

144 PAGINE!

Mensile di progettazione elettronica, attualità scientifica, novità tecnologiche.

Elettronica In

www.elettronica.in.it

oltre l'elettronica

€ 6,00

Anno XIX - n. 177
Giugno 2013

FOCUS
STAMPA

3D



Rendiamo autonoma la

3D RAG



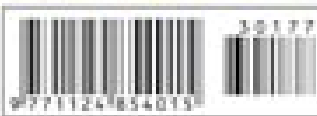
BOARD

Linux
Embedded
a confronto



- OpenWheels, le funzioni nello sketch
- Controller USB per 4 stepper motor
- Gestire i moduli GSM con Arduino

- Telemetria real-time per radiomodelli
- myKIT, misura di accelerazione con sensore triassiale



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

GESTIRE I MODULI GSM CON ARDUINO

di MARCO MARTINES

Dopo aver mostrato diversi esempi circa l'utilizzo del GSM/GPRS/GPS shield attraverso chiamate ed SMS, presentiamo ora una serie di applicazioni che coinvolgono la comunicazione dati GPRS. Grazie alla classe `inet.h` è possibile interfacciare lo shield al mondo di Internet e alle sue svariate applicazioni. Vedremo come utilizzare il nostro Arduino come client per la richiesta di una pagina Internet, come server per la ricezione di comandi e, infine, per l'invio di email, grazie alle nuove funzioni della libreria.

ESEMPIO 1

Prima di descrivere questo esempio, ricordiamo cosa accade quando aprendo il browser premiamo invio

dopo aver digitato l'indirizzo della pagina richiesta (ad esempio nel caso fossimo interessati alla pagina 'Chi Siamo' del nostro sito digiteremo http://www.elettronica.in/MyMain/ChiSiamo/p_chisiamo.asp). In seguito a tale operazione, il browser stabilisce una connessione TCP/IP con il server (identificato con l'indirizzo IP o, attraverso i DNS, con la stringa come nel caso in esempio www.elettronica.in). Una volta accettata la connessione, il browser invia una richiesta per farsi restituire la pagina interessata. Tale richiesta prende il nome di 'http request' dove al suo interno troviamo - oltre a diverse informazioni di base - anche il path dove è salvata la pagina che intendiamo visualizzare. Solitamente il path coincide con tutto quello che viene dopo il

Listato 1

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

InetGSM inet;
int k=0;
int j=0;
char str[]="Date: ";
char msg[50];
boolean found=false;
char data;
int numdata;
char inSerial[50];
int i=0;
boolean started=false;

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

  if(started){
    if (inet.attachGPRS("internet.wind", "", ""))
      Serial.println("status=ATTACHED");
    else Serial.println("status=ERROR");
    delay(1000);
    numdata=inet.httpGET("www.google.com", 80, "/", msg, 50);
  }
};

void loop()
{
  data=gsm.read();
  if(data>0){
    //Serial.print(data);
    if(data==str[k]){
      k++;
      if(k+1==strlen(str)){
        found=true;
        msg[0]='\0';
        j=0;
      }
    }else{
      k=0;
    }
  }

  if(found){
    if(data!='\n'){
      if(data>0){
        msg[j]=data;
        j++;
      }
    }else{
      msg[j]=data;
      msg[j+1]='\0';
      found=false;
      Serial.print("DATA: ");
      Serial.println(msg);
    }
  }
};
```

.it, .com e così via (in questo caso il path è MyMain/ChiSiamo/p_chisiamo.asp).

Il server, ricevendo tale richiesta, va alla ricerca della pagina seguendo il path indicato e restituisce una prima risposta con l'esito della ricerca (in caso di pagina non trovata restituirà il classico '404 Not Found') e altre informazioni come l'orario GMT impostato sul server o l'ultima modifica a tale pagina. A seguito di questa risposta, in caso di esito positivo della ricerca, il server comunicherà il codice HTML relativo e il browser si occuperà di interpretarlo per generare la pagina formattata come siamo abituati a vedere.

Tutto questo procedimento, sino al codice HTML, grazie allo shield e alla sua relativa libreria, è possibile replicarlo con un paio di funzioni.

In questo sketch ci occuperemo di richiedere una specifica pagina ad un server (è stata scelta la pagina principale del motore di ricerca Google), analizzare la risposta e, a partire da questa, estrapolare l'orario GMT.

A questo punto possiamo procedere ad osservare il relativo codice.

Per semplicità si è scelto di includere la fase di attivazione della connessione dati nel setup dello sketch. Tale scelta non è obbligatoria, ma è stata preferita per motivi di chiarezza, facendo così risaltare la parte vera e propria del programma all'interno della funzione di loop.

Dunque nel setup, dopo aver inizializzato lo shield, verrà eseguita la routine di inizializzazione della rete dati mobile. Ricordiamo che per tale operazione è sufficiente richiamare la seguente funzione

```
inet.attachGPRS(char* APN, char* username, char* password)
```

Negli esempi troverete la funzione settata con i parametri per il gestore WIND.

Anche la richiesta di connessione al server, essendo effettuata una sola volta, è stata inclusa nel setup.

Con tale chiamata viene richiesto al server, attraverso la porta specificata, la pagina relativa al path definito dal terzo parametro. Gli ultimi due parametri servono a memorizzare, all'interno della stringa passata con il quarto parametro, l'intera risposta o - eventualmente - una sua parte, la cui lunghezza viene definita con l'ultimo parametro.

La funzione restituisce il numero di caratteri salvati.

```
inet.httpGET("www.google.com", 80, "/", msg, 50);
```

In questo modo i primi 50 caratteri della risposta verranno salvati nella variabile msg. Questo, non

Abbiamo parlato di GSM...

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



Fig. A

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

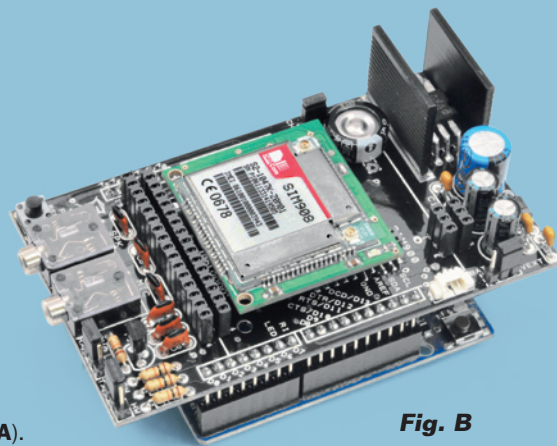


Fig. B

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

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

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

ha alcun scopo in questo esempio e viene eseguito solamente a fini dimostrativi. Potrete benissimo impostare questo parametro a 0. L'importante è che la porzione di testo che ci interessa analizzare (quella contenente la data e l'ora) non rientri tra i caratteri salvati in quanto effettueremo il controllo sui caratteri rimanenti.

In questo caso, i primi 50 caratteri vengono utilizzati dal server per comunicare altre informazioni. Vediamo ora di capire cosa succede alla parte restante della risposta che non viene salvata.

La libreria prevede l'utilizzo di un buffer, di dimensione definita all'interno della libreria stessa, impostabile a piacimento. Tale buffer viene utilizzato per memorizzare temporaneamente qualunque comunicazione sulla seriale software da parte del modulo SIM900, tra cui anche le risposte del server.

Attraverso il comando `gsm.read()` è possibile leggere (se presente) il primo dato (inteso come il meno recente in ordine di arrivo) memorizzato sul buffer. Se il buffer è vuoto verrà restituito il valore 0 (nota bene che è diverso dal valore '0' codificato in ASCII con il numero 48).

Dunque ad ogni ciclo del loop leggeremo il primo valore presente nel buffer.

```
data=gsm_read();  
if(data>0){ [...] }
```

Una volta verificato che effettivamente all'interno della variabile data sia presente un carattere non nullo, procederemo con la ricerca della stringa. La stringa contenente la data e l'orario è preceduta dal testo "Date:".

Noi ci occuperemo di ricercare tale valore, e una volta identificato, di memorizzare i successivi caratteri sino al carattere di 'a capo', identificato con '\n'. L'algoritmo di riconoscimento della stringa è molto semplice: ogni carattere viene confrontato con il primo della nostra stringa (in questo caso una 'D' maiuscola), quindi se c'è corrispondenza tra i due caratteri -quello letto e quello atteso - tramite l'incremento di una variabile contatore si comunica all'algoritmo che alla prossima iterazione dovrà andare a confrontare non più il primo carattere, bensì il secondo. Allo stesso modo per i successivi, sino a quando il numero di corrispondenze positive (contenuto all'interno della variabile contatore k) non è uguale alla lunghezza della parola attesa.

In caso di un carattere intermedio diverso, l'algoritmo si azzererà ricominciando a ricercare il primo valore della stringa attesa.

Nel caso venga trovata la parola richiesta, viene azzerato l'algoritmo e una variabile booleana dal nome found, viene settata a true.
A questo punto non ci resta che stampare a schermo tutti i successivi valori della risposta, sino al raggiungimento del carattere terminatore '\n', che ci comunica la fine della riga.

ESEMPIO 2

Spesso le pagine che visitiamo hanno form per consentire all'utente di inviare dati, tramite il browser, al server. Questi dati possono essere inviati in chiaro o criptati per questioni di sicurezza. I metodi più semplici per l'invio di dati in chiaro sono GET e POST.

Il metodo GET utilizza il path per l'invio di dati. Infatti, spesso osservando i link delle pagine durante la navigazione, ci accorgiamo che oltre al link della pagina compaiono le informazioni che avevamo appena digitato nel form precedente.

Un esempio può essere banalmente l'interrogazione di un motore di ricerca.

L'indirizzo di base di Google è *www.google.it* (notiamo che per le pagine principali non vi è da specificare il path, è sufficiente inserire '/' come path o non definirlo affatto), mentre dopo aver effettuato una ricerca diventa *http://www.google.it/#hl=it&tbo=d&output=search&sclient=psy-ab&q=elettronica&fp=1*. Vediamo come all'interno del link, più precisamente all'interno del path, sia presente la parola che abbiamo cercato: "elettronica".

Questo perché abbiamo comunicato al server, tramite il form di ricerca nella pagina principale di Google, di voler cercare tale parola e questo motore di ricerca utilizza il metodo GET per passare i para-

metri alla pagina del server che compie la ricerca. Quindi notiamo che non vi è niente di più semplice che effettuare una ricerca con lo shield, richiama la stessa funzione utilizzata in precedenza, scegliendo opportunamente il path.

```
inet.httpGET("www.google.com", 80, "/#hl=it&tbo=d&output=search&sclient=psy-ab&q=elettronica&fp=1", msg, 50);
```

Il metodo POST nasce per evitare di comunicare i dati in maniera così visibile, celando i parametri all'interno dell'header della richiesta HTTP spiegata precedentemente.

Per questo esiste la funzione *httpPOST* appartenente alla classe *InetGSM* che svolge questa operazione

```
inet.httpPOST(char* server, int port, char* path, *char param, *char risposta, int len_risposta);
```

Come è possibile notare, è molto simile alla chiamata di GET, dove in aggiunta specifichiamo i parametri con una stringa apposita e questi non appariranno nel link.

Vediamo ora come realizzare un programma che comunichi dei dati, tramite metodo POST, ad una pagina in PHP su un server che li riceve e li interpreta fornendo una risposta.

Per prima cosa riportiamo il codice della pagina PHP. Non essendo un argomento direttamente legato a questo articolo riporteremo semplicemente il codice senza descriverlo.

Il suo scopo è ricevere i dati, come nome ed età, tramite metodo POST e generare un messaggio di benvenuto personalizzato.

A questo punto non ci resta che, a partire dall'esempio precedente, sostituire la funzione per la richiesta di connessione al server in modo opportuno e rimuovere la parte per la ricerca della data e dell'ora. Una volta caricato lo sketch, potremo vedere dall'output seriale di Arduino la seguente scritta: "Welcome Marco
 You are 24 years old."

I caratteri
 fanno parte del codice HTML e sono interpretati dal browser come un 'a capo'. La libreria non prevede l'interpretazione del codice HTML, dunque sulla seriale potrete vedere tutto il codice relativo alla pagina.

Un'applicazione del genere risulta particolarmente utile laddove si voglia monitorare l'andamento nel tempo di alcuni sensori remoti salvandone le misurazioni ad intervalli prestabiliti.

L'utilizzo di un server per tale compito risulta adeguato e di semplice configurazione. Grazie a quanto mostrato con questi esempi potrete passare i parametri al sever nel modo che voi preferite e svi-

Listato 2

```
<?php
if( $_POST["name"] || $_POST["age"] )
{
    echo "Welcome ". $_POST['name']. "<br />";
    echo "You are ". $_POST['age']. " years old.";
    exit();
}
?>
<html>
<body>
<form action="<?php $_PHP_SELF ?>" method="POST">

Name: <input type="text" name="name" />
Age: <input type="text" name="age" />

<input type="submit" />
</form>
</body>
</html>
```

luppare le applicazioni più disparate dalla semplice raccolta dati alla pubblicazione di un 'tweet'.

NOVITA' SOFTWARE

Con l'ultima versione della libreria - giunta alla versione 3.05 al momento in cui scriviamo - sono state inserite diverse nuove funzioni tra cui alcune di notevole importanza per quanto riguarda l'invio della posta elettronica. Grazie a tali funzioni è possibile inviare email utilizzando la maggior parte dei provider di posta elettronica.

Essendo il tutto basato su Arduino, un dispositivo dalle risorse limitate, sono supportati tutti i provider che non richiedono una connessione protetta tramite SSL. Vediamo nella tabella quali sono le principali funzioni aggiunte: per l'invio delle email sono necessarie due funzioni. La prima consente, una volta impostati i parametri del nostro account di posta, di effettuare il login e inizializzare un messaggio definendo il destinatario e l'oggetto.

In seguito a tale operazione attraverso la funzione SimpleWrite sarà possibile definire il testo del messaggio.

Una volta comunicato il corpo dell'email, chiamando la funzione closemail si potrà concludere il messaggio e inviare l'email.

Ad esempio:

```
inet.openmail("smtp.email.it", "userbase64", "passbase64", "prova@email.it",
info@elettronica.in.it ", "Questo è l'oggetto");
gsm.SimpleWrite("Questo è il corpo del messaggio");
inet.closemail();
```

ESEMPIO 3

Vediamo ora un semplice esempio che utilizza le

Listato 3

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

InetGSM inet;
int k=0;
int j=0;
char msg[150];
boolean found=false;
char data;
int numdata;
char inSerial[50];
int i=0;
boolean started=false;

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

  if(started){
    if (inet.attachGPRS("internet.wind", "", ""))
      Serial.println("status=ATTACHED");
    else Serial.println("status=ERROR");
    delay(1000);

    numdata=inet.httpPOST("www.evildeejay.it", 80, _
"/test/test.php", "name=Marco&age=24",msg, 50);
  }
};

void loop()
{
  serialswread();
};

void serialswread(){
  gsm.SimpleRead();
}
```

<pre>int openmail(char* server, char* lo- ginbase64, char* passbase64, char* from, char* to, char* subj)</pre>	Serve per inizializzare l'invio di un'email. Il primo parametro è una stringa contenente il server smtp che utilizzeremo per l'invio del messaggio. A seguire troviamo username e password in base 64. Per il campo from solitamente è richiesto che sia uguale al proprietario della casella di posta elettronica che stiamo utilizzando. Gli ultimi due parametri sono il destinatario dell'email e l'oggetto del messaggio. Una volta chiamata questa funzione il modulo resterà in attesa del corpo del messaggio finché non viene chiamata la funzione di chiusura ed invio. In caso di successo la funzione restituisce 1, 0 altrimenti.	<pre>inet.openmail("smtp.email.it", "userbase64", "passbase64", "prova@email.it", "info@elettronica.in.it ", "Questo è l'oggetto");</pre>
<pre>int closemail()</pre>	Dopo aver digitato il testo del messaggio, viene chiamata questa funzione per chiudere l'email e inviarla. In caso di successo restituisce 1, altrimenti 0.	<pre>inet.openmail("smtp.email.it", "userbase64", "passbase64", "prova@email.it", "info@elettronica.in.it ", "Questo è l'oggetto"); gsm.SimpleWrite("Questo è il corpo del messaggio"); inet.closemail();</pre>

Listato 4

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

InetGSM inet;
int k=0;
int j=0;
char msg[150];
boolean found=false;
char data;
int numdata;
char inSerial[50];
int i=0;
boolean started=false;

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

  if(started){
    if (inet.attachGPRS("internet.wind", "", ""))
      Serial.println("status=ATTACHED");
    else Serial.println("status=ERROR");
    delay(1000);
  }
};

void loop()
{
  serialhwread();
  serialswread();
};

void serialhwread(){
  i=0;
  if (Serial.available() > 0){
    while (Serial.available() > 0) {
      inSerial[i]=(Serial.read());
      delay(10);
      i++;
    }

    inSerial[i]='\0';
    if(!strcmp(inSerial, "/END")){
      Serial.println("\n");
      inSerial[0]=0x1a;
      inSerial[1]='\0';
      gsm.SimpleWriteln(inSerial);
    }

    //Send a saved AT command using serial port.
    if(!strcmp(inSerial, "EMAIL")){
      inet.openmail("smtp.email.it", "userbase64", _
        "passbase64", "prova@email.it", "info@elettronica.it", _
        "Questo è l'oggetto");
      gsm.SimpleWrite("MESSAGGIO DI PROVA");
      inet.closemail();
    }
    else{
      Serial.println(inSerial);
      gsm.SimpleWriteln(inSerial);
    }
    inSerial[0]='\0';
  }
}

void serialswread(){
  gsm.SimpleRead();
}
```

nuove funzioni all'interno della libreria sviluppate per l'invio di messaggi di posta elettronica.

Utilizzeremo questo esempio per mostrare qualche altra funzione di base della libreria per chi si fosse perso le prime puntate dove è stato presentato lo shield. Per questo sketch ci occuperemo di stabilire una comunicazione base con il modulo in modo da poter comunicare con esso semplicemente attraverso il Serial Monitor di Arduino.

Per tale scopo è necessario che ogni comando inviato sulla seriale hardware dal PC ad Arduino venga inoltrato sulla seriale software con cui Arduino comunica con il modulo, e viceversa per le risposte del modulo. Per tale operazione è sufficiente inserire nel loop le due funzioni:

