro costanti. Senza ripetere quest'operazione per ogni caso della struttura, potete selezionare con lo strumento *Positioning* tutto l'array appena creato e, con i comandi <Ctrl-C> e <Ctrl-V> (copia e incolla), copiare e poi incollare in ognuno dei tre casi rimanenti l'array costruito per il caso "0, Default" (prima di incollare l'array all'interno di ogni caso, cliccate con il mouse sul punto dove volete incollarlo e solo dopo premete <Ctrl+V>: ciò evita di incollare l'array in un punto a caso del diagramma a blocchi).

Per associare il valore corretto a ogni costante booleana dell'array di ogni caso, utilizzate lo stru-

mento *Operating* e aiutatevi con la **Fig. 8**. Ricordatevi inoltre di collegare con lo strumento *Wiring* ogni stringa e ogni array al lato destro di tutti i casi della struttura dove comparirà un quadratino (tunnel), altrimenti il VI non potrà andare in esecuzione. Notate che fino a quando non collegherete le stringhe di tutti i casi al lato destro della struttura *case* il quadratino viola (tunnel) risulterà bianco al suo interno.

Ora dovete inserire il secondo DAQ Assistant che si occupa della gestione dei quattro LED montati sulla breadboard. Le seguenti istruzioni vi guideranno passo-passo nella configurazione del secondo DAQ Assistant.

1. Assicurarsi che myDAQ sia collegato al PC;
2. Premete Ctrl-Spazio per aprire la finestra Quick Drop (al primo utilizzo ciò richiederà circa un minuto);
3. Cercate la voce DAQ Assistant e fate doppio clic su di essa quando appare nella lista;
4. Trascinate DAQ Assistant sul diagramma a blocchi;
5. Quando appare un nuovo riquadro per le attività di configurazione rapida, selezionate:
 - a. Generate Signals;
 - b. Digital Output;
 - c. Line Output;
 - d. Dev 1 (NI myDAQ) * Nota: Se si dispone di un altro hardware NI installato, il myDAQ non sarà Dev1;
 - e. Port0/line2, Port0/line3, Port0/line4, Port0/line5, (per selezioni multiple tenere premuto

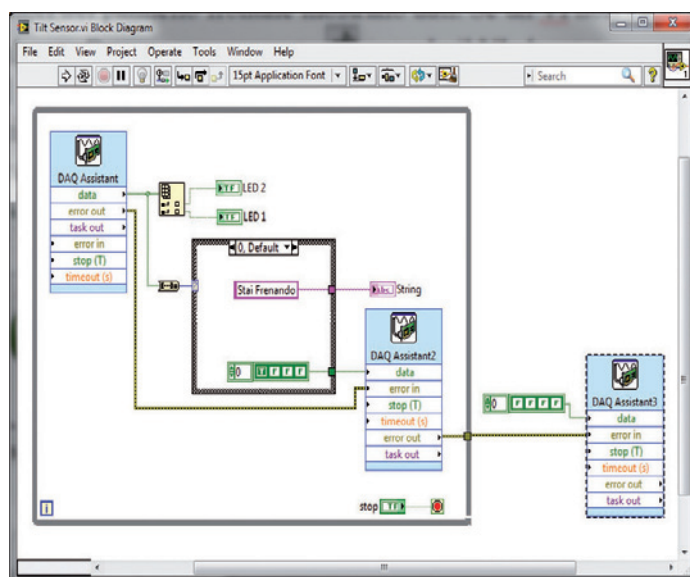


Fig. 10 - Diagramma a Blocchi LabVIEW.

- <Ctrl>);
 f. Finish;
 6. Lasciare le impostazioni di temporizzazione a:
 a. 1 Sample (On Demand);
 7. Fate clic su OK.

Adesso con lo strumento *Positioning* e *Wiring* sistemate e collegate tutti i blocchi inseriti finora come in **Fig. 10**. In particolare collegate il quadratino viola e il quadratino verde situati sul bordo destro della "Case Structure" al terminale "String" e all'ingresso "data" del secondo DAQ Assistant rispettivamente. In seguito, nella palette *Functions* alla voce *Programming»Structures* selezionate *While Loop* e inglobate tutto il codice inserito finora all'interno del ciclo *While*. Collegate con lo strumento *Wiring* il terminale del controllo di Stop al *Loop Condition* del ciclo *while* nell'angolo in basso a destra. Per quanto riguarda il terzo DAQ Assistant, che si occuperà di spegnere tutti i LED posti sulla breadboard una volta premuto il pulsante Stop per arrestare l'esecuzione del VI, non dovrete fare altro che copiare e incollare all'esterno del ciclo *While* il secondo DAQ Assistant creato (utilizzando i comandi illustrati in precedenza oppure trascinando il secondo DAQ Assistant e tenendo premuto il tasto <Ctrl>). Questo perché le funzionalità e quindi le configurazioni di questi due DAQ Assistant sono le stesse. Per finire, dovete collegare in ingresso al terzo DAQ Assistant un array di costanti booleane tutte impostate al valore "false" che serviranno a spegnere i LED sulla breadboard. Per fare ciò copiate e incollate un array di costanti booleane (preso da uno dei casi della struttura case) vicino al terzo DAQ Assistant, con lo strumento *Operating* impostate tutte le costanti booleane al valore "false" e con lo strumento *Wiring* collegatelo all'ingresso *data* del terzo DAQ Assistant. Come seconda e ultima cosa collegate tra loro tutti gli ingressi *error out* e *error in* dei tre DAQ Assistant; questo per evitare la condizione di parallelismo introdotta prima. Se avete agito come spiegato, il vostro diagramma a blocchi si presenterà all'incirca come in **Fig. 10**.

Quando avrete finito di costruire il vostro VI andate su *File»Save as* e scegliete un nome e una destinazione per salvarlo.

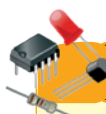


Fig. 11 - VI in esecuzione.

Ora cliccate sul pulsante *Run* nella barra degli strumenti per mettere il VI in esecuzione. Mentre il VI è in esecuzione, l'icona del pulsante *Run* si trasforma nella seguente: . Una volta terminata l'esecuzione, il pulsante *Run* torna al suo stato originale e gli indicatori del Pannello Frontale mantengono i dati visualizzati. Mentre il VI è in esecuzione, appare il pulsante *Abort Execution* (); facendovi clic potete fermare il VI immediatamente. Il pulsante *Abort Execution* va usato soltanto se non riuscite ad arrestare il VI in altro modo.

Notate che se un VI non va in esecuzione, vuol dire che degli errori

non lo rendono eseguibile. Se il VI che state creando o modificando contiene errori, il pulsante *Run* appare spezzato (); in questo caso, anche dopo aver finito i collegamenti nel diagramma a blocchi, il VI non è eseguibile. Generalmente l'errore può consistere in un ingresso richiesto non collegato o in un collegamento spezzato. Per risolvere il problema, fate clic sul pulsante *Run* spezzato in modo da accedere alla finestra di dialogo *Error List*, dove viene elencato e descritto ogni errore riscontrato; qui, facendo doppio clic su un errore andate direttamente alla posizione in cui esso si trova nel diagramma a blocchi.



per il MATERIALE

Questo progetto fa parte del set di esperimenti didattici realizzato da IRS-ElettronicaIn per la board NI myDAQ con il supporto del software LabVIEW. Il set (cod. SETmyKIT, Euro 232,00) comprende tutti i componenti per realizzare i 22 esperimenti, i VI relativi ed una protoboard. È possibile richiedere protoboard aggiuntive (cod. PROTOBOARDmyDAQ) al prezzo di 52,00 Euro. Tutti i prezzi sono comprensivi di IVA e non comprendono la board NI myDAQ né il software LabVIEW. Futura Elettronica distribuisce questi prodotti presso gli utenti privati mentre la distribuzione presso gli Istituti Scolastici è a cura di IRS srl (Via Vigonovese 81 - 35127 PADOVA - tel: +39 049 8705156 - info@irsweb.it).

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)
 Tel: 0331-799775 - Fax: 0331-792287
www.futurashop.it

INTERFACCIA 4 CANALI I/O BLUETOOTH RN-42 ANDROID BASED



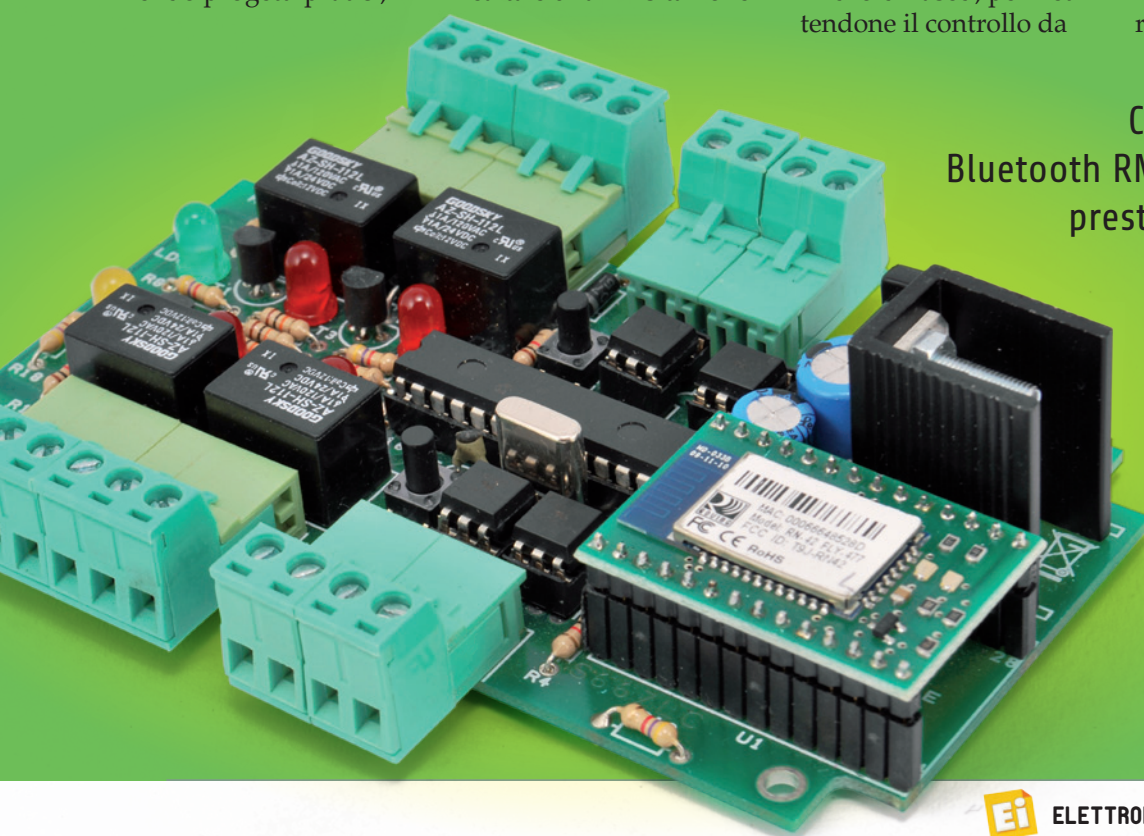
degli Ingg. TOMMASO E ALESSANDRO GIUSTO

Il protocollo Bluetooth è molto utilizzato nelle comunicazioni wireless a corto raggio tra apparati come telefoni cellulari, stampanti, auricolari, dispositivi di acquisizione. Per questa ragione, più volte ce ne siamo occupati, proponendo progetti pratici,

prima con il modulo BISM2 della Ezurio, poi con il nuovo RN-42 della Roving Network RN-42, con il quale in queste pagine vi proponiamo una nuova applicazione: si tratta di una scheda di gestione input/output che in realtà è una rivisitazione

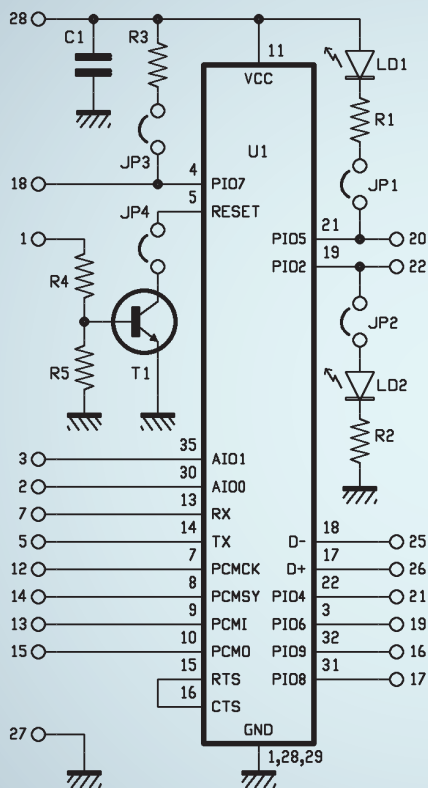
di quella dell'Interfaccia Bluetooth 4 canali In/Out presentata nel fascicolo di febbraio 2007, che abbiamo predisposto per ospitare l'RN-42 invece del BISM2. Inoltre abbiamo riprogettato il firmware per rendere la scheda un sistema Android Based, permettendone il controllo da

qualsiasi dispositivo con sistema operativo Android munito di hardware Bluetooth. Per prima cosa descriviamo le caratteristiche hardware e software della scheda Interfaccia Bluetooth 4 Canali In/Out: la scheda è provvista di quattro uscite a relé ed altrettanti ingressi



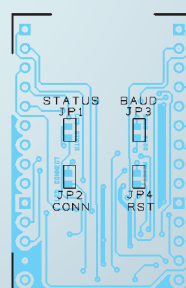
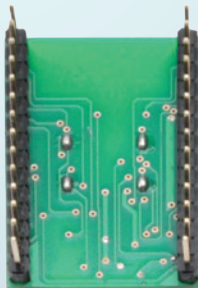
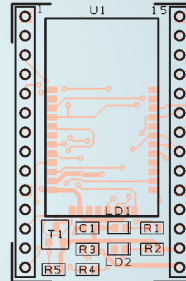
Con il nuovo modulo Bluetooth RN-42 miglioriamo le prestazioni della scheda per telecomando a 4 canali trasformandola in un sistema Android Based.

Il socket per l'RN-42



Elenco Componenti:

- R1: 220 ohm (0805)
- R2: 220 ohm (0805)
- R3: 1 kohm (0805)
- R4: 3,3 kohm (0805)
- R5: 10 kohm (0805)
- C1: 100 nF multistrato
- LD1: LED arancio (0805)
- LD2: LED verde (0805)
- U1: RN-42
- Varie:
 - Strip femmina 14 poli (2 pz.)
 - Circuito stampato



L'RN-42 è un modulo Bluetooth 2.1 di classe 2 con velocità di comunicazione di circa 3 Mbps dotato di antenna ricetrasmittente integrata nel PCB, che consente un range di trasmissione e di utilizzo teorico di circa 15÷20 metri; questa distanza può salire anche fino a 1.000 metri nella versione con antenna esterna.

Il modulo supporta le modalità di funzionamento denominate SPP (Serial Port Profile) e HID (Human Interface Device); inoltre implementa un efficace meccanismo di riduzione del consumo che lo porta, nelle migliori condizioni, ad assorbire solo 26 µA (nelle normali condizioni di utilizzo il consumo aumenta).

Nella modalità SPP il baud rate di comunicazione è di 240 kbps per quanto riguarda i dispositivi slave e di 300 kbps per i master; queste velocità sono più che sufficienti per le più svariate applicazioni Bluetooth (anche per quelle più esigenti).

L'RN-42 dispone di una semplice interfaccia hardware e software UART che può essere utilizzata per il collegamento a qualsiasi microcontrollore, allo scopo di realizzare sistemi embedded; inoltre, avendo integrato al proprio interno lo stack software Bluetooth, non richiede il collegamen-

toisolati a livello di tensione, comandabili tramite Bluetooth. A livello software, l'unità prevede tre modalità di funzionamento: manuale, automatica e semi-automatica (tra uno spegnimento e il successivo riavvio viene memorizzata l'ultima modalità selezionata). Nella prima, tramite apposito software Android può essere collegata ad un dispositivo mobile e, tramite appositi comandi, permette sia di attivare o disattivare i relé distintamente, sia di acquisire e visualizzare la condizione degli input. Nella seconda (automatica) quando l'unità intercetta, nel proprio campo di copertura Bluetooth, un cellulare preventivamente abbinato mediante l'apposita procedura (numero massimo di dispositivi abbinabili: 5), viene attivato un relé (RL1 più preci-

samente), che resta eccitato fin quando lo stesso apparecchio non esce dal campo. Quest'ultima modalità può essere utilizzata come RFID: quando ci si avvicina con in tasca un dispositivo di riconoscimento, si attiva un utilizzatore o si dà il consenso ad altri apparati per l'esecuzione di determinate operazioni. La terza ed ultima modalità (semiautomatica) è simile, ma differisce per il fatto che la prossimità di un cellulare Bluetooth (anche in questo caso il numero massimo di dispositivi abbinabili è 5) non attiva direttamente un determinato relé, ma dà il consenso per il comando diretto di tutti i relé (RL1, RL2, RL3, RL4) mediante un livello di tensione applicato agli input: la presenza di tensione ad un ingresso determina l'attivazione dell'uscita

corrispondente, nel senso che l'input 1 agisce su RL1, il 2 su RL2 e via di seguito. In pratica in questa modalità, quando ci si avvicina con un dispositivo Bluetooth abilitato, è possibile ottenere l'accesso a determinate funzioni.

SCHEMA ELETTRICO

Lo schema elettrico della scheda è sviluppato intorno al microcontrollore U2 (un PIC16F876A di casa Microchip) e al modulo Bluetooth RN-42. Il PIC controlla i relé mediante quattro linee di I/O configurate come uscite, ognuna delle quali pilota un transistor NPN utilizzato come amplificatore di potenza e interruttore statico; i quattro fotoaccoppiatori vengono letti dal micro mediante altrettante linee di I/O e possono essere utilizzati

to ad MCU particolarmente potenti. Come prevede il protocollo Bluetooth, l'RN-42 implementa la crittazione della comunicazione (vengono utilizzati algoritmi a 128 bit) e la correzione automatica degli errori di trasmissione; queste ultime due caratteristiche rendono l'RN-42 ideale anche per applicazioni particolarmente critiche in cui viene richiesta la massima sicurezza e trasmissioni dati senza perdite.

Il modulo RN-42 viene fornito dalla Roving Network in tecnologia SMD e funziona senza bisogno di alcun componente esterno. Per questo motivo Futura Elettronica ha progettato e rende disponibile un socket adattatore munito di doppio strip femmina passo 2,54 mm, che sicuramente risulta più maneggevole e di più semplice montaggio su un circuito tradizionale; il socket dispone di 4 piccoli ponticelli sotto lo stampato predisposti per differenti configurazioni ma non realizzati. In questo progetto i ponticelli vanno chiusi tramite delle piccole gocce di stagno.

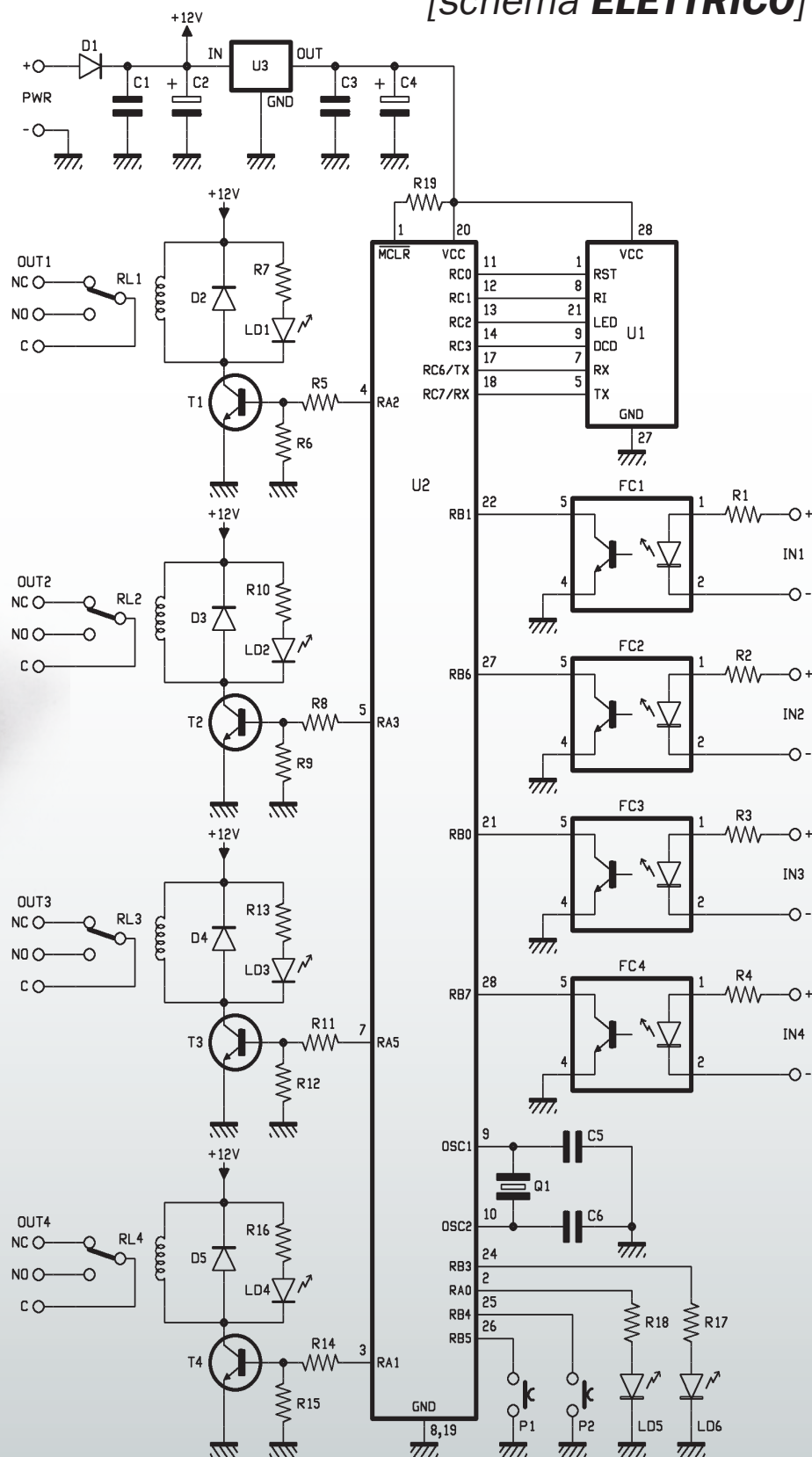
Il modulo completo si chiama FT1018M ed è fornito in versione montata e collaudata.



per rilevare lo stato degli ingressi a livello di tensione: quando ai morsetti IN1÷IN4 viene applicata una tensione di valore 5÷30 V, l'uscita dei fotoaccoppiatori si porta in saturazione e chiude quasi a massa gli ingressi RB1, RB6, RB0, RB7 del PIC, che così ricevono il livello logico basso. Il circuito si alimenta mediante il plug siglato PWM, al quale va collegato lo spinotto di un alimentatore capace di fornire una tensione di valore compreso tra 12 e 15 Vcc. Nel circuito è presente un regolatore di tensione LD1086-3.6 (U3) che fornisce i 3,6 volt al micro e al modulo Bluetooth.

Completano lo schema elettrico della scheda base, i pulsanti P1 e P2 e due LED (giallo e verde)

[schema **ELETTRICO**]



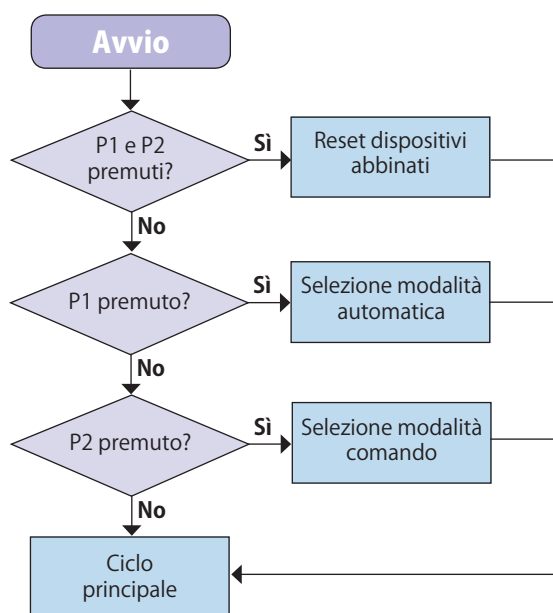
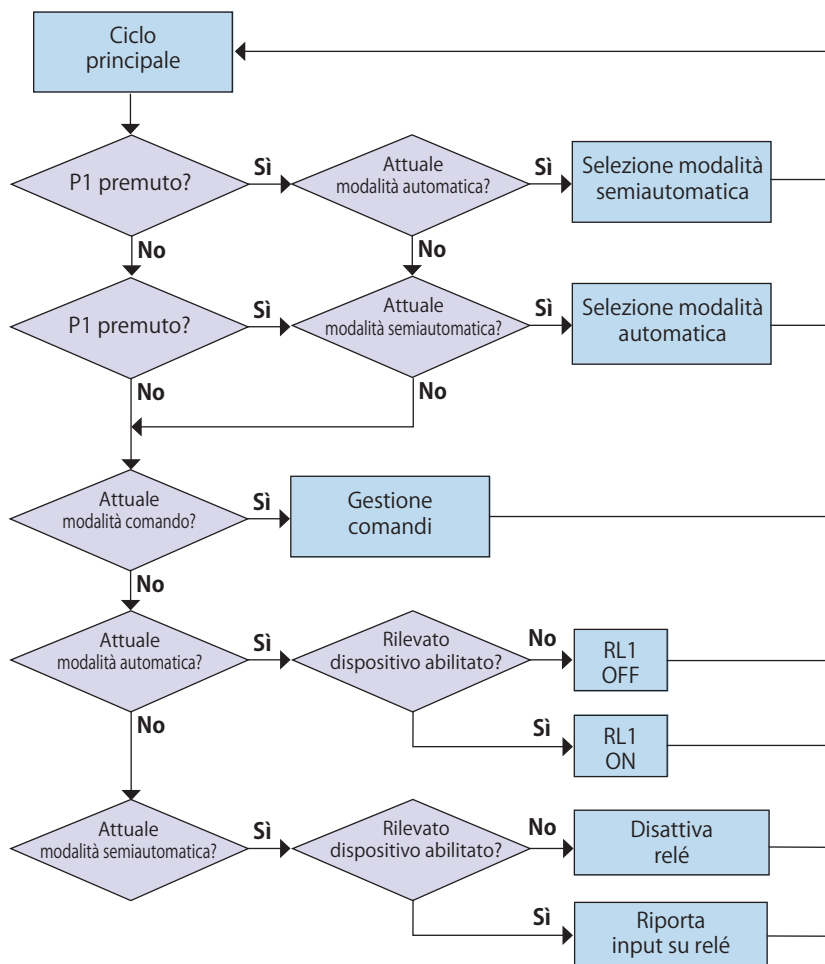


Fig. 1
Il flow chart
del firmware caricato nel
microcontrollore.



utilizzati per la selezione/segnalazione delle modalità di funzionamento.