```
serialhwread();
serialswread();
```

con la prima ogni dato che viene letto dalla seriale hardware (quella tra PC e Arduino) viene inoltrato al modulo di SIMcom, con la seconda avviene il contrario: ogni carattere inviato dal modulo ad Arduino viene inoltrato sulla seriale hardware.

Per fare una semplice prova è sufficiente, una volta caricato e avviato lo sketch, conclusa la fase di inizializzazione, inviare il comando "AT" e attendere la risposta "OK" da parte del modulo. Con questo metodo potete comunicare direttamente con il modulo usando i comandi AT.

Alcune volte è necessario inviare un carattere terminatore, che tramite seriale non è possibile inviare, dunque all'interno della funzione serialhwread() troviamo un controllo che nel caso riconosca l'invio da parte dell'utente del comando "/END" lo interpreta inviando al modulo il carattere terminatore. Sfrutteremo proprio questo sistema di interpretazione dei comandi inviati per l'invio delle email. Infatti, nel caso venga riconosciuto il messaggio "MAIL" inviato su seriale dal PC, Arduino provvederà ad inviare un'email con i parametri pre-impostati.

Viene lasciato al lettore, nel caso sia interessato, di modificare uno degli esempi visti nelle precedenti puntate per l'invio automatico di un'email in caso venga verificata una determinata condizione sugli input di Arduino.

Per la codifica dello username e della password in base 64 si può utilizzare uno dei numerosi strumenti messi a disposizione da Internet per effettuare la codifica/decodifica. Uno dei tanti è <http://www.base64decode.org/> che consente di convertire una stringa ASCII in base 64.



CONTROLLER USB PER 4 STEPPER MOTOR

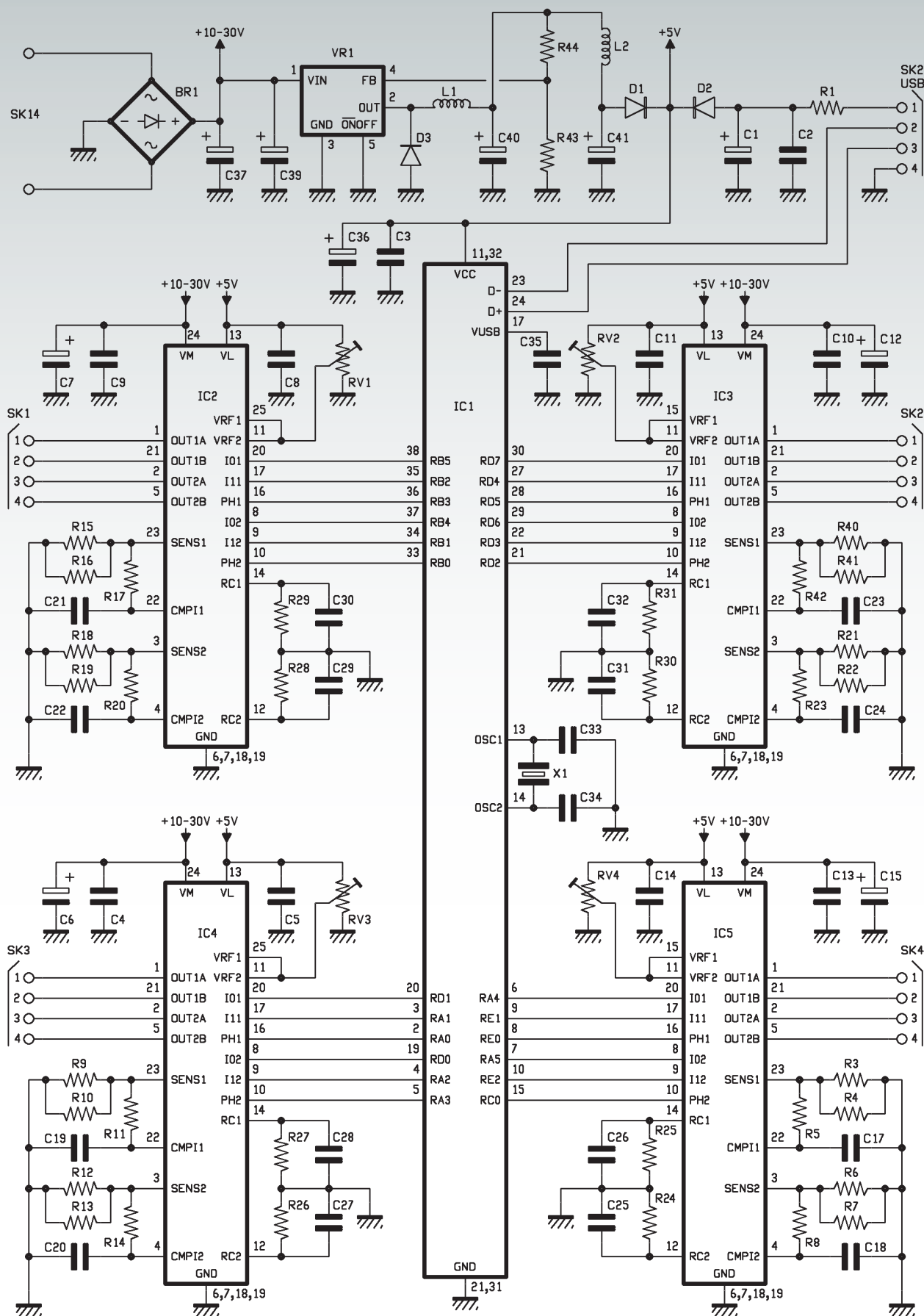
di DAVIDE SCULLINO

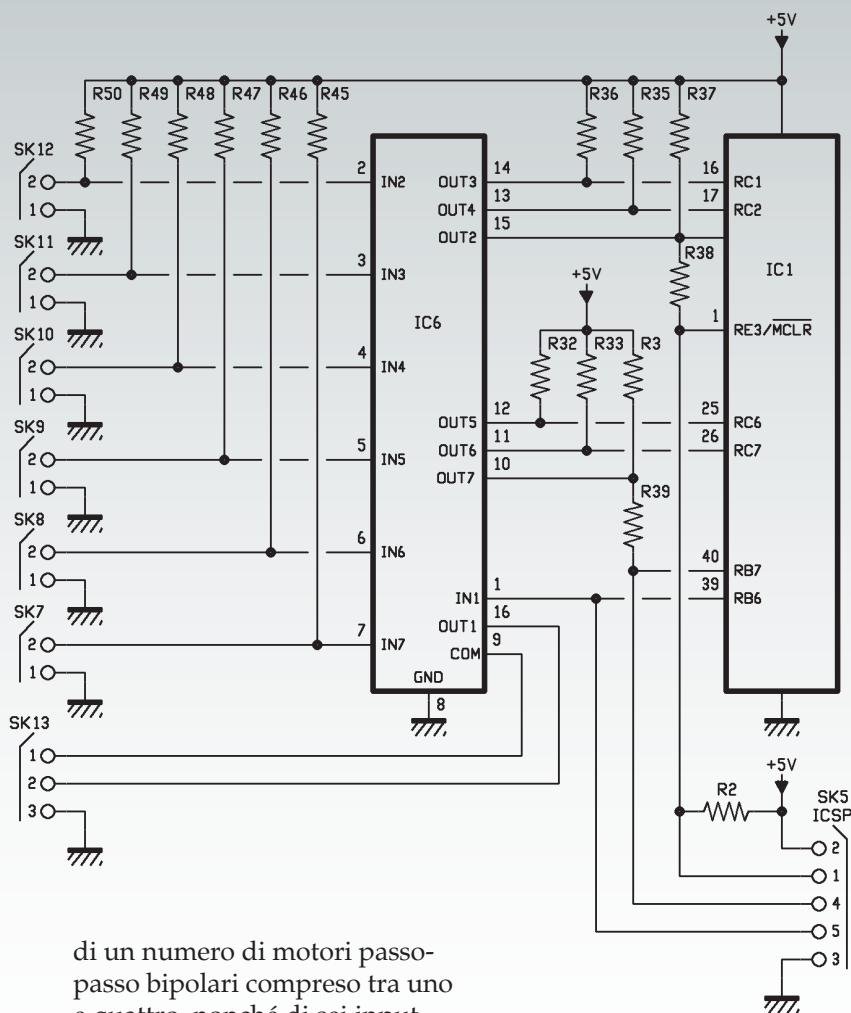


Permette di controllare da qualsiasi computer dotato di presa USB fino a quattro motori passo-passo bipolari, ma anche di attivare un utilizzatore mediante un'uscita open collector e leggere lo stato di 6 ingressi per contatti.

Sono moltissime le macchine, comprese quelle da ufficio (stampanti, scanner...) in cui per muovere meccanismi si impiegano i motori passo-passo, in virtù della possibilità che questi offrono di poter posizionare con precisione un carrello o un leveraggio in quanto il loro funzionamento prevede l'avanzamento a passi e l'arresto dell'albero del rotore in posizioni determinabili con assoluta precisione e stabilità. Il controllo di questi motori è un po' più complesso di quello del classico motore in corrente continua e richiede circuitazioni specifi-

che o la gestione da computer attraverso interfacce come quella qui descritta, che nasce per svolgere due funzioni: controllare il movimento di un massimo di quattro stepper motor bipolari (i motori bipolari sono quelli nei cui avvolgimenti, durante il funzionamento, scorre corrente in entrambi i versi) e gestire l'attività di sei ingressi per contatti e di un'uscita open collector. La scheda di cui parliamo è sostanzialmente una completa interfaccia di controllo per una macchina a controllo numerico (CNC) giacché dispone delle uscite per controllare l'attività





di un numero di motori passo-passo bipolari compreso tra uno e quattro, nonché di sei input con cui si può leggere l'attività di interruttori finecorsa, pulsanti di vario genere (ad esempio quello per lo stop di emergenza) attivati dall'operatore, sensori con uscita a contatto pulito o NPN open collector e via di seguito. Inoltre, l'uscita open collector può essere impiegata per azionare un utilizzatore su comando diretto da computer, ovvero dare e togliere l'alimentazione, tramite un relé, all'intera macchina o al gruppo dei motori.

Ma vediamo subito in cosa consiste l'hardware della periferica, rimandando il discorso sul software di controllo da PC alla fine di questo articolo.

SCHEMA ELETTRICO

La scheda è costituita da un microcontrollore dotato di inter-

faccia USB e da quattro driver per motori passo-passo bipolari o motori in continua tradizionali, ciascuno dei quali contiene due ponti ad H. Completano l'insieme un line-driver ULN2003 che serve a far leggere al microcontrollore lo stato dei cinque ingressi (oltre che a comandare l'uscita open collector) ed uno stadio di alimentazione che permette di ricavare le tensioni occorrenti a tutto il circuito partendo da un'alimentazione esterna, che può essere fornita sia in continua che in alternata.

Il microcontrollore è un PIC18F4450 della Microchip, caratterizzato da un'architettura ad 8 bit, dotato di memoria di programma di tipo Flash da 16 kB,

RAM da 768 byte, EEPROM dati da 256 byte, quattro timer (uno da 8 bit e 3 da 16 bit), un ADC 10 bit a 13 canali con campionamento a 100 kpsps ed un modulo PWM; inoltre questo micro integra periferiche di comunicazione quali un USART e ad una SPI/I2C-Bus, ma, cosa più importante, dispone di una porta USB Full Speed (12 Mbps) compatibile USB 2.0 dotata di buffer RAM bidirezionale da 256 byte.

Dovendo gestire una comunicazione veloce come quella in USB 2.0 Full-Speed, il PIC18F4450 prevede un'elaborazione veloce, che necessita di un clock relativamente elevato per un microcontrollore Microchip, che può arrivare a 48 MHz; nel nostro caso optiamo per un quarzo da 12 MHz, che fa lavorare l'oscillatore interno alla frequenza base di 12 MHz, moltiplicata internamente con l'aiuto di un PLL fino ad ottenere i 48 MHz.

Il micro è interfacciato al computer mediante la connessione USB, che esternamente fa capo ai piedini 23 e 24.

Del funzionamento del PIC ci occuperemo più avanti; ora vediamo come funzionano i controller dei motori passo-passo, che sono tutti L6219. L'integrato L6219 è un driver monolitico (incapsulato in contenitore DIP plastico) prodotto dalla ST, che contiene due ponti ad H a transistor BJT, adatto al controllo dei motori passo-passo bipolari, ma anche utilizzabile per il comando di motori in continua tradizionali con inversione del verso di rotazione.

Il componente ST è costruito per essere gestito tramite linee TTL-compatibili, ossia con tensioni 0/5 V, il che ci permette di controllarlo direttamente con gli I/O del microcontrollore. Ciascuno dei ponti può sopportare tensioni

CARATTERISTICHE TECNICHE

- Tensione di alimentazione c.a.
- Tensione di alimentazione c.c.
- Corrente assorbita
- Corrente continuamente erogata ai motori
- Corrente di picco per motore
- Protezione da sovracorrente
- Protezione termica
- 6 ingressi per contatti
- 1 uscita open collector 500 mA

9÷24 Veff.
10÷30 V
max 8A
2x750 mA
1 A

continue fino a 46V ed integra quattro diodi di protezione dei transistor di commutazione, utili a sopprimere le correnti inverse che si generano negli avvolgimenti all'interruzione dell'alimentazione.

L'integrato è provvisto di una protezione termica, una da sovracorrente ed un'altra che previene la conduzione incrociata, ossia la conduzione simultanea dei due lati di ogni ponte durante l'inversione di direzione del motore (che comporta un'inversione del verso della corrente erogata). Quanto alla protezione termica, interdice gli stadi di potenza (più esattamente, interdice i transistor superiori dei ponti) nel caso la temperatura del chip ecceda il margine operativo; la protezione da sovraccarico

fa altrettanto se viene superata la corrente limite, che è 750 mA a regime continuativo e fino ad 1 A nei picchi.

I ponti dell'integrato vengono pilotati dagli impulsi prodotti da un modulatore PWM (Pulse Width Modulation) interno, che a sua volta pilota i comparatori utilizzati per impostare la corrente di uscita e la protezione in corrente che il componente prevede. Attraverso il piedino PHASE corrispondente (PHASE1 vale per il ponte 1 e PHASE2 per il ponte 2) si imposta il verso di rotazione del motore, ovvero la polarità degli impulsi alle metà di ciascun ponte ad H.

Il funzionamento dell'L6219 è praticamente basato sul modulatore PWM, che in un comparatore confronta una forma d'onda triangolare con una tensione di riferimento proveniente dall'esterno attraverso i piedini VREF1 (per

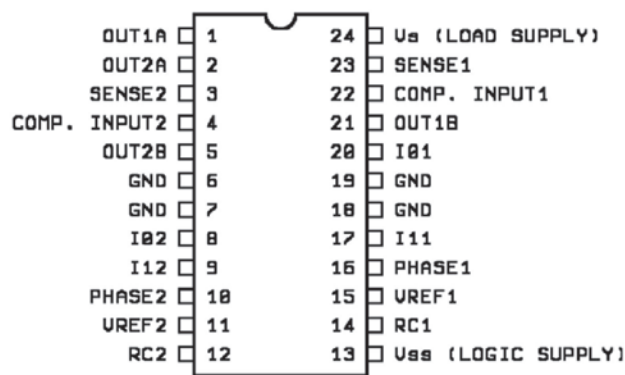
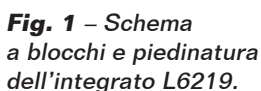
il ponte 1) e VREF2 (per il ponte 2), che nel nostro circuito sono uniti e polarizzati dalla tensione fornita da un unico trimmer; quest'ultimo è RV1 per IC2, RV2 per IC3, RV3 per IC4 ed RV4 per IC5. Regolando tale trimmer è possibile impostare la massima corrente che i ponti potranno erogare; più esattamente, maggiore è Vref, minore sarà la corrente erogata e viceversa.

I piedini I01 e I11, nonché I02 e I12, intervengono su una logica che interagisce con il comparatore di limitazione della corrente e permettono di definire la soglia di corrente erogabile dai transistor del ponte; quando entrambi sono a livello alto il ponte si disattiva (ciò si realizza interdicendo i transistor superiori del ponte corrispondente). Con I0 a livello basso e I1 allo stato logico alto, il ponte eroga 1/3 della corrente massima, mentre con l'impostazione contraria (1 0) la corrente erogata raggiunge i 2/3 del massimo; con entrambi i piedini a livello basso, si abilita il ponte ad erogare il massimo della corrente permessa.

La limitazione della corrente viene ottenuta intervenendo sulla soglia di commutazione del comparatore del modulo PWM, ovvero facendo interdire i transistor prima o dopo, a seconda del valore di corrente desiderato. Sullo stesso comparatore interviene la protezione in corrente, realizzata rilevando la corrente erogata da ciascun ponte (il rilevamento si effettua sull'emettitore) mediante le resistenze collegate tra i piedini SENSE1 (per il ponte 1) e SENSE2 (per il ponte 2) e retrocedendone parte agli ingressi COMPARATOR INPUT 1 e COMPARATOR INPUT 2: quando la tensione determinata dalla corrente di uscita ai capi delle predette resistenze supera



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

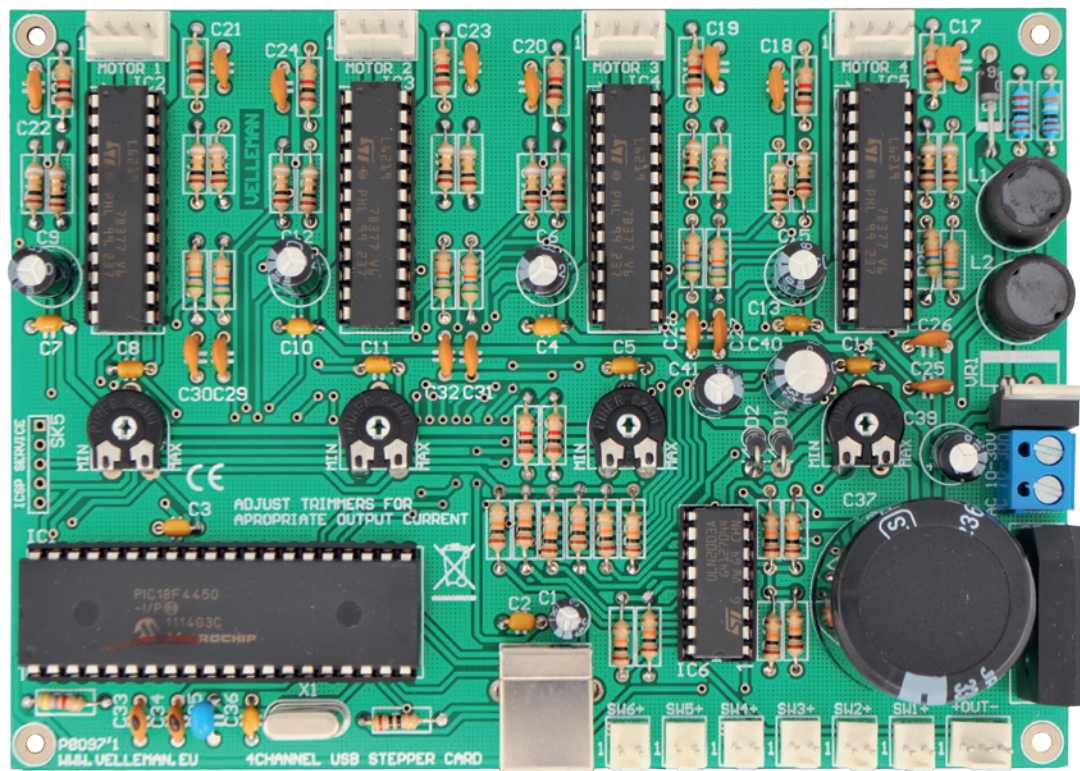
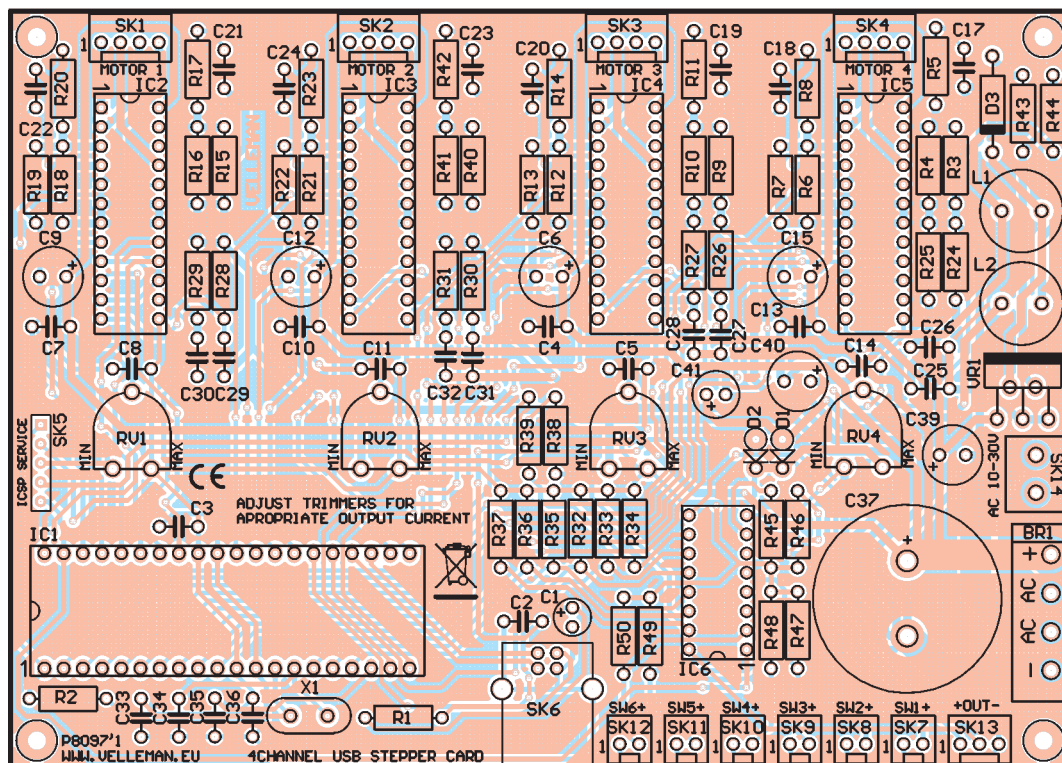


con quello degli altri diodi e si connette internamente al piedino COM D. I diodi servono, quando si pilotano carichi induttivi, a sopprimere le tensioni inverse generate dall'interruzione della corrente che l'apertura dei transistor determina. I collettori dei Darlington sono tutti collegati insieme al piedino GND dell'integrato. Nel nostro circuito usiamo i line-driver dal 2 al 7 per leggere lo stato di altrettanti ingressi; ogni input dell'ULN2003 è provvisto di resistore di pull-up in modo da mantenere, a riposo (cioè ad ingresso aperto) il relativo Darlington in saturazione e quindi la corrispondente uscita a zero logico. Le uscite dei line-driver 2÷7 vengono lette dalle linee RE3, RC1, RC2, RC6, RC7, RB7 e sono tutte fornite di resistore di pull-up; notate che sia RE3 che RB7 hanno in serie all'uscita una resistenza da 1 kohm, perché si tratta di linee che nella programmazione in-circuit sono in comune con il programmatore collegato al connettore ICSP. Infatti RE3,

che nel normale funzionamento è un I/O senza pull-up, in programmazione diventa /MCLR dell'ICSP, mentre RB7, anch'esso I/O senza pull-up, in programmazione funziona da linea PGD dell'ICSP.

Siccome in condizione di riposo gli ingressi della scheda, che sono provvisti di pull-up, sono tutti polarizzati e causano la saturazione dei Darlington interni all'ULN2003, le uscite OUT2, OUT3, OUT4, OUT5, OUT6 e OUT7 si trovano normalmente a zero logico, pertanto se si collegasse il programmatore le linee /MCLR e PGD sarebbero fisse a zero e la programmazione in-circuit non potrebbe avere luogo correttamente perché verrebbero praticamente cortocircuitate. Inserendo R38 ed R39, indipendentemente dal livello logico assunto dai piedini 10 e 15

[piano di **MONTAGGIO**]



Elenco Componenti:

R1: 10 ohm
R2: 4,7 kohm
R3, R4, R6, R7, R9, R10, R12, R13, R15, R16, R18, R19, R21, R22, R40, R41: 1 ohm
R5, R8, R11, R14, R17, R20, R38, R39, R42: 1 kohm
R23÷R31: 56 kohm
R32÷R37: 10 kohm
R43: 1,2 kohm
R44: 3,9 kohm
R45÷R50: 10 kohm
RV1÷RV4: Trimmer 10 kohm MO
C1: 4,7 µF 50 VL elettrolitico
C2÷C5: 100 nF ceramico
C6: 100 µF 35 VL elettrolitico
C7, C8: 100 nF ceramico
C9: 100 µF 35 VL elettrolitico
C10, C11: 100 nF ceramico
C12: 100 µF 35 VL elettrolitico
C13, C14: 100 nF ceramico
C15: 100 µF 35 VL elettrolitico
C16: -
C17÷C32: 820 pF ceramico
C33, C34: 22 pF ceramico
C35: 470 nF ceramico
C36: 100 nF ceramico
C37: 3300 µF 35 VL elettrolitico
C38: -
C39: 100 µF 35 VL elettrolitico
C40: 330 µF 16 VL elettrolitico
C41: 100 µF 16 VL elettrolitico
D1, D2: 1N4007
D3: SB130
X1: Quarzo 12 MHz
VR1: LM2575TADJ
L1, L2: Bobina 330 µH
BR1: Ponte diodi RS603
IC1: PIC18F4450-I/P (VK8097)
IC2÷IC5: L6219
IC6: ULN2003

Varie:

- Morsetto 2 poli
- Zoccolo 8+8
- Zoccolo 12+12 (4 pz.)
- Zoccolo 20+20
- Connettore USB-B da CS
- Connettore ad innesto 2 vie (6 pz.)
- Connettore ad innesto 3 vie (1 pz.)
- Connettore ad innesto 4 vie (4 pz.)
- Piedini adesivi (4 pz.)
- Circuito stampato

dell'ULN2003, /MCLR e PGD possono assumere sia lo stato zero, sia lo stato 1.

Il primo line-driver è invece usato come uscita, quindi il suo ingresso viene comandato dal PIC attraverso la linea RC7; la sua uscita è disponibile sul contatto centrale dell'apposita morsettiera, alla quale si deve collegare, tra il morsetto 1 ed il 3, la stessa tensione che alimenta il carico da comandare con l'uscita open collector.

Stabilito ciò, possiamo analizzare il funzionamento dell'intero circuito, ossia spiegare come lavora il microcontrollore: dopo il power-on-reset, il firmware che gira in esso inizializza gli I/O impostando RB0÷RB5 come uscite dedicate al controllo dell'L6219 IC2, RD2÷RD7 come output per il controllo dell'IC3, RD0, RD1, RA0, RA1, RA2 e RA3 come uscite per gestire IC3 e RA4, RA5, RE0, RE1, RE2, RC0 come uscite per comandare IC5. Questo è quanto riguarda gli L6219 e quindi la gestione dell'attività dei motori passo-passo.

La linea RB6 è inizializzata come uscita per comandare l'IN1 dell'ULN2003, mentre le linee RE3, RC1, RC2, RC6, RC7 ed RB2 vengono anch'esse inizializzate come ingressi senza pull-up interno e servono a leggere lo stato dei line-driver 2, 3, 4, 5, 6 dell'ULN2003.

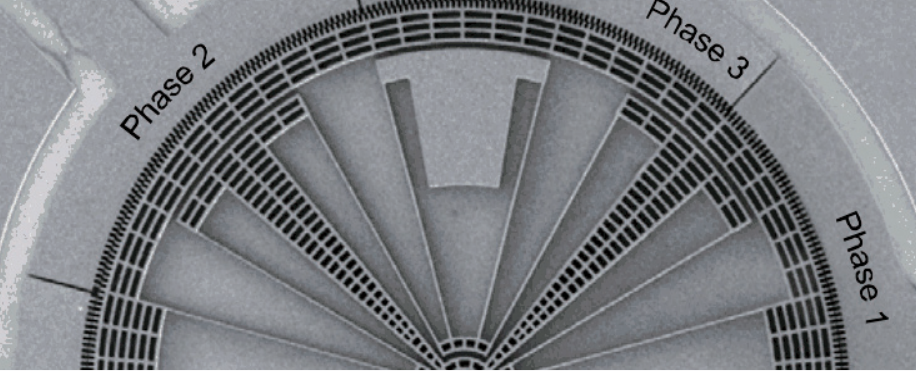
Quanto alle linee dedicate al controllo del motore passo-passo, RB5 ed RB2 impostano la corrente limite nel motore o il blocco del driver del primo ponte dell'IC2, mentre RB4 e RB1 fanno altrettanto con il secondo ponte dello stesso integrato; invece RB3 fornisce il segnale che imposta la polarità degli impulsi di corrente forniti dal ponte 1 ed RB0 l'analogo segnale che definisce il verso della corrente fornita dal ponte

2. Per IC3, RD4 e RD7 impostano la corrente nel primo ponte, RD3 e RD6 quella nel secondo ponte, mentre RD2 ed RD5 gestiscono il verso di rotazione. Per IC4, RA0 ed RA3 comandano il verso di rotazione, mentre RD1, RA1 ed RD0, RA2 impostano la corrente degli impulsi rispettivamente per il ponte 1 e 2; infine, per IC5, il controllo della corrente viene effettuato da RA4, RA5, RE1, RE2 e quello del verso di rotazione da RE0 ed RC0.

Le altre linee del microcontrollore sono dedicate all'interfaccia USB (RC4/D-/VM è il canale dati D-, mentre RC5/D+/VP è il canale D+) ed alla programmazione in-circuit (RA3/MCLR/VPP); notate che RA1 ed RA2, che nel normale utilizzo gestiscono il traffico dei dati sull'USB, in programmazione servono a programmare il micro (mediante il connettore ICSP).

Una volta completata l'inizializzazione degli I/O, il firmware gira in loop attendendo l'arrivo di dati dal computer via USB ed acquisendo ciclicamente lo stato dei sei ingressi, che invia al computer; quando arriva un comando da PC riguardante l'uscita open collector o l'azionamento di uno dei motori passo-passo, ne avvia l'esecuzione. Per quanto riguarda l'uscita open collector, agisce sulla linea RB6, mentre se i comandi riguardano il motore imposta le linee di controllo PHASE o I a seconda che debba regolare il verso di rotazione o la corrente.

Giunti a questo punto, non ci resta che descrivere l'ultimo stadio del circuito: quello di alimentazione. L'intera scheda richiede una tensione d'alimentazione continua o alternata: nel caso della continua, servono da 11 a 30 volt, mentre applicando l'alternata occorrono da 9 a 24 Veff.



Non solo per lavori pesanti

Quando si pensa agli stepper motor, solitamente si immagina qualcosa di grande, massiccio e destinato a macchine che spostano carrelli e bracci di una certa importanza. L'utilità di questi motori arriva però anche in settori dove la possibilità di definire un angolo di rotazione o uno spostamento molto preciso e su scale molto ridotte risulta indispensabile per migliorare le attuali tecnologie. Ecco quindi i nano stepper motor all'interno dei pickup per BluRay, minuscoli e con soli 20 step per una rotazione completa, ma con 0,4 grammi di massa; così piccoli e leggeri sono integrati nel pickup e permettono di spostare con precisione la lente che focalizza il raggio a seconda della lunghezza d'onda necessaria (infrarosso, rosso o blu). Ancora più minuscoli sono quelli realizzati sperimentalmente con tecnologia MEMS per inclinare le testine degli hard disk direttamente sulla punta del braccio, compensando la perdita di tangenza rispetto alla traccia quando la testina si sposta verso le zone più interne o esterne del disco; il tutto con dimensioni minuscole e risposta adeguata alla velocità con cui il braccio attuatore si sposta.

L'alimentatore della scheda è composto da due stadi: un raddrizzatore a ponte di Graetz integrato, che raddrizza la tensione d'ingresso, ottenendo impulsi che, filtrati dal C37 e dai condensatori elettrolitici posti a valle, sviluppano una componente continua che fa funzionare il driver per gli stepper-motor; un regolatore DC/DC switching il cui ingresso è alimentato dalla tensione presente a valle del solito ponte a diodi e che serve a fornire i 5 Vcc stabilizzati alla logica. Notate che alimentando la scheda in continua, il ponte raddrizzatore consente di applicare la tensione senza riguardo per la polarità, dato che il ponte di Graetz presenta la stessa polarità tra i punti di uscita + e - indipendentemente da qual è la polarità all'ingresso (punti ~). Lo stadio DC/DC impiega il regolatore switching LM2575-ADJ

della National Semiconductors per ricavare circa 5,3 volt stabilizzati; l'integrato riceve dal ponte di Graetz la tensione non regolata, filtrata dall'elettrolitico C37; i piedini 3 e 5 sono collegati a massa, mentre il 4 riceve la tensione di feedback dall'uscita mediante il partitore resistivo formato da R44 ed R43, che determina la tensione che l'LM2575-ADJ deve fornire. Più esattamente, la Vout è legata al valore di dette resistenze dalla relazione:

$$V_{out} = V_{ref} (1 + R44/R43)$$

dove Vref è la tensione del riferimento interno e vale 1,23 V. Il piedino 2 dell'integrato fa capo al transistor di uscita, che fornisce impulsi di corrente all'induttanza L1 intervallati da pause nelle quali si interdice; durante tali pause, L1 restituisce l'energia

immagazzinata forzando un flusso di corrente nel condensatore C40, che pertanto viene caricato e assume una tensione continua di 5,3 volt. La funzione del diodo veloce D3 è impedire che, durante le pause, l'extratensione inversa dovuta alla scarica dell'induttore L1 danneggi lo stadio finale dell'LM2575-ADJ; ma non solo, perché l'SB130, proprio andando in conduzione in tali fasi, consente di chiudere il circuito della bobina su C40, consentendone la carica che altrimenti sarebbe impossibile.

La componente continua presente ai capi dell'elettrolitico C40 viene filtrata dalla cella LC formata dall'induttore L2 e dal condensatore C41, il cui scopo è sopprimere eventuali residui di commutazione in modo da fornire al PIC una tensione pulita e priva di disturbi che potrebbero alterarne il funzionamento.

Il diodo D1 porta la tensione ricavata dall'LM2575-ADJ alla linea di alimentazione a 5 V - per il microcontrollore e la logica - ed è stato inserito perché il circuito prevede la possibilità di essere alimentato, almeno per le sole funzioni di dialogo con il PC, acquisizione dei cinque input e controllo dell'uscita on/off, tramite la connessione USB. A tal proposito la linea 5V VBUS in arrivo dal connettore USB, filtrata da R1, C1 e C2, viene portata al catodo del D1 mediante il diodo D2. I due diodi permettono di fornire al microcontrollore poco meno di 5 volt (nel caso dell'alimentazione tramite un alimentatore, si tratta di 5,3-0,7 V della caduta sul diodo, mentre con l'USB la tensione è 5-0,7 V) indipendentemente da quale sia la fonte di alimentazione.

REALIZZAZIONE PRATICA

Bene, giunti a questo punto possiamo vedere come si costruisce

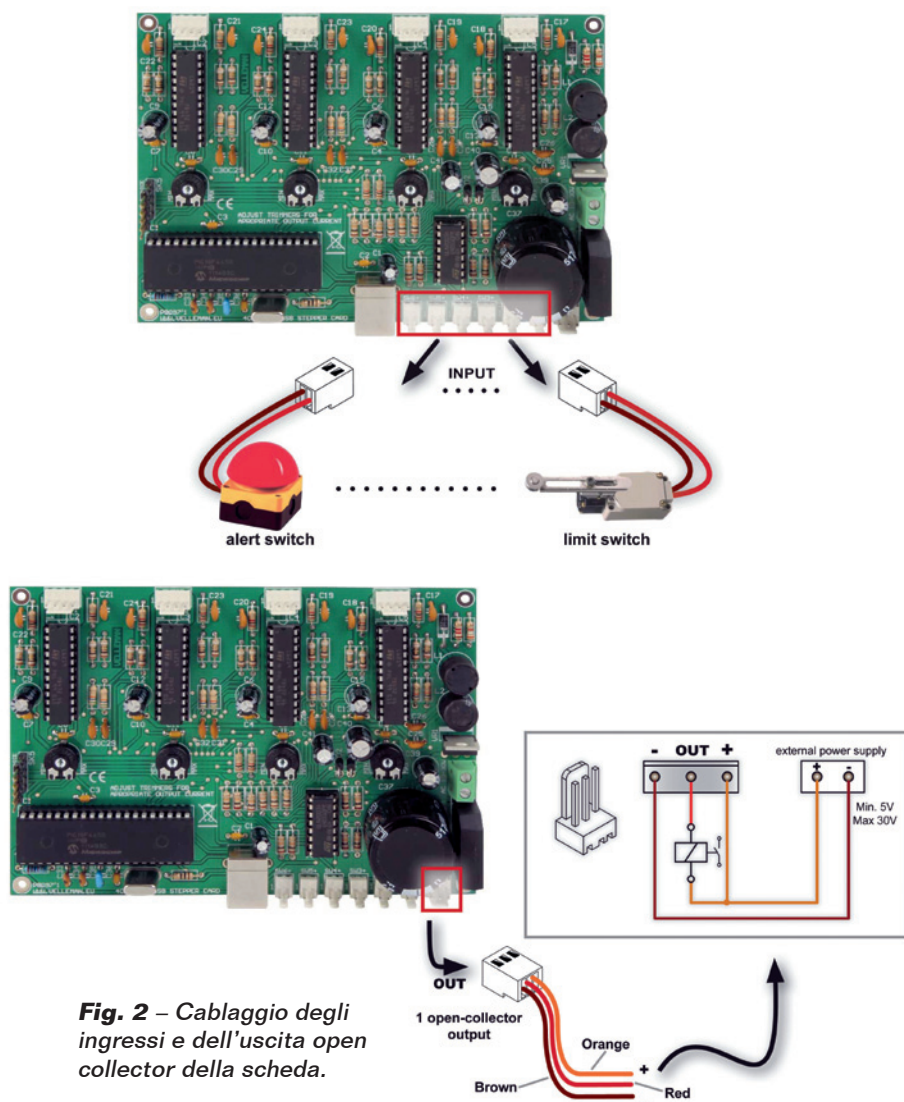


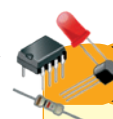
Fig. 2 – Cablaggio degli ingressi e dell'uscita open collector della scheda.

la scheda USB descritta in queste pagine, per la quale è stato previsto un circuito stampato a doppia faccia ottenibile per fotoincisione e del quale trovate nel nostro sito le tracce lato rame in formato stampabile. Una volta in possesso della basetta, non resta che saldarvi i componenti a partire da quelli a più basso profilo, ossia resistenze, diodi, trimmer e zoccoli per microcontrollore, ULN2003 e per gli L6219, quindi i condensatori, partendo da quelli non polarizzati. Proseguite con il quarzo, la presa USB e il ponte a diodi. L'integrato LM2575ADJ deve essere montato in verticale e senza alcun dissipatore, dato

che la corrente che dovrà erogare è piuttosto ridotta. Per le connessioni con le bobine dei motori passo-passo bisogna utilizzare dei connettori maschio quadripolari tipo JST o delle morsettiere miniatura passo 2,54 mm; quanto agli ingressi per contatti e all'uscita on/off, potete usare morsettiere miniatura o connettori a passo 2,54 mm rispettivamente bipolari e tripolare. Per l'alimentazione è invece prevista una morsettieria bipolare da c.s. a passo 5 mm. Non dimenticate di inserire e saldare negli appositi fori un pin-strip a 5 poli che vi servirà per la programmazione in-circuit del microcontrollore o

per l'eventuale debug. Per tutte le connessioni tra il motore, gli switch degli ingressi, l'uscita ecc. fate riferimento alla Fig. 2 e alla Fig. 3, che illustrano il cablaggio delle varie sezioni usando i connettori preposti. Completato il circuito e inseriti gli integrati nei rispettivi zoccoli, dopo la programmazione del PIC (che va fatta senza fornire alimentazione in quanto provvede la connessione USB) potete alimentare il tutto con un alimentatore da rete capace di fornire la tensione e la corrente richiesta dal motore passo-passo che intendete controllare. Se usate l'uscita on-off, l'alimentazione verrà fornita dallo stesso circuito di alimentazione dell'utilizzatore che andrete a controllare con essa.

Diciamo che per alimentare la scheda basta una corrente di 2 ampere. Ora accendete il computer e installate in esso il file eseguibile contenente il software di controllo, ottenuto scompattando l'archivio *motorsetup.zip*

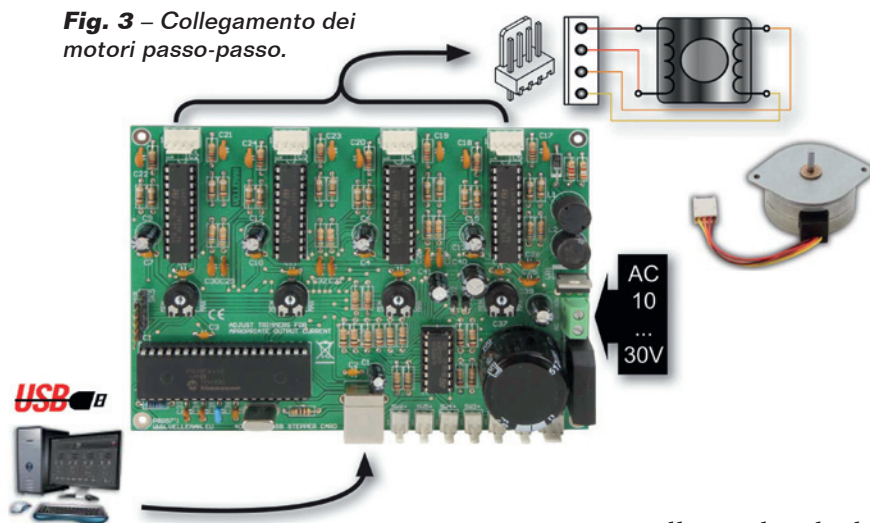


per il MATERIALE

Questo controller USB per step-motor è disponibile in scatola di montaggio. Il kit - prodotto da Velleman e distribuito da Futura Elettronica - comprende tutti i componenti, la basetta forata e serigrafata, il micro programmato e le minuterie. Non sono compresi l'alimentatore e i motori passo-passo. Il kit (cod. K8097) costa 84,00 Euro IVA compresa.

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

Fig. 3 – Collegamento dei motori passo-passo.



scaricabile dal nostro sito www.elettronica.in.it; vi sarà altresì necessario installare i driver della scheda, che poi si identificano con i driver Microchip del microcontrollore contenente la porta USB, opportunamente customizzati. I driver verranno richiesti al primo collegamento della scheda al computer, quindi suggeriamo di installarli prima del software di controllo, così da fare in modo che una volta installato, questo sarà in grado di rilevare l'unità.

IL SOFTWARE DI CONTROLLO

Si tratta di un'utility che gira in ambiente Windows (compatibile con Windows 2000/XP, Vista, Windows 7) e che una volta installata ed avviata mostra una schermata con il pannello di controllo visibile in Fig. 4. Nel menu a tendina Card Type dovete scegliere K8097 e, nella casella sottostante (Port), definire la porta di comunicazione (COM virtuale) corrispondente all'USB utilizzato dal computer per connettersi alla scheda di controllo. Fatto ciò potete iniziare a verificare come funziona l'insieme, anche aiutandovi con la funzione DEMO, che permette di emulare le funzionalità anche

senza aver collegato la scheda. Se quest'ultima è effettivamente collegata al computer, facendo clic sul pulsante Connect avviate il dialogo con essa e potete controllare il motore passo-passo e l'uscita on/off open collector, ovvero acquisire lo stato degli ingressi. Notate che il software di controllo è universale e vale sia per la scheda qui descritta, sia per l'unità di cui ci siamo già occupati in passato e che è in grado di gestire un solo motore. I comandi sono intuitivi; ciascun "Motor" dispone dei pulsanti Left e Right facendo clic sui quali potete impartire i