IL FIRMWARE

Il firmware del microcontrollore è stato progettato per gestire il protocollo di comunicazione seriale (via UART) del nuovo modulo utilizzato; comunque tutte le precedenti funzionalità sono state mantenute e si è cercato di mantenere anche la piena corrispondenza nella selezione/passaggio tra le modalità.

Il firmware del micro gestisce la comunicazione con il modulo U1, legge ciclicamente lo stato logico dei fotoaccoppiatori corrispondenti agli ingressi e, a seconda della modalità di funzionamento e dei comandi ricevuti, pone a livello alto o basso gli output cui sono affidati i relé di uscita.

All'avvio, il PIC inizializza le proprie linee impostando RA1, RA2, RA3 ed RA5 come uscite per il comando dei relé, RB0, RB1, RB6 ed RB7 come ingressi (pull-up interno) per la lettura dei livelli di uscita dei fotoaccoppiatori, quindi RA0 ed RB3 come uscite usate per accendere i LED di segnalazione. RB4 ed RB5 vengono settati come input (con pull-up interno) per la lettura dei pulsanti; infine, il firmware utilizza l'UART interna per dialogare con il modulo Bluetooth.

Il diagramma di flusso visibile nella Fig. 1 mostra il funzionamento del firmware.

La prima operazione eseguita all'avvio è la verifica di quali pulsanti sono premuti: se risultano premuti entrambi, viene azzerata la struttura di memorizzazione dei MAC address delle unità slave Bluetooth abilitate; se viene premuto solo P1 si seleziona la modalità di funzionamento automatica, mentre se è il solo pulsante P2 a essere premuto, si

Il sistema presentato in questo articolo può funzionare secondo 3 distinte modalità:

- **Manuale** (comando): l'unità esegue i comandi impartiti da un dispositivo Android attraverso il protocollo di comunicazione Bluetooth. È possibile attivare o disattivare i singoli relé, oppure leggere e visualizzare graficamente lo stato logico degli ingressi optoisolati.
- **Automatica**: se uno o più dei 5 dispositivi Bluetooth memorizzati entra nell'area di copertura, il PIC attiva il relé RL1, che resta eccitato fino a quando tutti i dispositivi abilitati non escono dall'area.
- **Semiautomatica**: gli input fotoaccoppiati determinano la condizione dei relativi relé (RL1 per IN1, RL2 per IN2, RL3 per IN3 e RL4 per IN4) ma solo quando almeno un dispositivo Bluetooth abilitato entra nel campo di copertura; la presenza di una sufficiente differenza di potenziale a un ingresso forza l'innescio del relé corrispondente.

imposta la modalità manuale (comando). Altrimenti non viene eseguita alcuna operazione, ma si mantiene l'ultima modalità memorizzata.

In seguito, attraverso gli opportuni comandi (vedere i precedenti articoli relativi al componente Bluetooth), viene configurato il modulo RN-42: in particolare se debba funzionare come slave (comando "SM,0") o come master (comando "SM,1") e il pin code predefinito ("1234").

Segnaliamo che se all'avvio viene selezionata la modalità automatica, durante il normale funzionamento è possibile passare dall'automatica alla semiautomatica e viceversa, semplicemente premendo P1. Invece, sempre in queste due modalità, P2 viene utilizzato per avviare la procedura di abbinamento Bluetooth, durante la quale il modulo RN-42 esegue una scansione e memorizza i MAC Bluetooth dei primi 5 dispositivi rilevati.

Infine si entra nella sezione principale di funzionamento: nella modalità comando, attraverso

i dati inviati/ricevuti da RN-42 vengono gestiti gli IN/OUT hardware e la comunicazione con il software Android; nelle altre due modalità, il PIC comanda ciclicamente il modulo per eseguire successive scansioni Bluetooth e, a seconda delle risposte ricevute e dello stato degli ingressi hardware, attiva o disattiva le relative uscite.

SOFTWARE ANDROID

Il software sviluppato per smartphone Android ("littleBlueControl") permette il collegamento e la gestione da remoto della scheda nel caso in cui si selezioni la modalità comando: è possibile verificare lo stato corrente degli ingressi digitali, rendersi conto dello stato delle uscite (ON o OFF) e modificarne il valore secondo le proprie necessità (in sostanza si può dire che si tratta di una versione rivista del software "blueControl" presentato nel numero di Luglio/Agosto 2012).

All'avvio, dopo una schermata iniziale di presentazione del sof-

tware (Fig. 2), vengono avviati la scansione e rilevamento di tutti i dispositivi Bluetooth presenti all'interno della zona di copertura dello smartphone.

Nella lista visualizzata a destra in Fig. 2, vengono mostrati sia tutti i sistemi già accoppiati, sia i nuovi rilevati; cliccando sulla voce della scheda (*Vecchia Scheda I/O*, nel nostro esempio) la si seleziona e si richiede l'avvio del collegamento.

Inoltre è presente un pulsante per richiedere una nuova esecuzione della scansione e l'aggiornamento della lista.

A questo punto, se è la prima volta che il dispositivo viene rilevato, viene richiesta la digitazione del PIN code (quello predefinito dal firmware è 1234) come mostra la Fig. 3. In seguito, si passa nella schermata principale del programma (Fig. 4) nella quale, partendo dall'alto viene visualizzato lo stato delle quattro



Fig. 2 - I sistemi Bluetooth abbinati e rilevati dall'unità.

uscite a relé e degli altrettanti ingressi fotoaccoppiati. Per ogni singola uscita viene utilizzata la seguente convenzione: sulla sinistra è presente un'etichetta di testo che indica lo stato corrente; più a destra sono invece presenti 2 pulsanti (etichettati con ON e OFF di cui solo 1 risulta abilitato) utilizzabili per modificare lo stato corrente.

Se l'uscita è attiva, l'etichetta di testo riporta ON e risulta abilitato il pulsante OFF, altrimenti l'etichetta riporta OFF ed è abilitato il pulsante ON.

Ciò perché nel primo caso, essendo l'uscita attiva, sarà possibile solo richiederne la disattivazione, pertanto è corretto che sia abilitato il solo pulsante OFF; viceversa, nel secondo caso il solo comando plausibile è l'attivazione e pertanto risulta abilitato il pulsante ON. Dalla Fig. 4 è possibile capire che sono attivi i relé 1 e 3, mentre il 2 e il 4 sono OFF.

Per ogni ingresso fotoaccoppiato è invece presente una sola etichetta di testo riportante lo

stato corrente: ON nel caso che all'ingresso fotoaccoppiato sia applicata una tensione sufficiente ad attivare lo stesso; OFF in caso contrario (nella solita schermata di Fig. 4 si vede che tutti e quattro gli ingressi sono OFF).

Tornando ancora alle uscite, per richiedere la modifica dello stato corrente è sufficiente fare clic sul relativo pulsante attivo; a questo punto il comando selezionato viene inviato alla scheda e sullo smartphone appare una finestra che illustra l'invio del comando (Fig. 5); al completamento della procedura, viene aggiornata l'interfaccia grafica.

Un ultimo dettaglio da segnalare circa il software Android, è la presenza di un timer di aggiornamento della grafica: ciclicamente il software si collega alla scheda, ne legge lo stato corrente e in conseguenza alle informazioni ricevute, aggiorna l'interfaccia grafica (in particolare se è variato lo stato degli ingressi, anche la relativa sezione grafica viene modificata).

IL PROTOCOLLO DI COMUNICAZIONE

Quando viene selezionata la modalità comando, il protocollo di comunicazione tra scheda e smartphone prevede che quest'ultimo agirà da master Bluetooth, mentre la scheda si comporterà da slave. Il protocollo prevede trame composte da 4 byte, strutturate come di seguito.

- Primo byte = tipo di operazione (lettura o impostazione stato I/O scheda); GETINFO_CMD (byte 0x41) per lettura; SETINFO_CMD (byte 0x42) per impostazione;
- secondo byte = identificativo tipo della risorsa. Sono previste 3 possibilità: stato completo scheda (STATUS_CMD; byte 0x40); singola uscita a relé (OUTRLE_CMD; byte 0x41) ed infine singolo ingresso digitale (INPDIG_CMD; byte 0x44).
- Terzo byte = identificativo numero della risorsa.
- Quarto byte = stato di impostazione risorsa (solo per settaggio uscite relé).



Fig. 3 - L'inserimento del PIN Bluetooth.



Fig. 4 - La finestra che riepiloga lo stato degli ingressi e delle uscite a relé.



Fig. 5 - L'invio del comando via Bluetooth alla scheda.

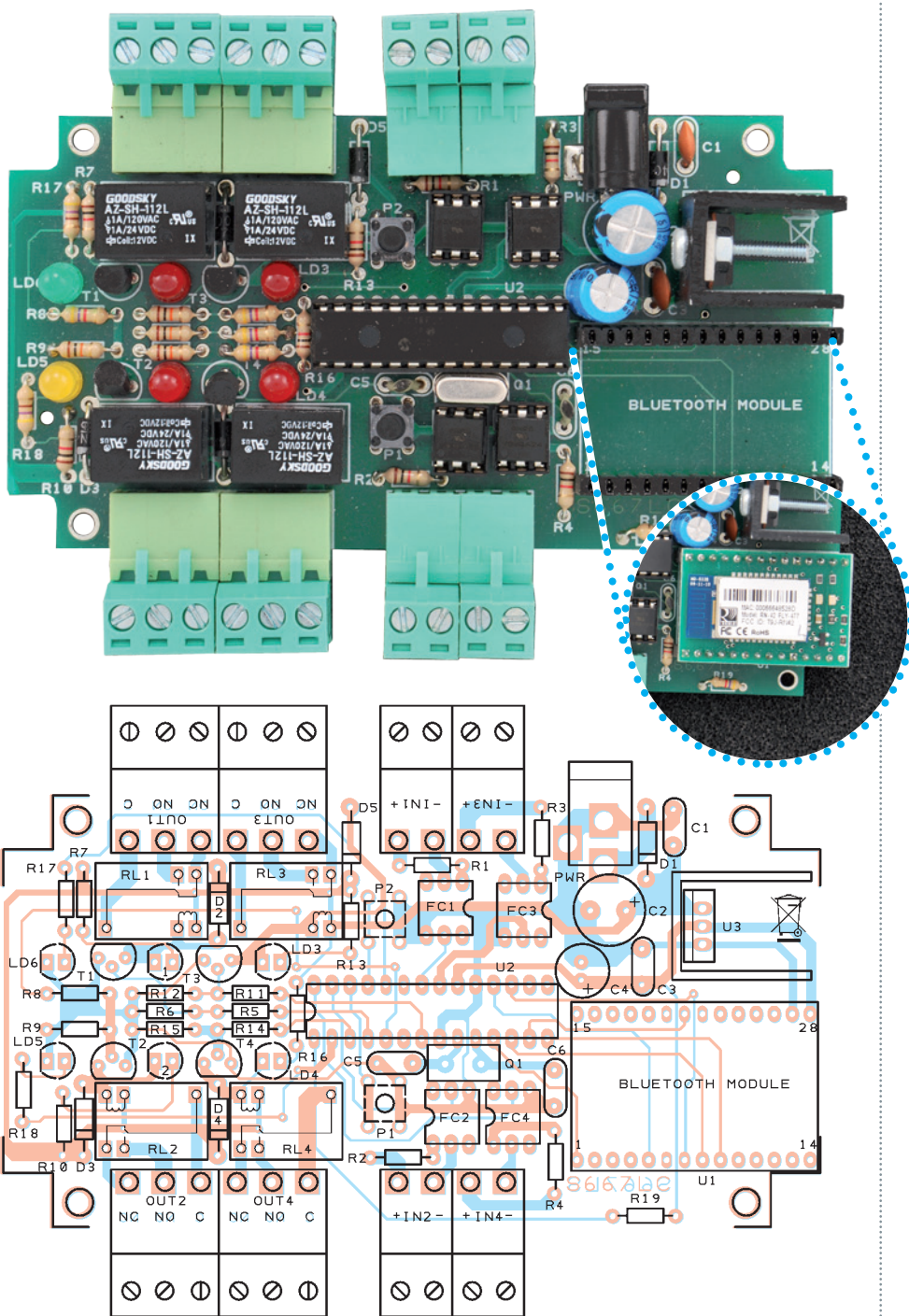
[piano di **MONTAGGIO**]

Elenco Componenti:

R1÷R4: 1 kohm
 R5: 4,7 kohm
 R6: 10 kohm
 R7: 1 kohm
 R8: 4,7 kohm
 R9: 10 kohm
 R10: 1 kohm
 R11: 4,7 kohm
 R12: 10 kohm
 R13: 1 kohm
 R14: 4,7 kohm
 R15: 10 kohm
 R16: 1 kohm
 R17, R18: 470 ohm
 R19: 1 kohm
 C1, C3: 100 nF multistrato
 C2: 470 µF 25 VL elettrolitico
 C4: 470 µF 16 VL elettrolitico
 C5, C6: 10 pF ceramico
 U1: Socket RN-42 (FT1018M)
 U2: PIC16F876A (MF1095)
 U3: LD1086-3.6
 D1÷D5: 1N4007
 Q1: quarzo 20 MHz
 LD1÷LD4: LED 5 mm rosso
 LD5: LED 5 mm giallo
 LD6: LED 5 mm verde
 T1÷T4: BC547
 RL1÷RL4: Relé 12V 1 scambio
 P1: microswitch
 P2: microswitch
 FC1÷FC4: 4N25

Varie:

- Plug alimentazione
- Morsetti componibile 2 poli 90° (4 pz.)
- Morsetti componibile 3 poli 90° (4 pz.)
- Zoccolo 3+3 (4 pz.)
- Zoccolo 14+14
- Strip femmina 14 poli (2 pz.)
- Vite 10 mm 3 MA
- Dado 3 MA
- Dissipatore (ML26)
- Circuito stampato codice S667



Il protocollo prevede inoltre che, ad ogni richiesta di comandi dal master, lo slave risponda con pacchetti di conferma.

REALIZZAZIONE PRATICA

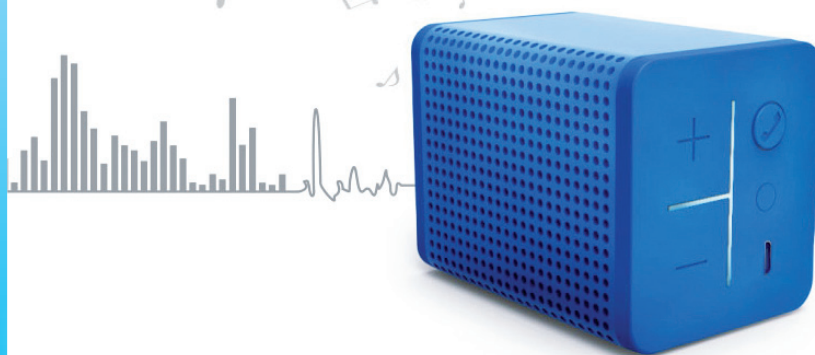
L'hardware del sistema è fonda-

mentalmente lo stesso di quello presentato nel numero di febbraio 2007, con la differenza che il modulo Bluetooth è basato sul nuovo RN-42; in altre parole, la scheda base resta invariata e ciò che cambia è il c.s. che ospita l'RN-42, da montare

sulla scheda base ad innesto, dopo averlo provvisto di due file di pin-strip a passo 2,54 mm da 14 contatti ciascuna. Il socket dispone di 4 piccoli ponticelli sotto lo stampato predisposti per differenti configurazioni ma non realizzati. In questo

progetto i ponticelli vanno chiusi tramite delle piccole gocce di stagno. Per il montaggio, diciamo che tutto il circuito è composto da componenti elettronici in formato tradizionale, pertanto l'operazione è semplice e alla portata di tutti: è sufficiente un saldatore da 25÷30 W e un po' di filo di lega saldante da 0,75 mm. Al solito, disponete per primi i diodi al silicio e le resistenze, quindi gli zoccoli per i fotoaccoppiatori e il microcontrollore, nonché i pulsanti miniatura; quindi passate ai transistor e ai LED, poi ai condensatori (prima i non polarizzati e poi gli elettrolitici) ed alla presa plug per l'alimentazione. Montate quindi due file di contatti femmina a passo 2,54 mm da 14 contatti ciascuna in corrispondenza dei fori per l'inserzione del modulo Bluetooth, i quattro relé e le morsettiere, concludendo con il regolatore di tensione U3, che dovete mantenere verticale e sollevato quanto basta a fissarlo ad un radiatore di calore ad U avente resistenza termica di circa 20 °C/W (va bene il classico ML26). Per l'orientamento dei componenti polarizzati (elettrolitici, transistor, LED, optoisolatori, regolatore di

Il box Bluetooth mini dalle prestazioni maxi



Il mondo degli accessori per smartphone e tablet sforna novità a ciclo continuo; una delle più recenti è questa cassa acustica miniatura completamente autonoma e collegabile via Bluetooth a tutti gli smartphone, tablet e PC notebook. Si chiama Mipow The Boom ed è uno speaker con cui è possibile ascoltare musica senza alimentazione esterna ed effettuare chiamate in vivavoce per un massimo di 12 ore.

Il prodotto, realizzato da Mipow (www.mipow.com), è disponibile anche in versione Mini, dalle dimensioni ancora più compatte, che consente fino a 9 ore di conversazione e di riproduzione musicale continua. Entrambi i box supportano la tecnologia bluetooth V4.0 a basso consumo e possono essere associati anche a due dispositivi (smartphone, tablet...) contemporaneamente.

per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono di facile reperibilità. Il circuito è anche disponibile in kit (cod. FT1095K) al prezzo di 64,00 Euro. Il prezzo comprende il circuito base in kit ed il modulo Bluetooth FT1018M già montato. Quest'ultimo è disponibile anche separatamente al prezzo di 29,00 Euro. I prezzi sono 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>

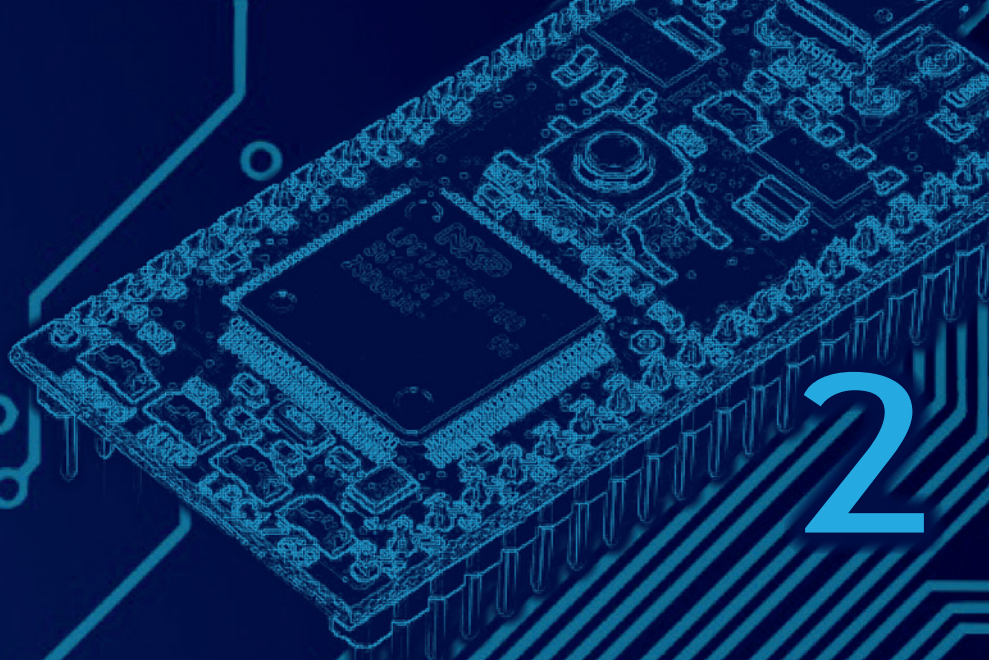
tensione e microcontrollore) riferitevi al piano di montaggio pubblicato in queste pagine.

Quanto al modulo Bluetooth, per mantenere la compatibilità con quello dell'Ezurio (montato su una basetta e commercializzato da Futura Elettronica come FT622M) abbiamo previsto di montare anche l'RN-42 su un c.s. (socket) dotato di due pin-strip da 14 contatti ciascuno, mantenendo sia il passo, sia la pin-out compatibile con il vecchio; pertanto è sufficiente inserire il nuovo modulo nello zoccolo previsto. Sia sullo stampato della scheda principale che su quello del nuovo modulo Bluetooth, è riportata la numerazione dei pin del socket (sono

indicati i pin numero 1, 14, 15 e 28), pertanto se inserite il nuovo modulo avendo cura di far coincidere i pin andrà tutto bene.

Quanto al firmware del PIC, chiaramente nonostante le due versioni della scheda abbiano le stesse funzionalità, il firmware tra le due non può essere lo stesso in quanto è stato aggiornato il modulo Bluetooth. Pertanto se eseguite l'aggiornamento partendo da una precedente scheda, oltre a sostituire il modulo Bluetooth cambiate anche il PIC. Se invece avete costruito adesso la scheda, programmate il vostro PIC con il firmware scaricabile insieme agli altri file di questo progetto e non preoccupatevi di null'altro. ■

Scopriamo una scheda di interfaccia e prototipazione che permette di utilizzare con mbed gli shield pensati per Arduino. 2^ puntata.



2

Conoscere e usare

mbed

dell'ing. Vincenzo Mendola

Come abbiamo già spiegato nella prima puntata di questo corso, l'mbed ha un package DIP di 40 pin, con passo 0.1", che misura soltanto 54x26mm, quindi non molto più grande di un Arduino Nano. Per poter utilizzare in maniera proficua tutte le caratteristiche del nostro sistema di prototipazione, è conveniente disporre di un circuito stampato cui poter connettere in maniera stabile e affidabile il nostro mbed e che sia in grado inoltre di fornirci

un supporto meccanico adeguato. Il codice per sviluppare applicazioni con l'mbed è diverso da quello utilizzato da Arduino, tuttavia nella prima puntata abbiamo mostrato che è possibile eseguire in maniera equivalente le stesse funzioni di gestione dei GPIO, ADC, ecc. Allora, perché non realizzare un unico PCB che, oltre a fornire un utile supporto alle connessioni del nostro mbed, ci permetta anche di impiegare gli innumerevoli shield di Arduino? Ci siamo posti questa domanda e in risposta ab-

Listato 1

```
#include "mbed.h"
#include "ds1307.h"

Serial pc(USBTX, USBRX);

DS1307 my1307(p9,p10);

int sec = 00;          // Values to set time
int min = 00;
int hours = 12;
int day = 7;
int date = 3;
int month = 3;
int year = 13;

void test_rw(int test)
{
    if (test == 0)
    {
        pc.printf("%0.2D:",hours);
        pc.printf("%0.2D:",min);
        pc.printf("%0.2D \n\r",sec);
        pc.printf("%0.2D/",date);
        pc.printf("%0.2D/",month);
        pc.printf("%0.2D \n\r",year);
        pc.printf("\n\r");
    }

    else pc.printf("Last R/W operation failed!\n\r");

    wait(1);
}

int main ()
{
    //test_rw(my1307.settime( sec, min, hours, day, date, month, year)); wait(3); // Required only during
    the first RTC connection to set the RTC.

    while(1){
        test_rw(my1307.gettime( &sec, &min, &hours, &day, &date, &month, &year)); // Read the time and display on
        screen
    }
}
```

questo caso automaticamente ridotta per tensioni d'ingresso maggiori di 10 V.

In entrambi i casi la massima tensione applicata all'ingresso del regolatore non può eccedere i 15 V, pena il danneggiamento irreversibile del chip. Nel nostro circuito possono essere utilizzate entrambe le versioni, dato che hanno la stessa piedinatura; per quanto riguarda la corrente di esercizio, difficilmente verrà superato il valore di 500 mA, che poi è la corrente standard erogabile da una comune porta USB. Per la stessa ragione abbiamo scelto il più compatto case SOT223, che difficilmente sarà sottoposto a grandi stress termici, meglio tollerati dalla più massiccia versione in contenitore DPAK.

Come è visibile dallo schema elettrico, sia il plug DC che la USB sono protetti dai cortocircuiti mediante protezioni autoripristinanti ottenute mediante l'uso

di poliswitch da 500 mA, che abbiamo voluto inserire anche se generalmente le porte USB dei PC sono già munite di una loro protezione; in questo modo si può stare tranquilli anche qualora si decida di alimentare il circuito mediante USB non standard o presenti su alimentatori autocostruiti come quelli che presenteremo nei prossimi numeri della rivista. I diodi Schottky D1 e D2 servono per separare le sorgenti di alimentazione dato che, come abbiamo visto, il nostro mbed può essere alimentato sia tramite USB, sia tramite il DC plug.

Sempre sulla linea di alimentazione, possiamo notare D3, il quale, in maniera analoga a quanto appena visto, ha la funzione di garantire che il terminale d'uscita a 5 V del regolatore LDO presente sul PCB dell'mbed possa solo erogare corrente e non diventare un carico. I catodi di questi tre diodi sono fra loro connessi, garan-

Listato 2

```
#include "mbed.h"

int main() {

    // get the current time from the terminal
    struct tm t;
    printf("Enter current date and time:\n\r");
    printf("YYYY MM DD HH MM SS[enter]\n\r");
    scanf("%d %d %d %d %d %d", &t.tm_year, &t.tm_mon, &t.tm_mday, &t.tm_hour, &t.tm_min, &t.tm_sec);

    // adjust for tm structure required values
    t.tm_year = t.tm_year - 1900;
    t.tm_mon = t.tm_mon - 1;

    // set the time
    set_time(mktime(&t));

    // display the time
    while(1) {
        time_t seconds = time(NULL);
        printf("%s \r", ctime(&seconds));
        wait(1);
    }
}
```

do in questo modo una corretta tensione di alimentazione al pin Vin del mbed. Alla luce di quanto appena visto, al fine di ridurre le cadute di tensione ai loro capi, i tre diodi devono essere necessariamente di tipo Schottky, come ad esempio l'PME-G2005AEA della NXP da noi scelto, che

garantisce una piccola caduta di tensione (400mV@IF=500mA); C1 e C2 sono i due condensatori di filtro al tantalio da 10 µF, della stessa tipologia consigliata dal datasheet; R1 e R2 (da 180 ohm) sono le due resistenze di limitazione della corrente che scorre nei LED inseriti all'interno

Listato 3

```
#include "mbed.h"
#include "EthernetInterface.h"
#include "NTPClient.h"

EthernetInterface eth;
NTPClient ntp;

int main()
{
    eth.init(); //Use DHCP

    eth.connect();

    printf("Trying to update time...\r\n");
    if (ntp.setTime("ntp1.inrim.it") == 0) // INRIM italian NTP server
    {
        printf("Set time successfully\r\n");
        time_t ctTime;
        ctTime = time(NULL);
        printf("Time is set to (UTC): %s\r\n", ctime(&ctTime));
    }
    else
    {
        printf("Error\r\n");
    }

    eth.disconnect();

    while(1) {
    }
}
```

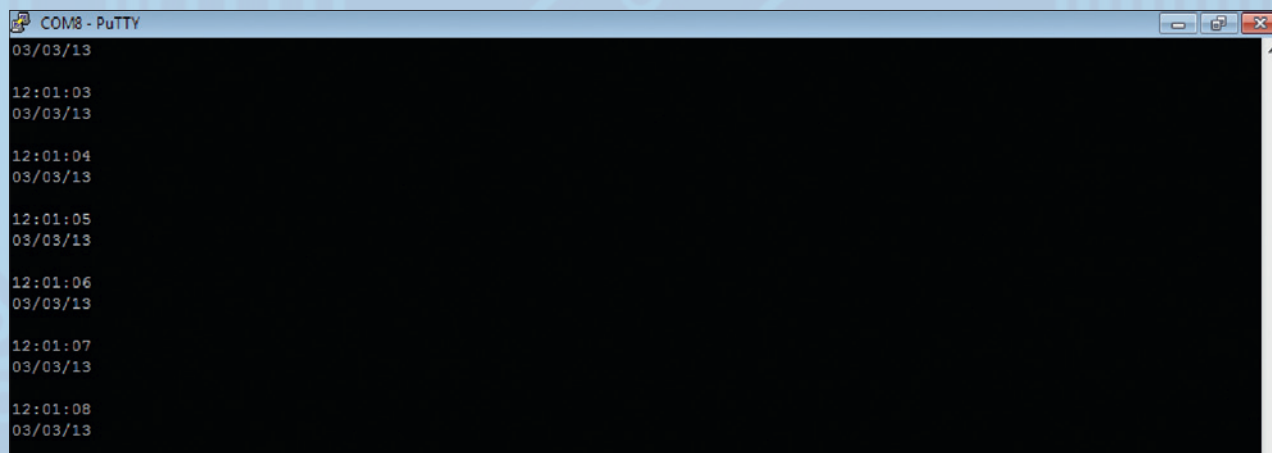



Fig. 3

del connettore RJ45 dotato di filtri magnetici (si può acquistare da Futura Elettronica, www.futurashop.it, con il codice 7300-RJ45EM) che possono essere utilizzati per segnalare la connessione dell'mbed ad una rete ethernet e per evidenziare lo scambio di dati su di essa. C3 e C4 servono per disaccoppiare la tensione di 3,3 V come riportato nella pagina del cookbook dedicata (punto [1] dei Riferimenti).

Il pcb è stato disegnato per ottenere la massima compatibilità con gli shield di Arduino; la **Tabella 1** illustra la corrispondenza tra i pin dell'mbed con quelli di Arduino, un'informazione necessaria, durante la scrittura del codice, per garantire una perfetta equivalenza delle funzionalità con gli shield già esistenti.

Come si può notare, sono disponibili tutti i GPIO dell'Arduino Uno (D2-D13), i sei PWM (D3, D5, D6, D9, D10, D11), la seriale (D0, D1), la I2C (A4-A5), le connessioni all'ICSP e 4 ADC (A0-A3), tutti disponibili nella stessa posizione in cui lo sono sull'Arduino. Per l'mbed LCP1768 è anche disponibile il DAC, che risulta connesso al pin 18 dell'mbed e all'A3 del nostro PCB. Ora che abbiamo a disposizione questo comodo circuito, che ci permette di utilizzare con facilità gli shield di Arduino, possiamo ad esempio collegare all'mbed l'RTC realizzato per Arduino e presentato nel Fascicolo n° 163 (febbraio 2012) della nostra rivista, con l'unica accortezza di ponticellare con dei fili le due connessioni a SDA e SCL (replicate nelle ultime revisioni di Arduino UNO alle estremità del connettore dei GPIO, oltre il D13) con i

pin analogici A4 (SDA) e A5 (SCL) della Arduino stessa.

Ora, prima di procedere è importante fare una precisazione: gli mbed sono 5V-tolerant (cioè accettano livelli TTL, pur

ARDUINO	MBED	FUNCTION
D0	p14	
D1	p13	
D2	p8	
D3	p21	PWM
D4	p12	
D5	p22	PWM
D6	p23	PWM
D7	p27	
D8	p30	
D9	p24	PWM
D10	p25	PWM
D11	p26	PWM
D12	p28	
D13	p29	
A0	p15	
A1	p16	
A2	p17	
A3	p18	DAC
A4	P9	SDA
A5	P10	SCL
ICSP	P5	MOSI
ICSP	P6	MISO
ICSP	P7	SCK

Tabella 1

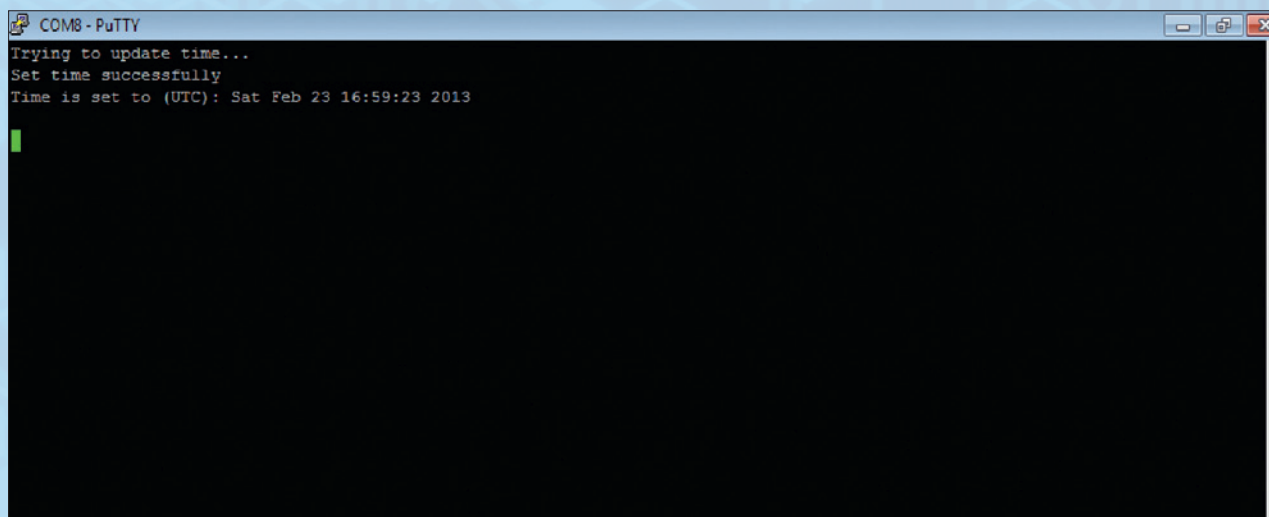


Fig. 4

prediligendo quelli 0/3,3V) ma, per essere certi di evitare danni irreparabili al chip, è sempre consigliabile utilizzare un traslatore di livello tipo quello presentato per Raspberry Pi nel fascicolo di febbraio di quest'anno, oppure lavorare direttamente con logica a 3,3 V, ovviamente anche per il Bus I²C.

Lo shield RTC di Arduino lavora con un I²C a livello TTL (0/5V) per cui è necessaria molta attenzione qualora lo si voglia connettere direttamente all'mbed.

Sappiamo che l'LPC1768, a differenza della versione LPC1114, incorpora già un modulo RTC, ma comunque ad entrambi è possibile connettere un DS1307 della Maxim, servendosi di un'apposita libreria (ad esempio quella disponibile all'indirizzo http://mbed.org/users/harrypowers/code/DS1307/docs/c3e4da8feb10/ds1307_8h_source.html) e del semplice codice riportato nel **Listato 1**.

Nella **Fig. 3** è visibile la schermata di terminale della seriale, contenente i dati in uscita dall'RTC. Notate che è possibile utilizzare anche l'RTC a bordo dell'LPC1768 e del nuovo modello FRDM-KL25Z, servendosi del codice (punto [2] dei Riferimenti) contenuto nel **Listato 2**.

Come si vede dalla lettura delle poche istruzioni necessarie, l'RTC on-board di mbed viene impostato direttamente inserendo la data e l'ora tramite seriale, sulla quale viene costantemente aggiornata e visualizzata ogni secondo.

Il nostro PCB di adattamento dispo-

ne anche di un connettore ethernet che consente di connettere l'LPC1768 alla rete LAN (Local Area Network) e WAN (Wide Area Network); perciò vi proponiamo un semplice esempio di utilizzo della libreria ETHERNET che permette di determinare l'ora esatta tramite un server NTP (Network Time Protocol) qualora abbiate connesso il vostro mbed alla rete INTERNET e vogliate sfruttare l'altissima precisione fornita dagli orologi atomici (il codice è illustrato nel **Listato 3**). Per i nostri test ci siamo serviti del primo dei due server NTP primari installati nel Laboratorio di Tempo e Frequenza Campione dell'INRIM, accessibili liberamente. Ulteriori informazioni sono disponibili alla pagina http://www.inrim.it/ntp/index_i.shtml. La libreria ed il relativo codice sono disponibili nella pagina http://mbed.org/users/donatien/code/NTPClient_HelloWorld/.

Nella **Fig. 4** si vede il risultato della sincronizzazione con il server NTP.

Bene, con questo abbiamo concluso anche questa puntata; nella prossima mostreremo ulteriori esempi di utilizzo del nostro circuito di adattamento e approfondiremo la conoscenza delle schede di prototipazione per mbed.

Riferimenti

- [1] <http://mbed.org/cookbook/Ethernet-RJ45>
- [2] <http://mbed.org/blog/entry/103/>

LDC: NASCE LA MISURA A SENSORI INDUTTIVI

di DAVIDE SCULLINO



Texas Instruments presenta un sensore induttivo che rileva metalli ed elementi conduttivi con una precisione mai vista: permetterà di sostituire molti dei tradizionali sensori usati per rilevare spostamenti, forze, consistenza di metalli e sistemi per cercare imperfezioni nei pezzi meccanici.

Il rilevamento mediante induzione elettromagnetica è una pratica comune a moltissimi campi: dalla prospezione mediante metal-detector per cercare metalli preziosi o ordigni bellici, al rilevamento della posizione o dello spostamento di particolari metallici nell'industria, fino ad arrivare al settore dell'automobile, dove viene impiegata nelle ruote foniche dell'ABS per verificare

la rotazione delle ruote o negli iniettori strumentati dei motori diesel (in cui consente di rilevare la discesa dello spillo durante l'iniezione). Per quanto sia una tecnica super-collaudata, finora è stata impiegata principalmente in applicazioni in cui serve rilevare lo spostamento o la presenza di oggetti di dimensioni relativamente grandi e per spostamenti abbastanza estesi, in quanto non è

CARATTERISTICHE TECNICHE dell'LDC1000

- Frequenza di lavoro: 5 kHz – 5 MHz
- Risoluzione corrente parassita (R_p): 16 bit
- Risoluzione Impedenza: 24 bit
- Tensione di alimentazione: 5 V
- Corrente assorbita: 1,7 mA
- Contenitore: SON-16
- Interfaccia: SPI

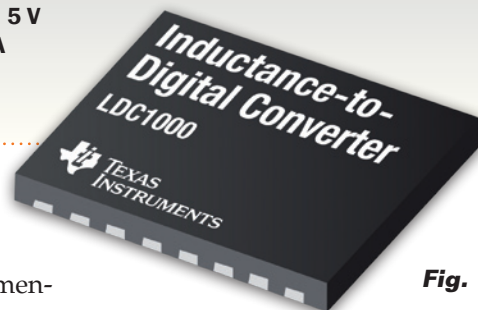


Fig. 1

semplice, con un sistema elettromagnetico, rilevare la presenza di elementi di dimensioni micrometriche o spostamenti micrometrici. Ciò, a causa del fatto che il rilevamento si effettua mediante un induttore inserito in un circuito elettronico, il quale irradia un'onda elettromagnetica che incontrando il materiale conduttore induce fenomeni riconducibili all'elettromagnetismo, in grado di variare, nel circuito, parametri come l'assorbimento della bobina o la frequenza del campo elettromagnetico generato. Queste variazioni sono analogiche e possono essere quantificate con una tolleranza tale da non consentire rilevamenti di estrema precisione.

Oggi, però, Texas Instruments ci informa che qualcosa è cambiato e ci mette a conoscenza della realizzazione di un nuovo circuito integrato in grado di effettuare rilevamenti di oggetti e spostamenti di oggetti, anche dell'ordine dei micron; la tecnologia proposta potrà essere utilizzata per realizzare svariati tipi di sensore, da quelli di movimento, rotazione e spostamento, sino ai sensori di flusso, pressione e forza che trovano applicazione in ambito industriale, automotive, consumer, medicale, mobile, eccetera.

COME FUNZIONA

Il principio di funzionamento del

dispositivo di Texas Instruments deriva da quello dei metal-detector: questi funzionano, a seconda del tipo di materiale da rilevare, con tre tecniche diverse che si basano sulla variazione di induttanza di una bobina inserita in un oscillatore, sulla variazione di flusso in un trasformatore e, infine, sulla variazione di induttanza e assorbimento in un oscillatore. Il primo e il secondo caso si applicano al rilevamento dei materiali ferromagnetici (es.: ferro) e paramagnetici (es.: acciai ad alto tenore di carbo-

nio, alluminio, platino) mentre il terzo permette di rilevare sia metalli (siano essi ferromagnetici, paramagnetici o diamagnetici), sia non metalli, cioè materiali elettricamente conduttivi ma che non sono propriamente metalli (per esempio soluzioni acquose contenenti ioni disciolti).

La nuova tecnologia si basa su un dispositivo a basso costo denominato LDC1000 (LDC sta per Inductance to Digital Converter), che TI definisce "convertitore analogico-digitale di induttanza". Ma che cos'ha di particolare? Semplice: rispetto ai circuiti analogici tradizionali, il componente permette di rilevare minime variazioni di induttanza (ma anche di corrente assorbita dall'induttore) con una precisione elevatissima, dato che queste variazioni dal lato analogico vengono convertite in valori numerici mediante un ADC che presenta una risoluzione che arriva a 24 bit (16 sulle misurazioni della corrente indotta nel target)! Questo significa poter apprezzare variazioni quasi infinitesime e perciò poter utilizzare il sensore per rilevare oggetti piccolissimi

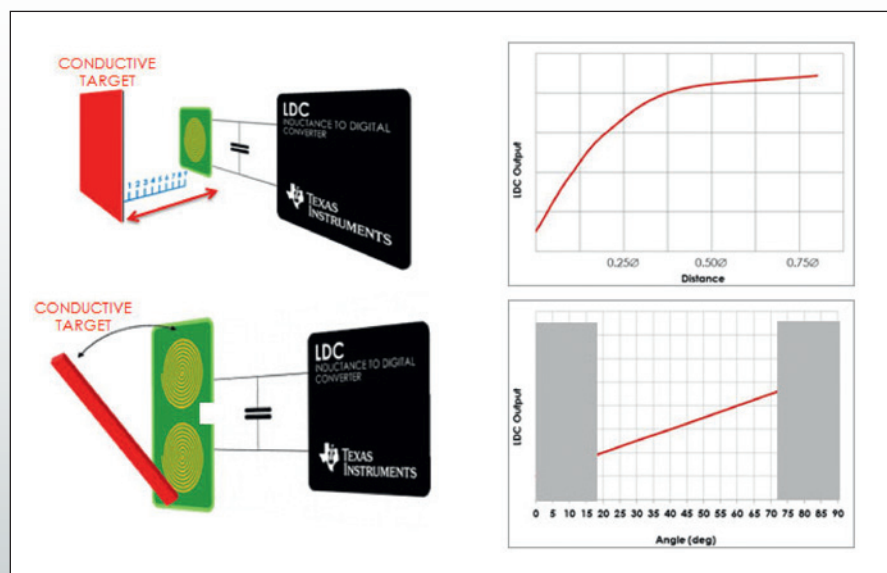


Fig. 2 - Rilevamento della distanza e inclinazione di oggetti target.

o che si spostano di pochissimo, ovvero analizzare con precisione la discontinuità nella struttura di un metallo e perciò poter realizzare con il componente TI apparati di verifica industriale capaci di equiparare i sistemi di analisi dei metalli mediante raggi X o ultrasuoni. La soluzione proposta da TI è sostanzialmente un metal-detector composto da un circuito risonante che utilizza una bobina, che crea un campo elettromagnetico entro il quale si deve porre ed eventualmente muovere il target (ossia l'oggetto da rilevare) conduttivo il cui movimento può essere rilevato con grandissima precisione. La bobina può anche essere sagomata a molla e in questo caso il sensore potrebbe rilevarne la compressione, trazione o torsione, con grande precisione, in quanto costituiscono una variazione di induttanza. La bobina può anche rilevare la deformazione di target metallici a forma di molla. Il sistema funziona analogamente ai metal detector per il rilevamento di tutti i metalli e viene proposto come soluzione low-cost per sostituire sensori tradizionali basati su vari principi, ovviando ai loro limiti fisici. Per esempio, i sensori di tipo ohmico (meccanico) non possono essere utilizzati in ambienti ostili e vanno opportunamente protetti; i trasduttori di forza (FSR) presentano una scarsa risoluzione e una limitata distanza di funzionamento, mentre i sensori ad ultrasuoni sono soggetti a disturbi ambientali e funzionano male sulle piccole distanze. Ancora, i sensori capacitivi, pur avendo una elevata sensibilità, presentano una scarsa selettività e quelli ad effetto di Hall richiedono l'impiego di componenti magnetici (non sempre compatibili con l'ambiente di lavoro)

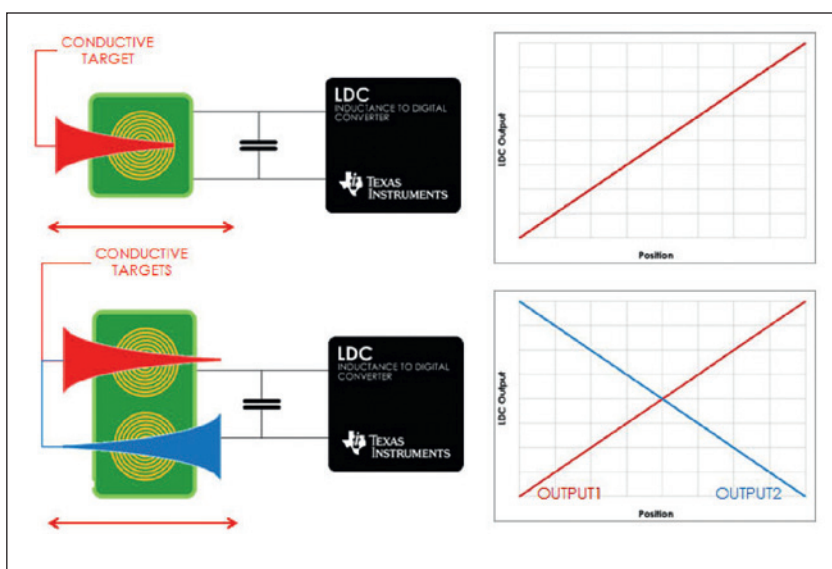


Fig. 3 - Rilevamento del movimento di un oggetto: i grafici in questa e in altre figure sono quelli visibili nella schermata del software di test.

e necessitano di una attenta calibrazione; infine, i sensori ottici non possono funzionare in ambienti inquinati. Le criticità di queste tecnologie possono essere superate facendo ricorso al sistema messo a punto da Texas Instruments, che può essere composto da uno o più convertitori induttivo-digitali LDC1000 dotati di una bobina che può assumere le forme più

diverse. Con il sistema di Texas Instruments, il target (cioè l'elemento da rilevare) si muove all'interno del campo creato dalla bobina dell'LDC e, grazie all'elevatissima risoluzione del dispositivo, viene rilevato con una precisione sub-micrometrica. Ciò significa, ad esempio, che questo sistema potrebbe rilevare le fratture micrometriche nella struttura degli aerei o di altri

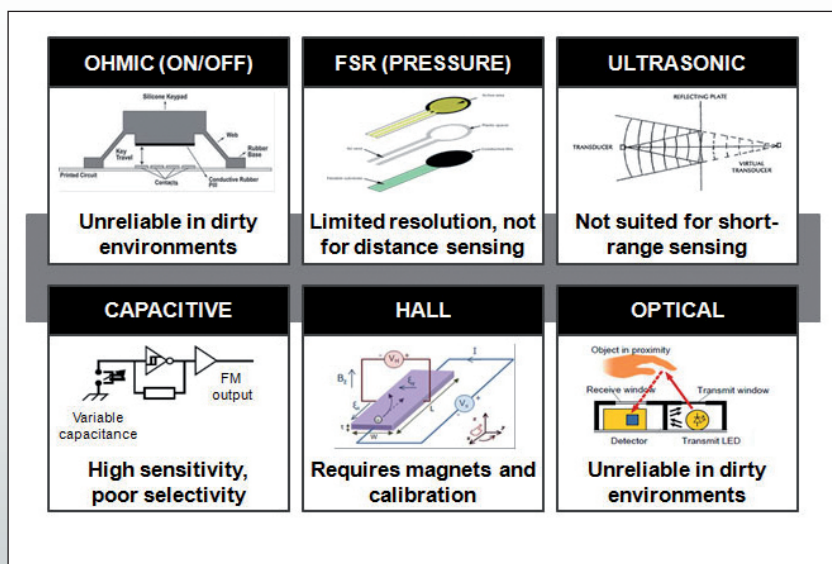


Fig. 4 - L'LDC1000 di Texas Instruments può sostituire sensori di vario tipo.

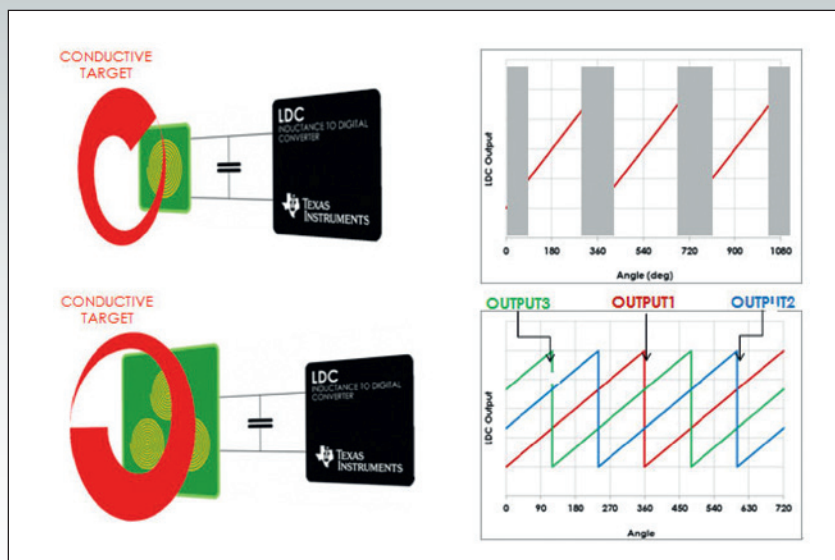


Fig. 5 - Rilevamento della posizione angolare di una manopola o di una ruota, ottenuto mediante l'applicazione di una striscia metallica.

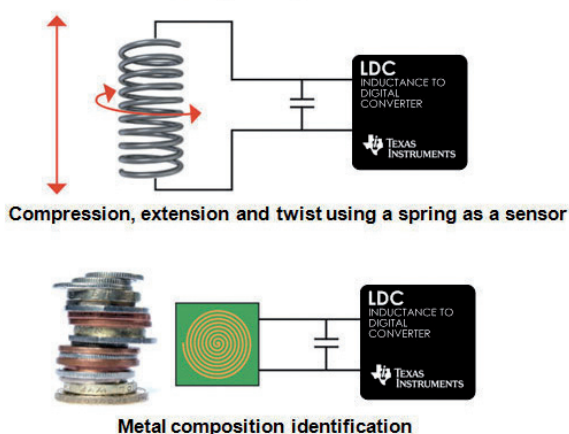


Fig. 6 - Rilevamento della deformazione della bobina sagomata a molla (in alto) e di metalli differenti in una pila di monete.

mezzi di trasporto, sostituendo i sistemi a raggi X attualmente in uso. Ma non solo: il sistema è in grado di rilevare la deformazione di una molla e quindi si presta a rilevare pressione e forza; inoltre può distinguere metalli di differente natura (Fig. 6). Tra le applicazioni ipotizzabili, valutando le caratteristiche dell'LDC1000, cercametri in grado di distinguere i metalli preziosi da quelli di scarso valore, oppure discernere e separare, in un sito di recupero, i vari metalli

per avviarli alla lavorazione. Il sensore di TI può essere usato da solo (è il caso del metal-detector e del rilevamento di oggetti singoli) ovvero in schiera: in questo caso, l'opportuna disposizione di più elementi su due o tre piani permette di compiere analisi tridimensionali e la verifica della consistenza e integrità di particolari meccanici, in luogo, come già detto, delle tradizionali tecnologie di analisi. Tra i principali vantaggi della tecnologia LDC:

- risoluzione sub-micrometrica nelle applicazioni con impedenza di risonanza a 16-bit e valori di induttanza a 24-bit;
- maggiore affidabilità, perché il rilevamento è senza contatto e immune agli inquinanti non conduttivi quali olio, sporcizia e polvere, che accorciano la vita dei sensori tradizionali;
- maggiore flessibilità, grazie alla possibilità di posizionare il sensore a distanza dall'elettronica di controllo;
- basso costo;
- possibilità illimitate, perché funziona anche con fogli di stagnola e inchiostri conduttivi, offrendo infinite possibilità di realizzare progetti innovativi e creativi;
- consumo limitato; l'LDC1000 consuma meno di 8,5 mW durante il funzionamento e meno di 1,25 mW in modalità standby.