comandi di rotazione rispettivamente levogira e destrogira, oltre che di Torque e Stop, che rispettivamente avviano la rotazione e la bloccano. Nelle caselle Steps e Speed potete definire il numero di passi/giro che il motore dovrà compiere (in accordo con le caratteristiche dello stepper-motor) e di giri/minuto (anche questo valore dovrà essere compatibile con i dati del motore passo-passo che userete).

Nell'archivio zip contenente il software di controllo e i driver, si trovano anche utili contenuti come la dll che potete incorporare in software personalizzati da voi realizzati, per integrare la gestione della scheda in altri programmi; il materiale fornito a supporto è presente insieme al software (completata l'installazione, nel menu Avvio/Start>Tutti i programmi, si trova la voce Velleman, dalla quale si accede ai vari contenuti) e permette anche di associare agli ingressi la gestione manuale del motore passo-passo.



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

LA CARICA DEI PINGUINI

di MARCO MAGAGNIN

Un'analisi dell'attuale situazione delle board con i microcomputer ARM RISC in grado di funzionare con GNU/Linux. E ancora una volta, questo è solo l'inizio. Prima puntata.



Chi ha seguito la serie di articoli sulla rivista dedicati a RaspberryPi si ricorderà che avevamo previsto la crescita e la diffusione dei dispositivi in miniatura ARM RISC che possono funzionare con GNU/Linux. Quello che non avevamo previ-

sto è stato il fatto che, più che di crescita, possiamo parlare di esplosione, cioè dell'arrivo sul mercato di dispositivi eterogenei, con caratteristiche diverse e che richiedono procedure diverse per "addomesticarli" all'utilizzo del sistema operativo

GNU/Linux. In questo articolo vi presentiamo da una parte gli elementi unificanti dei diversi dispositivi e, dall'altra, classifichiamo ciascun dispositivo in base alle migliori possibilità di impiego. Iniziamo a elencare i dispositivi dei quali ci

occupiamo in questo articolo, ben sapendo che al momento dell'uscita della rivista ne saranno già apparsi dei nuovi di cui ci occuperemo in seguito.

- RaspberryPi
- PcDuino
- BeagleboneBlack
- CubieBoard

Fig. 1

	RaspberryPi	PcDuino	BEAGLEBONE BLACK	CUBIEBOARD	MK802	MK802II	MK802III	MK802IIIS	ODROID-U2
SoC	Broadcom BCM2835	AllWinner A10	Sitara XAM3359AZCZ100 (Texas Instruments)	AllWinner A10 1GHz	AllWinner A10 1GHz	AllWinner A10/ 1GHz Cortex-A8	CPU RK3066 Dual Core - Cortex-A9 1.6GHz	CPU RK3066 Dual Core - Cortex-A9 1.6GHz	Samsung Exynos4412 Prime 1MB L2 cache
CPU	700 MHZ ARM1176JZF-S	ARM Cortex A8 1 GHz	ARM Cortex A8 1 GHz	ARM Cortex-A8 coprocessori VFPv3 e NEON	Cortex A8 single core	Single core 1Ghz- Cortex A8	Cortex-A9 dual core	Cortex-A9 dual core	Cortex-A9 quad Core
GPU o processore grafico	Broadcom VideoCore IV	OpenGL ES2.0, OpenVG 1.1 Mali 400 core	SGX530 3D, 20M Polygon/S	Mali 400 MP	GPU: AMDZ160 con accelerazione 2D/3D/ OpenGL ES2	MALI 400	Quad- Core 2D/ 3D/ OpenGL ES2.0(AMD Z430)/ OpenVG1	Quad-Core 2D/ 3D/ OpenGL ES2.0(AMD Z430)/ Open- VG1.1	3D accelerator Mali-400 Quad Core 440MHz
RAM	512MB (mod. A, rev. 2)	1GB	512MB	1GB	512MB 1GB	1GB	1GB	1GB	2GB LP-DDR2
RAM DATI		2GB Flash	2GB embedded MMC	NAND (max 64GB) SATA II, SD Card 3.0	MEMORIA: 4GB interna + slot SD fino a 32GB	MEMORIA: 4GB interna + slot SD fino a 32GB	8GB Flash	8GB Flash	
Video	HDMI Video composito	HDMI	HDMI	HDMI VGA, TV e Webcam tramite GPIO	HDMI	HDMI	HDMI	HDMI	1080p HDMI
Audio	HDMI Jack OUT	HDMI	HDMI	HDMI Jack IN Jack OUT	HDMI	HDMI	HDMI	HDMI	Standard 3.5mm headphone jack HDMI Digital
USB	2	2 porte standard e una mini	2	2/3	USB: 2 porte, una mini, l'altra standard	USB: 2 porte, una mini, l'altra standard	Ports (male),Micro SD slot, USB host (full sizeUSB), USB power port (micro USB); LED(Blue) I(male),Micro SD slot, USB host (full sizeUSB), USB power port(micro USB); LED(Blue)	Micro SD slot, USB host*1,USB OTG*1, USB power port; LED (Blue)	
Network	10/100Mbps RJ45	10/100Mbps RJ45	Ethernet 10/100, RJ45	Ethernet 10/100	WIRELESS: integrata Ralink8188	Wireless 802.11b/g/n, WAPI (Ralink8188)	WiFi 802.11 b/g/n Realtek RT-L8188EUS	Mediatek MT5931 (Bluetooth v2.1: Mediatek MT6622)	Ethernet 10/100
GPIO	8 GPIO pin, I ² C, SPI, UART	Compatibile Arduino	2 bus da 46 pin configurabili che includono McASP0, SPI1, I2C, GPIO(65), LCD, GPMC, MMC1, MMC2, 7 AIN(1.8V MAX), 4 Timers, 3 Serial Ports, CAN0,	2 bus da 48 pin configurabili che includono - I/O digitali 2c (twi) - spi - rgb/lvds - csi/ts - fm-in - adc - cvbs - vga - spdif-out - touch panel	NO	NO	NO	NO	NO
Dati	SD Card	microSD	microSD	microSD Porta SATA per Hard disk	microSD fino 32GB	microSD fino 32GB	microSD fino 32GB	microSD fino 32GB	microSD Card Slot eMMC module socket
Alimentazione	5V, 1A	5V, 700mA	5V, 700mA	5V 2A	5V	5V	5V	5V	5V, 2A
Dimensioni	85,60 x 53,98 mm	125 x 52mm	86 x 54 mm	100 x 60 mm	97 x 28 x 12mm	97 x 28 x 12mm	90 x 40 x 13 mm	90 x 40 x 13 mm	50 x 50 x 50 mm
Software predefinito		GNU/Linux	GNU/Linux Angstrom	Android	Android 4.0 ICS	Android 4.0	Android 4.1	Android 4.1	Nessuno
GNU/linux	Raspbian	Linux3.0 + Ubuntu 12.04 Android ICS 4.0	Angstrom Debian Ubuntu Fedora Archlinux Gentoo Sabayon	Supporta diverse distribuzioni di gnu/linux Permette installazione multi OS utilizzando BerryBoot	Debian Wheezy + BerryBoot	Debian Wheezy	Picuntu	Picuntu	Ubuntu Personalizzato
Installazione	SD Card BerryBoot	Predefinito	Predefinito	BerryBoot Immagine SD Card	BerryBoot Immagine SD Card	Immagine SD Card	Tool PicuntuKernel Installer	Tool PicuntuKernel Installer	Immagine SD Card
Applicazioni	Didattica base Applicazioni embedded Robotica	Didattica base Applicazioni embedded	Didattica base Applicazioni embedded Robotica	Didattica avanzata Applicazioni embedded Video base Robotica	Sistemi stand alone Networking	Sistemi stand alone Networking	Sistemi stand alone Networking Video	Sistemi stand alone Networking Video	Pc Desktop Server Grafica Video Elaborazione immagini
Facilità uso	Facile	Facile	Facile ma chiusa	Gestione periferiche non banale	Media	Media	Laboriosa	Laboriosa	Facile

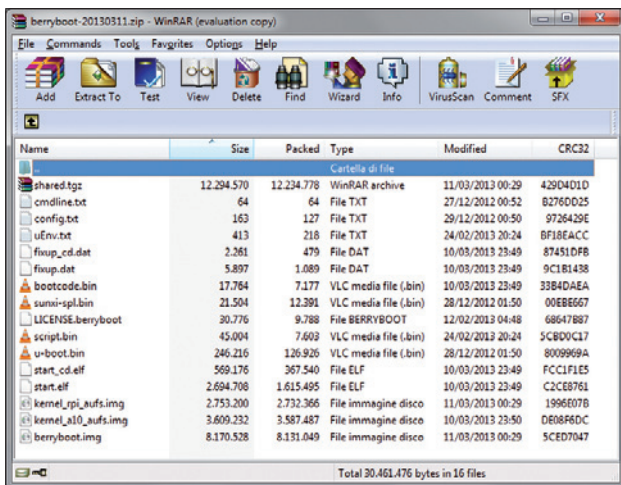


Fig. 2

- MK802
- MK802II
- MK802III
- ODDROID-U2

Una prima grande suddivisione tra i dispositivi presi in considerazione è fattibile in base ai processori a bordo degli stessi, o meglio dei SoC (System On a Chip). Questi sono delle specie di torte a strati composte da un integrato CPU, un integrato di gestione video GPU e uno o più banchi di memoria RAM. Queste "composizioni" rappresentano uno dei principali problemi e difficoltà nel realizzare le versioni dei sistemi operativi - sia Android, sia GNU/Linux - per questi dispositivi rispetto ai dispositivi dotati di processori tradizionali Intel x86.

I costruttori cercano di mantenere caratteristiche standard nello sviluppo delle CPU di base, mentre questo obiettivo di standardizzazione non è perseguito dagli assemblatori di SoC che, invece, tendono a ottimizzare le "ricette" di composizione dei SoC per ciascun tipo di dispositivo ai quali sono destinati. Dato che il kernel (nocciolo) dei sistemi operativi, sia Android, sia GNU/Linux - che condividono gran parte del Kernel - deve essere adattato alle caratteristiche dell'hardware che lo ospita. Ne consegue che il cosiddetto processo di porting deve essere svolto e personalizzato per ciascun tipo di SoC; questa attività che complica abbastanza la vita alle comunità di sviluppo.

A questa "biodiversità" si aggiungono le diverse modalità di gestione delle periferiche di I/O e comunicazione messe a disposizione in numero e specificità differenti su connettori dedicati, se ovviamente presenti. Possiamo quindi definire un secondo criterio di classificazione proprio in base alla presenza o meno di GPIO e periferiche di comunicazione su connettori dedicati.

Una terza tipologia di classificazione deriva dai differenti processi di boot svolti da ciascun dispositivo; que-

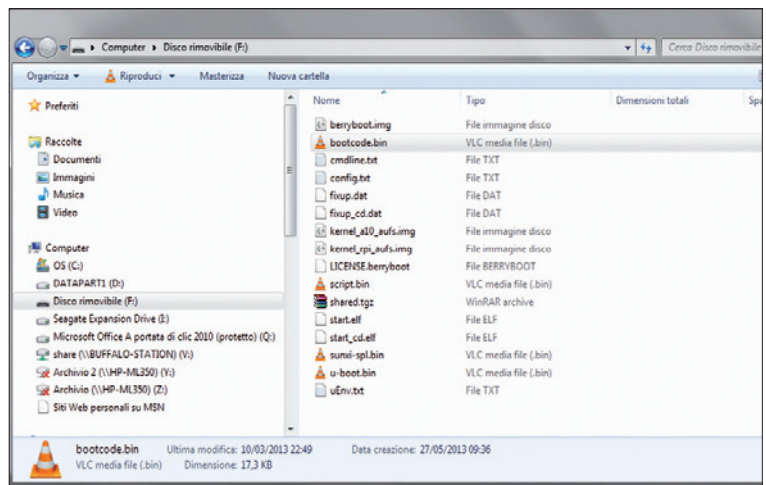


Fig. 3

sto comporta diverse procedure per installare GNU/Linux. Ve li presenteremo caso per caso.

In base alle caratteristiche citate possiamo quindi provare a determinare le "vocazioni" dei diversi dispositivi; queste destinazioni d'uso, secondo noi, sono: didattico, embedded, networking, lavoro d'ufficio. Torniamo ai nostri dispositivi.

Chi ci segue conosce già gran parte delle caratteristiche di RaspberryPi e le modalità di installazione e utilizzo di GNU/Linux, GPIO compreso. Rimane una delle scelte migliori per iniziare, soprattutto perché è praticamente impossibile fare danni con il software: il processo di boot parte direttamente dalla partizione di boot della SDCard e in caso di operazioni maldestre, si ricarica l'immagine del sistema operativo sulla SDCard e ... si riparte. Dal punto di vista hardware il GPIO è un po' più delicato, non sopporta, pena il danneggiamento del processore, tensioni superiori ai 3,3 V. Il grosso del gruppo dei dispositivi che abbiamo preso in considerazione è "motorizzato" dal SoC Allwinner A10, basato sul processore CPU ARM Cortex A8. A questo gruppo appartengono i dispositivi PcDuino, CubieBoard, MK802 e MK802II.

Beaglebone Black ha a bordo un SoC Texas Instruments Sitara XAM3359AZCZ100, mentre MK802III e MK802IIIS sono animate da SoC Rockchip RK3066. PcDuino, Beaglebone Black e CubieBoard, come già Raspberry Pi, sono dotati di connettori di espansione con GPIO e altre periferiche con caratteristiche che descriviamo per ciascuno.

Anticipiamo che la gestione del GPIO è diversa pressoché per ciascuno dei dispositivi che ne sono dotati, rendendoli più o meno adatti all'uso per chi si avvicina per la prima volta a questi sistemi.

MK802 e MK802II sono in formato USB "Stick" e non dotati di GPIO; MK802III e MK802IIIS, anch'essi nel formato USB "Stick", sono costruiti attorno al SoC Rockchip RK3066 che include la CPU ARM Cortex A9 dual core e costituiscono la prima apparizione di

sistemi multicore di dimensioni lillipuziane di prezzo contenuto. Infine chiudiamo questa prima rassegna con il dispositivo Odroid-U2: una specie di mostro da 5x5x5cm con a bordo un SoC Samsung Exynos4412 Prime; include un chip CPU ARM Cortex A9 quadricore, avanguardia di sistemi che probabilmente non avranno nulla da invidiare ai PC tradizionali che utilizziamo tutti i giorni. Anche in questo caso, non siamo che all'inizio.

Per organizzare e poter comparare le caratteristiche dei diversi dispositivi abbiamo realizzato la tabella visibile in Fig. 1, dove abbiamo raccolto sia le informazioni tecniche dei dispositivi, sia le nostre impressioni in termini di facilità d'installazione di GNU/Linux e di tipologia di impiego.

Dividiamo questa panoramica in due parti; nella prima includiamo le schede dotate di GPIO - escluso la BeagleBone Black sulla quale vogliamo approfondire alcuni aspetti - mentre nella seconda parte completeremo il quadro delle schede citate.

Raspberry Pi E BerryBoot

A Raspberry Pi abbiamo già dedicato ampio spazio nella rivista e ne dedicheremo ancora in futuro. Ricordare caratteristiche e impieghi in questo articolo risulterebbe un'inutile ripetizione. Cogliamo però l'occasione di questa panoramica trasversale per annunciare il rilascio

Fig. 4

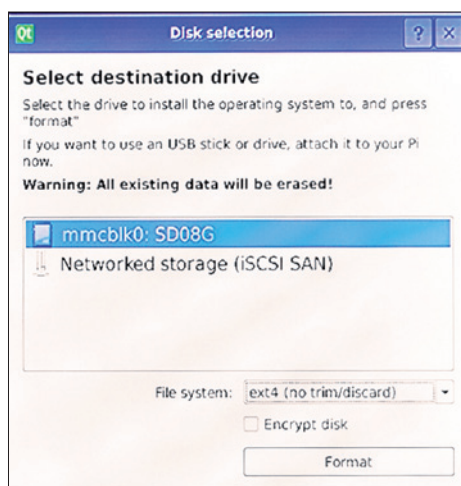
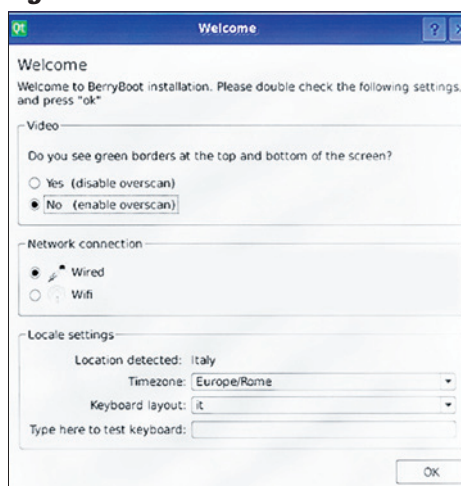


Fig. 5

di una nuova versione del sistema operativo Raspbian e introdurre due strumenti che permettono di caricare più distribuzioni di GNU/Linux su SD Card (o su disco fisso) per poter scegliere, in fase di boot, quale lanciare.

Si tratta dello strumento BerryBoot, scaricabile dal sito <http://www.berryterminal.com/doku.php/berryboot> e dello strumento New Out of Box Software (NOOBS), scaricabile allo stesso indirizzo della nuova versione di Raspbian <http://www.raspberrypi.org/downloads>.

BerryBoot è uno strumento molto versatile che può tornare utile in molte soluzioni applicative. Vediamo con ordine. La più immediata delle applicazioni è di permettere di caricare più distribuzioni di GNU/Linux in contemporanea sullo stesso dispositivo di memorizzazione di massa quali SD Card, chiavetta USB o disco fisso. Questo è un buon approccio, dal punto di vista didattico, per chi vuole sperimentare diverse distribuzioni di GNU/Linux senza dovere fare

incetta di SD Card. Vedremo tra breve come utilizzarlo e installarlo. Una volta lanciato, potremo notare che una delle possibilità è di installare il sistema operativo su una risorsa di rete, come un dispositivo SAN (Storage Area Network) in modo da trasformare Raspberry Pi in un terminale "tiny". In questo caso è sufficiente utilizzare una SD Card di minima capacità per conte-

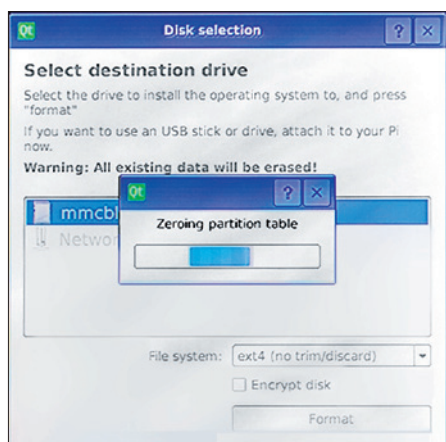


Fig. 6

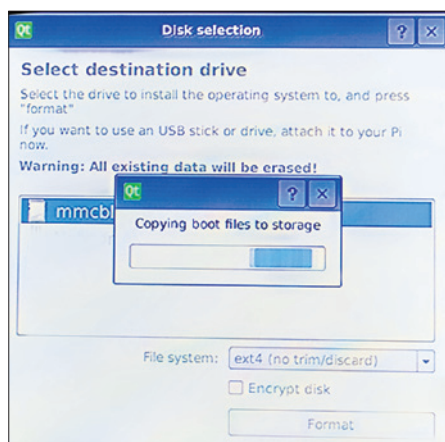


Fig. 7

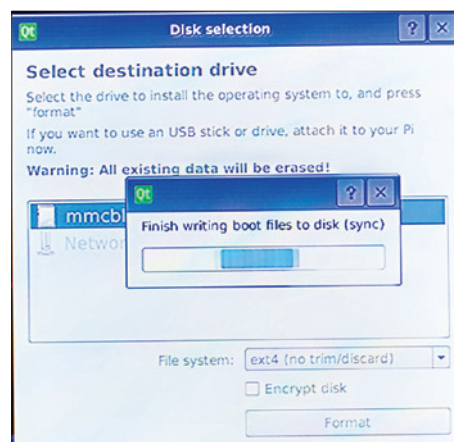


Fig. 8

nere la sola partizione di boot, e poi utilizzare come partizione principale l'unità di rete con anche tutte le garanzie di sicurezza dei dati che ne derivano. Aggiungiamo che l'utilizzo di BerryBoot, nonostante il nome, non è limitato al solo Raspberry Pi, ma supporta una buona parte dei dispositivi basati sul SoC Allwinner A10. Nella riga "Installazione" nella tabella di Fig. 1 abbiamo indicato su quali dispositivi può essere installato con successo BerryBoot.

INSTALLAZIONE SU Raspberry Pi

Un'altra piacevole sorpresa: nel numero 168 della rivista vi abbiamo descritto come trasferire l'immagine del sistema operativo Raspbian su SD Card utilizzando lo strumento WinDiskImager. Con la disponibilità di BerryBoot le cose si sono ulteriormente semplificate. Nel caso di Raspberry Pi è sufficiente scaricare dall'indirizzo <http://downloads.sourceforge.net/project/berryboot/> il loader di BerryBoot: `berryboot-20130311.zip` (o versione successiva, ma attenzione! NON il file che inizia con "berryboot-cubieboard"). Scompackare il file .zip in una cartella sul vostro PC e trasferire il contenuto copiandolo su una SD Card vuota, senza bisogno di formattazioni particolari o di utilizzare strumenti software (Fig. 2 e Fig. 3). Estraiete la SD Card dal PC, inseritela in Raspberry Pi, sinceratevi che sia collegata alla vostra rete locale via cavo e date tensione.

BerryBoot si carica in memoria e poi vi presenta la videata visibile in Fig. 4 dove vi chiede di controllare se sono visibili o meno delle barre di cornice sullo schermo, per poter configurare correttamente i parametri di overscan. Nella selezione successiva vi viene chiesto di specificare la tipologia della connessione di rete, se cablata o WiFi. In effetti potreste utilizzare un dongle WiFi e selezionare l'opzione corrispondente; questo però funziona solo se la vostra rete è libera o protetta con chiave WPA, altrimenti, almeno nella release provata da noi, non è possibile configurare correttamente

Negli ultimi mesi, come del resto descritto in questo articolo comparativo, il numero delle board che utilizzano come sistema operativo Linux Embedded è cresciuto a dismisura: RaspberryPi, Cubieboard, PCDuino e tante altre schede hanno invaso il mercato e sono state accolte con grande interesse sia dagli hobbysti che dagli utilizzatori professionali. Le elevate prestazioni di queste board e il loro basso costo hanno attirato l'attenzione di molti appassionati che, convinti di avere a che fare con la semplicità di Arduino, si sono spesso trovati di fronte ad un ostacolo insormontabile rappresentato dalla (oggettiva) maggior complessità di Linux Embedded che, al contrario di Arduino, è un vero e proprio sistema operativo. Per consentire a quanti non conoscono Linux di superare questo gradino iniziale, abbiamo realizzato questo volume che, prendendo per mano il lettore, lo guida passo dopo passo alla scoperta di questo mondo meraviglioso, fornendogli tutte le nozioni di base per proseguire il cammino verso mete ancora più ambiziose. Un viaggio in compagnia della più famosa board Linux Embedded, quella RaspberryPi di cui sono stati venduti in un anno oltre un

milione di esemplari. Un viaggio e un libro adatto a tutti, dagli hobbysti agli studenti, ma anche ai tecnici ed agli insegnanti che non hanno mai affrontato questo argomento. E proprio perché adatto a tutti abbiamo pensato ad una distribuzione in edicola, molto più capillare della distribuzione in libreria, iniziativa che ha anche consentito di limitare il prezzo di copertina a soli 13,90 Euro. Cosa aspettate? Corri in edicola ad acquistare una copia di "RaspberryPi, il mio primo Linux Embedded"!

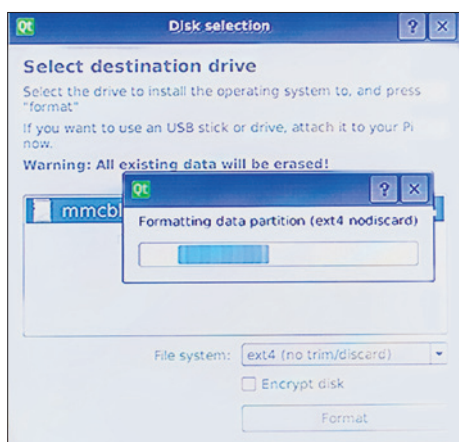
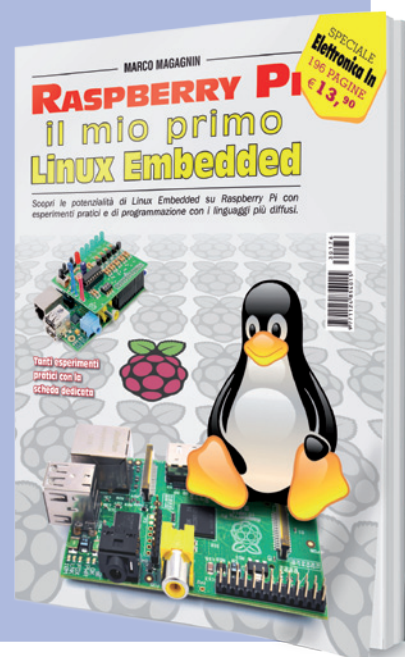


Fig. 9

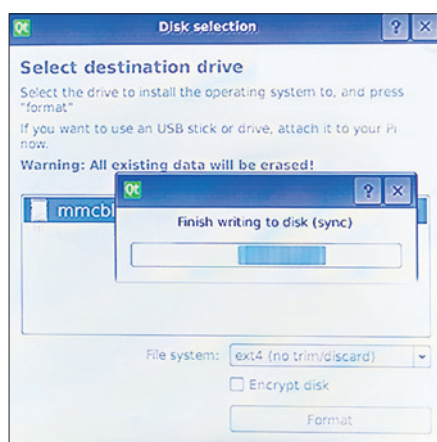


Fig. 10

la rete WiFi. Anche in questo caso l'aggiornamento è continuo, quindi scaricate sempre l'ultima versione e verificate. L'ultima selezione richiede di impostare la vostra Time Zone, per configurare correttamente l'ora di sistema. Probabilmente la troverete già impostata correttamente, in quanto BerryBoot cerca di determinare la zona geografica utilizzando l'indirizzo IP pubblico con il quale siete connessi a Internet. Premete il pulsante "OK" e passate alla pagina successiva (Fig. 5) dove viene chiesto di scegliere il



Fig. 11

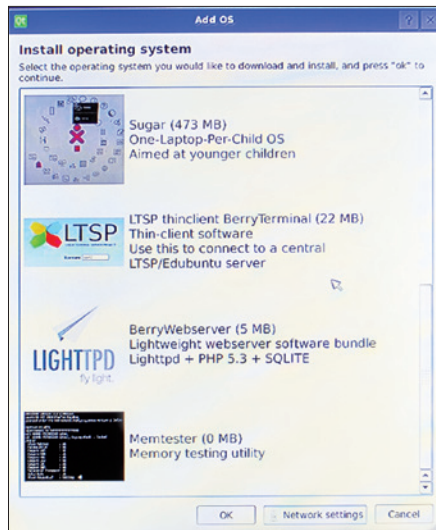


Fig. 12



Fig. 13



Fig. 14



Fig. 16

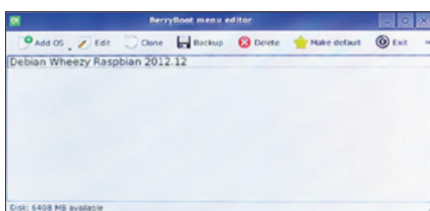


Fig. 17



Fig. 15

tipo di installazione che si vuole ottenere: su SD Card, su memoria esterna USB o su un dispositivo storage di Network. Per non complicarci la vita selezioniamo l'opzione SD Card.

Lasciamo inalterata la scelta del file system e premiamo il pulsante "Format" che - in realtà - mette in moto un processo automatico com-

posto di varie fasi e che permette di formattare correttamente la SD Card, preparare le partizioni di boot e di root, formattandole nei formati corretti (da Fig. 6 a Fig. 10). Al termine verrà presentata la videata di scelta del sistema operativo da installare (Fig. 11 e Fig. 12). In basso, in questa videata sono presenti i pulsanti di configurazione di un eventuale IP statico e del proxy, nel caso ne utilizzate uno (Fig. 13 e Fig. 14).

Scegliete un sistema operativo - per esempio Raspbian - e premete il pulsante "OK".

Ora il sistema viene scaricato direttamente dalla rete e installato su SD Card (Fig. 15).

Alla fine viene eseguito il reboot, che presenta la videata iniziale del bootloader di BerryBoot dove, in questo momento, appare un solo sistema operativo nell'elenco (Fig. 16). Se non facciamo nulla, dopo un certo numero di secondi, parte il sistema operativo predefinito che nel nostro caso è l'unico disponibile. Altrimenti, se premete il pulsante "Edit" potete accedere alle funzionalità di manutenzione del bootloader (Fig. 17). In questo modo potete aggiungere un nuovo sistema operativo alla collezione oppure clonare uno già caricato per eseguire prove senza intaccare quello di partenza per poi, eventualmente, cancellarlo. Potete modificare il sistema operativo predefinito, quello che parte al termine del conteggio alla rovescia. Infine, con più attenzione, potete accedere ai file di configurazione della partizione di boot di ciascun sistema operativo installato (da Fig. 18 a Fig. 22).

Con il pulsante "Exit" tornate alla funzione bootloader, dalla quale potete lasciare partire il sistema operativo predefinito, oppure un altro a vostro piacere selezionandolo con il mouse o con i tasti "freccia". Alla fine di tutto questo vedremo partire il nostro beniamino (Fig. 23

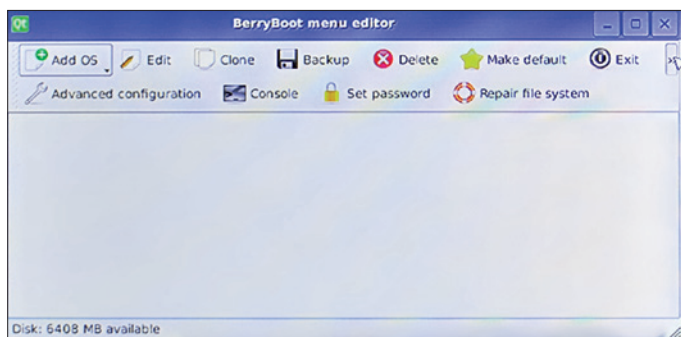


Fig. 18

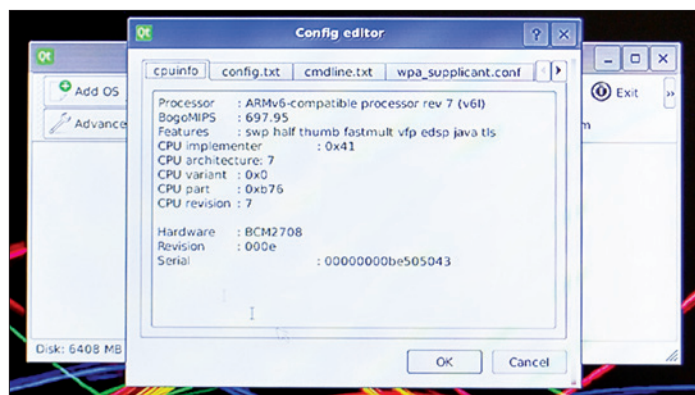


Fig. 19

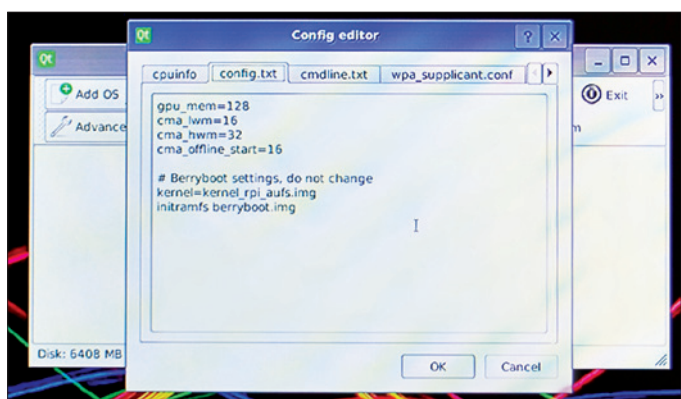


Fig. 20

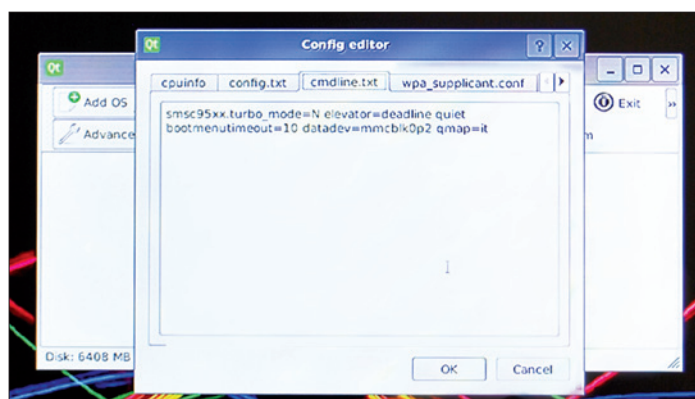


Fig. 21

e Fig. 24). Segnaliamo la possibilità di installare una videocamera sul connettore Jtag e la disponibilità della funzionalità di configurazione tramite lo strumento rasp-config.

BerryBoot può essere utilizzato anche con la Cubieboard e con i PC Stick MK802 e MK802II. Per questi ultimi due, però, è necessario utilizzare la connessione tramite WiFi interno con il vincolo della protezione di accesso WPA.

Passiamo ora all'ultima fatica della Fondazione che sviluppa Raspberry Pi, finalizzata a rendere il primo approccio verso il mondo GNU/Linux ancora più semplice e privo di ostacoli per i principianti. Dopo aver scaricato il file zippato recovery_v1_1_complete.zip che contiene il New Out of Box Software

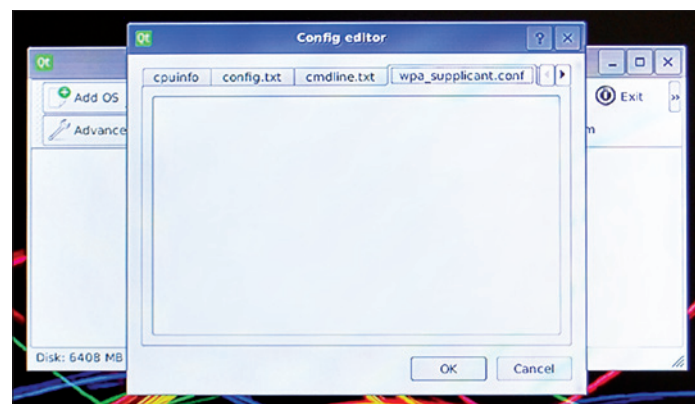


Fig. 22

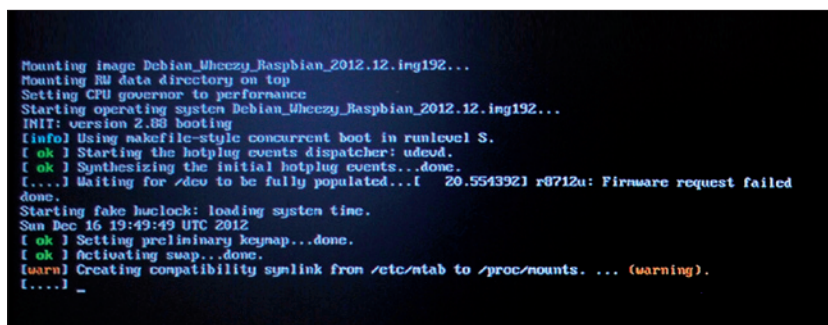


Fig. 23

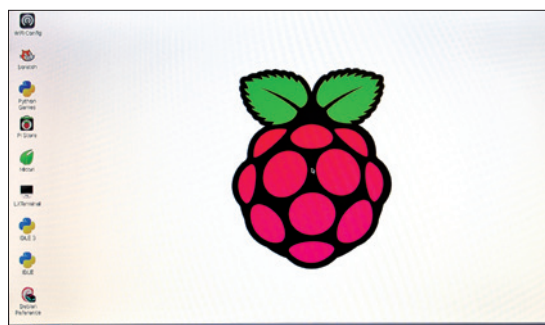


Fig. 24

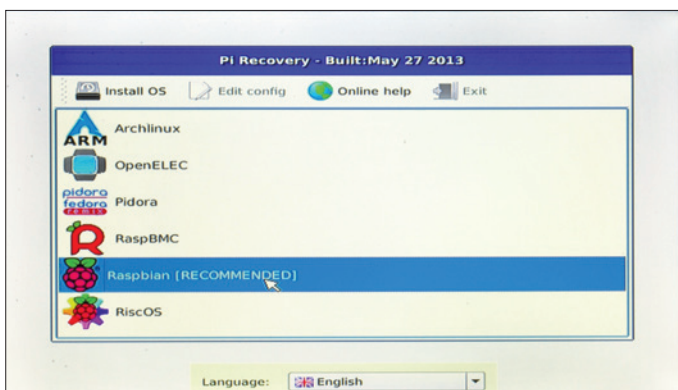


Fig. 25

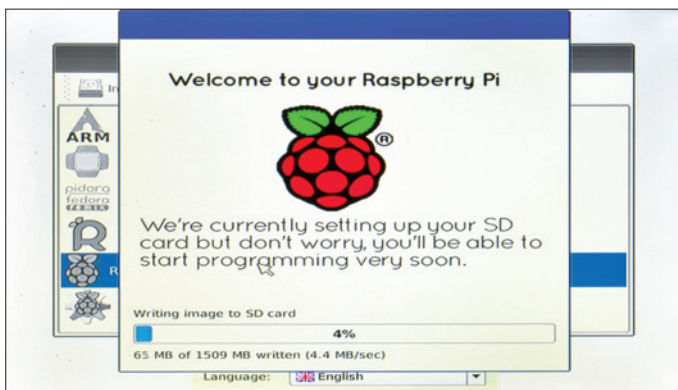


Fig. 26

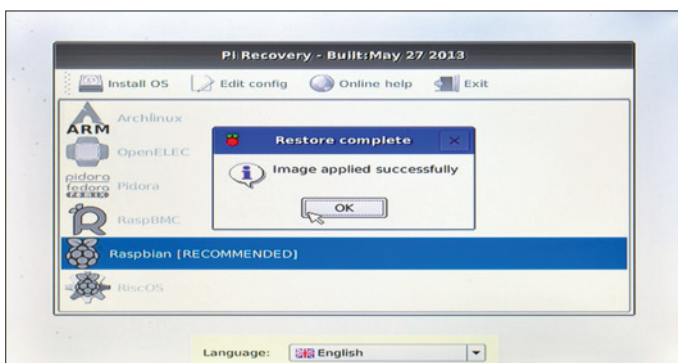


Fig. 27

(NOOBS), come nel caso di BerryBoot, copiate semplicemente il contenuto - dopo averlo scompattato - su una SD Card di almeno 4GB di capacità. Inseritela in Raspberry Pi e date tensione. Otterrete la videata visibile in Fig. 25 nella quale è possibile scegliere un sistema operativo e installarlo (Fig. 26 e Fig. 27). Se volete passare ad un sistema operativo diverso o recuperare una situazione corrotta è sufficiente tenere premuto il tasto "Shift" durante il bootload. In questo modo si torna alla videata iniziale di NOOBS e ... si ricomincia.

PCDuino

PcDuino è un mini PC, in formato scheda di sviluppo, basato sul SoC Allwinner A10, dotato di CPU ARM Cortex A8 da 1 GHz. La scheda nasce con installata già

a bordo una distribuzione di GNU/Linux della famiglia di Debian, che ci permette di mantenere valide pressoché tutte le informazioni che vi abbiamo trasferito negli articoli dedicati a RaspberryPi. È possibile anche installare, volendo, una versione di Android ICS, oppure aggiornare le versioni di GNU/Linux man mano che vengono rilasciate. Dal punto di vista dell'utilizzatore finale, secondo noi, è una bella comodità che permette di iniziare subito a sperimentare, come visibile in Fig. 28, con una dotazione di applicativi già disponibili e con prestazioni più che apprezzabili. Oltre alla scheda, nella confezione è fornito l'alimentatore che funziona a 220V, ma possiede una spina di tipo americano a cui va applicato un adattatore. Inoltre sono presenti un cavo HDMI e un HUB USB con un cavo USB. Sulla scheda sono presenti (Fig. 29):

- Due porte USB Host;
- La porta Ethernet RJ45;
- Il connettore di alimentazione;
- Un connettore seriale per la console di debug;
- L'alloggio per la micro SDCard;
- Un piccolo pulsante di "reset";
- I connettori del GPIO che descriviamo in seguito.

Ci soffermiamo immediatamente sulla caratteristica principale di questa scheda, che è la presenza di una struttura di connettori GPIO che ricalca il formato e le connessioni di quelli presenti sulle schede Arduino Uno. Come visibile in Fig. 30 sui connettori, che rispecchiano la tipologia ma non la disposizione di quelli presenti sulla scheda Arduino Uno sono disponibili gli stessi terminali di ingresso/uscita, ed in particolare:

- Fino a 13 pin di I/O digitali, molti dei quali ovviamente possono essere configurati per esprimere altre tipologie di interfacce;
- Fino a 6 pin di ingressi Analogici/Digitali, due dei quali, ADC0 e ADC1, hanno una risoluzione a 6 bit, gli altri hanno una risoluzione a 12 bit;
- Sono presenti due uscite PWM hardware, PWM1 e PWM2, che possono generare frequenze fino a 24Mhz, gli altri PWN sono simulati dal GPIO e possono supportare frequenze fino a 100 Hz, non molto accurate;
- Un'interfaccia I2C;
- Un'interfaccia SPI.

Una nota importante sulle tensioni di riferimento. I livelli di tensione dei pin GPIO e dei bus di comunicazione sono di 3,3V, pertanto è importante tenerne conto, soprattutto nel caso di utilizzo di shield compatibili Arduino. Ancora più importanti sono le tensioni di riferimento per i pin di conversione analogica digitale: i pin ADC0 e ADC1, con la risoluzione di 6 bit utilizzano

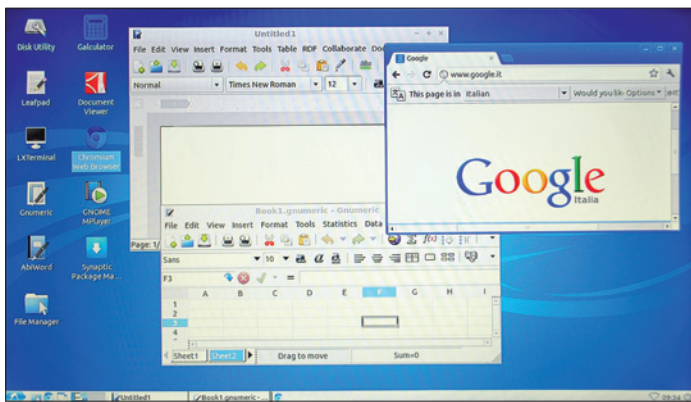


Fig. 28

una tensione di riferimento interna di 2 V, mentre gli altri pin utilizzano la tensione di riferimento di 3,3V. Per quanto riguarda l'utilizzo della scheda, abbiamo già anticipato che è equivalente all'utilizzo del sistema operativo di RaspberryPi, comandi a terminale e gestione dei pacchetti inclusi. Ci soffermiamo al momento solo sulla gestione del GPIO. Sono previste due possibilità principali. La prima prevede l'utilizzo dei file descrittori, per quanto riguarda i pin di I/O digitali e analogici. La seconda permette di recuperare programmi scritti per Arduino Uno grazie ad una libreria di API che implementano le principali funzioni presenti nel linguaggio di programmazione di Arduino. Nella prima opzione si accede ai pin GPIO utilizzando i file descrittori come nell'esempio di accensione del LED presentato nell'articolo sulla rivista n. 168 dedicato a RaspberryPi. Nell'implementazione di pcDuino i file descrittori dei pin del GPIO si trovano nelle cartelle

```
/sys/devices/virtual/misc/gpio/mode/
/sys/devices/virtual/misc/gpio/pin/
```

Nella cartella visibile in **Fig. 31**:

```
/sys/devices/virtual/misc/gpio/mode/
```

si trovano i file descrittori con i quali è possibile impostare le modalità di utilizzo di ciascun pin; sono previsti diversi valori, i principali sono:

- '0' - imposta il pin di I/O come INPUT;
- '1' - imposta il pin di I/O come OUTPUT;
- '8' - imposta il pin di I/O come INPUT, attivando la resistenza di pull up interna.

I valori dei pin di I/O vengono letti e scritti nei file descrittori corrispondenti nella cartella (**Fig. 32**):

```
/sys/devices/virtual/misc/gpio/pin/
```

Ricordiamo che il livello alto dei pin corrisponde a 3,3 V e che la corrente fornita da ciascun pin non è paragonabile a quella fornita dai pin di Arduino, ma è comunque sufficiente a mantenere acceso un LED.

Se vogliamo utilizzare i pin che possono essere utilizzati come convertitori ADC possiamo leggere i loro valori nei file descrittori contenuti nella cartella /proc/ con nome adcx (**Fig. 33**) dove "x" indica il pin ADC desiderato, da 0 a 5.

In **Fig. 34** possiamo vedere il risultato delle letture degli ingressi ADC dove è possibile verificare che ADC0 e ADC1 hanno una risoluzione a 6 bit mentre i rimanenti l'hanno a 12 bit. Infine possiamo impostare i valori dei pin che forniscono funzionalità PWM, utilizzando i file descrittori nelle cartelle