L'integrato LDC1000 è disponibile in commercio incapsulato in contenitore SON a 16 pin 5x4 mm. Una versione per impiego nel settore automotive sarà disponibile nella primavera del 2014.

STRUMENTI E SUPPORTO

Per i progettisti, Texas Instruments mette a disposizione una evaluation board (LDC1000EVM, Fig. 7), che include un microcontrollore MSP430F5528 e che viene offerta a soli 29 dollari: si tratta di una scheda composta da sezioni separabili fisicamente, dotata di connessione USB (si inserisce nel PC in una porta USB) e che dispone a bordo di tutta l'elettronica necessaria, compresa una bobina già tarata per rilevare monete, oggetti metallici e la rotazione di un cursore. Insieme a questa scheda, TI fornisce nel kit un supporto meccanico che riporta tre cave

per inserire la bobina della scheda stessa per testare l'LDC1000 in tre applicazioni: il rilievo di diverse monete, la presenza di metallo e la rotazione di una manopola.

Quest'ultima viene rilevata semplicemente perché sotto la manopola è applicata una striscia metallica sottile da un lato e che va inspessendosi man mano che si arriva all'altra estremità; così, il sensore LDC può rilevare la posizione sulla base della quantità di metallo che si trova in corrispondenza della bobina (Fig. 5). Questa applicazione permetterà di valutare l'affidabilità dell'LDC nell'uso al posto di potenziometri o encoder, rispetto ai quali non ha parti meccaniche soggette ad usura.

La bobina in dotazione sulla evaluation board può essere staccata e separata dallo stampato, così da collegarne una personalizzata; a tal proposito, per aiutare nel dimensionamento e configurare l'LDC, Texas Instruments mette a disposizione il WEBENCH® Inductive Sensing Designer (www.ti.com/lids/ti/analog/webench/inductive-sensing.page?DCMP=sva-psp-ssp-ldc1000-en&HQS=sva-psp-ssp-ldc1000-pr-webench-en) cioè uno strumento on-line che consente di progettare la bobina semplice-

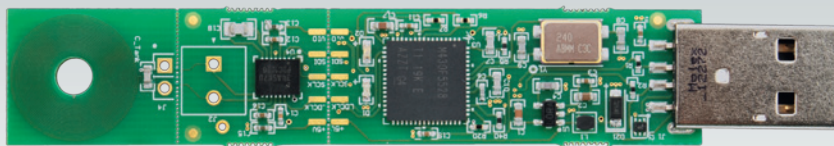


Fig. 7 - La scheda dell'evaluation kit fornito da TI.

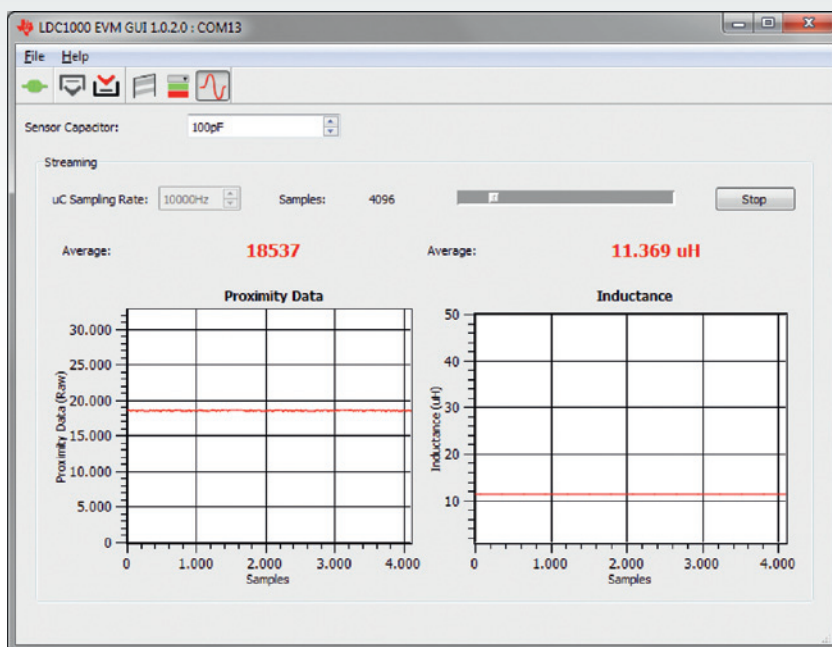


Fig. 8 - Schermata di lavoro del software di test per l'LDC1000.

mente specificando i parametri principali del sistema cui la si vuole applicare. Lo strumento semplifica il processo di progettazione della bobina e fornisce le impostazioni di configurazione per l'LDC in base ai requisiti

dell'applicazione e le esigenze del sistema. Il progetto può essere facilmente esportato in una varietà di programmi CAD per incorporare rapidamente la bobina del sensore nel proprio progetto.

RM ELETTRONICA

Forniture per hobbisti, scuole e industria

Distributore autorizzato:

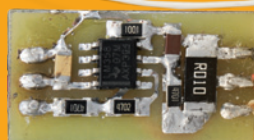
**FUTURA
ELETTRONICA**

**ELSE
Kit**

**NUOVA
ELETTRONICA**

Il tuo punto
di riferimento
per l'elettronica
a ROMA

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



Consente di monitorare i parametri di funzionamento dei modelli dinamici ad alimentazione elettrica, mediante un'unità fissa dotata di display LCD e una di bordo che si avvale di numerosi sensori. Seconda e ultima puntata.

TELEMETRIA REAL-TIME PER RADIOMODELLI

di PAOLO VASTA

Molti appassionati di radiomodellismo vogliono ormai dotare le proprie realizzazioni di telemetria, ossia di un sistema capace di verificare a distanza, in una postazione fissa, le condizioni operative del modello come ad esempio l'assorbimento di corrente dei motori, lo stato della batteria, la velocità, la temperatura ed altro ancora. Ciò vale sia per i radiomodelli terrestri, come le automobili (per simulare la telemetria usata nelle gare di F1) sia per gli aeromodel-

li, nei quali il controllo dell'assorbimento del motore e quello della carica residua della batteria assumono rilevanza fondamentale in quanto permettono di capire qual è l'autonomia residua di volo e far rientrare il velivolo prima che precipiti o vada incontro ad un atterraggio senza propulsione. Nel caso delle automobili, il controllo dell'autonomia residua serve a utilizzare al meglio la batteria nelle gare tra radiomodelli e quello della temperatura si può applicare al motore per evitare di

farlo riscaldare troppo. Nel fascicolo n° 177 abbiamo iniziato la descrizione di un sistema di telemetria realizzabile da chiunque, dedicato ai modelli a propulsione elettrica; con qualche modifica, la nostra soluzione potrebbe adattarsi ai modelli con motore termico: ad esempio per verificare il consumo si potrebbe realizzare un adattatore con rivelatore di passaggio del combustibile in grado di fornire un segnale elettrico al sensore 200A.

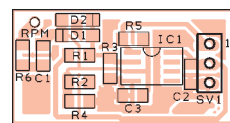
[piano di montaggio **SENSORI DI GIRI**]

Elenco Componenti:

C1, C2: 100 nF ceramico (1206)
 C3: 10 nF ceramico (1206)
 IC1: LM358
 R1÷R5: 47 kohm (1206)
 R6: 470 kohm (1206)
 R7: 100 kohm (1206)
 D1, D2: 1N4148 (DO214AC)

Varie:

- Connettore a crimpare 3 poli passo 2,54 mm Femmina
- Filo 15 cm giallo
- Filo 15 cm rosso
- Filo 15 cm marrone
- Circuito stampato



[piano di montaggio **SENSORE BEC**]

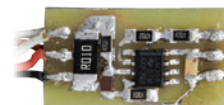
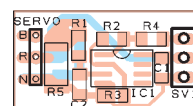
Elenco Componenti:

C1: 100 nF ceramico (1206)
 C2: 1 µF poliestere (1206)
 C3: 100 nF ceramico (1206)
 IC1: LM4558D (SO-8)
 R1: 4,7 kohm (1206)
 R2: 47 kohm (1206)
 R3: 1 kohm (1206)
 R4: 4,7 kohm (1206)

R5: 0,01 ohm (2010)

Varie:

- Connettore a crimpare 3 poli passo 2,54mm femmina
- Filo 15cm giallo
- Filo 15cm rosso
- Filo 15cm marrone
- Circuito stampato



Comunque, dopo una breve parentesi, in queste pagine riprendiamo e concludiamo il discorso descrivendo le fasi del montaggio e l'utilizzo del sistema.

REALIZZAZIONE PRATICA

Prima di costruire i singoli

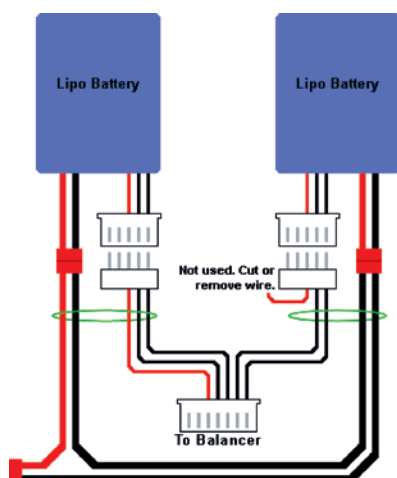


Fig. 1 - Connessione del connettore di bilanciamento del Modulo Sensori a due batterie LiPO 2s.

elementi costituenti il sistema di telemetria, bisogna decidere da quante parti sarà composto, ossia che sensori e accessori applicare al Modulo Sensori; la configurazione minima è composta, per l'unità remota (quella sul radiomodello) dal Modulo sensori, dal circuito BEC e dal sensore di corrente 200A, nonché dal modulo radio, senza il quale non può esserci telemetria. Il sensore BEC è indispensabile perché porta corrente dall'uscita di servizio dell'ESC al resto del sistema, ovvero al c.s. del Modulo Sensori; è necessario anche il sensore 200A, perché si presume che il radiomodello abbia almeno un motore. La costruzione di ciascuna parte richiede un circuito stampato, facilmente realizzabile per fotoincisione scaricando le tracce lato rame dal nostro sito www.elettronica.in.it; per tutte (Modulo Sensori, Ricevitore base, sensori vari, sezioni

radio) trovate in questo articolo i piani di montaggio, che vi aiuteranno nella corretta disposizione dei componenti. Completati i sensori, dovete interconnetterli al circuito del Modulo Sensori secondo lo schema di collegamento che trovate in queste pagine; inoltre, il sensore di giri va collegato con il punto RPM ad una fase del motore brushless (non dovete connettere la massa al negativo del motore, in quanto il ritorno è assicurato dal fatto

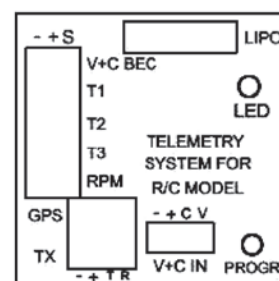


Fig. 2 - Disposizione dei collegamenti sul Modulo Sensori.

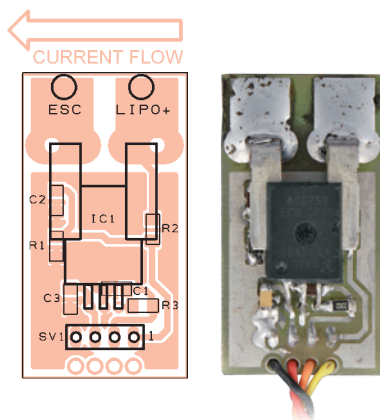
[piano di montaggio **SENSORE 200A**]

Elenco Componenti:

C1÷C2: 1 μ F poliestere (1206)
 C3: 100 nF ceramico (1206)
 IC1: ACS758ECB-200B-PFF-T
 R1: 4,7 kohm (1206)
 R2: 47 kohm (1206)
 R3: 10 kohm (1206)

Varie:

- Connettore a crimpare 4 poli passo 2,54mm Femmina
- Filo 15cm giallo
- Filo 15cm arancione
- Filo 15cm rosso
- Filo 15cm nero
- Circuito stampato



circuito, usate i connettori che trovate indicati in Fig. 2.

UTILIZZO DEL SISTEMA DI TELEMETRIA

Una volta assemblata l'elettronica e fissati i circuiti al telaio o al piano del radiomodello mediante fascette, nastro adesivo ecc., è il momento di configurare il tutto, iniziando dal trasmettitore RF dell'unità di bordo. Allo scopo si deve usare il pulsante di programmazione S1 del Modulo Sensori, che permette di cambiare il canale radio o visualizzare quello at-

[piano di montaggio **TRASMETTITORE RF**]

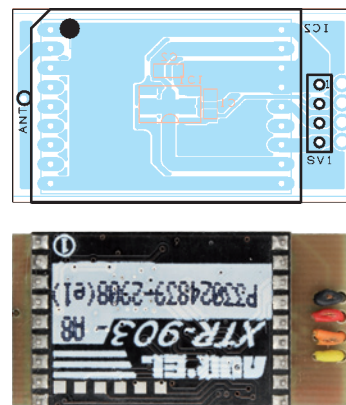
Elenco Componenti:

C1: 100 nF ceramico (1206)
 C2: 100 nF ceramico (1206)
 IC1: LD1117-3.3V (SO-08)
 IC2: XTR903-A8

Varie:

- Connettore a crimpare 4

- poli passo 2,54mm Femmina
- Filo 15cm giallo
- Filo 15cm arancione
- Filo 15cm rosso
- Filo 15cm nero
- Antenna accordata 868MHz
- Circuito stampato



che il Modulo Sensori ha la massa sul negativo della batteria, almeno se avete collegato tutto correttamente) mentre i sensori di temperatura possono essere montati su qualsiasi superficie di cui misurare la temperatura e vanno collegati ai connettori T1, T2 ecc. del Modulo Sensori.

Se volete tenere sotto controllo il bilanciamento della LiPO, dovete collegare con un apposito connettore volante il connettore di bilanciamento della batteria; per esempio, se usate due batterie 2S (a due celle) per formare una batteria 4S, il

connettore deve essere montato seguendo lo schema di Fig. 1. Una volta installati i sensori sul radiomodello, rimane da montare e collegare il trasmettitore, il quale va disposto in modo che l'antenna sia ben esposta; una buona posizione potrebbe essere la torretta degli ammortizzatori posteriori.

Anche l'eventuale modulo GPS va collocato in modo che sia esposto e che il lato componenti, contenente l'antenna, sia rivolto verso l'alto; diversamente non sarà possibile ricevere i segnali dei satelliti.

Per collegare i vari sensori al

tualmente usato. Per l'impostazione bisogna premere e mantenere premuto il tasto dando alimentazione al modulo RTX, allorché una serie di lampeggi del LED LD1 indicherà che è in corso la ricerca del canale libero, durante la quale il LED si spe-

Tabella 1

LAMPEGGI	CANALE RADIO
1	0
2	1
3	2
5	4
6	5
7	6
12	11

Elenco Componenti:

C1÷C3: 100 nF ceramico (1206)

C4, C6: 10 μ F poliestere (1206)

C5, C7: 100 nF ceramico (1206)

C8÷C14: 1 μ F poliestere (1206)

IC1: PIC18F4523 (TQFP44)

IC2: NCP511SN33 (SOT23-5)

IC3: 741G3157DBVR

R1, R2: 10 kohm (1206)

R3: 470 ohm (1206)

R4; R7, R9; R11; R13, R15: 4,7 kohm (1206)

R5, R5, R8, R10; R12, R17: 47 kohm

(1206)

R14: 10 kohm (1206)

R16: 10 kohm (1206)

S1: Microswitch

LD1: KA-3528ASYC

Varie:

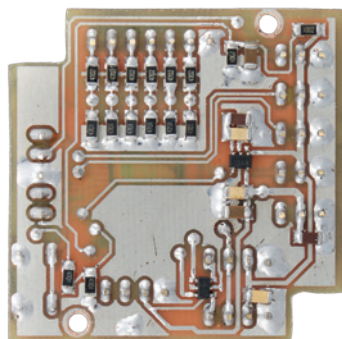
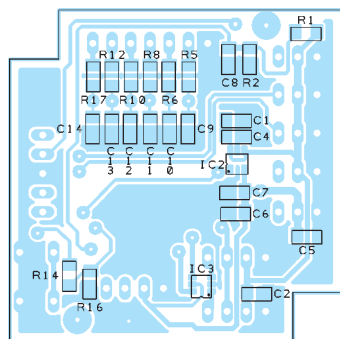
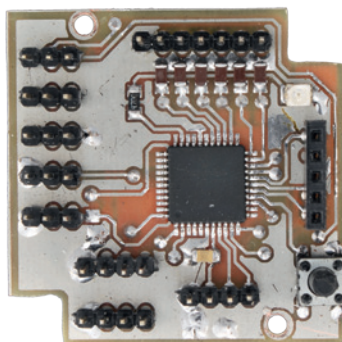
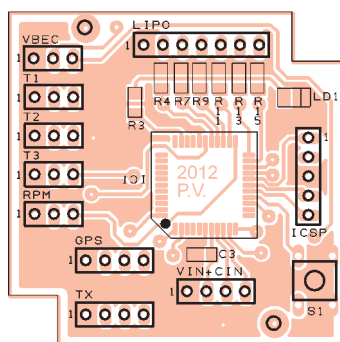
- Strip maschio 3 vie (5 pz.)

- Strip maschio 4 vie (3 pz.)

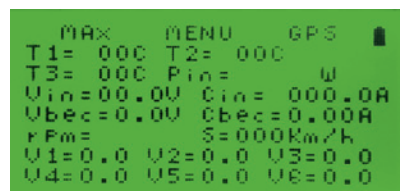
- Strip maschio 7 vie (1 pz.)

- Strip femmina 5 vie (2 pz.)

- Circuito stampato

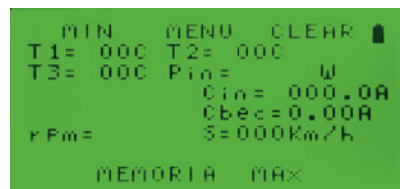


dopo la quale una nuova videata presenterà tutti i dati dei sensori rilevati sull'unità di bordo.



Notate che in questa schermata e nelle successive proposte, la prima riga indica le funzioni assegnate ai tre tasti: quello in linea con MAX sta per MAX, quello corrispondente a MENU attiva il menu, mentre quello vicino a GPS riguarda le funzioni GPS. MAX dà accesso alla schermata di memoria dei valori massimi rilevati dalla telemetria, mentre GPS apre la schermata che riepiloga i dati in arrivo dal ricevitore GPS.

Per "valori massimi" si intendono quelli di picco rilevati ad esempio dal BEC (corrente del motore e tensione di batteria), dal sensore 200A, da quello di giri (RPM) ecc.



Al riavvio della stazione base, tutti i valori massimi vengono azzerati; durante il funzionamento è possibile azzerare la memoria dei valori mediante il tasto abbinato alla funzione CLEAR (S1).

gnerà. Finita la ricerca, una serie di lampeggi comunicherà l'avvenuta impostazione del canale radio, secondo la **Tabella 1**. Durante il normale utilizzo la pressione di S1 permette di conoscere il canale attualmente in uso come da tabella.

Nel caso nello stesso campo operino più RTX uguali (ad esempio perché stiamo partecipando ad una gara tra radiomodelli tutti dotati dello stesso sistema di telemetria) la ricerca del canale li-

bero va effettuata con tutti questi moduli accesi.

CONFIGURAZIONE DELLA STAZIONE BASE

La stazione ricevente permette, usando i tre pulsanti e guardando le indicazioni del display, di configurare alcuni parametri della telemetria. Per prima cosa accendete l'unità, cosa che si fa mantenendo premuto per 5 secondi il tasto S3; apparirà la schermata di avvio (di seguito),

Elenco Componenti:

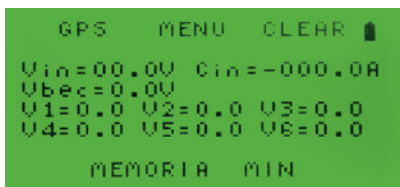
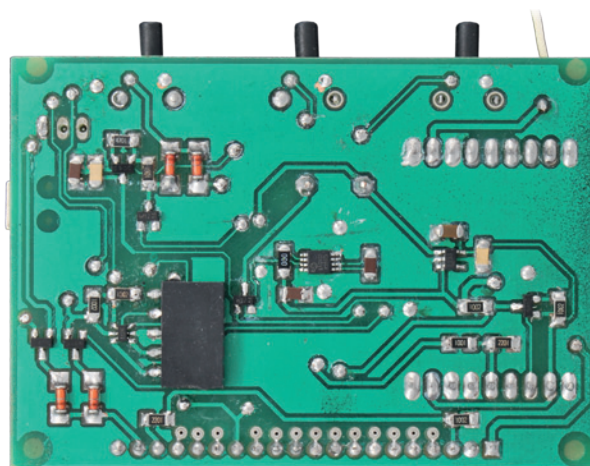
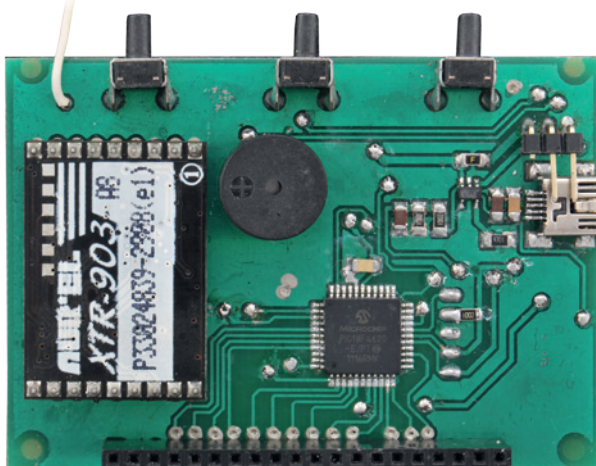
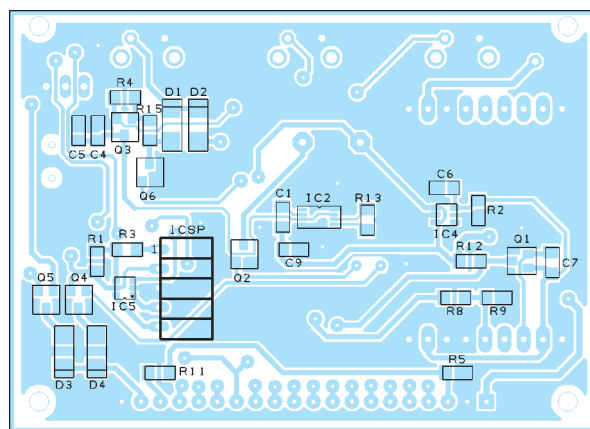
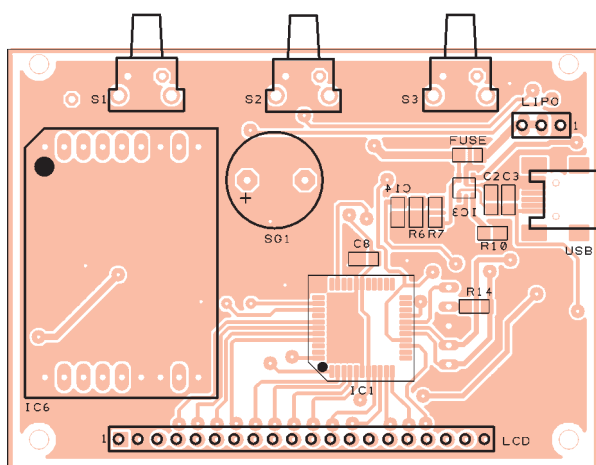
C1, C3, C5: 1 μ F poliestere (1206)
 C2, C4: 100 nF ceramico (1206)
 C6, C8: 100 nF ceramico (1206)
 C7, C10: 1 μ F poliestere (1206)
 C9: 10 μ F poliestere (1206)
 D1÷D4: ES2D (SMB)
 IC1: PIC18F4620 (TQFP44)
 IC2: MCP1253-33X50 (MSOP8)
 IC3: MCP73831 (SOT23-5L)
 IC4: NCP511SN33 (SOT23-5)
 IC5: SN74LVC1G332 (SOT23-6)
 IC6: XTR903-A8
 Q1: BSS123 (SOT-23)

Q2: MUN2212T1 (SC-59)
 Q3: BC857 (SOT-23)
 Q4÷Q6: MUN2214T1 (SC-59)
 R1, R2: 10 kohm (1206)
 R3: 1 kohm (1206)
 R4: 4,7 kohm (1206)
 R5÷R7: 10 kohm (1206)
 R8: 2 kohm (1206)
 R9: 1 kohm (1206)
 R10: 2 kohm (1206)
 R11: 2,7 kohm (1206)
 R12, R14: 10 kohm (1206)
 R13: 0 ohm (1206)

S1÷S3: Microswitch 90°
 LCD: Display Grafico (G12864)
 SG1: Buzzer F/HMB
 FUSE: Fusibile 500mA (0805)

Varie:

- Connettore Mini USB
- Batteria LiPo
- Antenna accordata 868MHz
- Strip maschio 3 poli 90°
- Strip maschio 20 poli
- Strip femmina 5 poli 90°
- Strip femmina 20 poli
- Circuito stampato

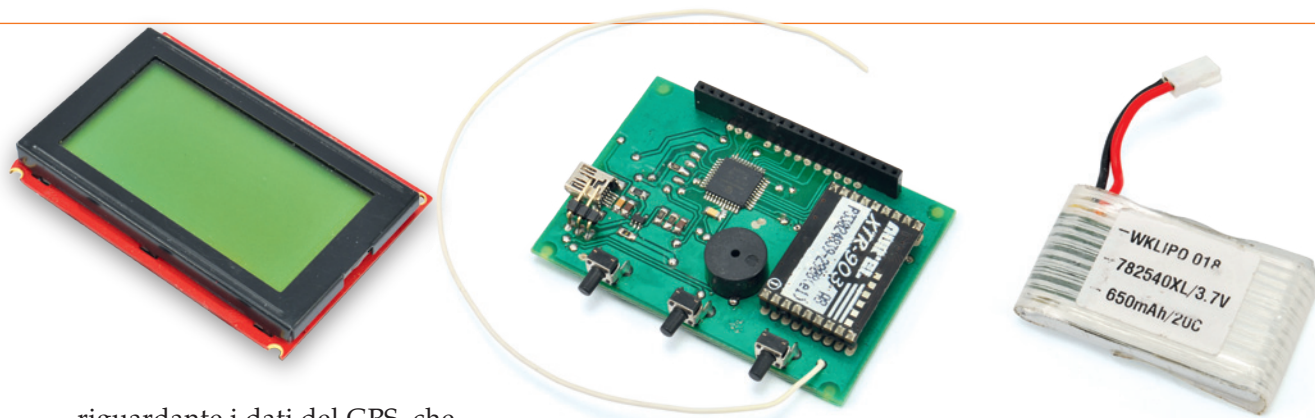


La pressione del tasto MIN

porta alla pagina successiva, la quale mostra la memoria dei valori minimi rilevati dai sensori di bordo, che possiamo resettare in ogni momento usando il solito pulsante corrispondente alla scritta CLEAR in prima riga.



La pagina successiva è quella



riguardante i dati del GPS, che mostra questi ultimi in tempo reale, così come il ricevitore li passa al microcontrollore del Modulo Sensori.

Il campo TIPO FIX indica con 1 se il ricevitore GPS non ha il fix dei satelliti, con 2 se ce l'ha e con 3 se dispone del fix 3D.

Premendo il tasto corrispondente alla voce PAGE1 in prima riga, il display torna alla pagina di avvio, contenente i valori attuali di telemetria.

Entrando nel menu da una pagina qualsiasi, è possibile operare le impostazioni del sistema.

```
CURSORE SELEZ. ESC
OFF * BUZZER ON
CERCA CH LIBERO
ALLARMIOFF
BATT : 4.0V
ILL. DISPLAY ON
COPPIE POLI 1
MENU
```

Con il tasto CURSORE ci si può spostare tra le varie voci, che vengono man mano evidenziate dal simbolo asterisco (*).

Il tasto SELEZ. permette di accedere alla voce selezionata dall'asterisco, mentre ESC riporta alla schermata di avvio. La prima voce di menu è quella di spegnimento (OFF): premendo il tasto in corrispondenza di SELEZ., il display mostrerà la schermata *Bye Bye* e l'unità si porterà in standby (il microcontrollore spegnerà il tutto mediante l'interdizione del Q6). Invece premendo il tasto CURSORE, si passerà ad evidenziare la voce BUZZER.

```
CURSORE SELEZ. ESC
OFF BUZZER ON *
CERCA CH LIBERO
ALLARMIOFF
BATT : 4.0V
ILL. DISPLAY ON
COPPIE POLI 1
MENU
```

Tramite il tasto SELEZ. possiamo impostare attivo o disattivo il segnalatore acustico, che normalmente segnala il raggiungimento di una soglia di allarme per un dato valore di telemetria; se durante il funzionamento del modello si presenta una situazione di allarme, essa viene sempre evidenziata da una segnalazione visiva, mentre la segnalazione acustica si ha solo se BUZZER è attualmente impostato ad ON. La successiva voce di menu è quella che permette di far cercare all'RTX un canale libero.

```
CURSORE SELEZ. ESC
OFF BUZZER ON *
CERCA CH LIBERO
ALLARMIOFF
BATT : 4.0V
ILL. DISPLAY ON
COPPIE POLI 1
MENU
```

Da essa si accede al menu di selezione del canale di trasmissione del Modulo Sensori; impartendo il relativo comando viene avviata la scansione dei canali, durante la quale il display mostra l'intensità (con un valore tra 0 e 9) del segnale sui possibili canali. Notate che 0 corrisponde a segnale nullo, mentre 9 alla massima intensità; normalmente il valore per una corretta connessione è compreso tra 4 e 6.

```
CH: 0 SEGNALE: 0
CH: 1 SEGNALE: 0
CH: 2 SEGNALE: 0
CH: 4 SEGNALE: 0
CH: 5 SEGNALE: 0
CH: 6 SEGNALE: 0
CH: 11 SEGNALE: 0
```

Una volta completata la scansione, il sistema sceglie e mostra il canale, tra quelli liberi, che presenta il miglior segnale; se ne volete impostare uno differente, premete il tasto corrispondente ad INC, mentre se accettate quello predefinito confermate. Alla prima accensione del sistema è consigliabile accettare il canale suggerito e successivamente programmare il Modulo Sensori. Se quest'ultimo venisse programmato prima della stazione base, essa troverebbe il canale occupato dalla portante dell'RTX sul Modulo Sensori e suggerirebbe di impiegare un canale differente, con il risultato che le due unità non riuscirebbero a comunicare.

Se comunque conosciamo il canale su cui trasmette il Modulo Sensori, possiamo impostare manualmente lo stesso canale sull'RTX della stazione base. In ogni caso, una volta selezionato il canale, con il tasto MEM lo si memorizza.

```
MEM ESC INC
TROVATO CANALE 0
CANALE SCELTO 0
```

Vediamo adesso l'impostazione degli allarmi, accessibile dall'omonima voce di menu; quando ALLARMI è OFF la funzione è disattivata, mentre se vale ON sul display appare la voce SOGLIE.

```
CURSORE SELEZ. ESC
OFF BUZZER ON
CERCA CH LIBERO
ALLARMI ON SOGLIE
BATT : 4.0V
ILL. DISPLAY ON
COPPIE POLI 1
MENU
```

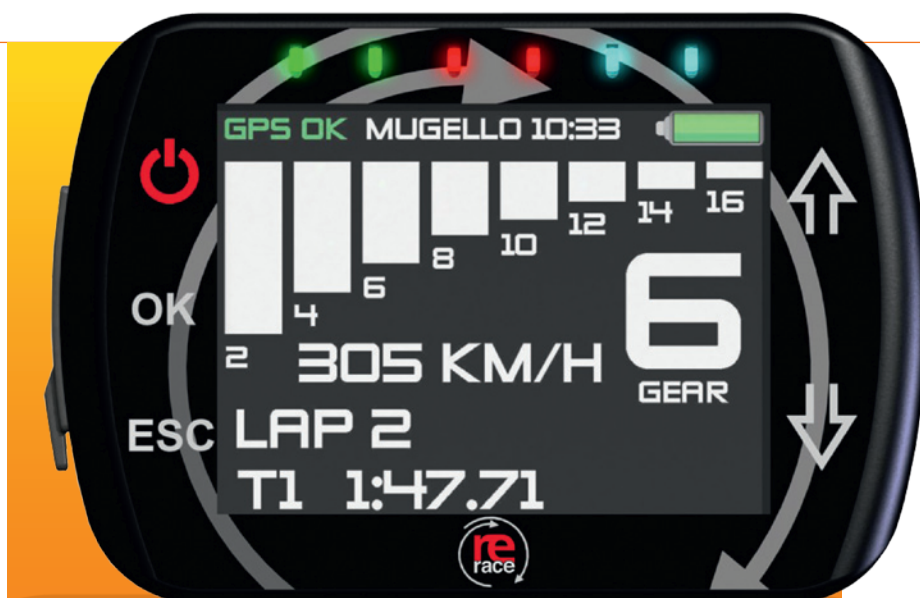
Selezionando SOGLIE (ci si arriva con il tasto CURSORE) si accede al menu di impostazione delle soglie d'allarme previste.

```
CURSORE SELEZ. ESC
OFF BUZZER ON
CERCA CH LIBERO
ALLARMI ON SOGLIE *
BATT : 4.0V
ILL. DISPLAY ON
COPPIE POLI 1
MENU
```

Notate che le soglie impostate automaticamente dal sistema corrispondono ai valori limite ammessi; per modificarle bisogna selezionare il valore di interesse mediante il tasto corrispondente a SEL e impostare i nuovi con il tasto MOD.

```
SEL MOD MENU
T1=130* T2=130
T3=130 Ubec=0.0
rPm=65000
Cin= 200.0
Cbec=9.99 Ucell=0.0
IMPOSTA SOGLIE ALLARM
```

Una volta entrati nella schermata di modifica, per cambiare un valore basta spostarsi con CURSORE sulla cifra da modificare e incrementare con INC; raggiunto il valore desiderato, con il tasto OK torniamo alla schermata precedente.



La telemetria va sul social network

Dedicato a chi ama correre in pista in moto, ReRace è un sistema di telemetria integrato con il mondo di Internet e dei Social Network: il dispositivo di bordo acquisisce i parametri del veicolo (come rollio e beccheggio della moto, angolo di piega o d'impennata, tempo sul giro, velocità attuale e media, accelerazione) e li memorizza su SD o li trasferisce al PC tramite WiFi. Permette poi di pubblicarli e condividerli via web sul nuovo social network ReRace Connect (appositamente creato da ReRace), così da confrontarli con amici e appassionati, sfruttando le informazioni sui vari circuiti contenute all'interno del grande database appositamente creato da ReRace (www.rerace.com). Il tutto è racchiuso in un'unità IMU collegata ad un ricevitore GPS a 10 Hz, abbinabile alla action-camera GoPro e adattabile ad automobili e go-kart.

```
CURSORE INC OK
T1=030
*
SOGLIA T1
```

Modificate le soglie che si intende cambiare, premendo il tasto MENU si torna alla schermata principale dei menu. Notate che una volta impostato il valore, esso resta memorizzato anche se la funzione ALLARMI dovesse essere disattivata. Vediamo adesso la gestione della

retroilluminazione del display, che si compie mediante la voce di menu ILL. DISPLAY.

```
CURSORE SELEZ. ESC
OFF BUZZER ON
CERCA CH LIBERO
ALLARMI ON SOGLIE
BATT : 4.0V
ILL. DISPLAY ON *
COPPIE POLI 1
MENU
```

Spostandosi con CURSORE sul valore asteriscato potete modificarlo in ON (retroilluminazione accesa) oppure OFF (retroilluminazione spenta).


```

CURSORE SELEZ.  ESC
OFF BUZZER ON
CERCA CH LIBERO
ALLARMION SOGLIE
BATT : 4.0V
ILL. DISPLAY ON
COPPIE POLI 1 *
MENU

```

Dopo l'impostazione dell'attività della retroilluminazione, nel menu troviamo la voce COPPIE POLI, il cui valore è necessario all'algoritmo di calcolo dei giri del motore; infatti, come spiegato a proposito del sensore RPM dell'unità remota, il microcontrollore desume i giri compiuti dal motore dalla frequenza della tensione di una fase del brushless.

Naturalmente, il calcolo richiede di conoscere il numero delle paia di poli, dato che più sono queste, più il motore gira velocemente a parità di frequenza della tensione trifase di comando.

Il valore di imposta tramite il tasto SELEZ., che permette solo l'incremento; raggiunto il massimo, un'ulteriore pressione fa ricomparire il valore minimo. Dopo la voce COPPIE POLI, il menu prosegue in una seconda pagina, la cui prima voce è SENSORI INST.

```

CURSORE SELEZ.  ESC
SENSORI INST.  *
DATI RX
ORA SOLARE
MENU

```

Tramite la relativa funzione si possono definire i sensori installati a bordo veicolo, in modo da togliere dalla schermata della telemetria (avvio) i valori dei sensori non installati o dei quali, pur essendo installati, non ci interessa conoscere lo stato.

```

SEL  ON/OFF  ESC

GPS ON *  Ucell ON
T1 ON    T2 ON
T3 ON    Uin+Cin ON
Cbec ON  RPM ON

```

Nella relativa schermata, con il tasto SEL ci si sposta tra i sensori e con ON/OFF abilitiamo/disabilitiamo di volta in volta quello selezionato. Con il tasto ESC si torna alla pagina precedente. Notate che l'unica voce non eliminabile è quella riguardante la tensione di alimentazione del Modulo Sensori. Torniamo adesso al menu e vediamo la voce DATI Rx, la quale fornisce i dati relativi al canale radio in uso e alla potenza del segnale in ricezione.

```

CURSORE SELEZ.  ESC
SENSORI INST.
DATI RX *
ORA SOLARE
MENU

```

Da qui, la pressione del tasto ESC riporta alla pagina precedente. Infine, l'ultima voce del menu è quella che permette di modificare l'ora, ossia impostare l'ora legale o quella solare; la selezione si opera con il pulsante SELEZ., che ad ogni pressione inverte l'impostazione visualizzata.

```

CURSORE SELEZ.  ESC
SENSORI INST.
DATI RX
ORA SOLARE *
MENU

```

Durante il normale utilizzo del sistema, se sono stati impostati degli allarmi e un valore supera la soglia corrispondente, la schermata di avvio diventa come segue e il cicalino, se attivato, emetterà una nota acustica.

```

MAX MENU ALLARM
T1= 010
Ubec=5.0V

```

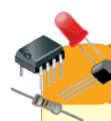
A questo punto, tramite il tasto corrispondente ad ALLARM è possibile accedere alla schermata degli allarmi, dalla quale si può vedere qual è stato l'allarme che si è verificato, ovvero il valore andato oltre la soglia definita.

```

PAGE1 MENU OFF
T1=130A T2=130
T3=130 Ubec=0.0
rPm=65000
Cin= 200.0
Cbec=9.99 Ucell=0.0
MEMORIA ALLARMI

```

I sensori che hanno dato allarme sono evidenziati da una lettera A posta dopo il valore della soglia; per tornare alla schermata principale è sufficiente premere PAGE1 oppure per escludere la funziona "allarmi" è possibile premere il tasto OFF. Tornando alla PAGE1 gli eventuali indicatori di allarme (buzzer o retroilluminazione) cesseranno non appena i valori rientrano nella normalità. ■



per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono di facile reperibilità. I master dei circuiti stampati ed il firmware possono essere scaricati dal sito della rivista www.elettronica.in.it.

I moduli XTR903-A8 sono reperibili presso Futura Elettronica al prezzo di 38,00 Euro cadauno 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>

RICONOSCIMENTO GESTUALE *TOUCH FREE*, RIDUCE LA POSSIBILITÀ DI DISTRAZIONE PER CHI GUIDA



..... di FRANCESCO DONI

Anche nelle applicazioni in auto, la tecnologia di riconoscimento gestuale GestIC di Microchip migliora la sicurezza e rende più piacevole la guida.

I produttori di autovetture hanno da sempre dovuto affrontare la sfida di minimizzare le distrazioni di chi guida, fornendo allo stesso tempo un'intuitiva interfaccia uomo-macchina (Intuitive Human-Machine Interface, HMI) al crescente numero di funzionalità e tecnologie presenti all'interno dei veicoli. Una nuova soluzione di Microchip Technolo-

gy, basata sul chip con tecnologia GestIC® (chiamato MGC3130 3D Tracking and Gesture Controller), insieme alla suite software Colibrì, ha le potenzialità per cambiare significativamente le interfacce HMI dell'automotive, utilizzando comandi gestuali anziché tattili. L'MGC3130 è il primo controller gestuale 3D al mondo ad utiliz-

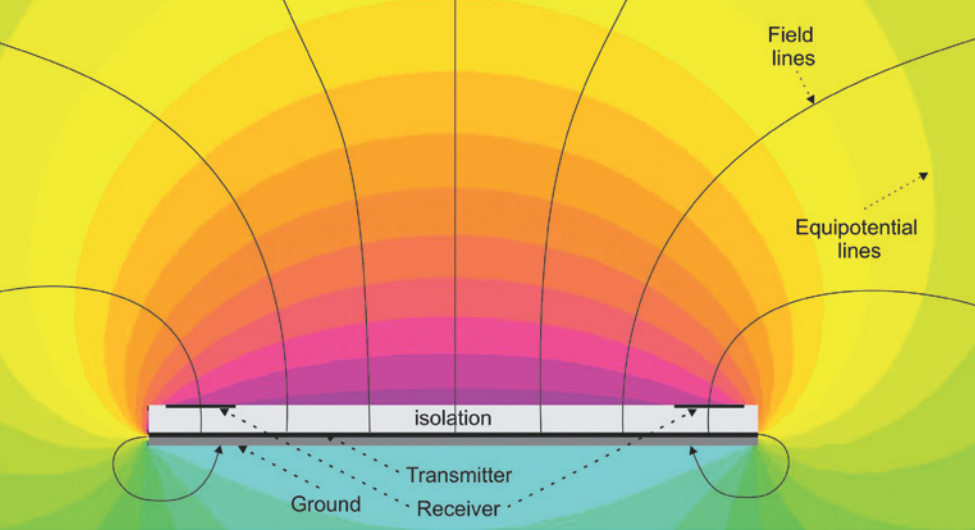


Fig. 1

ciechi, specialmente in ambienti come i veicoli dove gli utilizzatori siedono molto vicini al sensore. Di contro, è difficile per le soluzioni basate su telecamera eliminare falsi rilevamenti provenienti dai passeggeri per le funzioni specifiche del guidatore. La tecnologia GestIC® fornisce anche una uscita dati con una frequenza di refresh di 200 Hz, consumando nel contempo fino al 90% in meno di potenza rispetto alle soluzioni basate su sensore video.

LA FISICA SOTTESA

Il setup del sensore básico è descritto nella **Fig. 1**: un livello isolato separa un singolo elettrodo trasmettitore full-plane posizionato su un livello di terra dagli elettrodi ricevitori posizionati sul livello superiore. Controllato dall'MGC3130, l'elettrodo trasmettitore genera un campo elettrico con una frequenza di 100 kHz. Senza alcun disturbo esterno, questo campo elettrico è simile a quello uniformemente distribuito mostrato in **Fig. 1**. Qualora un oggetto entrasse in questo campo elettrico, questo disturberebbe le linee di campo. Un tipico esempio di questo disturbo è mostrato in **Fig. 2**.

A causa dei disturbi, la precedente distribuzione di linee equipotenziali uniformi, come pure di linee di campo, è significativamente mutata. Entrando nel campo elettrico, l'oggetto conduttore (nei casi in studio, una mano) assorbe il campo elettrico dato che conduce le cariche elettriche a terra, producendo un cambiamento sia del campo elettrico che delle linee equipotenziali. L'MGC3130, misura le minuscole deviazioni di segnale sugli elettrodi ricevitori generati dalla mano ed elabora i risultati. Più la mano si avvicina all'elettrodo,

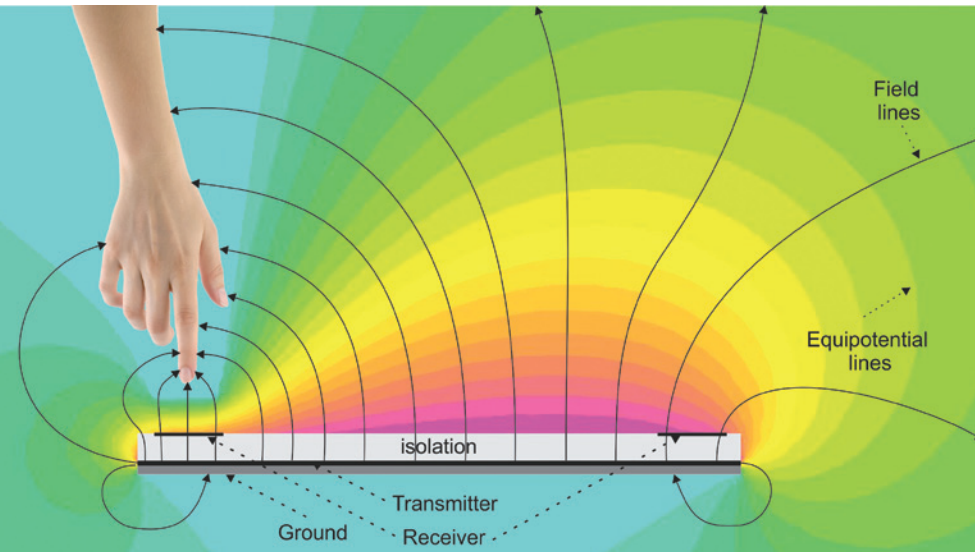


Fig. 2

zare campi elettrici (E-fields) per il rilevamento della posizione delle mani con riconoscimento 3D della gestualità, ed offre diversi vantaggi in particolare per l'ambiente automotive. Poiché l'MGC3130 rileva solo variazioni in prossimità dell'E-field causati da oggetti conduttori, come il corpo umano, esso è resistente alle influenze ambientali, come luci e suoni. Inoltre, poiché il suo raggio d'azione è di 15 cm, l'MGC3130 può garantire che solo i gesti degli utenti previsti siano rilevati, prevenendo in questo modo falsi rilevamenti di funzioni e comandi (riservati all'autista) provenienti invece dai passeggeri. Di converso, non ci sono angoli ciechi all'interno dei 15 cm, diversamente da altre tecnologie gestuali. Un altro grande vantaggio della tecnologia GestIC

è che essa utilizza elettrodi per il rilevamento che possono rimanere nascosti utilizzando il normale rivestimento Indium Tin Oxide (ITO) del display o che possono essere integrati dietro l'alloggiamento del dispositivo. Ciò è particolarmente importante in applicazioni automobilistiche; se confrontata con altre tecnologie gestuali 3D, come ad esempio l'infrarosso, gli ultrasuoni o soluzioni basate su sensori video, la tecnologia GestIC offre svariati vantaggi aggiuntivi per applicazioni Automotive. Per esempio, le soluzioni basate su sensore video necessitano di una certa quantità di luce per un corretto funzionamento richiedendo nel contempo una compensazione dinamica di luminosità. Oltre a ciò, una telecamera ha un angolo di visuale fisso, il che crea angoli

maggiore sarà l'influenza locale della stessa. Utilizzando quattro elettrodi ricevitori (Nord, Sud, Est e Ovest) l' MGC3130 consente il tracciamento delle posizioni della mano sugli assi X, Y e Z all'interno dell'area di rilevamento. In una fase successiva di elaborazione, l' MGC3130 identifica i gesti attraverso l'applicazione di un Hidden Markov Model (HMM) basato su un motore di riconoscimento gestuale che consente un riconoscimento eccezionalmente elevato ed indipendente dall'utente. Quando, per esempio, la mano scatta (flick) da destra a sinistra, inizialmente c'è una deviazione del segnale sull'area di destra. Questa deviazione di segnale sull'elettrodo destro decresce durante il movimento, mentre la deviazione di segnale sull'elettrodo di sinistra simultaneamente aumenta. Modelli di movimenti come questo vengono riconosciuti ed elaborati on-chip, e offerti in una modalità predefinita in uscita. La risoluzione di tracciamento X/Y/Z raggiunge i 150 dpi, a seconda dello schema e tipologia degli elettrodi e della posizione della mano. Più vicina è la posizione della mano rispetto all'area di rilevamento, maggiore è il rapporto segnale-rumore e maggiore risulta la risoluzione. La tecnologia GestIC consente questa precisione simil-mouse senza, praticamente, alcuna instabilità. Per scopi dimostrativi, il team di progettazione ha controllato il cursore di un notebook senza alcuna instabilità, solo muovendo la mano e senza usare il mouse.

GLI ELETTRODI

Gli elettrodi di ricezione sono sempre posizionati sopra l'elettrodo trasmettitore (Fig. 3). Per esempio, nell'installazione dimostrativa che Microchip ha

presentato a Electronica l'anno scorso, gli elettrodi ricevitori consistevano in uno strato di rame sulla faccia superiore di un PCB. Gli elettrodi possono essere realizzati usando qualsiasi tipo di materiale conduttore solido come PCB solidi, flexible printed circuit board (FPC), elettrodi LDS (laser direct structured), lamine conduttrici, ed il sopra menzionato strato ITO già presente nei display.

La tecnologia GestIC di Microchip è in grado di funzionare con sensori sottili che consentono una integrazione praticamente invisibile dietro l'alloggiamento del dispositivo target, senza aumentare lo spessore complessivo del prodotto industriale progettato. Pertanto, gli elettrodi non sono solo a basso costo ma anche a basso impatto in termini di schema generale. Tutto ciò è di grande importanza nell'elettronica automotive, dove questi elettrodi possono essere nascosti dietro aree come il cruscotto.

Come precedentemente specificato, l'utilizzo di strutture conduttive preesistenti, come il rivestimento ITO di un pannello

tattile di visualizzazione, fa della tecnologia GestIC una soluzione molto conveniente. Attualmente, Microchip sta collaborando con i maggiori produttori di display, sulla pre-integrazione della tecnologia GestIC in un modulo display completo. Collegando l'MGC3130 al rivestimento ITO, l'area tattile del display viene trasformata in un elettrodo sensore di campo elettrico senza disturbare le funzionalità multi-touch del sottostante touch display. A causa di questa perfetta integrazione, la tecnologia GestIC avvia la terza dimensione del rilevamento non appena le dita sono sollevate dalla superficie del display. Nelle applicazioni automotive, questo permette al sistema di visualizzare differenti elementi in base alla direzione dalla quale la mano si avvicina. Mentre un approccio verticale potrebbe attivare il menu telefonico, quello dal lato basso di sinistra potrebbe attivare il setup del sistema di navigazione. Un'altra possibilità è il passaggio tra i vari menu base sfogliando a gesti nell'aria.

Standard Electrode Design

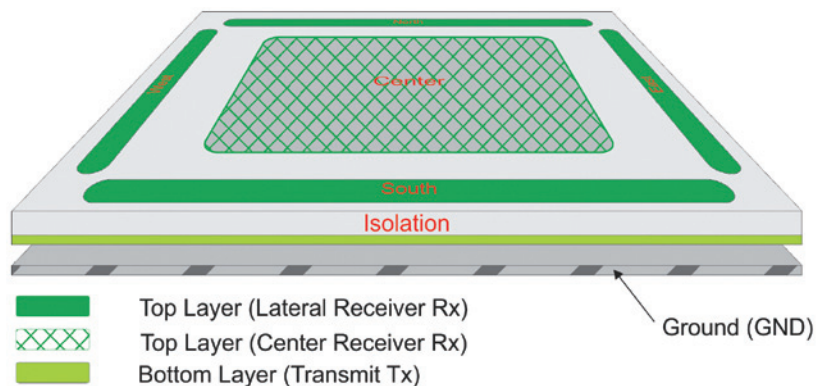
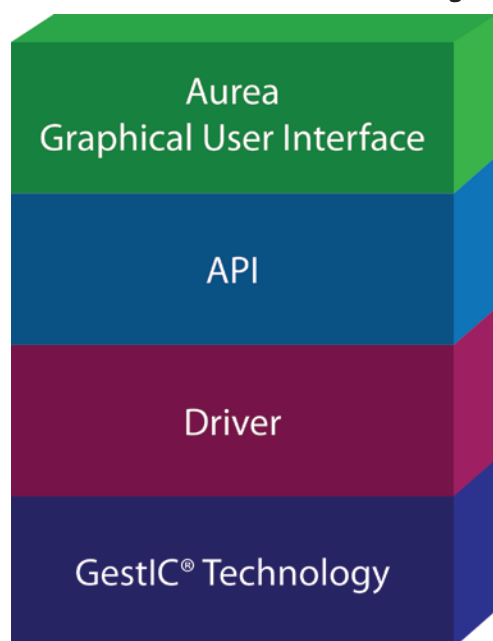


Fig. 3

Fig. 4



La tecnologia GestIC è molto flessibile, dato che non solo riconosce gesti lineari ma anche simboli gestuali, gesti circolari e altri. Nell'ambiente automobilistico, questa capacità potrebbe essere utilizzata al fine di aumentare/diminuire il volume del sistema di infotainment, disegnando una manopola del volume virtuale in aria. A prescindere dalla funzione assegnata ad un particolare gesto, il risultato finale è che il guidatore può prestare maggiore attenzione alla strada se tutto quello che deve fare è gesticolare in aria, invece di toccare un preciso tasto sullo schermo.