```
/sys/class/leds/pwm/x/brightness
```

(**Fig. 35**), dove anche in questo caso "x" indica il pin PWM desiderato, da 0 a 5.

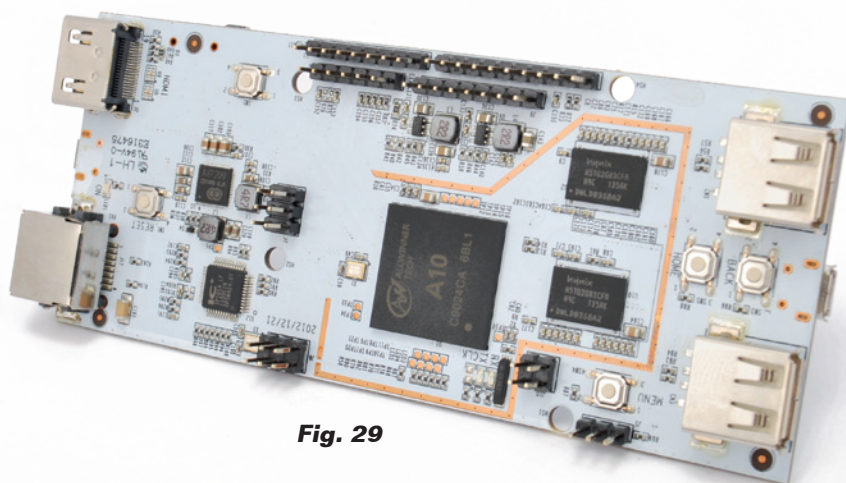
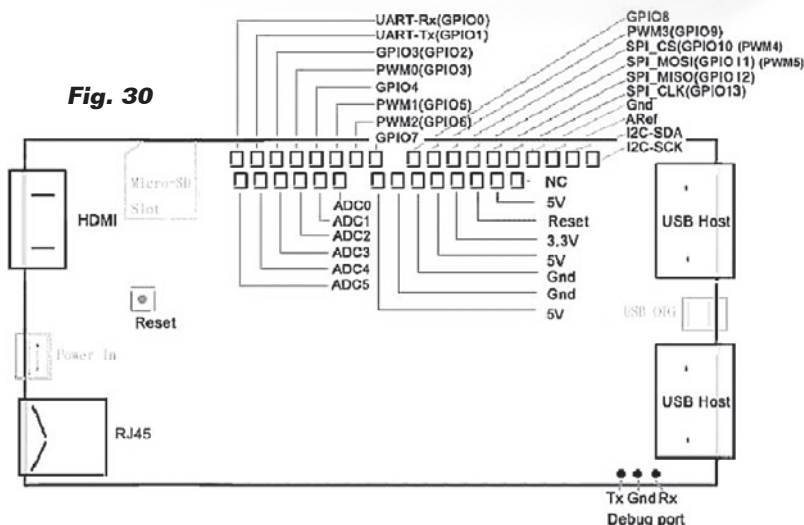


Fig. 29



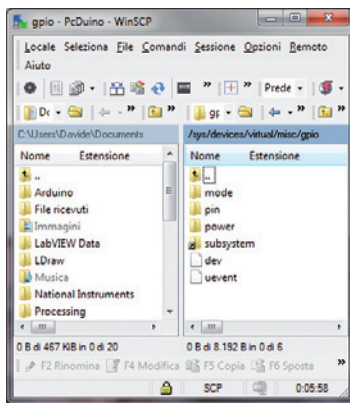


Fig. 31

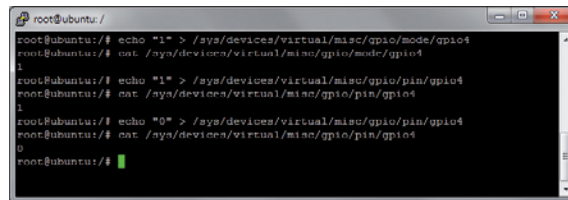


Fig. 32

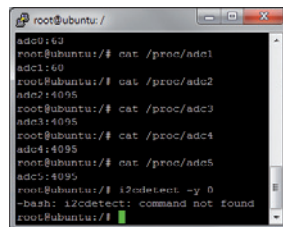


Fig. 33

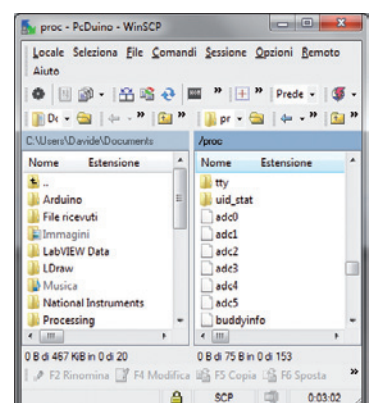


Fig. 34

Per impostare il valore PWM si scrive il valore desiderato nel file descrittore "brightness" nella cartella corrispondente al PWM richiesto (Fig. 36); per ciascun pin si possono impostare valori che vanno da 0 al valore contenuto nel file

`/sys/class/leds/pwmX/max_brightness`

diverso per ogni uscita (Fig. 37).

Per quanto riguarda l'utilizzo del bus i2C, utilizzando il linguaggio python vale quanto descritto negli articoli dedicati a Raspberry Pi, con l'utilizzo della libreria "smbus". Per la comunicazione SPI, sempre in python, è disponibile la libreria SPI-Py.

Veniamo alla seconda modalità di programmazione, che ricalca il modo di operare con Arduino. Il linguaggio è il C++, per il quale viene fornita una libreria di compatibilità con le funzioni di accesso al GPIO ed ai bus di comunicazione disponibili nell'IDE di Arduino. In particolare sono disponibili le funzioni:

Seriale

- if (Serial)
- available()
- begin()
- end()
- find()
- findUntil()
- flush()
- parseFloat()
- parseInt()
- peek()
- print()
- println()
- read()
- readBytes()
- readBytesUntil()
- setTimeout()
- write()
- serialEvent()

ADC

- analogReference()
- analogRead()

PWM

- analogWrite()

GPIO

- pinMode()
- digitalRead()
- digitalWrite()
- pulseIn()

I2C

- begin()
- requestFrom()
- beginTransmission()
- endTransmission()
- write()
- available()
- read()
- onReceive()
- onRequest()

SPI

- begin()
- end()
- setBitOrder()
- setClockDivider()
- setDataMode()
- transfer()

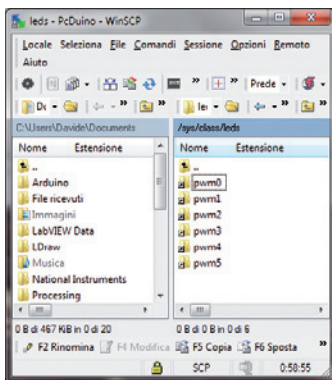


Fig. 35

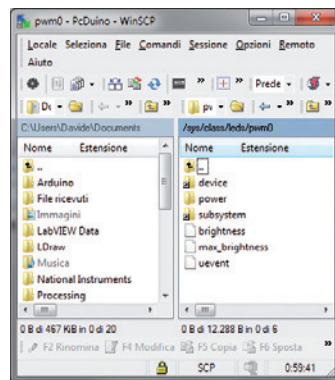


Fig. 36

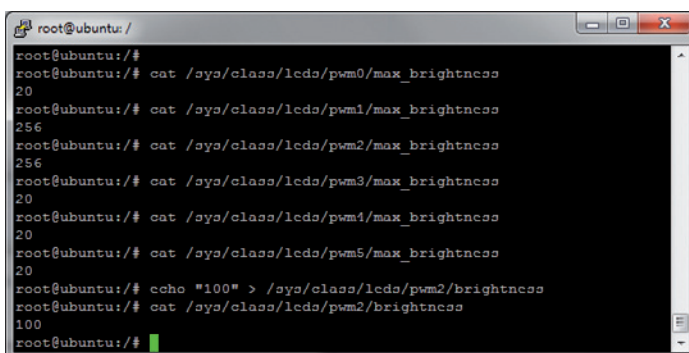


Fig. 37

CUBIEBOARD

Prima di tutto i complimenti ai progettisti: Cubieboard è, almeno a nostro parere, il massimo che si può ottenere in pratica dalle schede che adottano un processore ARM RISC a 32 bit. Volendo, potrebbe soddisfare quasi le esigenze del normale utilizzo di ufficio. Tra le caratteristiche che ci hanno favorevolmente impressionato segnaliamo la presenza di un connettore SATA per collegare un hard disk, poi la presenza di jack sia

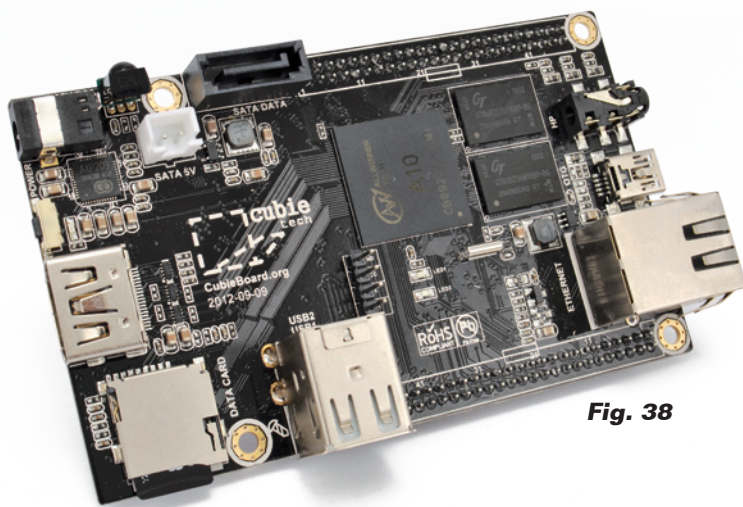


Fig. 38

per l'uscita, sia per l'ingresso audio. Infine due bus terminati su due connettori, ciascuno dotato di due file da 48 pin a passo 2mm (sigh!), per un totale di 96 pin che portano verso l'esterno le porte del processore. Con riferimento alla Fig. 38 vediamo più da vicino l'architettura della CubieBoard.

Partiamo dalla confezione: protegge adeguatamente la scheda e comprende un cavo di alimentazione costituito da un cavo collegabile a un alimentatore con uscita USB tipo A da una parte e un'uscita con un jack adatto alla presa sulla scheda dall'altra.

Sulla scheda notiamo i connettori Ethernet e USB, il doppio jack di ingresso ed uscita audio, il connettore per la micro SD Card che può essere fino a 64 GB. In alto nella figura c'è il connettore SATA per il disco fisso con l'alimentazione in grado di reggere un disco da 2,5 pollici. In caso si utilizzino dischi di dimensioni maggiori è necessario alimentarli esternamente. Anche nel caso di un disco da 2,5 pollici è opportuno essere generosi con l'alimentazione. Infine una piccola curiosità: in alto a sinistra possiamo notare un sensore all'infrarosso. Questo tipo di sensore era molto comune nei portatili e nei telefono cellulari d'annata, poi è scomparso. In questo caso però l'idea non è così peregrina, come potrebbe apparire a prima vista, in quanto ci permette di utilizzare nativamente un telecomando a infrarossi da integrare in applicazioni di automazione domestica e di controllo. La scheda Cubieboard nasce con a bordo, preinstallato nelle partizioni della memoria NAND, il sistema operativo Android. Se non è presente una SD Card con una partizione di boot valida, viene lanciato Android, altrimenti viene caricato e lanciato il sistema operativo presente su SD Card. Sono possibili due modalità per caricare un sistema operativo esterno sulla scheda Cubieboard. Il primo è scrivere una SD Card partendo dall'immagine di un sistema operativo. Il più recente e affidabile che abbiamo provato ad oggi lo potete scaricare dall'indirizzo: <http://dl.cubieboard.org/software/ubuntu/> ed è una distribuzione Linaro: una versione leggera di Ubuntu, che a sua volta è un derivato

di Debian. Questo sistema contiene già il supporto per il GPIO, che vi descriveremo tra breve. Il processo di preparazione della SD Card richiede di scompattare il file scaricato e di caricarlo sulla SD Card con lo strumento Win32DiskImager, come descritto nel numero 168 della rivista a proposito della preparazione della SD Card per Raspberry Pi. Al termine si inserisce la SD Card nell'apposito slot su Cubieboard e si dà tensione. Il risultato è visibile in Fig. 39. Abbiamo anche provato ad aprire alcune applicazioni con risultati soddisfacenti dal punto di vista delle prestazioni (Fig. 40). Da notare che è possibile accendere e spegnere la scheda con il pulsante vicino alla presa HDMI.

Il secondo metodo utilizza BerryBoot. A differenza di Raspberry Pi, nel caso di Cubieboard è possibile far partire il sistema Android, puntare il browser web di Android all'indirizzo <http://get.berryboot.com/> e scaricare il programma Berryboot.apk. Una volta scaricato installatelo come una normale App di Android. Inserite una SD Card nello slot e lanciate l'App. Verrà caricato BerryBoot configurato in modo opportuno per Cubieboard e vi ritroverete nella situazione descritta nel capitolo dedicato a Raspberry Pi (da Fig. 41 a Fig. 44). Una volta caricato uno o più sistemi operativi, potrete notare la notevole differenza di prestazioni rispetto a Raspberry Pi. Unico problema riscontrato - almeno fino al momento della chiusura del pezzo - è che nelle distri-



Fig. 39

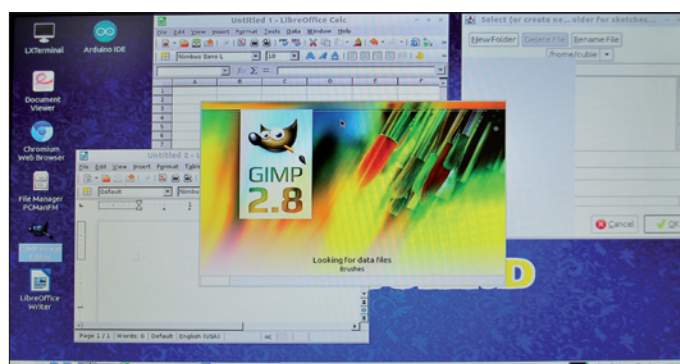


Fig. 40

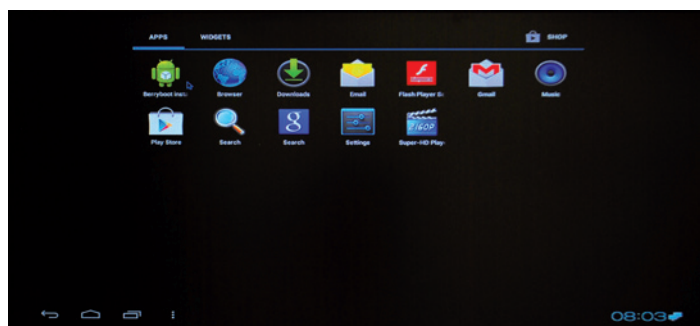


Fig. 41

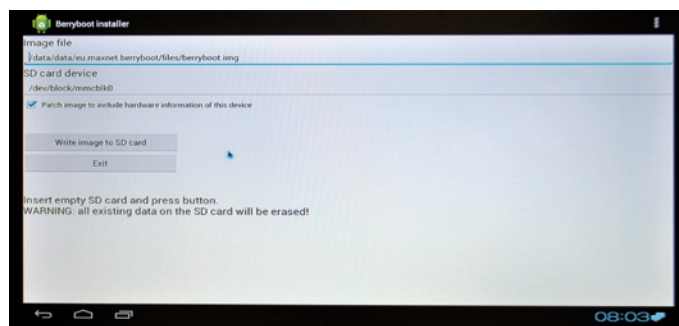


Fig. 42

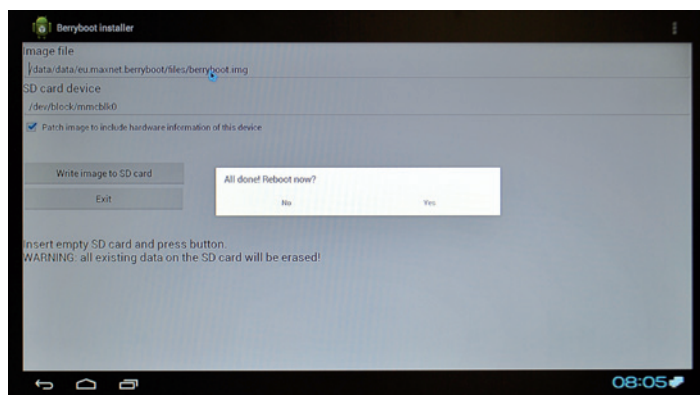


Fig. 43

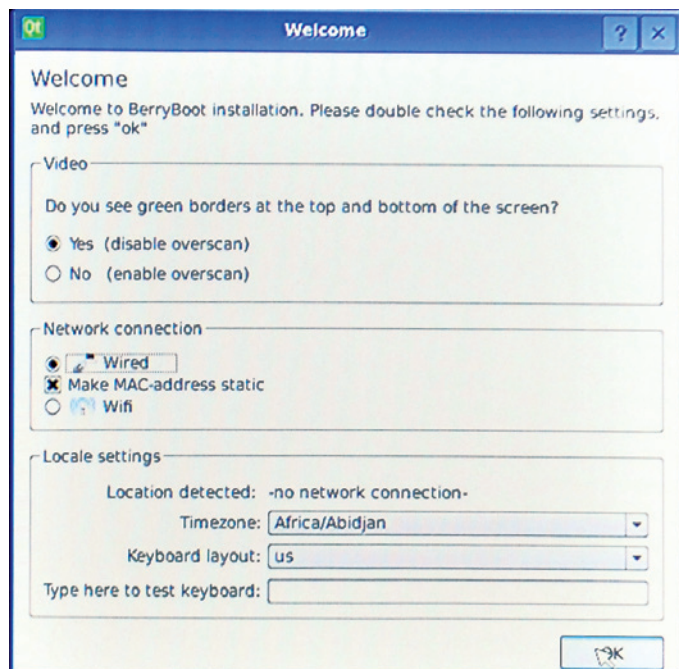


Fig. 44

buzioni caricate con BerryBoot non ci è stato possibile accedere ed utilizzare il GPIO, probabilmente per la configurazione utilizzata nella compilazione del kernel e la mancanza del driver per il GPIO.

Torniamo comunque alla gestione del GPIO che, al momento, è possibile solo con la distribuzione scaricata dall'indirizzo <http://dl.cubieboard.org/software/ubuntu/>. Le porte del processore possono essere assegnate ai pin dei connettori mediante un processo di configurazione un po' laborioso.

Un prezzo da pagare che permette una flessibilità d'impiego difficilmente replicabile. Proprio questa caratteristica la rende meno ... didattica rispetto a schede come RaspberryPi e PcDuino, tuttavia riteniamo che, per realizzare applicazioni di livello avanzato, valga la pena di affrontare l'impegno di acquisire le conoscenze aggiuntive necessarie a gestirla.

Il problema della maggior complessità di configurazione e utilizzo del GPIO sulla scheda Cubieboard è dovuta al fatto che Allwinner, produttore del SoC A10, non è particolarmente interessato al supporto del GPIO compreso nella CPU Cortex A8 (ARMv7); il loro mercato principale è rappresentato dai produttori di tablet e smartphone e quindi GPIO non è una priorità. Il progettista della Cubieboard, Tom Cubie, ex collaboratore di Allwinner, ha sviluppato il driver SUN4I-GPIO che non è flessibile come il driver disponibile per Raspberry Pi e non permette di gestire i pin GPIO altrettanto semplicemente. Con il driver SUN4I-GPIO

è necessario configurare in modo permanente ciascun pin che si vuole usare come ingresso od uscita nel file script.bin, che si trova nella partizione di boot. Per cambiare configurazione ad un pin è necessario disattivare il driver, trasformare il file script.bin in formato "leggibile" script.fex, eseguire le modifiche, ricompilare il file riottenendo script.bin e ricaricare nuovamente il driver. Ciò nonostante, a nostro parere, ne vale comunque la pena. Approfondiremo l'argomento in un prossimo articolo dedicato alla scheda Cubieboard. Per il momento diamo solo un assaggio del processo da seguire per la configurazione e l'utilizzo dei pin di I/O digitale. Innanzitutto, nella **Fig. 45** vediamo la mappatura dei pin del GPIO sui due connettori presenti sulla scheda. I nomi PE, PD, PG, ecc. seguiti da un numero si riferiscono ai pin delle porte di I/O del SoC A10, e sono descritti nel datasheet dello stesso.

La configurazione di ciascun pin si trova nel file script.bin, che si trova nella partizione di boot. Il file script.bin viene caricato durante il processo di boot per comunicare al kernel la configurazione dell'intera scheda,

con tutte le periferiche, tra cui la mappatura e configurazione dei pin dei connettori.

Il file è in formato esadecimale, per renderlo “leggibile” ed eventualmente modificarlo è necessario “decompilarlo” dal formato hex al formato testo. Si utilizzano gli strumenti scaricabili all’indirizzo <https://github.com/linux-sunxi/sunxi-tools/> e, seguendo le istruzioni, si ottengono i due tools necessari a decompilare e ricompilare il file script.bin.

Nel file script.fex ottenuto dalla decompilazione possiamo trovare la sezione [gpio_para], dedicata alla configurazione dei pin del GPIO. Ad esempio il GPIO1 è assegnato al pin PE1, la cui posizione fisica possiamo trovare in Fig. 45.

Possiamo vedere la sezione del file script.fex dedicata alla descrizione dei pin GPIO.

```
[gpio_para]
gpio_used = 1
gpio_num = 11
gpio_pin_1 = port:PE01<1><default><default><0>
gpio_pin_2 = port:PE02<1><default><default><0>
gpio_pin_3 = port:PE03<1><default><default><0>
gpio_pin_4 = port:PE04<1><default><default><0>
gpio_pin_5 = port:PE05<1><default><default><0>
gpio_pin_6 = port:PE06<1><default><default><0>
gpio_pin_7 = port:PE07<1><default><default><0>
gpio_pin_8 = port:PE08<1><default><default><0>
gpio_pin_9 = port:PE09<1><default><default><0>
gpio_pin_10 = port:PE10<1><default><default><0>
gpio_pin_11 = port:PE11<1><default><default><0>
```

Ora, utilizzando le informazioni descritte in precedenza, vediamo, in prima approssimazione, l’accesso al GPIO da linea di comando utilizzando i file descrittori che si trovano nella cartella /sys/class/gpio.

Come prima operazione dobbiamo caricare il driver che permette l’accesso al GPIO con il comando: *modprobe gpio-sunxi*

Facciamo creare i file descrittori per il gpio_pin4.

```
echo "4" > /sys/class/gpio/export
```

Se ispezioniamo la cartella /sys/class/gpio vedremo creata una nuova cartella di nome gpio4_pe4, combinazione della configurazione presente nel file script.bin (Fig. 46 e Fig. 47).

Ora possiamo usare i comandi consueti per impostare direzione e valore (Fig. 48).