IL CHIP MGC3130

L'MGC3130 è un controller mixed-signal configurabile consistente in un front-end analogico con un canale trasmettitore e cinque ricevitori e una unità Signal Processing Unit (SPU) digitale. Quattro di questi cinque canali vengono usati per riconoscere gesti e posizione delle mani, mentre il quinto canale consente il rilevamento tattile e migliora la precisione a breve distanza.

Ogni canale subisce condizioni di segnale. I segnali analogici pre-condizionati vengono quindi digitalizzati e processati dalla SPU integrata.

All'uscita, la SPU fornisce i risultati elaborati via interfaccia I2C o SPI. Microchip fornisce una API (application programming interface) che può funzionare sul controller dell'applicazione o dell'host. Questa API consente ai progettisti di mappare facilmente i segnali verso il target di destinazione.

Ciò significa che i progettisti non debbono preoccuparsi del condizionamento dei segnali, dato che Microchip pre-processa i dati sulle posizioni X/Y/Z della mano, così come offre il completo software di riconoscimento gestuale on-chip, ovvero la suite Colibri. Al fine di consentire ai progettisti di creare speciali funzionalità personalizzate attraverso il post-processing di specifiche applicazioni, Microchip fa transitare attraverso l'elettrodo filtrato anche segnali verso l'uscita.

LA SUITE SOFTWARE COLIBRI

La suite Colibri utilizza un motore di riconoscimento gestuale basato su un Hidden Markov model (HMM), in abbinamento a vettori X/Y/Z di posizione della mano. HMM offre la più elevata frequenza di riconoscimento user-independent per gestualità 3D di mani e dita.

Ciò significa che la tecnologia GestIC fornisce coerentemente una frequenza di riconoscimento gestuale eccezionalmente elevata, a prescindere da chi stia guidando il veicolo.

Sull'uscita digitale dell'MGC3130, la suite Colibri offre dati di tracciamento della posizione X/Y/Z della mano di elevata risoluzione come flick, cerchi e parole gestuali. Per i flick, la suite Colibri

non solo rileva i movimenti base, come sinistra, destra, su e giù, ma anche i flick più sofisticati, come da dentro verso fuori e viceversa, attraverso l'intera o parziale area di copertura.

L'utente è così in grado di attuare comandi come "apri applicazione", seleziona, click, zoom, scorri (scroll), mouse-over e molti altri, senza bisogno di toccare il dispositivo.

Se il progetto richiede gesti specifici che non sono inclusi nella suite, Microchip fornisce in breve tempo un registratore gestuale e modulo di addestramento che consente agli sviluppatori di aggiungere i loro specifici gesti alla libreria.

Grazie all'architettura flash dell'MGC3130, i progettisti possono fare il download direttamente nell'IC dei parametri di riconoscimento per qualsiasi nuovo gesto.

Inoltre, il rilevamento di avvicinamento è una funzione programmabile che esegue una scansione delle attività dell'utente mentre il chip è in modalità self-wake-up.

Se viene rilevata una interazione da un utente reale, il sistema commuterà automaticamente in "full sensing mode" tornando alla modalità self-wake-up a bassissimo consumo una volta che la mano dell'utente abbia lasciato l'area di rilevamento.

ADATTAMENTO FLESSIBILE ALL'AMBIENTE

La tecnologia GestIC funziona ad una frequenza di circa 100 kHz. L'MGC3130 adatta la frequenza di lavoro ad eventuali disturbi esterni provenienti da motori, inverter, caricatori e lampade fluorescenti, cambiando automaticamente la frequenza e spostandosi su un canale privo di rumore nel range compreso tra 70 e 130 kHz,

così da evitare interferenze RF, costituendo una soluzione molto efficace. D'altro canto, l'energia emessa è molto bassa.

Questo significa che la tecnologia GestIC non causa interferenze ad altri sistemi o elementi, come evidenziato dal fatto che ha superato vari test, come l'IEC 61000-4-3. Inoltre, mentre il consumo massimo di corrente dell'MGC3130 è di 70 mW, la maggioranza del suo consumo di corrente è necessaria per l'interpretazione, valutazione e classificazione dei segnali del campo elettrico disturbato, non per l'emissione di un campo elettrico in sé.

Dietro questa tecnologia c'è una storia molto lunga.

Per oltre 10 anni, la società Ident Technology con base a Monaco di Baviera (Germania) - che fu acquisita da Microchip Technology nel 2012 e integrata nella propria Human Machine Interface Division (HMID) - ha acquisito esperienza e know-how riguardo il tema dell'uso dei campi elettrici, e ha registrato diversi brevetti. Questo know-how include principalmente l'applicazione delle teorie dell'E-field in un ambiente reale, la progettazione di elettrodi di rilevamento, la progettazione dei chip, e gli algoritmi usati

per elaborare i dati grezzi del rilevamento.

La lunghezza dell'onda del segnale a 100 kHz usato nella tecnologia GestIC equivale a circa 3 km.

Questo significa che le dimensioni dell'area dell'elettrodo di rilevamento, che è tipicamente inferiore a 15 cm x 15 cm, è di svariate volte più piccola dell'ordine di grandezza della lunghezza d'onda del segnale emesso.

Questo consente di ottenere un E-field molto stabile, quasi statico durante il funzionamento, che può essere usato per rilevare oggetti conduttivi come ad esempio il corpo umano, mentre la componente magnetica è praticamente zero e non si verifica alcuna propagazione d'onda.

Pensando alle future applicazioni, e al fine di facilitare progetti applicativi, il team HMID di Microchip sta attualmente preparando un documento che descrive tutti i fattori rilevanti che debbono essere presi in considerazione durante la progettazione e il posizionamento degli elettrodi di rilevamento.

Questo consentirà agli sviluppatori automotive di progettare i propri elettrodi o implementarli utilizzando materiali già pre-

senti nei loro progetti. A partire da maggio, Microchip ha reso disponibile un kit di sviluppo chiamato Hillstar, che è destinato a supportare i clienti durante la fase di design. Hillstar utilizza la tecnologia GestIC, e include la suite Colibri, e si connette ad un PC attraverso una interfaccia USB, consentendo così agli ingegneri di connettere facilmente i loro schemi di elettrodi e parametrizzare il chip MGC3130 mediante PC.

La Graphical User Interface (GUI) di Microchip, ricca di funzionalità, chiamata Aurea, funziona su sistema operativo Windows 7 e offre il controllo di parametri e settaggi dell'MGC3130, rendendo facile l'aggiornamento e il salvataggio dei parametri stessi (Fig. 4).

Un video informativo può essere reperito all'indirizzo http://www.microchip.com/pagehandler/en-us/technology/gestic/gettingstarted.html#oid=YzM2J2NjqvN4ridRUOLzUItg0_urPvXZ.

Maggiori dettagli si trovano sulla seguente pagina web: www.microchip.com/gestic, mentre www.gesture-remote.com e www.gesture-cube.com offrono suggerimenti riguardo le possibilità di questa nuova tecnologia Microchip. ■

Elettronica In

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

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

COLLABORATORI HARDWARE della nostra Rivista

Realizzare un progetto per 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

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

RIPARTE LA STAGIONE DELLA ROBOTICA

Una lunga stagione, a cavallo tra la fine di questo 2013 e il prossimo 2014, attende gli appassionati di robotica: momenti di robotica educativa, fiere ed esposizioni per studenti, appassionati e professionisti, riempiranno i mesi a venire proponendo novità e stimolanti iniziative.



di DAVIDE SCULLINO

eu
Robotics
Week



Come ogni autunno, torna la rassegna degli eventi dedicati alla robotica nei suoi molteplici aspetti: non solo fiere e convegni che mostrano lo stato dell'arte, ma anche e sempre più momenti dedicati al fai da te tecnologico e ai makers; ma robotica significa anche eventi didattici rivolti agli studenti e in generale ai giovani, con gare in cui i partecipanti possono mettere a frutto la propria conoscenza e sfogare la propria passione.

Tra le competizioni, da segnalare sicuramente FLL Italia, edizione nostrana dell'affermata First LEGO League, a cura del Museo Civico di Rovereto (TN), le cui iscrizioni sono aperte fino al 20 novembre. La competizione che il network FLL propone quest'anno, prende il nome di Nature's Fury (ossia "furia della natura") e si svolgerà col seguente calendario.

Per le qualificazioni regionali:

- per il Centro Italia, il 18 gennaio

- 2014 a Pistoia;
- per il Nord-Ovest, il 26 gennaio 2014 a Cuneo;
- per il Nord-Est, il 2 febbraio a Udine;
- per il Sud Italia, il 4 o 6 febbraio (da confermare) a Catania.

Per la finale nazionale: il 14-15 marzo 2014 a Rovereto (Trento). I team selezionati potranno accedere all'edizione mondiale -World Festival- in programma dal 23 al 26 aprile 2014 a St. Louis (U.S.A.) e agli Open European Championship (dal 28 al 31 maggio 2014) previsti a Pamplona (Spagna).

Le competizioni locali, nazionali e internazionali sono aperte a ragazzi di età compresa tra 9 e 16 anni (anno di nascita compreso tra il 1997 e il 2004) organizzati in squadre con un "allenatore", che devono preventivamente iscriversi scaricando il modulo di adesione dal sito web del Museo Civico di Rovereto (il modulo va inviato a fll@museocivico.rovereto.tn.it entro il 20 novembre).

Il nome Nature's Fury non è casuale ma è attinente allo scopo della gara: infatti alle squadre è richiesto di identificare una comunità che potrebbe subire una possibile calamità naturale, identificare un problema che va affrontato quando si verifica il disastro e sviluppare una soluzione innovativa

che aiuti le persone a prepararsi all'evento, a rimanere al sicuro o a ricostruire. Le squadre devono anche condividere con gli altri e diffondere informazioni sul problema e su come intendono risolverlo.

Il tutto, utilizzando

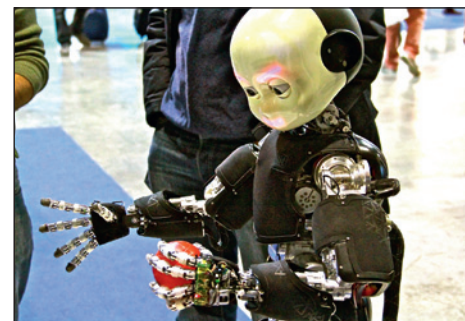
la piattaforma robotica Mindstorms di LEGO e accettando di seguire i valori cui si ispira la competizione.

Le sfide proposte dal torneo FLL nascono per incitare i ragazzi a ragionare come veri e propri scienziati; le squadre dovranno costruire, testare e programmare un robot autonomo usando la tecnologia LEGO Mindstorms per risolvere una serie di "missioni" nel Robot Game. Inoltre sceglieranno un problema che realmente affligge il nostro pianeta e ne spiegheranno la possibile soluzione con un Progetto ideato da loro stessi. Le competizioni avranno luogo su un campo di gara costituito da un tappeto posto su un tavolo riportante i modelli di missione; le squadre che devono prepararsi possono trovare sul sito www.fll-italia.it le caratteristiche per costruire un campo da allenamento simile a quello che i ragazzi troveranno in gara.

Il regolamento è disponibile, in pdf, alla pagina www.fll-italia.it/UploadDocs/4979_Nature_s_Fury_challenge_ITALIANO.pdf

Settimana europea robotica

Altro evento degno di nota è la eu-Robotic Week 2013 (<http://eurobotics-project.eu/eurobotics>) che, giunta alla sua terza edizione, si terrà dal 25 novembre all'1 dicembre 2013 per informare il pubblico sull'impatto che i robot hanno e avranno sulla società, incentivare studenti di tutte le età a occuparsi di scienza e tecnologia, e rafforzare la posizione dell'Europa nel mercato mondiale dell'ICT e della robotica. Infatti, uno degli scopi della Settimana Robotica Europea è far conoscere il settore ai giovani, in modo da favorire le iscrizioni degli studenti alle facoltà tecnico-scientifiche. Tra le altre finalità della Settimana Robotica Europa, far conoscere le realtà della robotica



nel nostro continente, con la molteplice varietà di applicazioni che le contraddistingue; le società di robotica e automazione, i centri di ricerca, le università, le scuole europee, avranno l'opportunità di far conoscere eccellenze, attività educative, attività di ricerca e di svago nei diversi settori applicativi della robotica. Durante la settimana si svolgeranno, in tutta Europa, circa 300 eventi dedicati alla robotica, che andranno da Open Lab (i laboratori, le aziende e i centri di ricerca apriranno i loro laboratori e gli impianti a visite guidate), a conferenze, corsi nelle scuole, manifestazioni artistiche legate alla robotics art, gare di robot, interviste e servizi sui media.

Per l'Italia, l'evento è coordinato da Scuola di Robotica, sul cui sito web <http://euroweek.scuoladirobotica.eu> sono pubblicate tutte le novità e sono disponibili tutte le informazioni per chi desidera proporre eventi durante la euroRobotics Week.

RoboCup Jr Italia

Come tutti gli anni, in autunno inizia la raccolta di adesioni alla RoboCup Jr., evento nazionale della RoboCup internazionale la cui fase finale mondiale è in programma, dal 19 al 25 luglio 2014,



a João Pessoa, in Brasile; la scelta del paese del Samba non è casuale, in quanto proprio in Brasile la prossima estate si terrà il Campionato del mondo di calcio - Copa do mundo 2014.

Per quanto riguarda la RoboCup Jr. Italia 2014 (VI^a edizione), in programma a Pisa dal 9 al 12 aprile, la Rete di scuole per la RoboCup Jr. Italia attenderà le adesioni delle squadre fino al 31 gennaio prossimo. I rappresentanti delle squadre delle scuole che hanno già aderito, potranno partecipare al consueto meeting nel cui contesto si riuniscono il Comitato di Gestione e il Comitato Tecnico della Rete di scuole per la RoboCup Jr. Italia; la data dell'incontro è fissata per venerdì 18 e sabato 19 ottobre 2013, presso il Serena Majestic Hotel & Residence di Montesilvano (PE).

La sesta edizione della RoboCup Italia di Pisa verrà curata dall'Istituto di BioRobotica della Scuola Superiore Sant'Anna e vedrà avvicinarsi le tradizionali competizioni Rescue A e B, Soccer, Dance e Theatre, suddivise in base all'età (Under 19 e Under 14); alla finale nazionale di Pisa prenderanno parte squadre provenienti da tutta Italia che si sfideranno, per la Under 19 nelle categorie Rescue A, Rescue B, Dance e Theatre, Soccer A Light, Soccer A e B Open (le prime tre sono valide per le qualificazioni alla finale mondiale) e per la Under 14 nelle categorie

Rescue, Dance e Theatre, entrambe valide come qualificazione per la finale mondiale in Brasile. Durante i giorni delle gare si terrà anche il convegno Roboscuola, con la presentazione da

Due novità per Robotica, il tradizionale appuntamento milanese dedicato alla robotica umanoide e di servizio: la prima è che la data, fino alle passate edizioni stabilita in novembre, viene spostata a marzo, forse per sfruttare il fiorire di idee che la primavera stimola. La seconda riguarda i format, che gli organizzatori annunciano rinnovato. Robotica, dopo i grandi successi degli anni passati torna per la quinta edizione dal 14 al 16 marzo 2014 a Fieramilano-Rho, in un nuovo contesto che ne favorirà l'aspetto divulgativo. Robotica 2014 si terrà in contemporanea con eventi rivolti al grande pubblico, per continuare a presentare la ricerca e i grandi progetti in ambito robotico, ma anche per far conoscere maggiormente la robotica a tutti coloro che, in prospettiva, ne diventeranno gli utilizzatori. L'operazione "robotica divulgativa" si svolgerà soprattutto sabato e domenica, mentre il venerdì continuerà a essere prevalentemente dedicato alla business community. In questa giornata -che prevede anche workshop di aggiornamento e dibattiti- gli operatori del settore potranno incontrare specialisti e ricercatori, ma anche imprenditori interessati a conoscere gli ambiti di applicazione e le prospettive industriali e commerciali del settore.

parte degli insegnanti che ne sono stati protagonisti, di numerose esperienze di Robotica Educativa nella scuola dell'obbligo e nella scuola secondaria tecnica e professionale. Tutti gli aggiornamenti sullo stato di preparazione della competizione su www.robocupjr.it.

Robotica 2014

Il mese di novembre vedrà, quest'anno, l'assenza di un evento protagonista dei quattro anni passati: Robotica prende tempo qualche mese per rinnovarsi e si ripresenterà in marzo (dal 14 al 16) nella solita location di Fieramilano-Rho. Tra le novità, è stata annunciata la dimostrazione, per tutti i giorni dell'evento, dell'esoscheletro EKSO presso lo stand di EKSO Robotics. Novità sarà anche il format dell'evento, che privilegerà l'aspetto divulgativo.

Due settimane dopo la città meneghina ospiterà una novità, probabilmente nata sull'onda del successo di Robotica: Robot & Makers Milano Show, prima edizione della mostra mercato dedicata alla robotica e ai makers che si svolgerà presso il Parco Esposizioni di Novegno il 29-30 Marzo 2014. Questo even-

to nasce come esposizione dei più importanti distributori italiani e internazionali di robot e di robotica, rivolto ai makers della robotica e proporrà banchi pieni di materiale per costruire il proprio robot in casa. L'organizzazione annuncia spazi dedicati alle realizzazioni dei Makers, workshop e laboratori aperti a tutti, sessioni didattiche, esibizioni di robot musicisti e ballerini, robot di pubblica utilità e di servizio, ma anche stampa creativa in 3D. Il tutto in un padiglione di 2.400 metri quadrati, all'insegna dell'innovazione tecnologica e del divertimento.

Robotica City Milano

In novembre si terrà a Milano l'evento Robotica City, aperto a tutti, e con il coinvolgimento di Comune, Museo della Scienza, Università Bocconi e Politecnico di Milano. La manifestazione prevede uno spazio in Galleria Vittorio Emanuele con workshop dimostrativi e divulgativi, un'area in piazza Duomo riservato a robot che circolano autonomamente, proposti dal Politecnico e non solo, nonché eventi all'interno del Museo della Scienza e della Tecnica. □



3.700 miglia in mare con Arduino

Scout è una barca robotica lunga circa 3,7 metri (12 piedi, per l'esattezza), realizzata da un gruppo di giovani appassionati del DIY per affrontare la traversata dell'Oceano Atlantico, pilotata da un computer (basato su Arduino!) che sulla base di una rotta prestabilita e delle condizioni di navigazione rilevate dai sensori di bordo e di waypoint predefiniti, naviga senza equipaggio. Superati i problemi iniziali dovuti a un errore del software che le ha fatto ignorare molti dei waypoint off-shore che erano stati programmati nel computer (il bug era nel codice destinato ad evitare l'effetto delle cor-

renti marine) ora la barca, partita il 24 agosto scorso da Rhode Island, sta viaggiando nella sua rotta transatlantica verso la Spagna. Lo scafo di Scout è una struttura in fibra di carbonio assemblata a sandwich con schiuma Divinycell e la propulsione è affidata a un motore elettrico montato sotto lo scafo; l'elettricità viene fornita da pannelli solari montati sulla parte superiore dello scafo, che caricano una batteria al litio-ferro-fosfato (LiFePO4). In condizioni ideali la batteria si carica di giorno, quanto basta per alimentare il motore della barca per tutta la notte seguente; se l'energia accumulata non

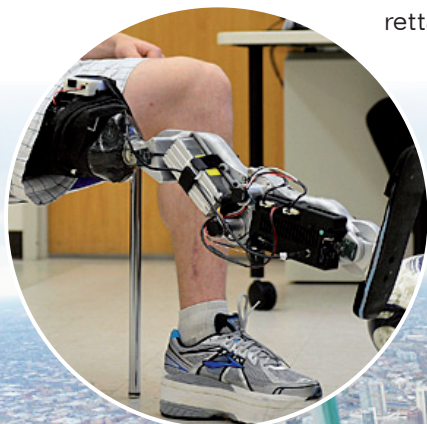
103 PIANI CON UNA GAMBA BIONICA

Provata su un paziente di 32 anni, la protesi di gamba bionica controllata dal pensiero ha dato i suoi primi risultati: costata 8 milioni di dollari e pesante 5 chili, alimentata a batterie e dotata di sensori come giroscopi e accelerometri, viene controllata direttamente

dagli stimoli nervosi del paziente quando pensa di camminare. Questa novità tecnologica è stata resa possibile grazie all'opera dei medici del Rehabilitation Institute of Chicago che, nel 2006, quando dovettero amputare una gamba al giovane, vittima di un incidente in moto, decisero di preservare alcuni nervi

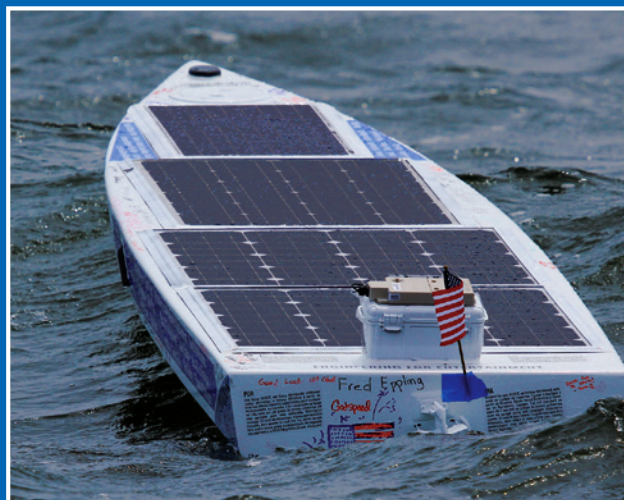
ancora funzionali nella speranza di poterli sfruttare con le protesi del futuro. Grazie ad un finanziamento dell'esercito U.S.A., particolarmente interessato al recupero dalle mutilazioni di guerra, gli studiosi hanno quindi potuto adottare un'innovativa tecnica chiamata TMR, acronimo di Targeted Muscle Reinnervation: durante un intervento hanno collegato le terminazioni nervose salvate dall'amputazione con i sensori della gamba artificiale, in modo da trasmettere gli impulsi nervosi e trasformarli in impulsi elettrici dell'arto robotico. Due motori azionano l'arto artificiale e permettono all'uomo di camminare, salire e scendere le scale. Per dimostrare l'efficacia dell'arto bionico, il paziente che l'ha indossato ha salito i 103 piani della Willis Tower di Chicago. Attualmente la gamba robotica ha un margine di precisione del 98% (superiore del 13% circa rispetto alle protesi robotiche standard) ma deve migliorare, soprattutto nella rumorosità e nell'autonomia, che è di 5.000 passi, contro i 10.000 che il programma dell'esercito richiederebbe.

<http://blogs.unb.ca/newsroom>



è sufficiente, il motore si spegne e l'imbarcazione va alla deriva e deve ritrovare la rotta appena in grado di muoversi. In previsione di questa evenienza, Scout è stata progettata per resistere con onde alte anche per un paio di giorni; la struttura del ponte di coperta e un peso in piombo sul fondo della chiglia permettono all'imbarcazione di raddrizzarsi se le onde laterali dovessero capovolgerla quando è alla deriva. Il computer di bordo è una scheda ad architettura Arduino e incorpora nel software funzioni di fail-safe come il riavvio automatico ogni due ore, per evitare avarie, e l'arresto ed il backup ogni cinque ore. Sul sito del progetto potete seguire la traversata in tempo reale.

<http://gotransat.com/>



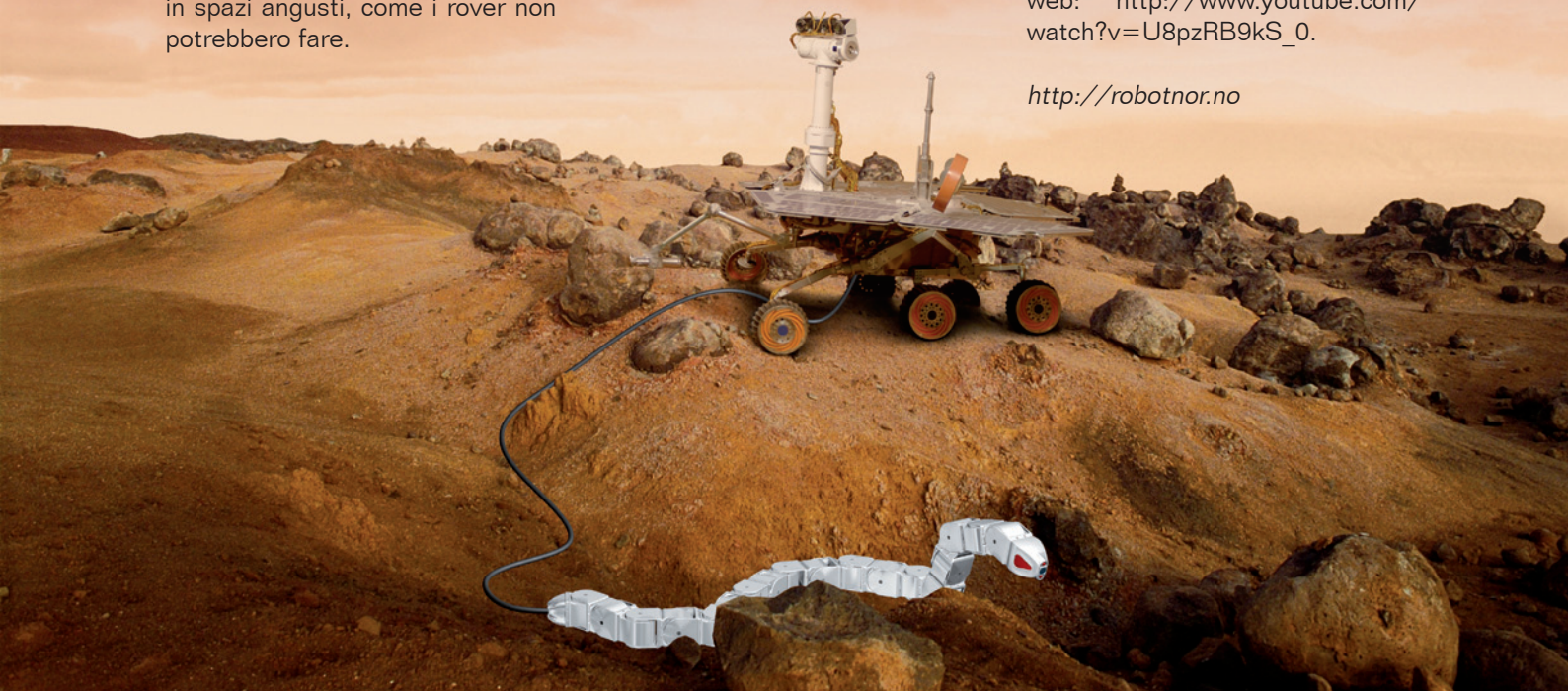
IL ROBOT SERPENTE ESPLORELLÀ MARTE

Il suolo del Pianeta rosso è da qualche anno percorso dai rover, robot su ruote e cingoli come Curiosity, che si muovono abbastanza agevolmente, almeno fin quando il terreno lo permette. Per ovviare alle loro limitazioni, l'ESA sta pensando a nuove forme per i robot esploratori: precisamente, ad automi a forma di serpente, composti da parti snodate in grado di scivolare tra gli ostacoli e passare in spazi angusti, come i rover non potrebbero fare.