```
cat /sys/class/gpio/gpio4_pe4/direction
in
echo "out" > /sys/class/gpio/gpio4_pe4/direction
cat /sys/class/gpio/gpio4_pe4/direction
out
echo "1" > /sys/class/gpio/gpio4_pe4/value
cat /sys/class/gpio/gpio4_pe4/value
```

U14 (Next to SATA connector)			
LCD			
1	PD0 (LCDD0/LVDS0P0)	2	Ground
3	PD2 (LCDD2/LVDS0P1)	4	PD1 (LCDD1/LVDS0N0)
5	PD4 (LCDD4/LVDS0P2)	6	PD3 (LCDD3/LVDS0N1)
7	PD6 (LCDD6/LVDS0PC)	8	PD5 (LCDD5/LVDS0N2)
9	Ground	10	PD7 (LCDD7/LVDS0NC)
11	PD9 (LCDD9/LVDS0N3)	12	PD8 (LCDD8/LVDS0P3)
13	PD11 (LCDD11/LVDS1N0)	14	PD10 (LCDD10/LVDS1P0)
15	PD13 (LCDD13/LVDS1N1)	16	PD12 (LCDD12/LVDS1P1)
17	PD15 (LCDD15/LVDS1N2)	18	PD14 (LCDD14/LVDS1P2)
19	PD16 (LCDD16/LVDS1PC)	20	Ground
21	PD18 (LCDD18/LVDS1P3)	22	PD17 (LCDD17/LVDS1NC)
23	PD20 (LCDD20)	24	PD19 (LCDD19/LVDS1N3)
25	PD22 (LCDD22)	26	PD21 (LCDD21)
27	PD27 (LCDVSYNC)-VGA-VSYNC	28	PD23 (LCDD23)
29	PD24 (LCDCLK)	30	PD26 (LCDHSYNC)-VGA-HSYNC
31	PB2 (PWM0)	32	PD25 (LCDDE)
33	YP_TP (TP-Y1)	34	XP_TP (TP-X1)
35	YN_TP (TP-Y2)	36	XN_TP (TP-X2)
37	PH7 (LCD0-BL-EN/LCD-PI00/UART5-RX/EINT7)	38	Ground
39	PB11 (LCD0-SDA/LCD-PIO2)	40	PB10 (LCD0-SCK/LCD-PIO1)
41	SPDIF	42	Ground
43	VCC-5V	44	3.3V (nc in 2012-08-08)
SPI0			
45	PI10 (SPI0-CS/UART5-TX/EINT22)	46	PI12 (SPI0-MOSI/UART6-TX/EINT24)
47	PI11 (SPI0-CLK/UART5-RX/EINT23)	48	PI13 (SPI0-MISO/UART6-RX/EINT25)

Fig. 45

U15 (Between Ethernet port and USB ports)			
CSI1/TS			
1	VCC-5V	2	PH15 (CSI1-PWR/EINT15)
3	CSI1-IO-2V8	4	PH14 (CSI1-RST#/EINT14)
5	PG0 (CSI1-PCLK/SDC1-CMD)	6	PB18 (TWI1-SCK)
7	PB19 (TWI1-SDA)	8	PG3 (CSI1-VSYNC/SDC1-D1)
9	PG2 (CSI1-HSYNC/SDC1-D0)	10	PG1 (CSI1-MCLK/SDC1-CLK)
11	PG4 (CSI1-D0/SDC1-D2)	12	PG5 (CSI1-D1/SDC1-D3)
13	PG6 (CSI1-D2/UART3-TX)	14	PG7 (CSI1-D3/UART3-RX)
15	PG8 (CSI1-D4/UART3-RTS)	16	PG9 (CSI1-D5/UART3-CTS)
17	PG10 (CSI1-D6/UART4-TX)	18	PG11 (CSI1-D7/UART4-RX)
19	Ground	20	Ground
Analog		SDIO3	
21	FMINL	22	PI4 (SDC3-CMD)
23	FMINR	24	PI5 (SDC3-CLK)
25	Ground	26	PI6 (SDC3-D0)
27	VGA-R	28	PI7 (SDC3-D1)
29	VGA-G	30	PI8 (SDC3-D2)
31	VGA-B	32	PI9 (SDC3-D3)
CSI0/TS			
33	LCD1-VSYNC	34	PE4 (CSI0-D0)
35	LCD1-HSYNC	36	PE5 (CSI0-D1)
37	Ground	38	PE6 (CSI0-D2)
39	AVCC	40	PE7 (CSI0-D3)
41	LRADC0	42	PE8 (CSI0-D4)
43	CVBS	44	PE9 (CSI0-D5)
45	HPL	46	PE10 (CSI0-D6)
47	HPR	48	PE11 (CSI0-D7)

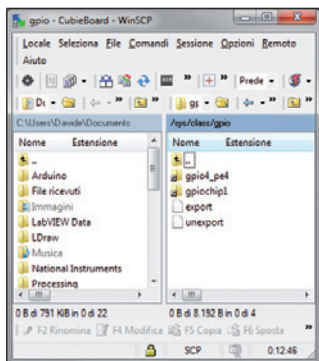


Fig. 46

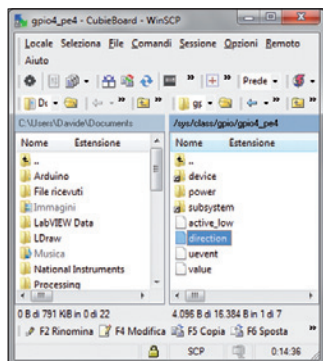


Fig. 47

1
echo "4" > /sys/class/gpio/unexport

Nella Fig. 49 possiamo vedere i nomi di device assegnati alle otto porte seriali disponibili. In Fig. 50, invece, possiamo vedere i device dei tre bus i2C, e in Fig. 51 il fatto che sono interrogabili nel modo usuale (dopo avere installato il pacchetto i2c-tools). Su questa scheda lavoreremo molto e vi presenteremo i risultati nei prossimi numeri.

CONSIDERAZIONI

Dopo aver visionato tutte le schede descritte nella tabella di Fig. 1 abbiamo provato a cercare il filo conduttore dell'evoluzione di questi dispositivi, senza volere classificare una scheda come migliore di un'altra. La qualità, la versatilità e la facilità d'uso di una scheda dipende molto, oltre che dalla ricetta dell'hardware, anche dalla maturità e completezza del o dei sistemi operativi disponibili; da questo punto di vista i diversi prodotti si trovano ancora a diversi stadi di evoluzione, dovuti alla già citata difficoltà di adattarsi ad hardware ancora troppo poco standardizzati, il che ne rende difficile la comparazione.

Per questa fascia di prodotti siamo ancora solo agli inizi, ma la certezza è che stanno diffondendosi e la loro destinazione è di integrarsi fortemente ai dispositivi a microcontrollore che siamo abituati a utilizzare. La seconda considerazione è legata al sistema operativo GNU/Linux. Anch'esso sta vivendo un momento di

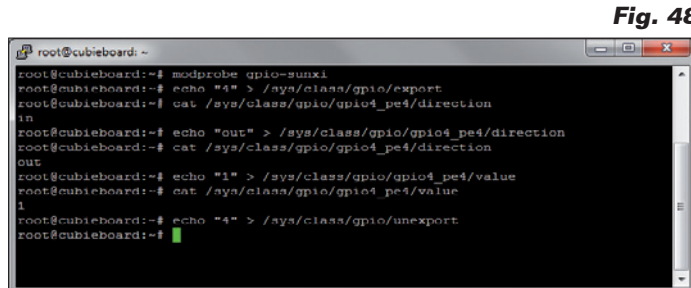


Fig. 48

diffusione che non ha precedenti. Dal punto di vista degli utilizzatori istituzionali, grandi amministrazioni statali come Francia e Germania sono migrati a GNU/Linux. In Italia è legge anche l'obbligo per le amministrazioni pubbliche di dotarsi di software open source. Recentemente anche la stazione spaziale sta migrando a GNU/Linux. Segno che la maturità è raggiunta. Il nostro compito infine, è quello di trovare la ... linea di interpolazione tra tutte le divergenze di realizzazioni hardware e specificità software in modo da proporre argomenti di divulgazione e percorsi di formazione che mantengano la loro validità nel corso delle evoluzioni che immancabilmente ci vedranno coinvolti. Ci abbiamo provato con gli articoli e poi con il libro dedicato a Raspberry Pi - che riteniamo ancora per lungo tempo il prodotto di riferimento per chi vuole cominciare - e al quale dedicheremo ancora la nostra attenzione proponendovi schede di espansione dedicate, di I/O come di interfaccia a sistemi GSM e GPS. Per le altre schede, della famiglia Cortex A8, vi proporremo dei progetti hardware e software un po' più sofisticati, provando a rimanere sempre il più semplice e descrittivi possibili, sempre in campi legati all'elettronica, come la robotica e le comunicazioni.

Nel prossimo numero vi presenteremo le schede che non siamo riusciti a inserire in questo, sicuri comunque che qualche novità tenterà di sorprenderci ancora. L'importante è non perdersi d'animo e continuare a studiare. I momenti di crisi sono accompagnati anche dalle opportunità derivanti dalle nuove tecnologie e innovazioni. Piuttosto che aspettare il futuro è meglio anticiparlo e prepararsi a viverlo da protagonisti e non da semplici spettatori.

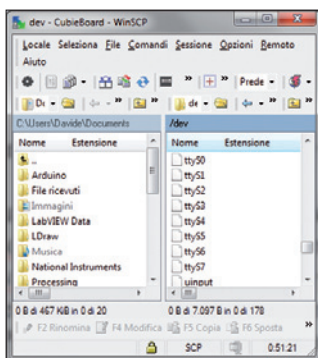


Fig. 49

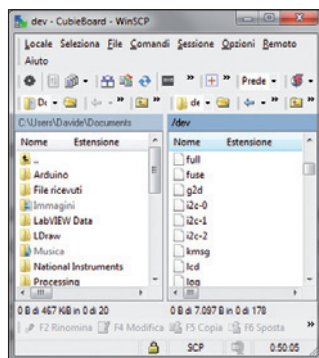


Fig. 50

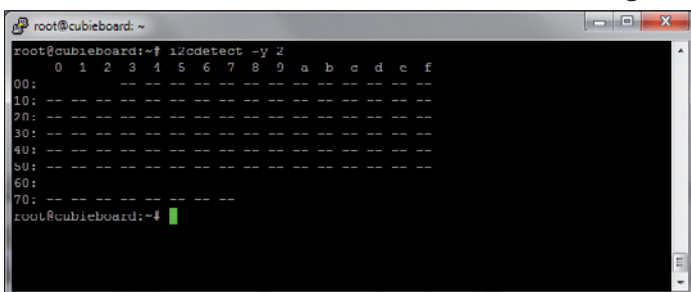


Fig. 51



Gli automodelli da corsa somigliano ormai a piccole vetture da gara; dotiamole di telemetria, così da renderle ancor più simili alle F1 che vediamo nei Gran Premi in TV.

TELEMETRIA REAL-TIME PER RADIOMODELLI

di PAOLO VASTA



Non è la prima volta che ci occupiamo di elettronica per radiomodelli e telemetria: lo abbiamo già fatto nel n° 157 con il progetto dell'UAV bimotore, per esempio, progetto basato sulla piattaforma ArduPilot Mega e sul software APM Planner. Lì si è trattato di lavorare su un prodotto open, con tutti i vantaggi del caso, ma va detto che non esiste solo la piattaforma Arduino-based: infatti, lo vedrete in queste pagine, si può realizzare un sistema di teleme-

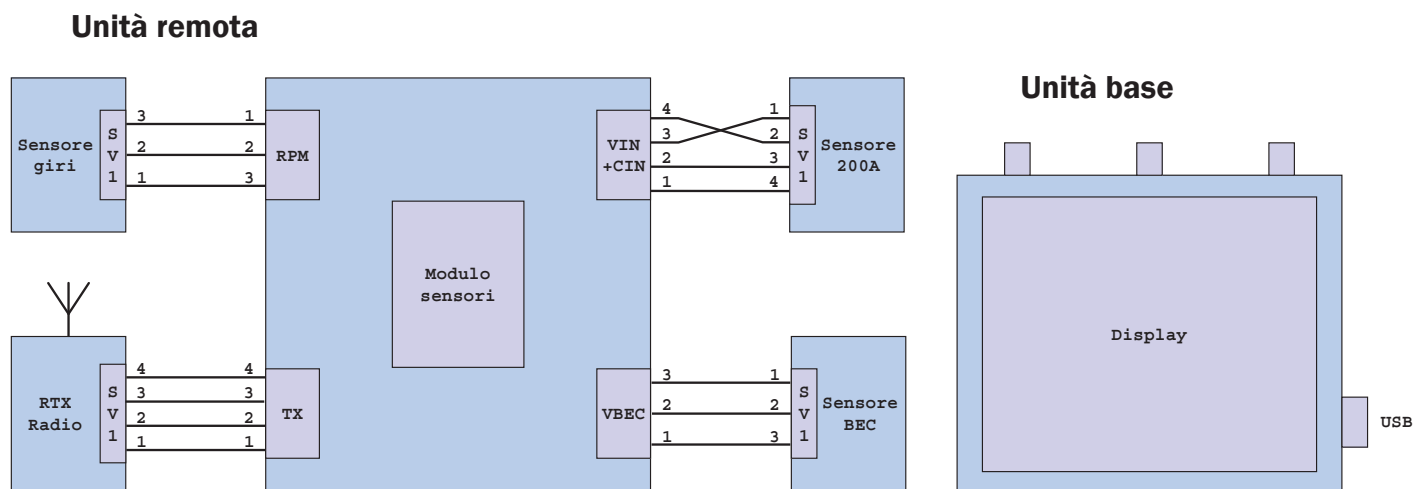


Fig. 1 - Schema a blocchi del sistema.

tria per radiomodelli anche con altre soluzioni. Il progetto che trovate qui realizza proprio il controllo a distanza di diversi parametri in modo stand-alone, cioè senza bisogno di alcun computer, come invece avviene con ArduPilot Mega e APM Planner: tutte le informazioni desunte dal radiomodello vengono visualizzate in un ampio display che si trova sulla consolle di monitoraggio.

Il progetto nasce prettamente per gli automodelli da corsa, che come i loro "fratelli maggiori" di Formula 1 possono essere dotati di un sistema di telemetria via radio per monitorarne a distanza i parametri operativi; nulla vieta, però, di applicarlo agli aeromodelli.

Il nostro sistema consta di due unità: quella base, che poi è la consolle di monitoraggio a terra o comunque la stazione base in cui visualizziamo i parametri; quella remota, che va a bordo del modello e che si avvale di unità periferiche, sempre a bordo, che riferiscono sui parametri richiesti. L'unità remota è corredata da un sensore di corrente, che misura l'erogazione della batteria e permette alla base di stimare l'autonomia del modello, oltre che da un senso-

re di tensione della batteria e da uno di giri del motore; è completata da una sezione radio ricetrasmittente per dialogare con la base, dotata anch'essa di un modulo RTX. La banda di frequenze utilizzata per il collegamento RF è quella degli 868 MHz.

Vediamo dunque nel dettaglio il nostro sistema di telemetria, dicendo che nasce per essere utilizzato su modelli elettrici di grossa scala al fine di tenere sotto controllo diversi parametri e la sua delicatissima elettronica. Motori, regolatori elettronici della velocità e batterie, infatti, sono sottoposti a stress causati da elevatissime correnti dell'ordine di centinaia di ampere di spunto, che causano surriscaldamento.

Il sistema proposto permette la lettura in real-time di tre diverse temperature con range 0÷120°, della corrente assorbita con range da 10÷200A e 0-10°, nonché la lettura di diverse tensioni. Inoltre è possibile misurare la velocità del modello tramite un GPS.

Si possono anche impostare diversi allarmi che, in caso di superamento delle soglie impostate, avvertono tramite segnalazione acustica. Sono presenti

anche le memorie dei valori massimi e minimi.

L'UNITÀ DI BORDO

È la parte del sistema che si installa a bordo del modello e che raccoglie le informazioni dai vari sensori e dall'eventuale GPS, preparando stringhe di dati che poi trasmette ciclicamente e frequentemente alla stazione base; è composta dal Modulo Sensori e da vari sensori, oltre che da un ricetrasmittente radio ad 868 MHz basato su un modulo Aurel.

A gestire tutti i sensori pensa un PIC18F4523, che lavora con clock di 8 MHz ottenuto tramite oscillatore interno e prescaler. La scelta è ricaduta su questo microcontrollore perché dotato di un convertitore A/D da ben 12 bit, che permette di realizzare una conversione molto precisa delle tensioni analogiche fornite dal modulo BEC e dal rilevatore di corrente. Inoltre, tramite la periferica hardware UART integrata, il microcontrollore comunica con il modulo radio ed il GPS.

Per sfruttare al meglio le potenzialità della periferica UART hardware (gli interrupt, per esempio...) è stato necessario utilizzare un sistema di commu-

tazione sulla linea RX del PIC, così da ovviare all'insufficienza di piedini del microcontrollore; in pratica, tramite il commutatore logico IC3 possiamo selezionare se ricevere i dati dal GPS o dal modulo radio. Nello specifico, il dispositivo porta alla linea RC7 dell'IC1 il canale dati in ricezione del GPS quando la sua linea S si trova a livello basso, mentre collega il ricevitore ad 868 MHz se lo stesso S è a livello alto.

La linea TX dell'UART va invece direttamente ai connettori dei moduli GPS e RTX.

Tramite il connettore LiPO è possibile leggere la tensione delle singole celle di una batteria attraverso il connettore per il bilanciatore di cui tutte le LiPO per modellismo sono dotate.

L'unità remota lavora a 3,3 V per facilitare l'interfacciamento con il GPS e con il modulo radio, i quali richiedono un'alimentazione continua di 3,3 V e operano con livelli logici 0/3,3 V. La tensione per il microcontrollore e tutta la logica viene ricavata localmente dal regolatore LDO siglato IC2, il quale prende tensione dal connettore del modulo BEC.

Per quanto riguarda i sensori, il BEC si collega al connettore VBEC, quello di giri va su RPM, il sensore di corrente del motore e della tensione di batteria si applica al connettore VIN+CIN, mentre i sensori di temperatura possono essere collegati ai connettori siglati T1, T2, T3.

Il connettore per la LIPO è stato concepito per consentire al microcontrollore di tenere sotto controllo la tensione di carica delle singole celle nei pacchi batteria al litio; ciò perché nelle LiPO è facile che le celle si carichino a tensione diversa le une dalle altre, condizione



Ed è Guinness World Record!

Con **11 grammi** di peso, **65 millimetri** di lunghezza e con telecomando a raggi infrarossi, il Nano-Falcon proposto online in Giappone da CCP, una sussidiaria di Namco Bandai, è ad oggi il più piccolo elicottero del suo genere. Se pensate che abbia dei seri limiti, vi sbagliate in quanto la tecnologia attualmente applicata agli smartphone ha consentito la nascita dei componenti miniaturizzati necessari a un simile micro elicottero. Il giroscopio e i tre microscopici motori riescono infatti a convivere con una batteria ai polimeri di litio e all'elettronica di controllo per offrire le stesse

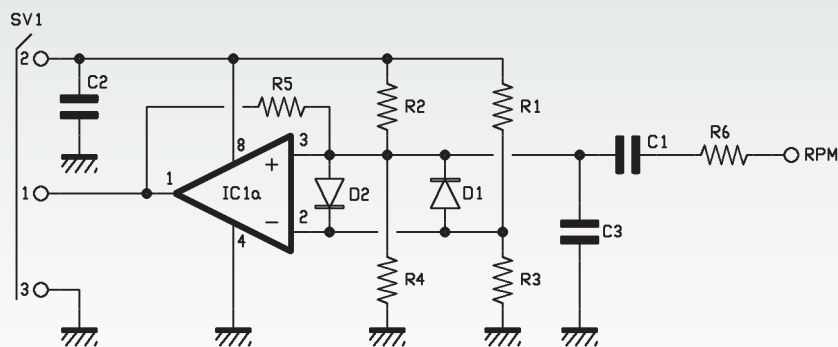
prestazioni e la stessa facilità di guida dei fratelli maggiori. Il raggio di copertura del sistema di controllo a infrarossi è di cinque metri ed evidentemente si tratta di un elicottero adatto esclusivamente al volo al

coperto. L'autonomia ottenuta con 30 minuti di ricarica è di circa 5 minuti di volo che, viste le dimensioni, è decisamente buona. Il primato non è inoltre solo un modo di dire, ma è stato certificato

proprio dallo staff del Guinness World Records; il prezzo di vendita online è di 4700 Yen circa, più le spese di spedizione. A quando una microtelecamera con microtrasmettitore da abbinare?



[schema elettrico **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

che con l'aumentare dei cicli di carica e scarica può portare ad avere celle caricate a tensioni che differiscono tra loro anche di alcuni decimi di volt. Questa situazione comporta da un lato l'incapacità della batteria di fornire la piena capacità (giacché le celle a tensione più bassa si caricano più di quelle che presentano una tensione maggiore) e dall'altro la possibilità di danneggiare, durante i cicli di carica, le celle, a causa dell'inevitabile eccesso di corrente nel pacco. Il controllo della tensione viene effettuato mediante l'ADC interno al microcontrollore, usando le linee RE0, RE1, RE2, RA2, RA3, RA5; può essere applicato a batterie fino alle 6S (formate, cioè, da sei celle). Affinché il microcontrollore possa testare il bilanciamento delle celle del pacco batteria (ossia accertare che le singole celle abbiano tensioni uguali tra loro) il connettore LIPO va collegato mediante un cavetto dotato dell'apposito connettore maschio compatibile con quello del

Balancer delle batterie standard per radiomodellismo. Il PIC montato nel circuito prevede la costante verifica della tensione che alimenta il circuito, fornita dall'ESC attraverso il sensore BEC; il monitoraggio avviene ad opera dell'ADC interno al microcontrollore, che, abbinato alla linea RB2, legge la tensione tramite il partitore R1/R2. Il valore desunto viene

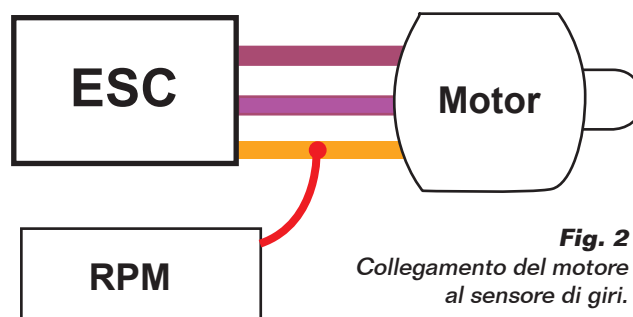


Fig. 2

Collegamento del motore al sensore di giri.

trasmissione insieme a tutti gli altri della telemetria.

I SENSORI

Al modulo sensori si connettono

altrettanti circuiti ed il ricevitore GPS, oltre alla sezione radio; vediamo subito i singoli sensori, partendo da quello che rileva i giri del motore e li comunica al microcontrollore IC1.

Sensore di giri

Questo circuito può essere composto anche solo da un qualsiasi sensore ad effetto di Hall o un sensore ottico collegato al connettore RPM del modulo sensori. Se però il nostro modello monta un motore di tipo brushless, è possibile utilizzare un altro metodo più elegante e non soggetto a problemi di installazione di magneti o di oggetti riflettenti sul motore; ciò perché i brushless da modellismo vengono alimentati da una

tensione trifase, quindi rilevando la frequenza e l'andamento dei segnali presenti sulle fasi è facile risalire al numero di giri del rotore. Infatti quando due

[schema elettrico **SENSORE BEC**]

fasi sono attive, sulla terza si vengono a creare delle correnti indotte che hanno frequenza correlata al numero di giri, perché nel motore trifase a magneti permanenti, quale è quello per modellismo, il rotore gira teoricamente a una velocità che dipende dalla frequenza di pilotaggio degli impulsi; è quindi sufficiente prelevare la tensione di una fase (uno dei tre fili che dal controller ESC vanno al motore brushless) e, mediante un filtro passa alto che elimini le componenti continue e un comparatore a isteresi, ottenere l'informazione sulla frequenza (**Fig. 2**). Tuttavia questo segnale, da applicare al contatto RPM del circuito, non permette di desumere immediatamente i giri del motore, i quali dipendono dal numero di coppie di poli presenti all'interno dei motori (le coppie di poli indicano quante coppie polari sono presenti nel rotore magnetico del motore); solitamente i motori per radiomodelli di automobili hanno rotori a 1 o 2 coppie di poli, mentre quelli per elicotteri, dovendo girare più veloce, hanno anche 6 o più coppie po-

Elenco Componenti:

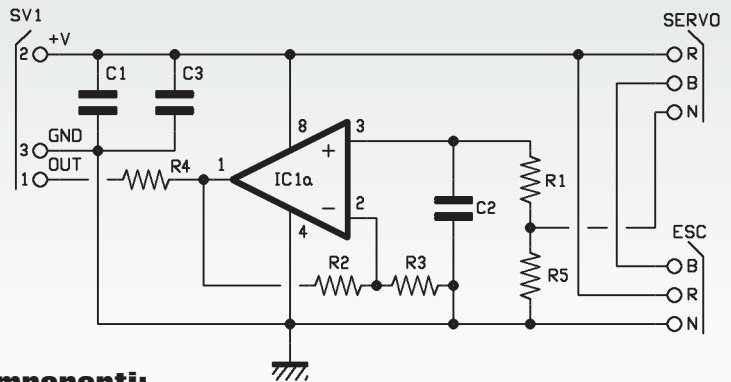
C1: 100 nF ceramico (1206)
C2: 1 μ F poliestere (1206)
C3: 100 nF ceramico (1206)
IC1: LM4558D (SO-8)
R1: 4,7 kohm (1206)
R2: 47 kohm (1206)
R3: 1 kohm (1206)
R4: 4,7 kohm (1206)

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

lari. Il circuito è quindi formato da un operazionale (IC1a) in configurazione da comparatore ad isteresi con i diodi D1 e D2 che, collegati in antiparallelo, limitano a $\pm 0,7$ V l'ampiezza del segnale di ingresso, così da non saturare né danneggiare l'operazionale, il quale, come tutti i sensori, è alimentato dai 3,3 V forniti dalla scheda Modulo Sensori.



Collegamento BEC

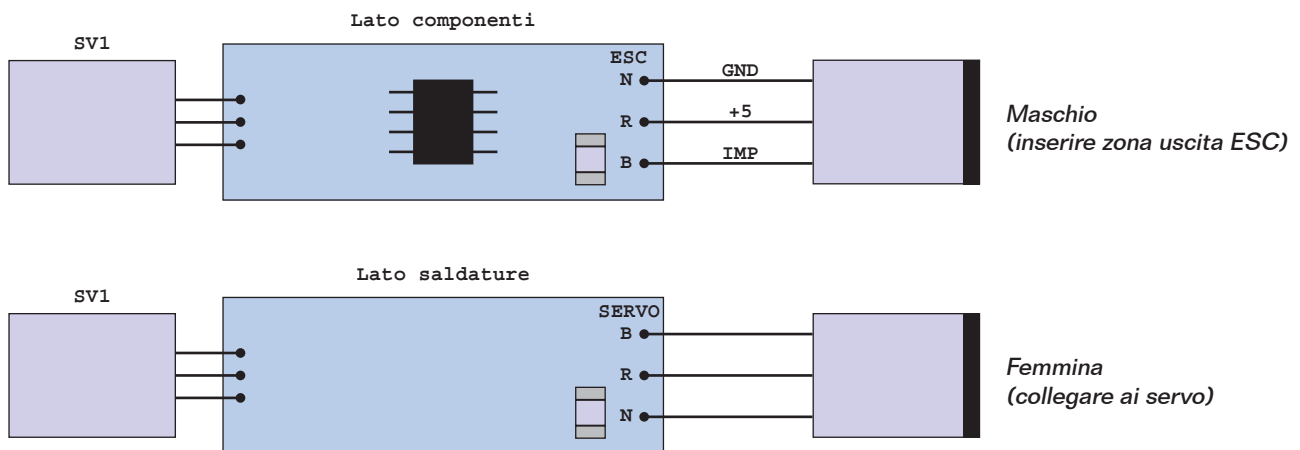


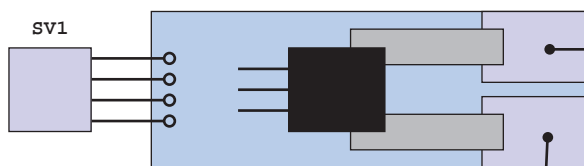
Fig. 3 - Collegamento del modulo BEC all'ESC.

Il comparatore ricava impulsi rettangolari 0/3 V in isofrequenza rispetto a quelli che pilotano il motore brushless, quindi li rende disponibili sul contatto 1 del connettore SV1, con il quale il modulo si collega al connettore RPM della scheda principale dell'unità remota; per l'esattezza, il contatto 1 finisce sul 3 del Modulo Sensori, il 3 va sull'1 di quest'ultimo e l'alimentazione (pin 2 dell'SV1) va collegata al contatto 2 dell'RPM sul Modulo Sensori.

Sensore BEC

Il sensore del BEC (Battery Eliminator Circuit) serve nel nostro sistema a misurare la corrente assorbita dai servomotori e da eventuali ventole o accessori vari; non dai motori, per i quali abbiamo previsto un apposito sensore che descriveremo tra breve. Anche qui il circuito è molto semplice: viste le correnti in gioco, dell'ordine di massimo 10 A, si è deciso di non usare sensori particolari ma una semplice resistenza di shunt, la quale è stata collegata in serie tra l'uscita dell'ESC e

l'ingresso del ricevitore radio, sul negativo dell'alimentazione. In altre parole, bisogna collegare al circuito il cavetto tripolare che dall'ESC porta i 5 volt e la massa agli utilizzatori quali i



servo, quindi connettere la massa dal lato dell'ESC a quella del nostro circuito e quella dal lato utilizzatori al nodo tra R1 ed R5, secondo il disegno di cablaggio visibile in Fig. 3. In questo modo, la corrente che fluisce nei servo e nei servizi attraversa R5, ai capi della quale ci sarà una caduta di tensione pari a:

$$V = R \times I$$

Alla corrente massima di 10 A avremo una caduta di:

$$V = 0,01 \, \Omega \times 10 \, A = 0,1 \, V$$

Collegamento sensore 200A

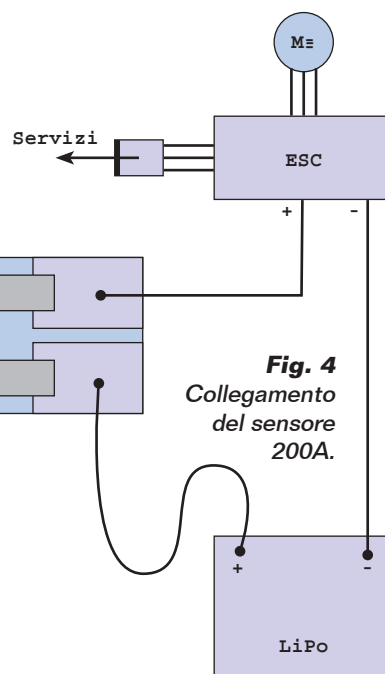


Fig. 4
Collegamento del sensore 200A.

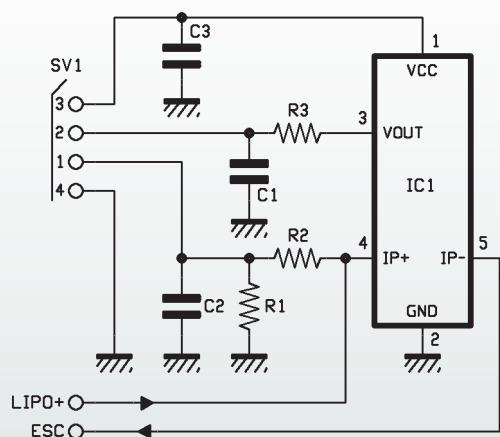
con una dissipazione di potenza pari a:

$$P = V^2 / R = 0,01 / 0,01 = 1W$$

Si è quindi deciso di usare una resistenza da 3W per evitare problemi di surriscaldamento. La caduta di tensione viene poi amplificata da un operazionale configurato come amplificatore non invertente (IC1a) che presenta un guadagno di $1 + R2/R1$, ossia di 48 volte.

La componente all'uscita dell'IC1a viene quindi inviata

[schema elettrico **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



Fig. 5
Il ricevitore GPS.



al modulo sensori mediante il contatto 1 del connettore SV1. Per quanto riguarda l'installazione, il sensore BEC va collegato con i contatti ESC direttamente all'uscita dell'alimentazione dei servizi dell'ESC, usando un cavetto con connettore maschio a passo 2,54 mm; il +5V va sul contatto R, il GND va sulla pista di massa del circuito N e il filo degli impulsi di comando va sul contatto B. Con un secondo cavetto, dotato di connessione femmina, portate ai servizi +5V, impulsi e GND, da collegare rispettivamente ai contatti R, B ed N del connettore siglato SERVO. Il connettore SV1 porterà l'alimentazione e il segnale di uscita dell'operazionale al Modulo Sensori. Per effetto di questo collegamento, tutti i circuiti di bordo appartenenti al sistema di telemetria vanno alimentati in parallelo al sensore BEC, ovvero con l'alimentazione uscente dall'ESC e non con quella

diretta al ricevitore del radiocomando e ai servo; se unite la GND di questi ultimi con il contatto 3 del connettore SV1 del sensore BEC, cortocircuitate R5 ed il sensore non rileverà alcunché.

Sensore da 200A

Per misurare le correnti di alimentazione del modello, che allo spunto possono arrivare a 150 A, abbiamo deciso di utilizzare un sensore specifico della Allegro Microsystems: si tratta dell'ACS758ECB-200B, un sensore ad effetto di Hall che può misurare correnti bidirezionali fino a 200 A. Questo elemento presenta in uscita una tensione proporzionale alla corrente che l'attraversa, pari a 10 mV/A; quando non è attraversato da alcuna corrente, il dispositivo presenta in uscita una tensione pari a metà di quella di alimentazione.

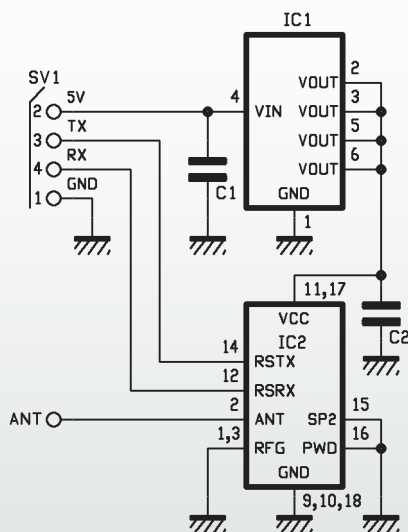
L'effetto di Hall si verifica nei

semiconduttori e determina l'addensamento dei portatori di carica elettrica lateralmente al flusso della corrente che fluisce; il sensore è sostanzialmente una barretta di semiconduttore che, quando viene attraversata da corrente, manifesta sulle due facce parallele al verso della corrente, una differenza di potenziale di valore proporzionale all'intensità della corrente elettrica, a seconda del verso della quale la polarità può essere positiva o negativa.

L'inserzione del sensore avviene tra i piedini 4 e 5, in serie al conduttore positivo di alimentazione del motore; per l'esattezza, il positivo della LiPo si applica al contatto 4 (IP+) ed il positivo dell'ESC che alimenta il motore va connesso al 5 (IP-).

Questo sensore, oltre alla corrente di ingresso misura anche la tensione del pacco batterie tramite il partitore di tensione formato da R1 ed R2, dal quale si ricava una differenza di potenziale direttamente proporzionale a quella che alimenta il motore, inviata al Modulo Sensori attraverso il piedino 1

[schema elettrico **TRASMETTITORE RF**]



Elenco Componenti:

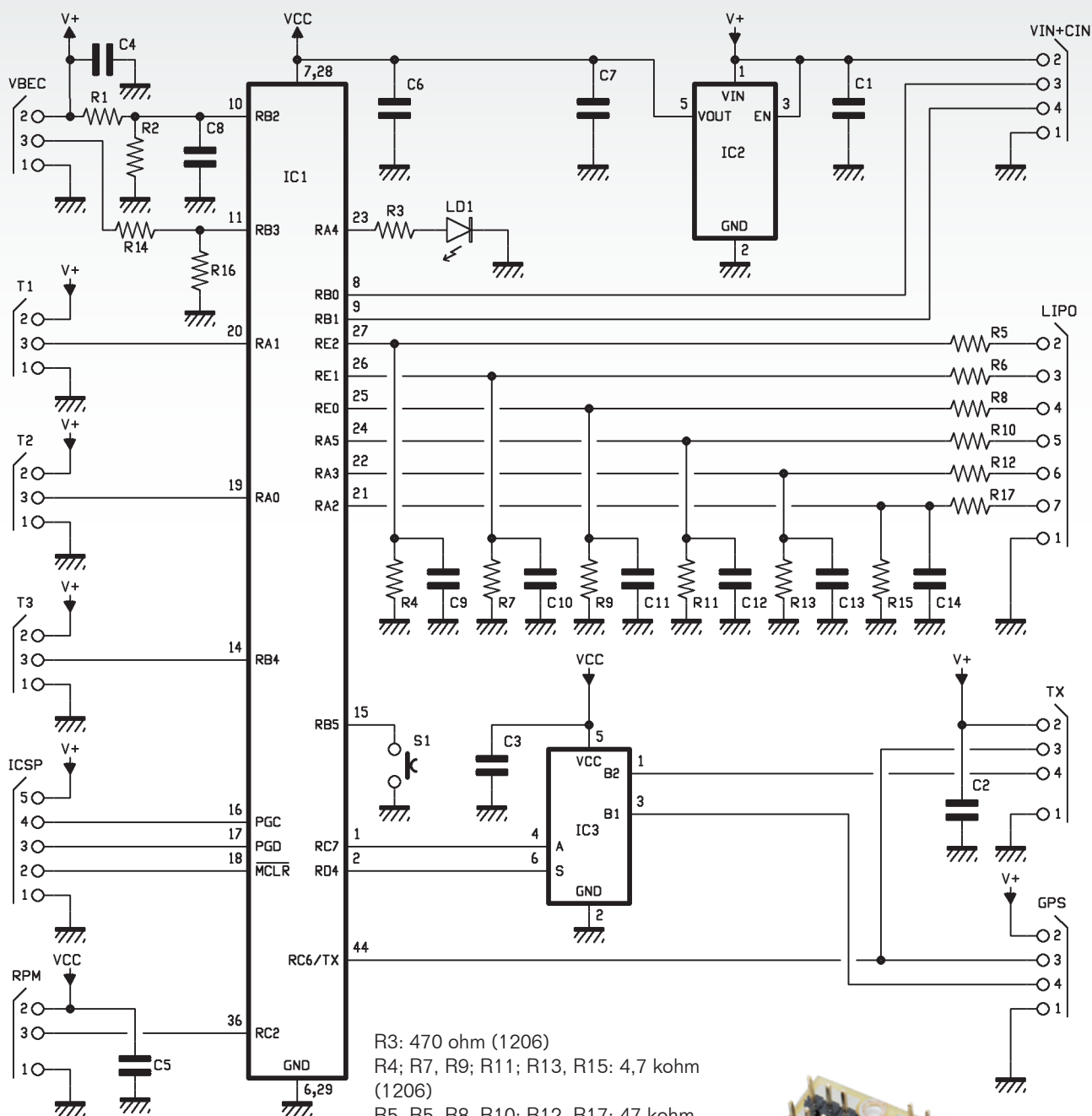
C1: 100 nF ceramico (1206)
C2: 100 nF ceramico (1206)
IC1: LD1117-3.3V (SO-08)
IC2: XTR903-A8

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



[schema elettrico **BASE SENSORI**]



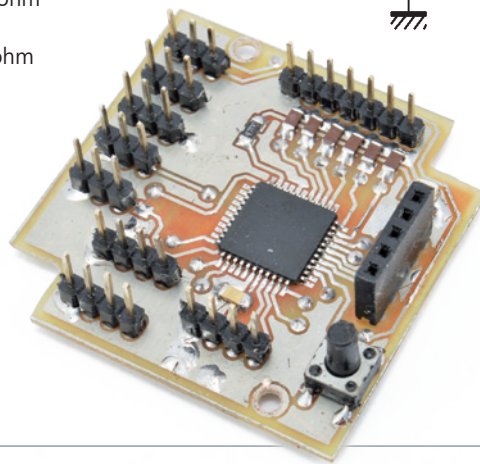
Elenco Componenti:

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



del connettore SV1 del circuito. La misura della corrente di ingresso e della tensione di ingresso permettono al circuito di determinare la potenza assorbita dalla batteria, in quanto la potenza è il prodotto della corrente per la tensione.

Attraverso l'SV1 si realizza anche la connessione di riferimento di massa con il negativo della LiPO: infatti, connettendo il sensore 200A al Modulo Sensori e il BEC a quest'ultimo, il BEC stesso, che è collegato alla massa dell'ESC e quindi al negativo della batteria, chiude il circuito sulla LiPO.

Sensori di temperatura

Sono sostanzialmente dei circuiti contenenti sensori termici analogici, ovvero semplicemente dei termistori inseriti in un partitore resistivo alimentato dalla tensione fornita dal Modulo Sensori; ne sono previsti fino a tre, da collegare ai connettori siglati T1, T2 e T3 dello stesso circuito.

I più classici sensori usati nel modellismo impiegano integrati come l'LM35 della National Semiconductors, che richiedono un'alimentazione tipica di 5 volt e forniscono su un contatto la tensione corrispondente alla temperatura rilevata; il nostro circuito è idoneo a leggere le informazioni che essi forniscono.

Un tipico sensore del genere è il TS-474002, prodotto dalla Futaba e reperibile, ad esempio, da Hobbyking (www.hobbyking.com); sempre sul sito di Hobbyking si trovano sensori composti da semplici termistori, collegabili tra il positivo di alimentazione e l'ingresso di misura, ovvero tra questo e massa.

Il ricevitore GPS

Il modulo GPS utilizzato dal sistema di telemetria è un MTK3329, capace di un refresh a 10 Hz (cioè aggiorna i dati di localizzazione ogni decimo di secondo) e di agganciare ben 66 satelliti. Il ricevitore GPS è montato su una schedina che incorpora un regolatore LDO capace di fornire la tensione di 3,3 V della quale necessita e un LED che indica quando il dispositivo sta effettuando il fix dei satelliti.

La versione da noi utilizzata monta un firmware modificato che si trova sul sito di aeromodellismo www.diydrones.com (cui raccomandiamo di dare uno sguardo) che permette di non utilizzare il solito protocollo NMEA ma un protocollo binario molto più veloce e versatile. Il collegamento al modulo sensori avviene quindi solo tramite la linea di alimentazione e le linee TX e RX del microcontrollore del Modulo Sensori, cui il GPS

si connette mediante il proprio connettore, che termina sui contatti GPS del Modulo Sensori.

Il trasmettitore

Il Modulo Sensori, man mano che acquisisce le informazioni dai vari sensori e dal GPS le trasmette in real-time tramite un modulo ricetrasmittente; più esattamente, nel sistema è stata utilizzata una coppia di moduli AUREL XTR-903-A8, operanti ciascuno a 868 MHz con 7 canali disponibili. Tramite semplici comandi AT impartiti serialmente attraverso la porta di comunicazione del microcontrollore, è possibile cambiare canale, leggere la potenza del segnale ricevuto o il canale utilizzato. I moduli lavorano nella modalità a comunicazione trasparente.

L'RTX Aurel utilizzato a bordo del radiomodello prende posto sulla scheda chiamata Trasmettitore ed è collegato al Modulo Sensori tramite i contatti 3 e 4 (rispettivamente TX ed RX) del connettore SV1; i suoi piedini RFG, PWD ed SP2 sono fissi a massa, quindi il modulo risulta sempre acceso e operativo. L'alimentazione viene ricavata, a partire da quella fornita dal Modulo Sensori attraverso i contatti 1 e 2 (GND e +5V) mediante il regolatore LDO (che nello schema è siglato IC1) che fornisce in uscita 3,3V stabilizzati, applicati tra i piedini 11,17 e la massa (pin 9, 10, 18 del modulo IC2).

Il collegamento al piedino Vin dell'EN, mantiene sempre operativo il regolatore a basso drop-out IC1.

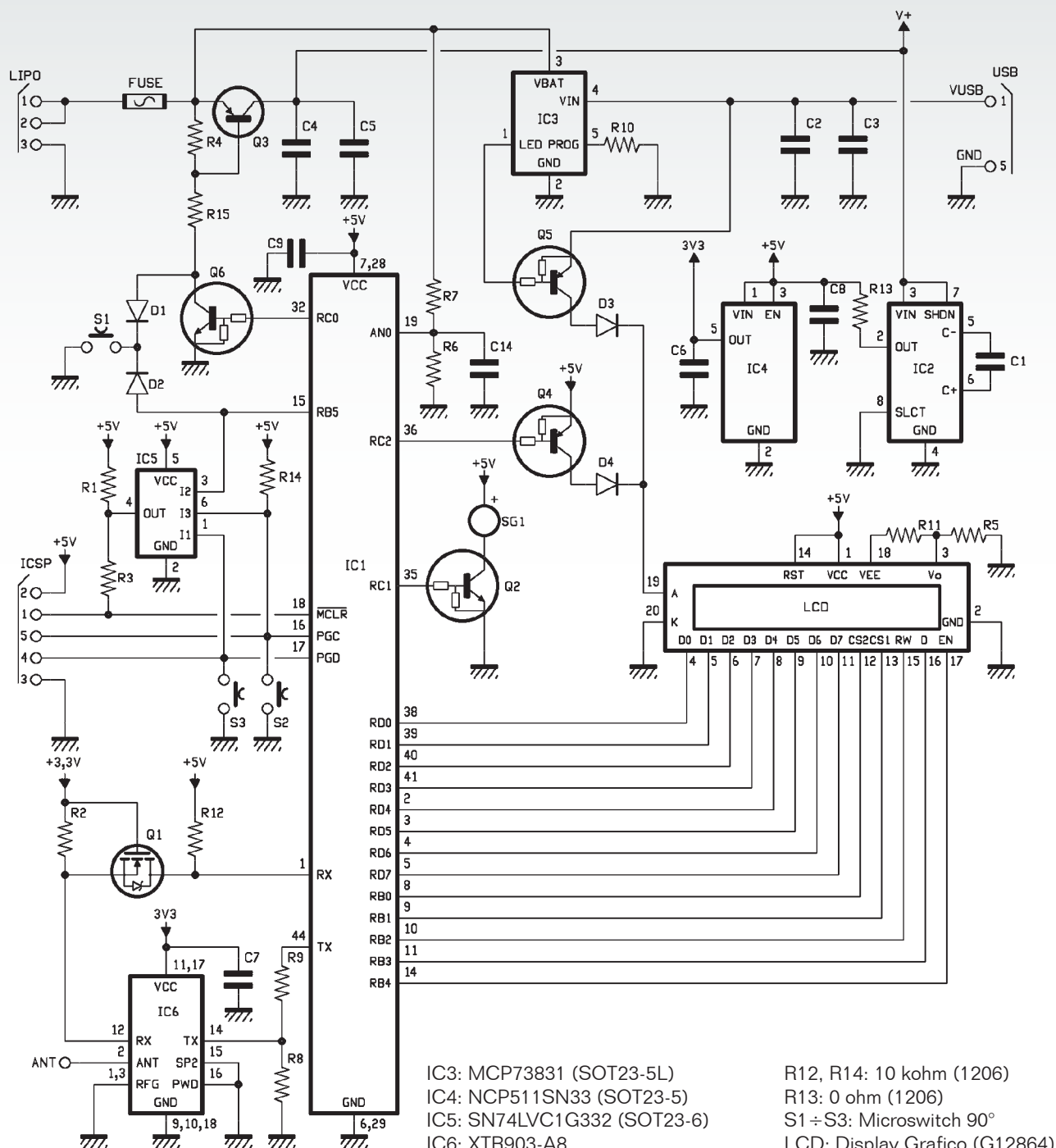
La linea ANT del modulo XTR-903-A8 va connessa ad un'antenna a stilo o ad un semplice filo in rame in guaina, lungo 9 o 18 cm (rispettivamente $\frac{1}{4}$ o $\frac{1}{2}$ d'onda). Tenete conto che migliore è l'antenna, più lontano il sistema potrà trasmettere i dati di telemetria alla stazione base situata a terra. In linea di massima, la copertura è di 500 metri, usando due antenne a stilo per 868 MHz.

IL FIRMWARE DEL MODULO SENSORI

Bene, descritti i singoli sensori e l'hardware del Modulo Sensori, possiamo adesso occuparci del firmware che gira nel microcontrollore, il quale è stato scritto in basic. All'accensione il sistema inizializza le porte e i registri, quindi se viene rilevata la pressione del tasto CH FREE parte la ricerca del canale radio libero, mentre se questo tasto non viene premuto, il firmware passa all'inizializzazione del GPS tramite i comandi AT:

HRSOut "\$PGCMD,16,0,0,0,0,0*6A\r\n"

[schema elettrico **RICEVITORE**]



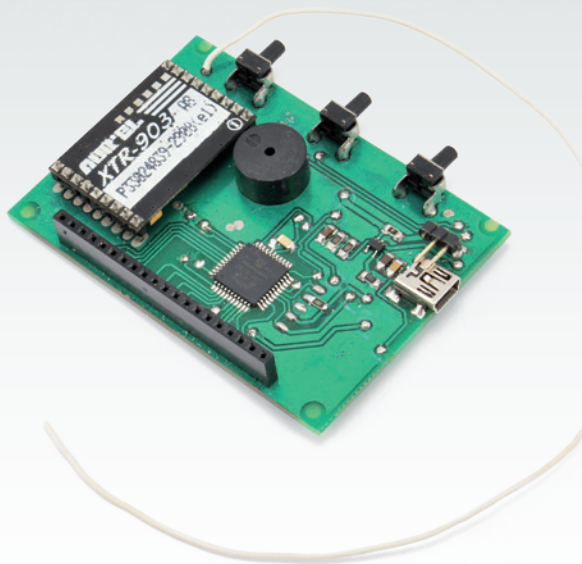
Elenco Componenti:

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



DelayMS 500