Il progetto si chiama SERPEX (Serpentine Robot for Planet Exploration) ed è in fase di sviluppo da parte del SINTEF, la Fondazione per la ricerca scientifica e industriale norvegese; finanziato con 85.000 dollari dall'Agenzia Spaziale Europea, il serpente robot potrebbe lavorare al fianco dei rover, strisciando e compiendo un'esplorazione parallela rispetto a quella del rover stesso. La

combinazione delle due tecnologie sfrutterebbe le qualità del rover di percorrere lunghe distanze ed esaminare i campioni di terreno, e quelle del serpente, che può essere trasportato da esso e sganciato quando raggiunge luoghi in cui non può arrivare. Non solo: se il rover resta incastrato, il robot serpente potrebbe aiutarlo a svincolarsi. Un video demo del prototipo realizzato è disponibile all'indirizzo web: http://www.youtube.com/watch?v=U8pzRB9kS_0.

<http://robotnor.no>



Continuiamo il nostro viaggio alla scoperta di Flowcode, il sistema di sviluppo grafico per microcontrollori proposto da Matrix Multimedia, in cui il codice viene scritto facendo uso di oggetti grafici. Quarta puntata.

corso di programmazione in **FLOWCODE** di Francesco Ficili e Daniele Defilippi

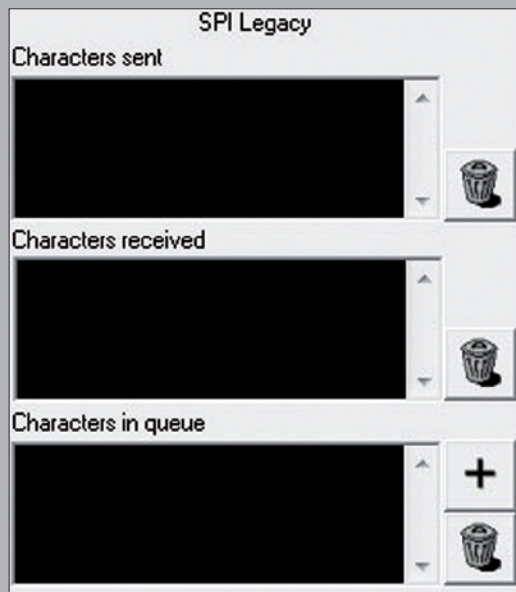
Nelle scorse puntate abbiamo spiegato la gestione degli interrupt in Flowcode e insegnato a utilizzare i componenti ADC, LCD e la memoria EEPROM interna. In questa puntata parleremo delle principali interfacce di comunicazione seriali a disposizione dei microcontrollori di bassa gamma: l'UART (o RS232), l'I²C, SPI e l'OneWire. Termineremo con un progetto pratico nel quale svilupperemo un controllo di una batteria di relé comandata tramite porta seriale.

Le periferiche di comunicazione di base

I sistemi che abbiamo trattato finora erano in grado di gestire solo sistemi chiusi, cioè

non potevano interfacciarsi con piattaforme esterne. Parleremo ora di altri componenti fondamentali di un microcontrollore, che sono le interfacce di comunicazione. I bus di comunicazione sono in continua evoluzione ed oggi ne sono di moltissime tipologie. Facciamo un piccolo passo indietro ed analizziamo brevemente le interfacce di comunicazione che sono state utilizzate in questi ultimi decenni. Con l'avvento dell'era digitale, il sistema di comunicazione maggiormente utilizzato è stato quello "parallelo", dove i dati sono scambiati a pacchetti, solitamente di 8 bit, ed uno o più segnali trasportano l'informazione del clock. Il trasporto dei dati in parallelo non è

Fig. 1
Pannello
SPI.



consigliabile sulle grandi distanze, in quanto richiederebbe molti fili e -per contrastare il degrado del segnale- l'amplificazione per ciascuno di essi; per questa ragione che sono state inventate le comunicazioni seriali, nelle quali il valore digitale da trasmettere è suddiviso in singoli bit, che sono trasmessi in ordine sequenziale sullo stesso canale di trasmissione, impiegando quindi due soli fili. Possiamo suddividere i sistemi di comunicazione in due grandi famiglie, in base alla distanza che riescono a coprire: quelli "interni" permettono la comunicazione tra sistemi vicini o comunque appartenenti allo stesso sistema, mentre quelli "tra sistemi" collegano apparati con compiti solitamente differenti su distanze anche molto ampie. Inizieremo parlando di quelli interni ed in particolare di SPI, I²C e OneWire. Successivamente passeremo alla descrizione della RS232, interfaccia con la quale realizzeremo un progetto pratico.

Serial peripheral interface (SPI)

L'interfaccia SPI (acronimo di Serial Peripheral Interface) è semplice da utilizzare ed affidabile e, per queste ragioni, è divenuta il sistema di comunicazione comunemente utilizzato da memorie, ADC e DAC esterni al microcontrollore.

La comunicazione mediante porta SPI necessita di tre collegamenti:

- SDO/MOSI (Serial Data Output) dal quale i dati vengono inviati;
- SDI/MISO (Serial Data Input) dal quale i dati vengono ricevuti;

- SCK/SCLK (Serial Data Clock) su cui viene trasmesso il segnale di clock.

I dati trasmessi e ricevuti passano su due connessioni differenti e, per questa ragione, la comunicazione è detta di tipo *full-duplex*. Più periferiche possono essere collegate allo stesso bus di comunicazione, ma, in questo caso, solo una di esse è di tipo "master", mentre tutte le altre sono di tipo "slave". Per poter utilizzare questo sistema di comunicazione non è comunque necessaria una conoscenza approfondita del livello fisico e di protocollo, perché Flowcode mette a disposizione una potentissima libreria di componenti che ci permette di utilizzarla e gestirla in modo ancora più semplice.

All'interno della barra dei componenti nel menu *Trasmissione dati* troviamo il componente *SPI*; se lo inseriamo nel nostro progetto mediante un clic sulla relativa icona, notiamo che appare, nel pannello, come una finestra suddivisa in tre parti. Partendo dall'alto verso il basso possiamo così esaminare i caratteri inviati, quelli ricevuti e in basso i caratteri in coda, come mostrato in Fig. 1.

Ora che il componente è stato inserito nel progetto, è necessario configurarlo in modo che possa comunicare correttamente con le periferiche. I campi della finestra delle proprietà, mostrata in Fig. 2, devono essere valorizzati esaminando il data-sheet del componente che vogliamo collegare esternamente, in modo che tutto possa funzionare correttamente.

Il box in cui si può indicare se trasmettere byte o caratteri non influenza la modalità

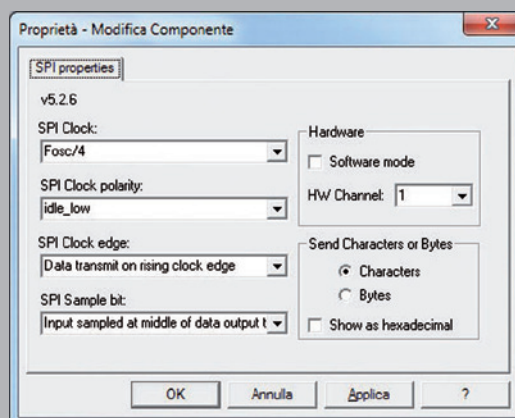


Fig. 2
Pannello di
configura-
zione del
bus SPI.

Tabella 1 - Parametri di configurazione del bus SPI.

Parametro	Descrizione
SPI Clock	Rappresenta la velocità di trasmissione dei dati. Deve obbligatoriamente essere minore di quella supportata dal componente più lento collegato al bus di comunicazione. Facciamo un semplice esempio supponendo che il componente slave supporti una velocità massima di comunicazione di 300 kHz ed il microcontrollore abbia una frequenza di oscillazione pari a 4 MHz. $\frac{F_{SPI(max)}}{F_{osc}} = \frac{300KHz}{4MHz} = 0,075 \rightarrow 0,075 < \frac{1}{16} \rightarrow \frac{F_{osc}}{16}$ Dividendo la frequenza massima della comunicazione per la frequenza di oscillazione del microcontrollore, otteniamo il rapporto massimo che vi può essere tra le due grandezze e scegliamo il valore più grande, tra quelli disponibili nel menu a tendina, che risulta minore del risultato della frazione.
SPI Clock polarity	È utilizzato per definire lo stato di inattività (idle) del segnale di clock.
SPI Clock Edge	Serve per scegliere se il dato è scritto durante il fronte di salita o discesa del segnale di clock.
SPI Sample bit	Indica quando il dato deve essere campionato per poter effettuare una corretta trasmissione/ricezione del dato da tutti i dispositivi collegati. In pratica, modifica il punto di campionamento del dato.

di trasmissione/ricezione dei dati sul bus di comunicazione, ma unicamente la loro visualizzazione sul componente nel pannello di Flowcode.

Al termine della configurazione possiamo inserire le macro gestite del componente dal menu delle icone, come fatto per il componente KeyPad e per l'ADC esaminati nelle puntate precedenti. In questo caso le principali macro a nostra disposizione sono quelle elencate nella **Tabella 2**.

Vi sono altre macro specifiche, mostrate in **Tabella 3**, che ci possono essere di aiuto nel

caso volessimo utilizzare la scheda EB013, che contiene al suo interno una memoria FRAM (rimpiazzabile con una memoria EEPROM) ed un convertitore digitale/analogo, entrambi su bus SPI.

Per poter utilizzare questi componenti è inoltre necessario configurare le connessioni esterne dei piedini di abilitazione. Utilizzando queste semplici macro è quindi possibile utilizzare una memoria EEPROM/FRAM od un DAC in modo davvero semplice ed intuitivo. Qualora il componente scelto non appartenesse a questa lista o non

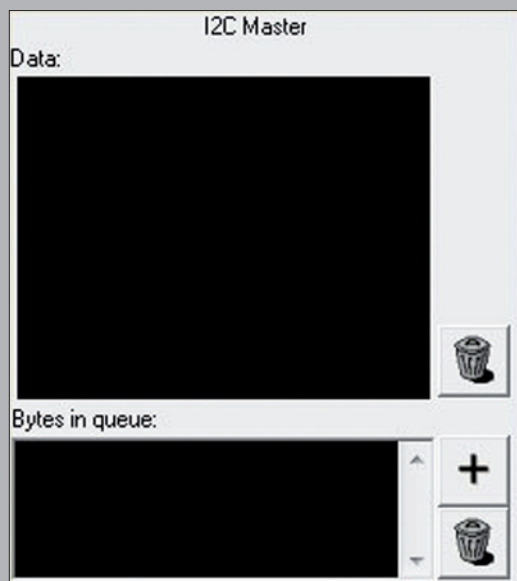
Tabella 2 - Macro disponibili per il controllo del bus SPI.

Nome Macro	Descrizione
SPI_Init	È utilizzata per inizializzare le periferiche di comunicazione SPI. Solitamente è posta ad inizio programma e viene chiamata una volta sola.
SPI_Send_Char	È utilizzata per inviare un carattere.
SPI_Get_String	È utilizzata per ricevere una stringa di caratteri.
SPI_Send_String	È utilizzata per inviare una stringa di caratteri.
SPI_Get_Char	È utilizzata per ricevere un carattere.
SPI_Unint	È utilizzata per rimuovere le inizializzazioni della connessione SPI e poter quindi utilizzare i pin del microcontrollore come normali bit della porta di I/O.

Tabella 3 - Macro supplementari per componenti specifici su bus SPI.

Nome Macro	Descrizione
DAC_Send_Char	Invia un carattere/byte al convertitore digitale/analogo che lo trasformerà direttamente in un valore analogico visibile sui morsetti esterni.
NVM_Send_Char	Invia un dato alla memoria EEPROM che verrà caricato all'interno della cella selezionata.
NVM_Get_Char	Legge il valore memorizzato all'interno della cella di memoria indicata.
FramOutput	Invia un valore alla memoria FRAM.
EnableFRAM	Abilita la memoria FRAM/EEPROM. Prima di ogni comando verso la memoria, è necessario abilitarla altrimenti tutti i comandi verrebbero ignorati.
DisableFRAM	Disabilita la memoria FRAM/EEPROM in modo da poter comunicare con altri dispositivi collegati sullo stesso bus SPI.

Fig. 3
Pannello del
componente
I²C Master.



fosse compatibile, basterà semplicemente utilizzare le macro generiche in modo da realizzare il protocollo corretto.

La porta I²C

Passiamo ora a descrivere il bus di comunicazione I²C, il cui protocollo richiede unicamente due linee di comunicazione:

- SDA (Serial Data Line) per la trasmissione/ricezione dei dati ;
- SCL (Serial Clock Line) per la trasmissione del clock di sincronizzazione.

Sullo stesso bus possono essere collegati fino a 112 dispositivi differenti. A differenza dell'SPI, sia i dati in trasmissione che quelli in ricezione viaggiano sulla stessa connessione fisica. Il bus SPI è di tipo *half-duplex* perché solo un componente alla volta può trasmettere i dati; inoltre, la comunicazione può essere iniziata solo dal master, mediante una stringa che contiene un bit di start, l'indirizzo dello slave con il quale vuole comunicare e un bit che indica se è richiesta

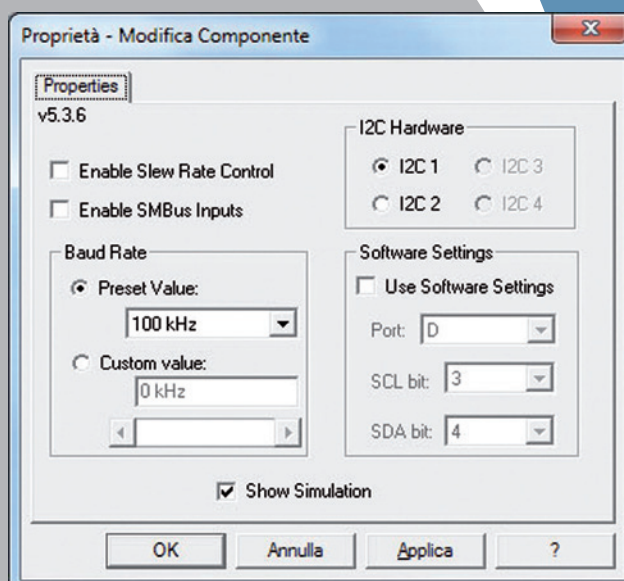


Fig. 4
Proprietà del
componente
I²C master.

una risposta. Successivamente vengono inviati, in sequenza, i dati veri e propri, iniziando sempre dal bit più significativo. All'interno della barra dei componenti possiamo selezionare il controller per il bus I²C ed inserirlo all'interno del pannello di Flowcode. Il pannello appare come in Fig. 3 e ci permette di visualizzare le comunicazioni presenti sul bus e poter interagire aggiungendo byte nella comunicazione in modo da rendere la simulazione interattiva. Per utilizzare il componente è necessario configurarlo come abbiamo fatto precedentemente con il bus SPI, ma i parametri da settare sono differenti; nel caso del bus I²C, essi sono quelli elencati nella **Tabella 4**. Invece le macro a disposizione sono riassunte in **Tabella 5**. Continuando nel nostro percorso di descrizione di bus di comunicazione, passiamo al bus con il minor numero di connessioni possibili: una connessione unica per la trasmissione/ricezione di dati e segnale di sincronizzazione: il bus OneWire.

Tabella 4 - Parametri di configurazione componente I²C.

Parametro	Descrizione
Enableslew rate control	Attiva il controllo sullo slew-rate (velocità dei fronti) del segnale trasmesso sul bus.
Enable SMBus Inputs	Abilita l'utilizzo del SMBus sulla connessione I ² C (modalità di compatibilità).
Baud rate	Imposta la frequenza con la quale vengono trasmessi i dati sul bus di comunicazione. Deve obbligatoriamente, come visto per il bus SPI, essere minore della velocità massima supportata dal componente più lento collegato al bus. Possono essere utilizzati i valori pre-configurati (preset), oppure dei valori custom scelti utilizzando la barra a scorrimento.
IC2 hardware	Seleziona quale periferica, messa a disposizione dal microcontrollore, è coinvolta nella trasmissione dei dati. Qualora ve ne fossero più di una disponibili, come mostrato in Fig. 4, possono essere inseriti due componenti I ² C, ciascuno su un differente canale in modo da avere due bus distinti di comunicazione.

Tabella 5 - Macro disponibili per la gestione della connessione I²C master.

Nome Macro	Descrizione
MI2C_Init	Inizializza la periferica interna di comunicazione I ² C. Deve essere chiamata una volta prima di eseguire le altre macro altrimenti non verrebbe trasmesso/ricevuto alcun byte.
MI2C_Start	Invia il segnale di start della comunicazione sul bus I ² C.
MI2C_Restart	Invia il segnale di riavvio della comunicazione sul bus I ² C.
MI2C_Stop	Invia il segnale di stop della comunicazione sul bus I ² C.
MI2C_Transmit_Byte	Trasmette un byte sul bus di comunicazione. Restituisce zero se rileva la ricezione di una conferma di ricezione "acknowledge".
MI2C_Receive_Byte	Riceve un byte dal bus di comunicazione. Necessita della valorizzazione del parametro che, se posto a uno, indica che il byte da ricevere è l'ultimo.
MI2C_Send_Byte_Transaction	Invia ad uno specifico dispositivo il valore che deve essere posto all'indirizzo indicato.
MI2C_Receive_Byte_Transaction	Riceve da uno specifico dispositivo il valore caricato all'indirizzo di memoria scelto.

La porta OneWire

Il bus OneWire è stato realizzato da Dallas Semiconductor per consentire comunicazioni tra due o più dispositivi attraverso un unico filo di interconnessione. Non essendoci un filo dedicato per la trasmissione del segnale di sincronizzazione, quest'ultimo deve essere trasmesso assieme ai dati.

Per evitare problemi di cortocircuiti sulla linea è presente un pull-up, cioè una resistenza collegata tra la linea e l'alimentazione, e tutti i dispositivi possono imporre uno zero sulla linea attivando il transistor/MOSFET di pilotaggio della porta connessa. La velocità di comunicazione non è elevatissima (è prefissata a 16 kbps), tuttavia permette comunicazioni tra dispositivi in modo davvero semplice. Si tratta di un bus di tipo Master/Slave, dove ogni dispositivo ha un indirizzo univoco e questo deve essere utilizzato dal master per iniziare la comunicazione con lo slave.

La codifica dei simboli differisce notevolmente dai casi esaminati in precedenza, perché sono codificati in base alla lunghezza dell'impulso, come indicato in **Tabella 6**.

Nel caso di ricezione, cioè durante la comunicazione di dati dallo slave verso il master, lo slave mantiene la linea al valore 0 per 60 µs qualora voglia inviare un valore logico zero mentre non fa nulla se intende inviare il valore logico uno a seguito di ogni impulso generato dal master.

Tabella 6 - Tempi di impulso per la trasmissione dei dati su bus OneWire.

Reset	Linea a 0 per almeno 480 µs
Valore logico uno	1+15 µs linea a 0
Valore logico zero	60 µs linea a 0

Per accertarsi che il pacchetto di dati sia stato correttamente ricevuto/inviato, è presente il controllo mediante CRC a 8 bit che è trasmesso come primo byte. Nella finestra di configurazione del componente, mostrata in **Fig. 5**, possiamo unicamente scegliere se utilizzare o no il controllo CRC.

Le macro disponibili in Flowcode per gestire questa porta sono riassunte nella **Tabella 7**.

In **Tabella 8** sono riportate altre macro che potrebbero risultare utili qualora si volesse utilizzare il dispositivo DS1820.

Prima di poter utilizzare il componente è

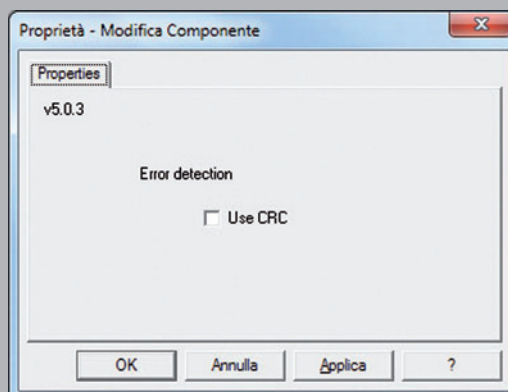


Fig. 5
Proprietà del
componente
OneWire.

Tabella 7 - Macro disponibili per la gestione del bus OneWire.

Nome Macro	Descrizione
oo_BusReset	Invia un comando di reset e restituisce uno se nessun dispositivo risulta resettato oppure non vi sono dispositivi collegati al bus.
oo_GetPadByte	Restituisce il dato letto dal dispositivo identificato dal parametro passato alla macro; I dispositivi letti sono al massimo 9 ed il parametro può variare da 0 a 8 se un dispositivo risulta resettato oppure non vi sono dispositivi collegati al bus.
oo_Tx_Byte	Trasmette un byte sul bus.
oo_Rx_Byte	Riceve un byte dal bus.
oo_ScanBus	Esegue una scansione del bus per l'identificazione del numero di dispositivi. Restituisce 1 se vi sono stati dei problemi durante la scansione.
oo_Get_DeviceCount	Restituisce il numero di dispositivi rilevati sul bus dopo un comando di scansione del bus.
oo_ReadDevice	Legge il dispositivo identificato dal numero in ordine di scansione.

Tabella 8 - Macro per la gestione del componente DS1820 su bus OneWire.

Nome Macro	Descrizione
DS1820_Start_Conversion	Avvia la conversione del dato del dispositivo DS1820. Restituisce uno se il dispositivo è entrato in time-out senza terminare la conversione.
DS1820_Read_Scratchpad	Legge i valori dal dispositivo DS1820. Può essere utilizzata unicamente qualora vi sia un unico dispositivo collegato al bus altrimenti è necessario utilizzare la macro oo_ReadDevice.
DS1820_Get_Temp	Legge la temperatura del dispositivo DS1820. Il valore letto è rappresentato su un intero con risoluzione di 0,0625 °C.

necessario configurare il piedino che sarà utilizzato per trasmettere/ricevere i dati, come indicato in Fig. 6.

Un semplice esempio di diagramma di flusso per il controllo del bus OneWire per la gestione del componente DS1820 è raffigurato in Fig. 7: con l'utilizzo di soli quattro blocchi possiamo comunicare con il componente dimenticandoci completamente tutte le problematiche inerenti al livello fisico e del protocollo di comunicazione. Ora che abbiamo esaminato tre esempi di

bus di comunicazione interni, passiamo a descrivere la porta RS232, che è una tra le più usate e semplici interfacce seriali.

La Porta RS232

La RS232 è una linea di comunicazione asincrona, in quanto il segnale di temporizzazione non viene trasmesso, ma generato in modo autonomo da entrambe le periferiche collegate. La connessione è di tipo punto-punto, quindi permette di far comunicare tra loro solo due dispositivi.

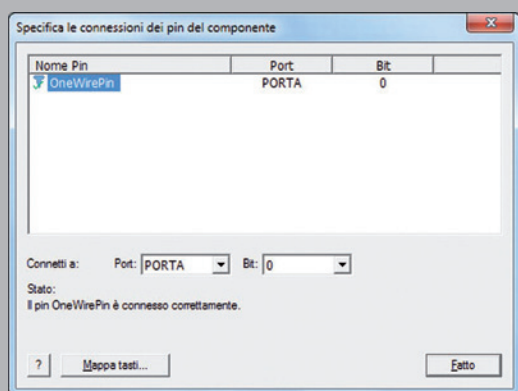


Fig. 6 - Configurazione connessioni bus OneWire.

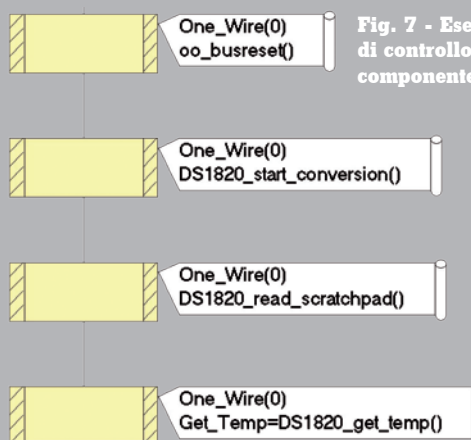


Fig. 7 - Esempio di flusso di controllo per il componente DS1820.

Tabella 9 - Macro disponibili per la gestione della connessione RS232.

Nome Macro	Descrizione
SendRS232Char	Invia un dato a 8-bit sulla seriale.
SendRS232String	Invia una stringa di caratteri sulla seriale.
ReceiveRS232Char	Riceve un carattere sulla seriale. La macro accetta come parametro il valore del tempo massimo di attesa del dato e qualora non venga ricevuto entro il tempo massimo indicato restituisce il valore INVALID_RETURN (255).
ReceiveRS232String	È simile alla ReceiveRS232Char ma legge una stringa di caratteri la cui lunghezza è indicata dal parametro Length.
ChangeHWBaud	Modifica la velocità di trasmissione dei dati. Accetta un valore numerico intero compreso tra 0 e 7 codificato come nella Tabella 10 .

I dati vengono trasmessi in modo seriale, in pacchetti da 8-9 bit racchiusi tra un bit di start ed uno di stop. La durata di ogni bit dipende dalla velocità di comunicazione; affinché tutto funzioni correttamente, quest'ultima deve essere uguale per entrambi i dispositivi.

Flowcode dispone di un componente specifico per la gestione della RS232, le cui macro sono riassunte in **Tabella 9**. Per poter utilizzare il componente dobbiamo configurarlo (come fatto per gli altri visti in precedenza) attraverso la finestra di configurazione, accessibile dal menu a tendina del componente nel pannello; qui abbiamo a disposizione i controlli per impostare la velocità di comunicazione (baud-rate), il numero di bit del frame, nonché la possibilità di utilizzare le connessioni per il controllo di flusso.

Non ci soffermeremo sui dettagli di ogni singola configurazione, perché il nostro obiettivo è dare a tutti le nozioni base per utilizzare i componenti trattati in modo rapido e efficace; qualora si volessero approfondire alcune caratteristiche e funzionalità, è sempre possibile utilizzare l'help di Flowcode, che consigliamo di tenere sempre sottomano, perché potrebbe, soprattutto all'inizio, permettervi di imparare molto.

Questo tipo di connessione è molto semplice da utilizzare e permette comunicazioni con dispositivi anche molto complessi come ricevitori GPS o display grafici. Per comprenderne meglio le potenzialità, passiamo ora alla realizzazione di un semplice progetto pratico, che ci permetterà di gestire una batteria di relé tramite l'interfaccia seriale del PC.

Tabella 10 - Valori disponibili per la modifica al volo della velocità di connessione.

Parametro	Velocità
0	1200 bps
1	2400 bps
2	4800 bps
3	9600 bps
4	19200 bps
5	38400 bps
6	57600 bps
7	115200 bps

Gestione di una scheda a relé tramite interfaccia RS-232

Nella realizzazione di questo semplice progetto tralasceremo i passaggi base che abbiamo già spiegato nelle puntate precedenti e ci soffermeremo maggiormente sui nuovi concetti.

Il nostro sistema presenta le caratteristiche seguenti.

- Comunica via RS232 bidirezionale tra PC e scheda di controllo a relé.
- Deve supportare i seguenti comandi:
 - "A": Attiva relé;
 - "D": Disattiva relé;
 - "R": Leggi stato relé.
- La sintassi dei comandi è <xp>, dove x identifica il codice del comando (che può essere uno dei tre comandi visti al punto precedente), mentre p identifica il relé controllato, ed è rappresentato da un numero con codifica ASCII. Quindi, ad esempio, per attivare il relé 3 inviamo il comando "<A3>", per spegnere il relé 2 inviamo "<D2>" mentre per leggere lo stato del relé 1 inviamo il comando "<R1>".
- La sintassi delle risposte ai comandi è la

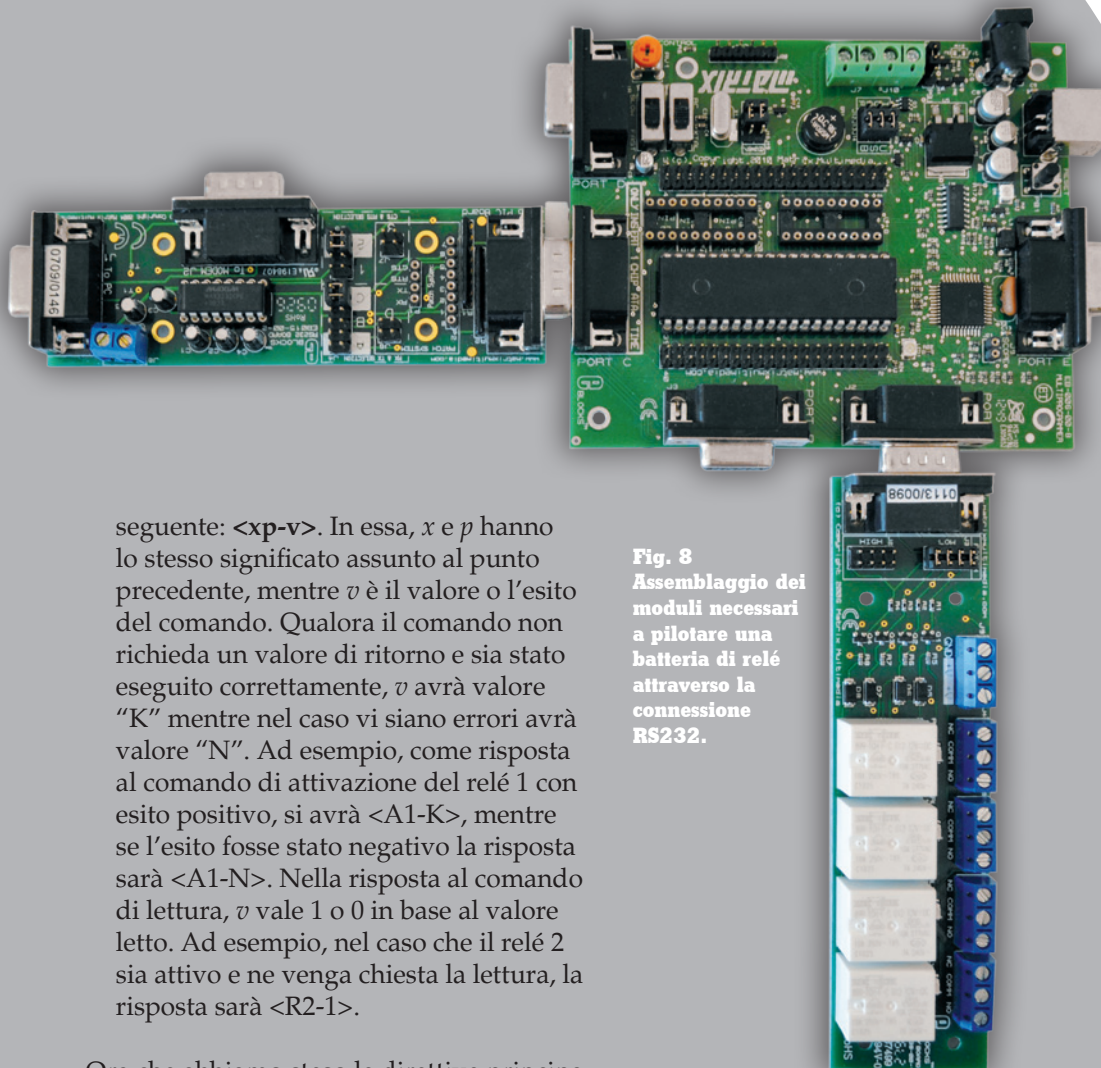


Fig. 8
Assemblaggio dei
moduli necessari
a pilotare una
batteria di relé
attraverso la
connessione
RS232.

seguente: $\langle xp-v \rangle$. In essa, x e p hanno lo stesso significato assunto al punto precedente, mentre v è il valore o l'esito del comando. Qualora il comando non richieda un valore di ritorno e sia stato eseguito correttamente, v avrà valore "K" mentre nel caso vi siano errori avrà valore "N". Ad esempio, come risposta al comando di attivazione del relé 1 con esito positivo, si avrà $\langle A1-K \rangle$, mentre se l'esito fosse stato negativo la risposta sarà $\langle A1-N \rangle$. Nella risposta al comando di lettura, v vale 1 o 0 in base al valore letto. Ad esempio, nel caso che il relé 2 sia attivo e ne venga chiesta la lettura, la risposta sarà $\langle R2-1 \rangle$.

Ora che abbiamo steso le direttive principali per realizzare il nostro progetto, passiamo all'identificazione dei componenti necessari per la realizzazione, che sono:

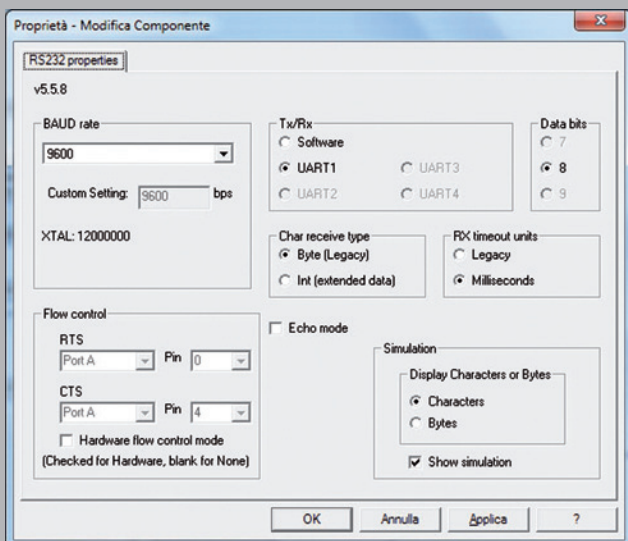


Fig. 9 - Configurazione del componente RS232.

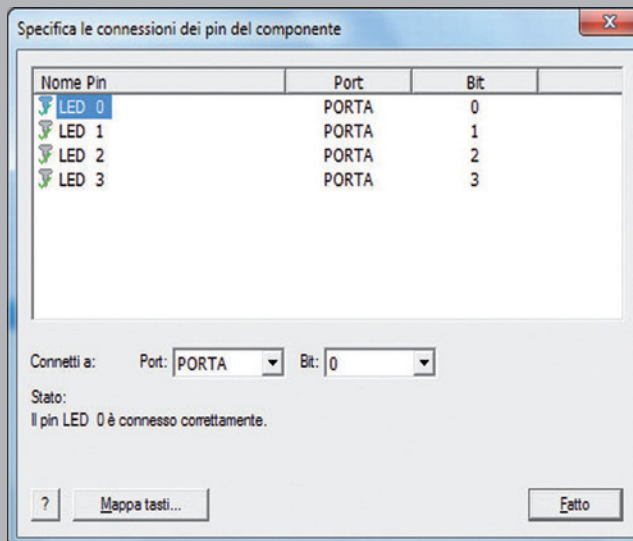
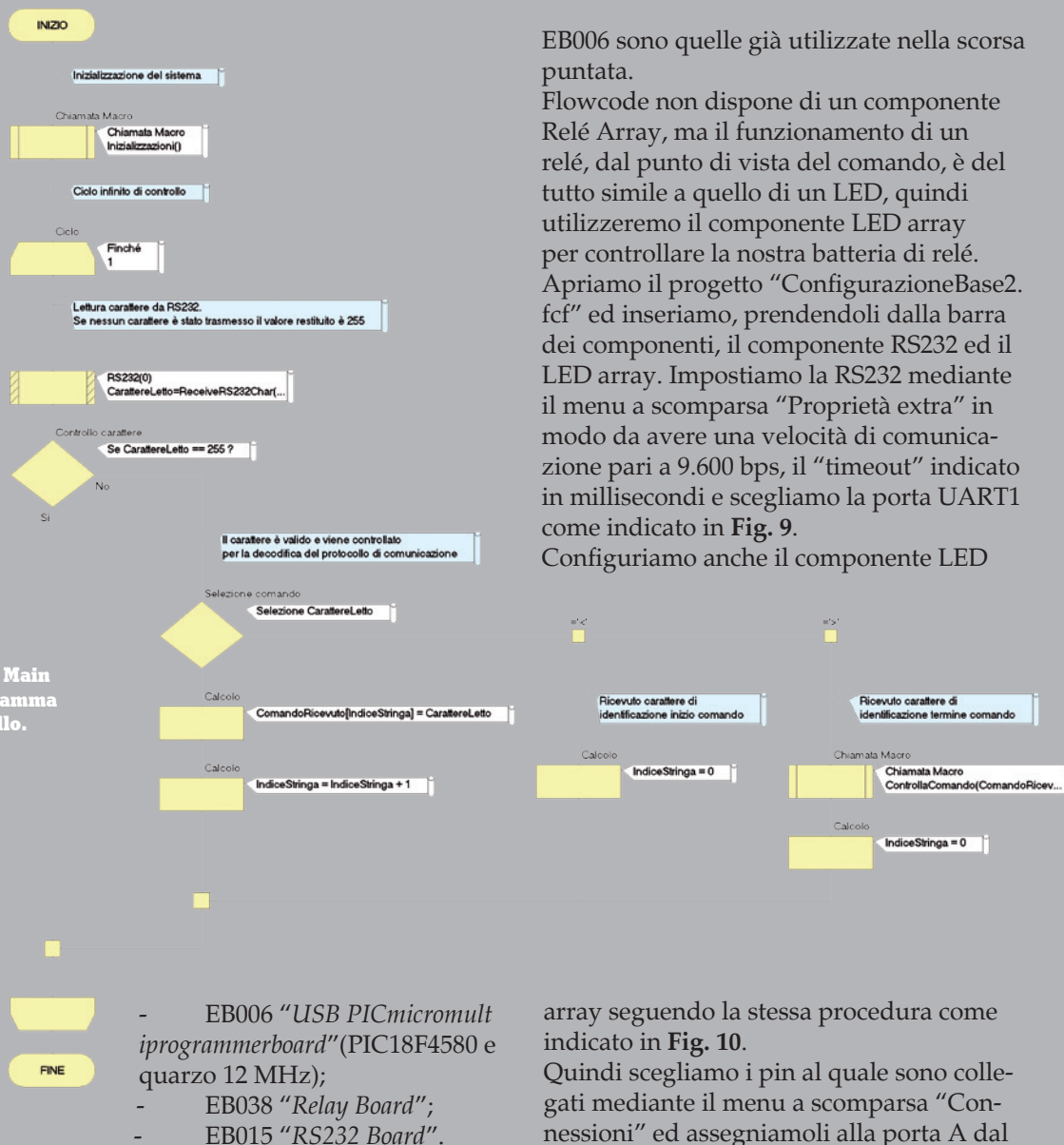


Fig. 10 - Configurazione del componente LED array.



Collegiamoli in modo da avere la scheda EB038 sulla porta A e la EB015 sulla porta C, come mostrato in Fig. 8.

Le schede devono essere alimentate per poter funzionare correttamente, perché attraverso il connettore DB9 passano solo le connessioni dati ed il riferimento di massa perciò, utilizzando dei fili da prototipazione, colleghiamo tutti i morsetti con indicato "+V" al corrispondente morsetto della scheda EB006. Posizioniamo, inoltre, i connettori di patch in posizione C e 2 per la scheda EB038, e in posizione LOW per la scheda EB015. Le altre configurazioni della scheda

EB006 sono quelle già utilizzate nella scorsa puntata.

Flowcode non dispone di un componente Relé Array, ma il funzionamento di un relé, dal punto di vista del comando, è del tutto simile a quello di un LED, quindi utilizzeremo il componente LED array per controllare la nostra batteria di relé. Apriamo il progetto "ConfigurazioneBase2.fcf" ed inseriamo, prendendoli dalla barra dei componenti, il componente RS232 ed il LED array. Impostiamo la RS232 mediante il menu a scomparsa "Proprietà extra" in modo da avere una velocità di comunicazione pari a 9.600 bps, il "timeout" indicato in millisecondi e scegliamo la porta UART1 come indicato in Fig. 9.

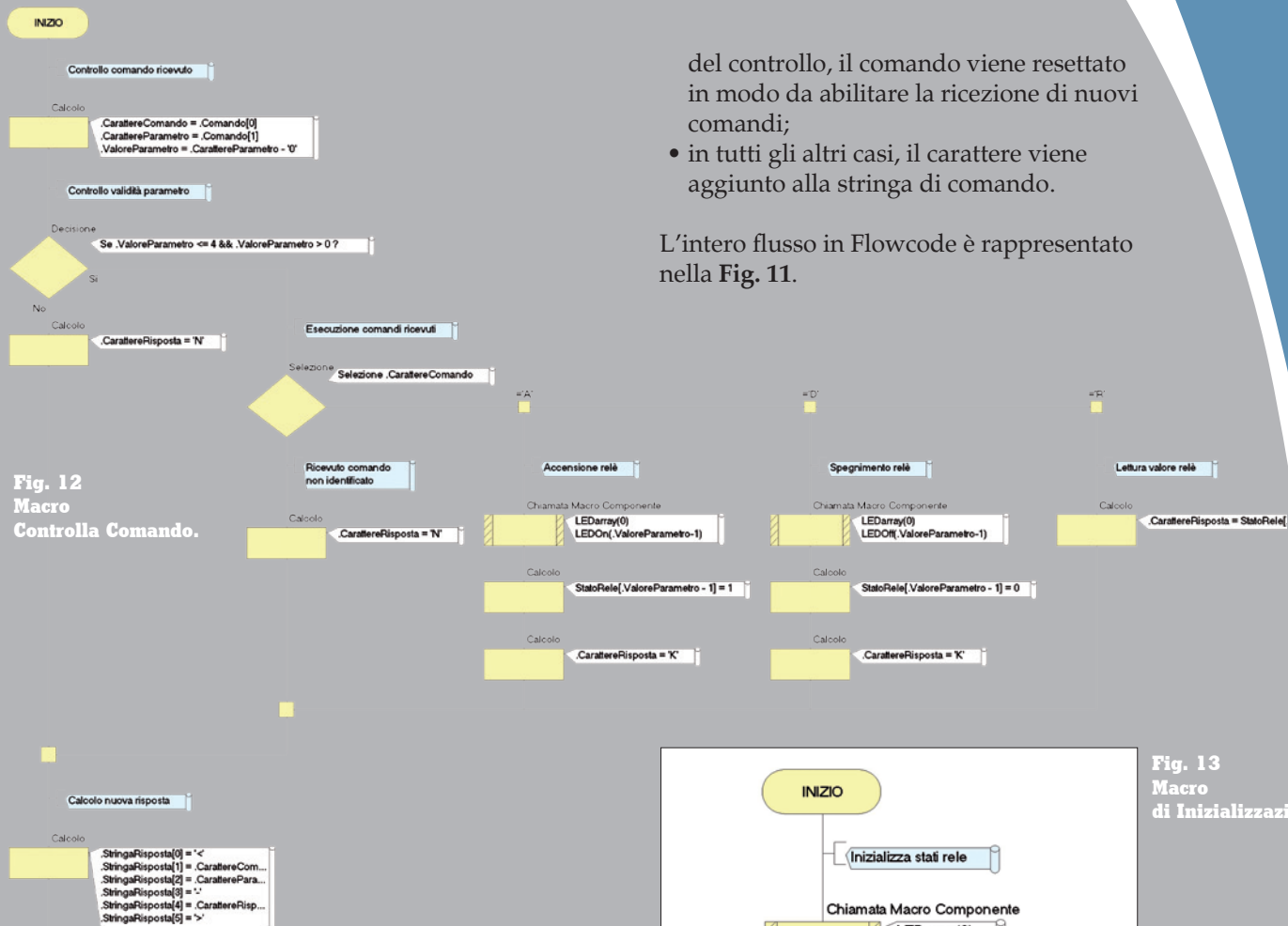
Configuriamo anche il componente LED

array seguendo la stessa procedura come indicato in Fig. 10.

Quindi scegliamo i pin al quale sono collegati mediante il menu a scomparsa "Connessioni" ed assegniamoli alla porta A dal pin 0 al pin 4.

Ora che è tutto pronto, possiamo dedicarci alla stesura del diagramma di flusso. Per ottimizzare l'uso di Flowcode possiamo pensare di suddividere, utilizzando le macro, l'intero programma in tre parti principali:

- **Inizializzazione** = inizializza tutto ciò che deve essere preimpostato prima che il ciclo principale inizi;
- **Main** = chiama l'inizializzazione del sistema, legge i caratteri ricevuti da seriale e li aggrega in un'unica stringa in modo da esaminarla mediante un semplice controllo di correttezza;
- **ControllaComando** = esegue il controllo

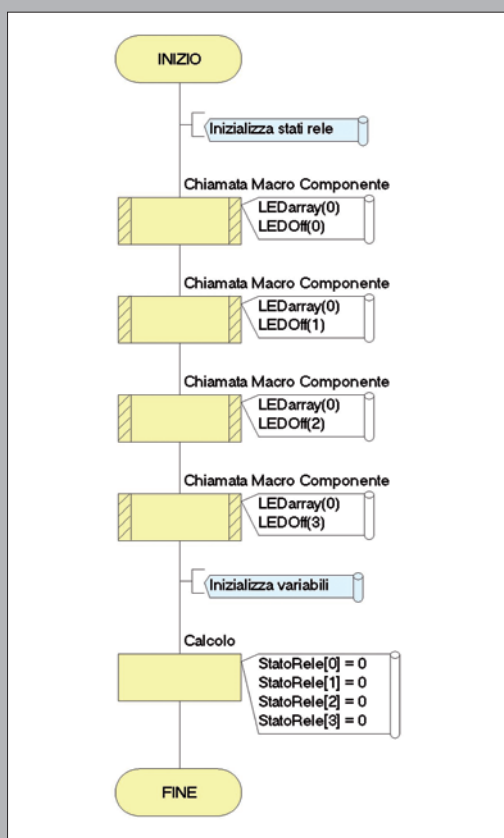


del controllo, il comando viene resettato in modo da abilitare la ricezione di nuovi comandi;

- in tutti gli altri casi, il carattere viene aggiunto alla stringa di comando.

L'intero flusso in Flowcode è rappresentato nella Fig. 11.

Fig. 13 Macro di Inizializzazione.



sulla correttezza del comando ricevuto, esegue il comando ed invia la risposta.

Main

Il main contiene la chiamata alla macro di inizializzazione, oltre ad un ciclo infinito nel quale viene richiesta una lettura da seriale ed è analizzato il carattere ricevuto: quindi, ogni volta che viene ricevuto un carattere da seriale, esso viene analizzato. Dopo la ricezione e l'analisi ci sono tre possibilità:

- il carattere è "<", allora viene azzerata la stringa che contiene il carattere ricevuto;
- il carattere è ">", allora la stringa di comando è stata ricevuta completamente e quindi viene controllato il comando attraverso la specifica macro; al termine

Tabella 11 - Formato del file .cvs.

Tempo di esecuzione [ms]	Stato Relé 1	Stato Relé 2	Stato Relé 3	Stato Relé 4
--------------------------	--------------	--------------	--------------	--------------



Fig. 14 - Sezione "Manual Control" dell'interfaccia sw.

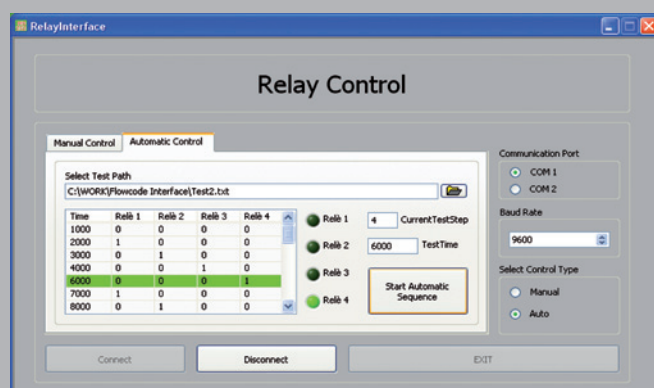


Fig. 15 - Sezione "Automatic Control" dell'interfaccia sw.

Macro "Controlla Comando"

Realizza il controllo del comando ricevuto come parametro, lo esegue nel caso sia stato ricevuto correttamente e, in ogni caso, invia una risposta sulla linea seriale. Nella prima parte della macro vengono estrapolati dalla stringa il carattere di comando ed il parametro associato. Viene fatto un controllo sul valore del parametro (che nel nostro caso deve essere compreso tra 1 e 4) e successivamente, in base al carattere del comando, viene eseguita l'operazione di accensione, spegnimento o lettura del relé. Gli stati dei relé sono memorizzati in un vettore globale di appoggio chiamato "StatoRele". Al termine dell'esecuzione, prima di ritornare il controllo alla funzione chiamante, la macro invia la risposta utilizzando la macro SendRs232String. L'intero diagramma è visibile in Fig. 12.

Macro "Inizializza"

È utilizzata per inizializzare lo stato dei relé e le variabili di appoggio del programma. In questo caso l'unica variabile che deve essere azzerata è il vettore con lo stato dei relé "StatoRele" che, per essere concorde allo stato fisico, deve essere impostato a zero. Il diagramma Flowcode è rappresentato nella Fig. 13.

Interfaccia Software

Per l'utilizzo della scheda relé appena realizzata è stata sviluppata un'apposita interfaccia software, scritta in Labview 2010. L'interfaccia permette sia il controllo manuale dei 4 relé, sia l'esecuzione di sequenze automatizzate. Subito dopo l'avvio dell'interfaccia possiamo scegliere tra le due modalità, mediante un radio button posto in basso, sulla destra del pannello. Uno screenshot

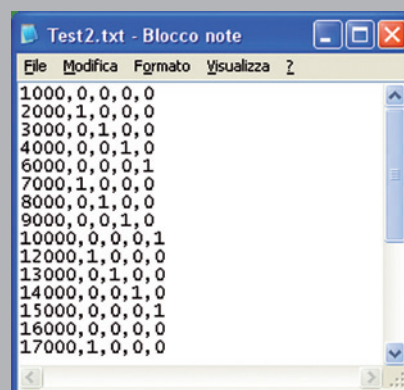


Fig. 16
Esempio di file
.csv.

dell'interfaccia software è visibile in Fig. 14. Come si può notare, è possibile selezionare la porta RS-232 (COM1 o COM2), selezionare il baud-rate (l'impostazione predefinita è 9600) mentre sono presenti due tab, "Manual Control" e "Automatic Control", contenenti i comandi specifici per le due modalità. Per la modalità di controllo manuale sono presenti 4 pulsanti, con i quali è possibile controllare lo stato dei relé. Due parole in più merita la sezione automatica, della quale in Fig. 15 riportiamo uno screenshot. Tale sezione permette di automatizzare il controllo della scheda da PC, impostando sequenze di controllo temporizzate. Per impostare le sequenze automatizzate è sufficiente generare un file .csv con la formattazione indicata nella Tabella 11. Come si può vedere, il primo campo indica il tempo (in ms) al quale verranno inviate le istruzioni presenti nei 4 campi successivi, che rappresentano lo stato dei 4 relé (0 o 1). Un esempio di file .csv è riportato in Fig. 16. Tramite questa modalità, è possibile impostare sequenze di controllo che possono essere impiegate per le più svariate necessità, dall'automazione alla domotica, al controllo di macchinari.