```
HRSOut "$PMTK220,100*2F\r\n"
```

Il primo commuta l'uscita dal formato NMEA a quello binario e il secondo impone un refresh di 10 Hz; la comunicazione seriale avviene a 38.400 bps. Una volta inizializzato il GPS, si parte con la scansione dei canali analogici corrispondenti al sensore BEC e a quello di corrente del motore.

Una volta ultimata la lettura di tutti gli ingressi, viene abilitato l'interrupt sulla UART in modo che la presenza dei dati sulla porta faccia avviare la bufferizzazione dei dati seriali.

Se viene calcolato un checksum uguale a quello ricevuto, i dati vengono considerati attendibili e si inizia l'invio tramite il modulo radio. Oltre all'interrupt sulla UART è stato necessario impostare un secondo interrupt sul canale Capture del PIC, cosa necessaria per determinare la frequenza del segnale proveniente dal sensore RPM.

Per evitare problemi, i due interrupt sono stati definiti con priorità differenti: quello relativo al GPS ha priorità maggiore di quello dell'RPM, secondo quanto descritto dalle righe di codice seguenti.

```
intrpm:
```

```
Context Save
```

```
If b = 0 Then
```

```
oldcap.HighByte = CCPR1H
```

```
oldcap.LowByte = CCPR1L
```

```
EndIf
```

```
If b = 1 Then
```

```
capture.HighByte = CCPR1H
```

```
capture.LowByte = CCPR1L
```

```
diff = capture - oldcap
```

```
Clear TMR1H
```

```
Clear TMR1L
```

```
vis = 1
```

```
EndIf
```

```
Inc b
```

```
a = 0
```

```
Clear CCP1IF
```

```
Context Restor
```

Non appena si verifica il primo interrupt sul canale CCP1, viene memorizzato il valore attuale del registro CCPR1, che al secondo interrupt viene sottratto al valore letto; questa differenza rappresenta proprio la frequenza del segnale.

Se il sistema viene acceso mentre il tasto CH FREE risulta premuto, viene avviata la scansione dei canali radio alla ricerca di quello libero.

La scansione avviene impostando i canali uno alla volta, leggendo l'intensità del segnale corrispondente e scrivendone il valore numerico in un array; arrivati all'ultimo canale, il firmware effettua una verifica sugli elementi dell'array ed il primo che ha valore 0 viene usato come canale radio.

Una volta avviato il sistema, la pressione del tasto CH FREE permette di visualizzare, tramite lampeggi del LED, il canale attualmente utilizzato.

Per quando riguarda l'invio dei dati acquisiti, questi vengono ordinati in un frame formato da alcuni caratteri speciali che identificano i vari dati, secondo questa struttura:

```
HSerOut [" *"]
```

```
HSerOut ["A",@ ampere,"-"]
```

```
HSerOut ["B",@ volt,"-"]
```

```
HSerOut ["C",@ sign,"-"]
```

```
HSerOut ["D",@ temp1,"-"]
```

```
HSerOut ["E",@ temp2,"-"]
```

```
HSerOut ["F",@ temp3,"-"]
```

```
HSerOut ["G",@ vbec,"-"]
```

```
HSerOut ["H",@ cbec,"-"]
```

```
HSerOut ["I",@ rpm,"-"]
```

```
HSerOut ["L",@ cella1,"-"]
```

```
HSerOut ["M",@ cella2,"-"]
```

```
HSerOut ["N",@ cella3,"-"]
```

```
HSerOut ["O",@ cella4,"-"]
```

```
HSerOut ["P",@ cella5,"-"]
```

```
HSerOut ["Q",@ cella6,"-"]
```

```
HSerOut ["R",@ speed,"-"]
```

```
HSerOut ["S",@ time,"-"]
```

```
HSerOut ["T",@ day,"-"]
```

```
HSerOut ["U",@ fix,"-"]
```

```
HSerOut ["V",@ sat,"-"]
```

```
HSerOut ["Z",@ quota,"-"]
```

```
HSerOut ["K",@ chk,"-"]
```

```
HSerOut ["# "]
```

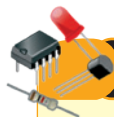
La trasmissione inizia con un "*" e ciascun dato è preceduto da una lettera che ne identifica il sensore corrispondente; **chk** è un checksum calcolato come segue:

```
chk = temp1 + temp2 + temp3 +  
vbec + time + day
```

La trasmissione viene chiusa con il carattere "#".

L'UNITÀ RICEVITORE

A visualizzare tutti i dati inviati dall'unità remota provvede il ricevitore, dotato di un display grafico a matrice di punti da 128x64. Come mostra il relativo schema elettrico, questa unità dispone di un microcontrollore PIC18F4620, un modulo RTX radio uguale a quello collegato all'unità remota, un sistema di carica per batterie LiPO tramite USB ed un circuito di alimentazione necessario per il display. L'alimentazione del ricevitore è



per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono di facile reperibilità. I master dei circuiti stampati ed il firmware utilizzato per programmare i microcontrollori 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>

affidata ad una batteria lipo da 3,7 V – 650 mAh; considerando un consumo massimo con la retroilluminazione del display accesa di 110 mAh, la batteria può arrivare tranquillamente a oltre 5 ore di utilizzo, che aumenterebbero in caso di uso con la retroilluminazione spenta. In linea di massima, lasciando spenta la retroilluminazione l'autonomia dell'unità aumenta del 25%.

L'integrato IC3, l'MCP73831 della Microchip, si occupa della carica con ciclo CC-CV della LiPO. La corrente di carica è determinata dalla resistenza R10, secondo la formula $I_c = 1.000/R$, dove, per avere la corrente in mA, R deve essere espressa in kohm.

Tale corrente non è proprio quella di carica che in realtà viene diminuita a causa del fatto che durante la carica, tale stato viene indicato dalla retroilluminazione del display che resta acceso per tutta la durata del processo. La corrente reale sarà quindi circa 420 mA. Lo stadio successivo è formato da R4, R15

e Q3 che formano l'interruttore digitale che serve ad accendere e spegnere la parte elettronica a valle. In condizioni normali R4 assicura l'interdizione del Q3; se però viene premuto il tasto S5, questo porta la base a massa tramite R15 e Q3 si porta in saturazione, applicando l'alimentazione a valle. Per mantenere il sistema acceso sarà sufficiente protrarre tale stato sulla base, cosa che viene garantita dal Q6, portato in saturazione dal PIC non appena avviato, provvede a mantenere il sistema acceso. Quando deve spegnere il circuito, il micro lascia interdire Q6. Il microcontrollore necessita di una tensione stabilizzata, quindi non possiamo alimentarlo direttamente con la batteria, la cui tensione varia, tra lo stato di carica e quello di scarica, tra 4,2 V e 3,3 V.

Questa variazione così ampia rende impossibile l'alimentazione diretta del PIC e del display che, inoltre, ha bisogno di 5 V per funzionare in modo corretto. La soluzione è stata trovata in un regolatore DC/DC a pompa capacitiva siglato MCP1253, che sfrutta la carica e scarica del condensatore da 1 µF collegato tra i suoi piedini 5 e 6 per ricaricare una tensione stabilizzata del valore di 5 V ed erogare una corrente di oltre 100 mA.

Questa tensione servirà ad alimentare tutta l'elettronica compreso il modulo radio Aurel tramite il regolatore lineare a basso drop-out LD1117, il quale fornisce 3,3 V stabilizzati. Nel circuito, il microcontrollore si occupa della gestione del display, del buzzer e della retroilluminazione, del controllo di carica della batteria tramite il partitore R7-R6 e della gestione dei tasti. Come già detto, durante la fase di carica la retroillumi-

nazione del display viene accesa dall'MCP73831 tramite il pin LED, che pilota Q5.

Per permettere al PIC di comunicare correttamente con il modulo radio, che è alimentato a 3,3 V, abbiamo inserito un circuito traslatore di livello dei segnali seriali: i livelli logici in uscita dal PIC e diretti al modulo sono semplicemente stati abbassati tramite un partitore a 2/3 (portandoli da 0/5V a 0/3,3V) mentre quelli diretti dal modulo RTX al microcontrollore è stato realizzato un doppio stadio invertente realizzato con due transistor, l'ultimo dei quali è alimentato a 5 volt, in modo da amplificare i segnali 0/3,3V per trasformarli in 0/5V.

Notate che, avendo esaurito le linee disponibili nel PIC, per ottenere il reset abbiamo usato un artificio: se tutti e tre i tasti presenti nel circuito vengono premuti contemporaneamente, il micro viene resettato.

Per realizzare questa funzione logica, considerando che i tasti premuti hanno livello logico 0 e l'MCLR del PIC è attivo con livello logico 0 si è deciso di utilizzare una porta OR a tre ingressi, che, nello specifico, è contenuta in un integrato SN-74LVC1G332.

La ricarica della batteria è ottenuta prendendo alimentazione dal connettore miniUSB di cui l'unità è dotata, quindi allo scopo il circuito va collegato ad un PC o ad un caricabatterie universale a 5 V dotato di attacco miniUSB.

Durante la carica, il circuito accende la retroilluminazione del display, facendola spegnere a carica terminata; è consigliabile che, durante la carica della batteria, il circuito venga tenuto spento, così da velocizzare il procedimento. ■



OSIMONE

Il tema delle stampanti 3D occupa sempre più spazio sulla stampa generalista per i motivi più vari: dalla realizzazione di organi umani (e quindi salvare vite) alla realizzazione di armi che sparano veri proiettili (e quindi tolgono la vita). Fra questi due estremi

si trovano un'infinità di realizzazioni che sfruttano sia le stampanti come la nostra 3Drag, sia le stampanti professionali utilizzabili in service ricorrendo ad aziende quali Shapeways, Pono-ko o Sculpteo. Mentre il mondo acquisisce una crescente

consapevolezza dell'esistenza delle stampanti 3D e di come queste possano impattare sulla vita di tutti i giorni, quelli che hanno deciso di equipaggiarsi con una nostra 3Drag stanno già creando oggetti e progetti basati sulla tecnologia della fabbricazione con

filamento fuso. In questi mesi anche noi abbiamo lavorato per migliorare sempre più le nostre abilità nel gestire la

Focus Stampa 3D

Da qualche mese non affrontiamo più su queste pagine il tema della stampa di oggetti 3D, ma non per questo ci siamo astenuti dal valutare e fare stampe di svariati oggetti. Condividiamo con voi le nostre "scoperte" e anche dieci oggetti che potrete usare per spiegare cosa può fare la vostra 3D.

COME AVERE IL MASSIMO DALLA TUA

3DRAG

stampante e allo stesso tempo abbiamo potuto fare tutte le valutazioni sulle sue prestazioni, individuando i pregi e i difetti rispetto alle altre stampanti disponibili sul mercato.

Quello che comunque abbiamo capito è che non è sempre tutto ovvio e che uno stesso modello può risultare perfetto o disastroso pur essendo stampato con la medesima stampante. La differenza sta tutta nei parametri con cui viene realizzato lo slicing e – conseguentemente –

come viene pilotato l'estrusore per realizzare fisicamente gli strati in materiale plastico.

IL SOFTWARE "STATE OF THE ART"

Al momento in cui scriviamo, i software che utilizziamo per la stampa sono Slic3r 0.9.9 e Repetier-Host 0.85b; gli altri pacchetti disponibili per effettuare lo slicing e la stampa stanno seguendo le loro strade, con Skeinforge ormai in uno stato di scarso aggiornamento per disinteresse degli utenti e le interfacce come Pronterface che hanno ceduto

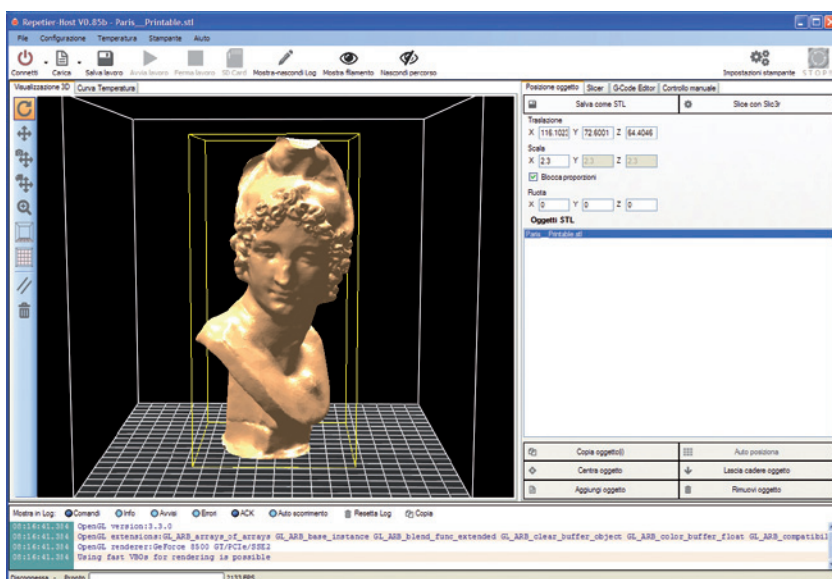
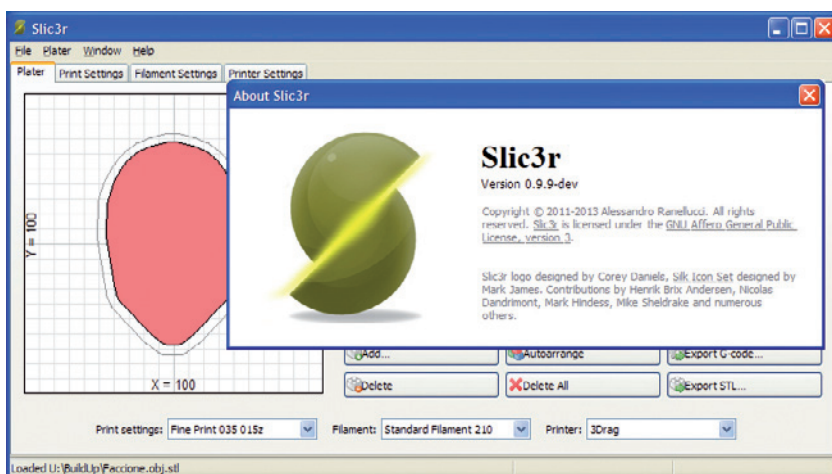
il passo a soluzioni o proprietarie o più evolute. Sul fronte del firmware, Marlin prosegue nella sua evoluzione aggiungendo e togliendo prestazioni a seconda dei report degli utenti. Il tema degli archi, ad esempio, è stato temporaneamente disabilitato per problemi di correttezza e stabilità nel funzionamento di questa specifica caratteristica. Un'altra interessante informazione trovata ultimamente fra forum e commenti è la presenza di qualche problema occasionale (molto occasionale) nella matematica per il calcolo degli step



del motore con il risultato che in situazioni molto particolari, il piatto di stampa può perdere degli step e quindi non effettuare il corretto spostamento del piatto di stampa, creando quell'antipatico gradino che rovina senza appello la stampa. Ci è capitato più di una volta (e spesso con delle specifiche stampe) di avere la perdita secca di qualche step con uno sfasamento fra il mezzo e i 3 millimetri su uno dei due assi (solitamente la X), me senza l'informazione sui problemi del firmware abbiamo ritenuto che la cosa fosse da addebitare all'abbinata meccanica/driver. Ora sappiamo che ci si può mettere anche il firmware. Provate a spostare il modello sul piatto di stampa, rilanciate lo slicing e senza modificare parametri vedrete che la perdita di step potrebbe essere eliminata. Ovviamente questo ragionamento non vale se gli step sono persi in situazioni di stampa estreme, ovvero ad alta velocità e con piccoli spostamenti alternati come direzione su entrambi gli assi (come ad esempio il fili di piccoli interstizi fra pareti).

QUALCHE CONTROLLO PRIMA DI PARTIRE

Prima di precipitarvi a stampare i dieci modelli che abbiamo



scelto per aiutarvi a stupire e a stupirvi, vogliamo ripassare con voi qualche elemento generale relativo alla vostra 3Drag e alle sue impostazioni.

Partiamo dal presupposto che la meccanica sia a posto, con le cinghie tese (senza che diventino corde di chitarra), i vari bulloni serrati, il piatto di stampa livellato correttamente attraverso le 4 rotelle sotto ai quattro angoli e che la piastra in vetronite o il piatto riscaldato siano correttamente trattati per far aderire il PLA o l'ABS che userete per le

stampe. Il corretto livellamento del piatto va fatto portando a "0" l'asse Z – con l'estrusore che tocca il piano di stampa - quindi si ruota di un mezzo giro la vite dell'asse Z per alzare la punta di quel decimo di millimetro necessario a non farlo sfregare sul piatto di stampa. Spostate manualmente il piatto e controllate che questa distanza resti costante in tutte le posizioni. Aiutatevi con un foglio di carta fra piatto e punta dell'estrusore per sentire quando il foglio viene bloccato o non ha attrito quindi

regolate le viti di conseguenza. Può capitare che il piatto abbia delle deformazioni che creano avvallamenti o dossi al centro del piano o a metà dei lati, dove le regolazioni non possono agire. Potete smontare la piastra e a mano cercare di riportarla in forma, oppure potete usare degli spessori fatti con un pezzetto di carta ripiegato una o due volte, infilati fra il supporto di allu-



Cuscinetto a ingranaggio

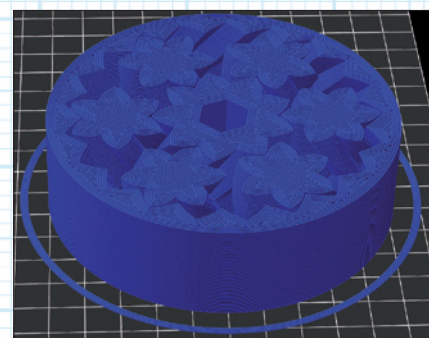
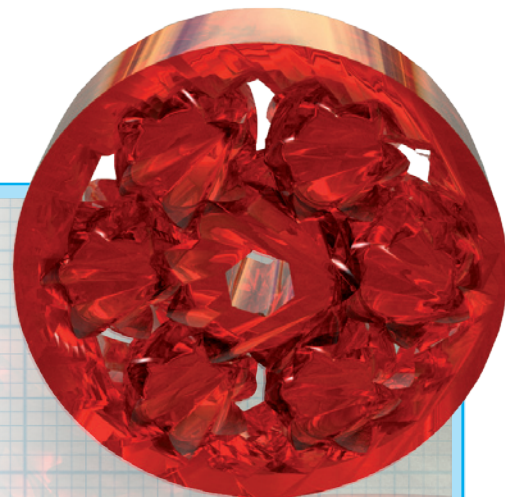
Questo modello è molto particolare in quanto fa parte di quella categoria delle stampe che si auto assemblano e che, se realizzate a singoli pezzi, non possono essere montate. In pratica gli ingranaggi che compongono questo cuscinetto hanno dentature a coda di rondine. Questo evita che i sei ingranaggi attorno a quello centrale possano uscire dalla loro sede, ma allo stesso modo rende impossibile un assemblaggio partendo dai singoli pezzi. Un ottimo esempio di meccanismi che sono pronti all'uso immediatamente dopo la fine della stampa in 3D. Le dimensioni del pezzo sono "importanti" e per questo i parametri di slicing sono "grezzi" privilegiando il tempo di stampa e l'impiego di materiale rispetto a un inutile dettaglio. Se intendete utilizzare questo meccanismo in qualche applicazione, allora dovete

intervenire sulla sua densità lavorando sullo spessore dei perimetri, che devono essere saldati fra loro, senza che gli ingranaggi si tocchino (e conseguentemente si fondono). Questo modello è scalabile verso il basso (riduzione) purché si diminuisca il diametro di estrusione in proporzione, attraverso la scheda ADVANCED dei parametri di Print Setting di Slic3r. Controllate da Repetier - nel visualizzatore G-Code - che gli ingranaggi non si tocchino saldandosi; lo spazio deve essere ben visibile, altrimenti anche il contatto porta alla fusione. In caso di dubbio, avviate la stampa vera e propria e controllate che al quarto o quinto layer, gli ingranaggi siano visibilmente separati fra loro. Se invece ingrandite il modello, non dovrete incontrare problemi.

Nome File: **bearing6.gcode**

PARAMETRI DI STAMPA:

layer_height =	0.2	travel_speed =	150
perimeters =	4	scale =	1
top_solid_layers =	4	nozzle_diameter =	0.5
bottom_solid_layers =	4	filament_diameter =	3
fill_density =	0.5	extrusion_multiplier =	1
perimeter_speed =	90	perimeters extrusion width =	0.80mm
infill_speed =	90	infill extrusion width =	0.80mm
		first layer extrusion width =	0.60mm



minio e la vetronite. Purtroppo dovete fare il possibile per avere il piatto di stampa perfettamente in piano perché il primo stato condiziona la tenuta e la riuscita della maggior parte degli oggetti; gli spessori in gioco vanno da 2 a 4 decimi di millimetro per poi stabilizzarsi fra 15 e 20 centesimi negli strati successivi. Un controllo anche alla punta dell'estrusore e possiamo partire. Questa deve essere priva di ammaccature, con il foro ben centrato sulla punta e di 0,5 mm

di diametro esatti. Se potete, aiutatevi con uno specchio per evitare contorsioni o smontaggi, ma verificate che sia in ordine. Rimuovete l'eventuale accumulo di plastica fusa e bruciata così da evitare che ci possano essere agganci e trascinamenti di eventuali bave di PLA che emergono dalla stampa in corso. Se il foro risulta per qualche motivo non perfetto o ammaccato, potete aiutarvi a ridargli la forma circolare con uno spillo/ago d'acciaio purché sia da 0,5mm e non oltre

(accertatevi con un calibro prima di usarlo). Alla fine del vostro intervento, il foro deve essere piatto e senza sbavature; non utilizzate lime perché compromettereste l'estrusore che è in ottone e quindi molto morbido. Potete sincerarvi dello stato dell'estrusore provando ad estrarre a vuoto un po' di plastica: se il filo che si crea scende dritto verso il basso, senza fare ricciolini e senza deviare in una qualsiasi direzione laterale, il foro è perfetto. Se ci sono deviazioni costanti,

Artisti, architetti, medici, e...

La Marina Militare statunitense ha di recente pubblicato sull'Armed Forces Journal un articolo a firma del Lt. Cmdr. Michael Llenz dal titolo "Print when ready, Gridley". Il tema dell'articolo è quello della razionalizzazione – per non dire rivoluzione – della catena dei rifornimenti inserendo stampanti 3D alla fine del processo o in prossimità dei teatri d'azione. In pratica, l'accelerazione delle tecnologie per la produzione non solo con metalli, ma anche combinando parti in materiali diversi ha reso possibili degli scenari che solo tre anni fa non sarebbero stati immaginati come possibili. Quando la digital fabrication si avvale delle risorse militari i risultati che vengono perseguiti non si fermano al piccolo oggetto o alla parte di ricambio: qui si parla di assemblare droni con tanto di componenti elettronici e combustibile direttamente "stampato" e auto assem-



blato. Il tutto ridisegnando la catena degli approvvigionamenti in termini di "materie prime" da far pervenire ai centri di stampa. Ovviamente il tutto rigorosamente nella filosofia del "print on demand" di tutti i vari dispositivi che possono servire sia come ricambi, sia come parti finite. L'articolo è molto recente e affronta anche il tema degli utilizzi da "lato oscuro", come l'iniziativa di Cody Williams per la sua pistola Liberator; la "cattiva stampa" generata da questo genere di iniziative è – secondo l'autore – abbastanza fisiologica, ma i timori e le problematiche che le tecnologie di stampa e acquisizione 3D comportano sono da considerare molto seriamente. Rubare un progetto top secret – e Wikileaks al proposito ci ha fatto capire che i dati su sistemi collegati in Rete sono sempre in pericolo – è un paio di maniche, ma farsi rubare la

"ricetta" per la stampa in 3D di un'arma top secret è decisamente peggio. Quello che sfugge a noi civili è la portata delle ricerche in corso: mentre noi ci studiamo per le applicazioni nel campo della pasticceria o della medicina, i militari lavorano per stampare in 3D le munizioni direttamente sul campo, cambiando completamente le modalità produttive. E che la stampa 3D non sia una moda lo confermano anche le prese di posizione di personaggi quali Jeffrey Immelt – CEO di General Electric – che ha definito la stampa 3D "meritevole dei miei sforzi, del mio tempo e dei miei soldi": un'affermazione molto forte da un'azienda che è famosa per la sua efficienza aggressiva. Vi invitiamo a leggere l'articolo nella sua interezza per avere un ulteriore punto di vista sul futuro di questa tecnologia. Lo trovate qui: <http://armedforcesjournal.com/2013/05/13520067/>

allora c'è un problema. Un altro indicatore di estrusione ottimale è la formazione di un anello abbastanza omogeneo che tende a crescere in modo regolare mentre fate estrudere a vuoto. Più largo è l'anello formato dalla plastica e più questa è viscosa, mentre più l'anello è piccolo più liquida è la plastica.

COME DOMARE IL PLA

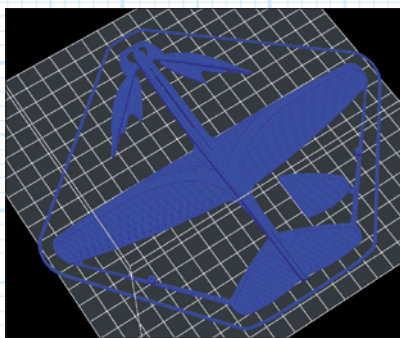
Quando abbiamo iniziato a stampare, pensavamo che la temperatura avesse un'importan-

za relativa, immaginando anche un intervallo di temperature di "stampa" abbastanza ampio. Il PLA – materiale che noi preferiamo per molte delle nostre stampe grazie a come si presenta alla vista e al tatto – in effetti inizia ad ammorbidire già a 70/80 gradi centigradi, diventando fluido a 175 gradi circa e liquido a circa 220. L'esatta temperatura per avere la fluidità ideale per la stampa dipende da come il PLA neutro è stato "caricato" con altre sostanze

per alterarne il colore e la trasparenza. Dalla nostra esperienza vi possiamo anticipare che i PLA opachi tendono ad avere bisogno di un po' più di calore, mentre quello oro e quello fosforescente hanno richiesto una temperatura sensibilmente minore per avere una stampa di qualità. Per dare qualche numero, l'oro a 193 gradi era già molto fluido, mentre quel-

Monarch IX - Aeroplanino

Il nome originale di questo file è "Monarch_IX_with_tags"; si tratta di un derivato da un altro progetto di aeroplanino che già offriva dei buoni risultati (il modello si chiama glider). Rispetto alla versione di partenza, le forme sono più arrotondate e la composizione sul piatto di stampa è tale da garantire una



maggiore resistenza ai problemi di distacco, offrendo allo stesso tempo degli appigli con cui sollevare l'oggetto finito senza rovinarlo.

La parte del timone di coda e quella della cabina sono da montare manualmente in apposite fessure nella parte posteriore e quella frontale. Un punto di colla ciano acrilica può rendere il posizionamento permanente. Per volare, questo modello ha i pesi abilmente calibrati e presenta una struttura a strati che fornisce alle ali la giusta leggerezza e rigidità, mentre il corpo è rigido e robusto. Solo due strati nella parte principale delle ali è una sfida non da poco per la vostra stampante e questo modello è il banco di prova ideale per la messa a punto di allineamento, livellamento e omogeneità della superficie. Dalla tabella dei parametri si nota che lo spessore degli strati è 0.18mm,

ma l'informazione più rilevante riguarda il diametro del filo estruso, pari a 0.35 mm sia per il primo strato, sia per i successivi. Questo parametro si regola dal pannello "Advanced" del profilo di stampa. La velocità di stampa è impostata a 90 mm/sec e può essere gestita in tempo reale tramite il cursore dei controlli manuali per il "Feedrate". In termini di utilizzo, vi consigliamo il lancio manuale e non con l'elastico, anche se la parte da inserire come "cabina di pilotaggio" prevede il gancio proprio per questo scopo.

Nome File:
Monarch_IX_with_tags.gcode

PARAMETRI DI STAMPA:

layer_height =	0.18	scale =	1
perimeters =	4	nozzle_diameter =	0.5
top_solid_layers =	4	filament_diameter =	3
bottom_solid_layers =	3	extrusion_multiplier =	1
fill_density =	0.2	perimeters extrusion width =	0.35mm
perimeter_speed =	90	infill extrusion width =	0.35mm
infill_speed =	90	support material extrusion width =	0.35mm
travel_speed =	140	first layer extrusion width =	0.35mm



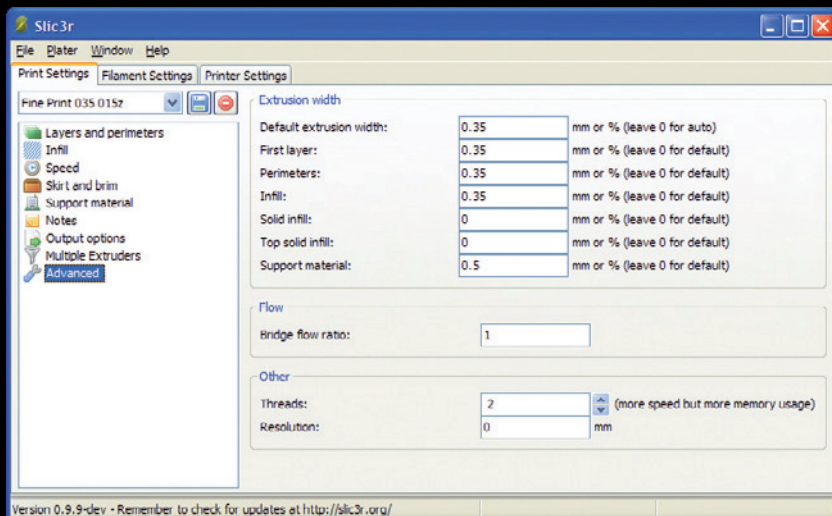


Fig. 1 - I parametri di "extrusion width" permettono di ottenere dal nostro estrusore da 0,5mm dei fili di diametro inferiore o superiore, migliorando o adattando la risoluzione di stampa al tipo di modello.

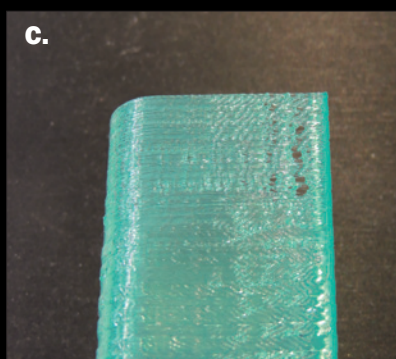
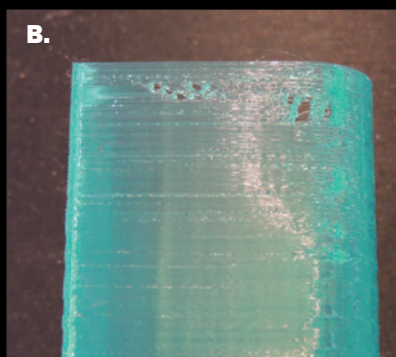
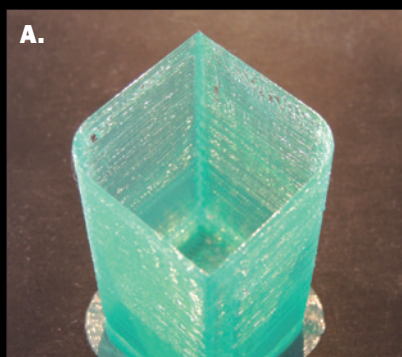
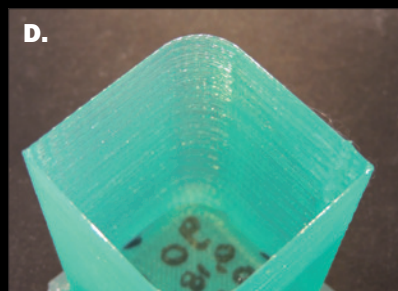
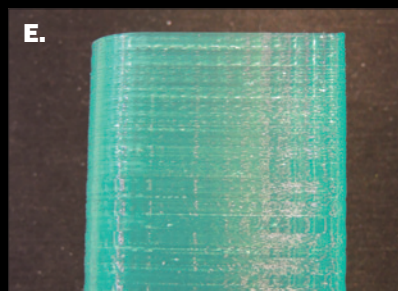


Fig. A-E - Questa stampa di test evidenzia in A, B e C i difetti che derivano da una errata combinazione fra velocità e temperatura: gli strappi nel filo sono sintomo di temperatura errata; in D ed E si vede invece la stampa senza difetti grazie alle impostazioni ottimali; la parete ha uno spessore costante di 0.35mm ed è quindi intenzionalmente critica per la stampa.

lo fosforescente a 195 gradi era perfetto. Il classico verde o rosso trasparente sono da usare fra i 198 e i 205, mentre quelli opachi come l'azzurro o l'arancione e il rosa lavorano bene anche a 205 – 215 gradi.

Come capire la migliore temperatura di lavoro? Intanto vale la pena capire come questa plastica si comporta all'aumentare della temperatura: passa dalla fase vetrosa a quella fluida attorno ai 65

gradi centigradi e diventa liquida attorno ai 220 gradi. Nell'intervallo fra questi due valori il comportamento non è però solo di un progressiva diminuzione della viscosità, ma anche dell'insorgere di particolari comportamenti quali la creazione di fili



Music box

Questa è una bella prova di pazienza in quanto la stampa dell'intero set di pezzi richiede diverse ore. Il cilindro con le tacche corrispondenti alle note da suonare è il pezzo più impegnativo (lungo) da stampare, mentre la struttura con le lamelle è quella più delicata e critica. Gli ingranaggi sono invece relativamente semplici. Per aumentare la loro solidità abbiamo scelto di stampare impostando a 0.35mm lo spessore dei perimetri. Come effetto questo porta ad avere un maggiore dettaglio nel disegno dei denti degli ingranaggi che, allo stesso tempo, vengono realizzati con perimetri molto più vicini fra loro e quindi più robusti. Provate a fare lo slicing impostando valori da 0.5mm a 0.3mm a passi di 0.05mm per lo spessore dei perimetri e capirete cosa intendiamo per "maggiore dettaglio". Ovviamente non dovete stampare, ma basta osservare i singoli layer da Repetier nella finestra dell'editor G-Code. Dopo che avrete stampato i vari pezzi, potrete assemblarli partendo dalla rotellina più piccola, da infilare nel foro tondo e poi da far scivolare in alto e scattare sul pernino predisposto; poi inserite la ruota più grande, con l'ingranaggio più grande verso l'esterno e quello più piccolo all'interno. L'ultimo pezzo da mettere in posizione è il cilindro. Fatelo con estrema attenzione

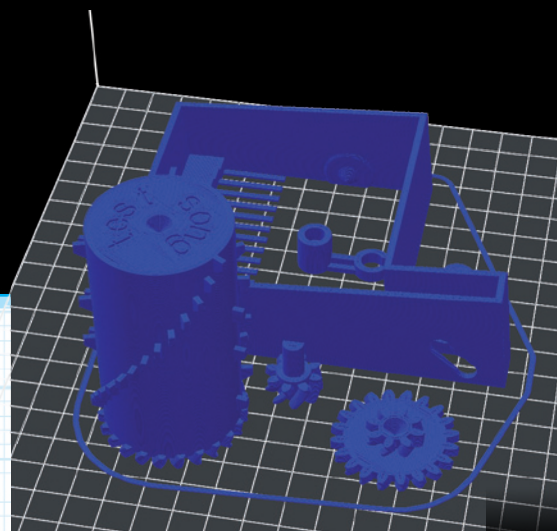
sottili come quelli della tela di un ragno, che si estendono da un punto all'altro delle nostre stampe, quando l'estrusore si sposta fra parti non connesse dello strato in fase di creazione.

Non siamo esperti del comportamento delle materie plastiche in fase fluida, ma abbiamo dedicato un po' di tempo all'osservazione sperimentale del PLA durante le stampe e i risultati sul campo ci portano ad affermare che sopra una certa temperatura, il PLA tende a formare delle goccioline piuttosto che restare un filamento di diametro costante. In pratica la tensione superficiale

e la natura viscosa della fase fluida portano alla creazione di tante sfere collegate da un filo sottilissimo. Il fenomeno è accentuato dallo stress meccanico dovuto dall'estrusione ad elevata velocità, dove la lunghezza del segmento estruso sopra la temperatura ottimale è apprezzabile. In altre parole, ipotizzando che a 210 gradi il PLA produca questo filo sottile con i pallini al posto del filo del diametro previsto, si può diminuire il fenomeno rallentando la stampa, ovvero avendo un segmento di PLA sopra la temperatura critica più corto. Nelle stampe, si nota questo

effetto nelle pareti che risultano come strappate o segmentate e fragili.

La soluzione che nel tempo si è rivelata più efficace è l'abbassamento della temperatura di stampa, lasciando solo al primo strato una temperatura in cui il PLA è molto fluido per consentirgli una buona aderenza, ma subito dopo



perché se un dente si punta su una lamella questa si romperà (noi ad oggi ne abbiamo rotte già 3). La manovella va girata in senso orario. NON lasciate il carillon con una o più lamelle piegate o a contatto con un dente perché tendono a deformarsi: spostate il rullo in un punto in cui non ci sono denti a contatto con le lamelle e il vostro carillon funzionerà a lungo. Sul sito Thingiverse (<http://www.thingiverse.com/thing:53235>) trovate l'applicazione per realizzare altri cilindri musicali con note definite da voi.

Nome File:

FullyPrintableParametricMusicBoxV3_
completePlate.gcode

PARAMETRI DI STAMPA:

layer_height = **0.18**
perimeters = 4
top_solid_layers = 4
bottom_solid_layers = 3
fill_density = **0.2**
perimeter_speed = 90
infill_speed = 90

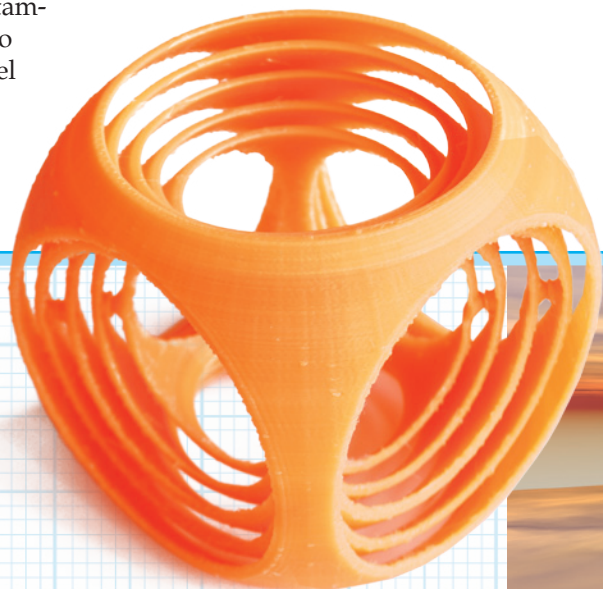
travel_speed = 140
scale = 1
nozzle_diameter = 0.5
filament_diameter = 3
extrusion_multiplier = 1
perimeters extrusion width = **0.35mm**
infill extrusion width = **0.35mm**
first layer extrusion width = **0.35mm**



che ci permette di andare ad alta velocità senza che il filo si trasformi in questa specie di collana di perline. In alcuni casi l'accensione della ventola cura il difetto in quanto lo scambio di calore fra il filo già deposto e quello in fase di deposizione raffredda il PLA sufficientemente in fretta ed impedisce la creazione dei mini grumi a perlina. In altri casi si può semplicemente scendere di temperatura finché questo comportamento sparisce. Non abbiate paura a scendere perché il PLA – a differenza dell'ABS – offre una ottima adesione fra

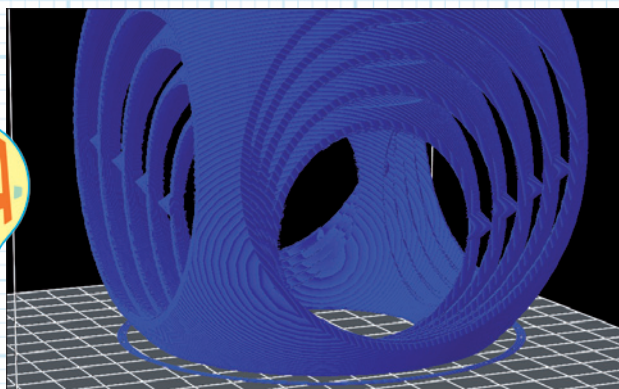
strati anche se la temperatura può sembrare bassa. Nelle nostre prove, siamo partiti da PLA che avevamo valutato come adatti ai 215 gradi per poi scoprire che a 205 tenevano bene anche con velocità oltre i 100mm/sec. Ricapitolando, se il PLA tende a non fare più il filamento omogeneo, ma si trasforma in pallini, o trattini, o comunque perde di continuità, la temperatura è troppo alta, soprattutto se state stampando applicando il trucco della riduzione virtuale del diametro di estrusione (Fig. 1); in questo caso il PLA viene "tirato" per

ridurre il diametro del filo estruso e questo porta allo "strappo" anche a velocità relativamente basse. Osservate le immagini che abbiamo preparato per capire di quale effetto stiamo parlando (Fig. A-E). In questo caso la sperimentazione è indispensabile per conoscere il comportamento specifico di ciascun tipo di PLA che state



Hypercube

Questa non è una stampa facile, anzi! Ha diversi accorgimenti "onerosi" in termini di tempo e di materiale per garantire il più possibile una buona riuscita. Il primo accorgimento è un'estrusione intenzionalmente più ricca per dare maggiore robustezza ai vari pezzi che – da progetto – sono abbastanza sottili e delicati. Il diametro a 2,8 mm del filo di plastica da estrarre e la moltiplicazione per 1.3 della quantità di materiale sono il metodo che abbiamo adottato. Nelle varie prove di stampa abbiamo anche riscontrato che l'ugello tende a urtare le parti ogni volta che passa da un pezzo all'altro dei quattro elementi che salgono in verticale per ciascun cubo, sottoponendo l'oggetto a vibrazioni e urti che con grande probabilità porteranno al distacco dal piano di stampa. La soluzione al problema è stato attivare il sollevamento dell'estrusore durante gli spostamenti e questo metterà a dura prova la barra filettata del vostro asse Z. Anche i tempi di stampa risentono MOLTO di questo movimento verticale aggiunto agli spostamenti orizzontali, ma gli



urti e le sollecitazioni meccaniche vengono ridotte moltissimo, evitando il pericolo del distacco. Dovete poi lavorare sulla temperatura per trovare quella che offre aderenza senza che si crei il filo sottilissimo durante il passaggio da un punto all'altro. Per motivi di matematica e di percorsi di estrusione, vengono realizzate delle bavette fra i pezzi: queste andranno eliminate con un taglierino, ma collegando le parti fra loro ci danno un ulteriore aiuto per la stabilità del pezzo durante la stampa. Con queste impostazioni siamo riusciti a stampare senza problemi, ma con tempi molto lunghi, ben due oggetti completi. Una volta realizzati, vi consigliamo di migliorare gli incavi in cui vanno alloggiati le punte dei perni di rotazione utilizzando la punta di un jack audio da 6 mm (quelli degli strumenti musicali). La punta va riscaldata con un accendino e poi premuta sull'incavo per dargli una forma migliore e più liscia; fate attenzione a non deformare l'incavo perdendo l'allineamento perpendicolare al montante del cubo. Noi abbiamo effettuato l'operazione a più riprese per non rovinare il pezzo; lo scopo di questa lavorazione è avere la rotazione di ciascun cubo senza attriti, ma senza che questo "balli" per l'eccessiva profondità degli incavi. Lavorate quindi l'incavo, provate ad assemblare quel cubo e poi ripetete se c'è ancora troppo attrito.

Nome file: **GyroCube_Up_Orange215C.gcode**



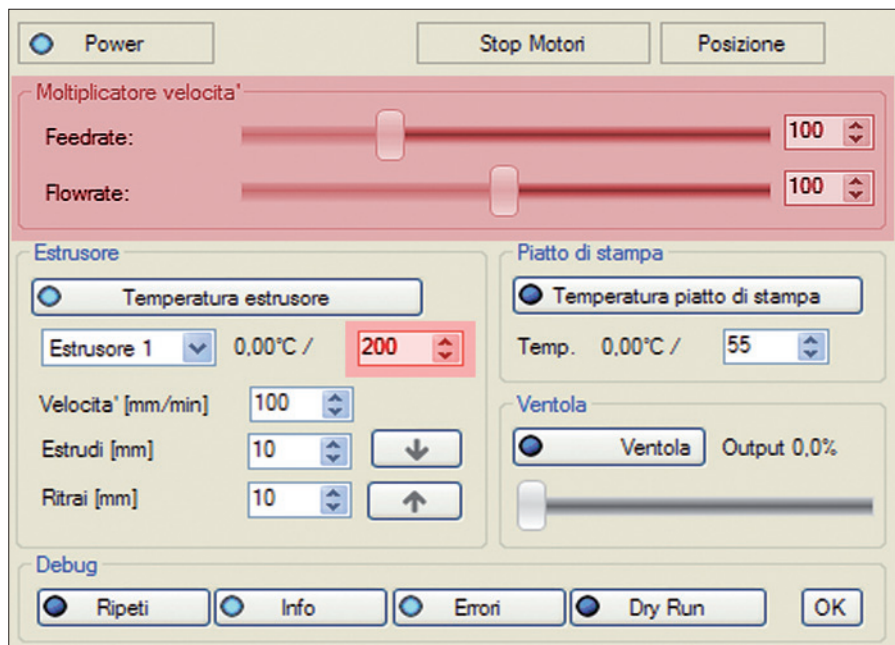
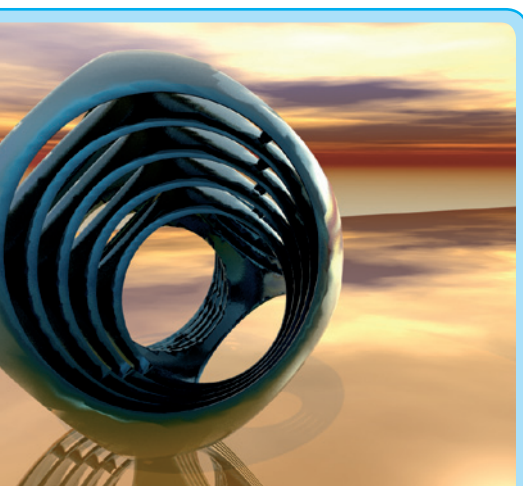


Fig. 3 - Nella scheda Controllo Manuale di Repetier si trovano le regolazioni – evidenziate in rosso – per variare la velocità di stampa, il flusso del materiale estruso e la temperatura di estrusione, tutto durante la stampa.



PARAMETRI DI STAMPA:

layer_height =	0.18
perimeters =	4
top_solid_layers =	4
bottom_solid_layers =	3
fill_density =	0.2
perimeter_speed =	90
infill_speed =	90
travel_speed =	140
scale =	1
nozzle_diameter =	0.5
filament_diameter =	2.8
extrusion_multiplier =	1.3
perimeters extrusion width =	0.80mm
infill extrusion width =	0.80mm
first layer extrusion width =	0.70mm

utilizzando. Trovate uno dei tanti modelli per test in cui viene stampato un singolo perimetro in una struttura verticale (thin wall test o simile) e individuate l'intervallo di temperatura in cui la parete cresce compatta e omogenea alle varie velocità; probabilmente dovrete creare una piccola tabella relazionando velocità a temperatura. La mezza giornata spesa per queste prove vi permetterà però d'intervenire in modo efficace sulle stampe utilizzando nel modo migliore le tre regolazioni (velocità, estrusione e temperatura) che Repetier-Host permette di gestire in tempo reale durante la stampa (Fig. 3).

LA PREPARAZIONE DEL PIATTO DI STAMPA

Le ultime considerazioni che vogliamo fare prima di passare alle stampe riguarda proprio il piatto dove la plastica viene depositata strato dopo strato. 3Drag si caratterizza per una soluzione basata su una basetta di vetronite, trattata con carta vetrata fine per scoprire le fibre di vetro che la compongono. La porosità della basetta permette alla plastica

fusa di trovare un "appiglio" e infatti serve un taglierino per staccare ogni stampa una volta terminato il processo. Se la forma del modello ha delle pareti spesse e una base di appoggio molto estesa, c'è la possibilità che raffreddandosi la plastica cambi le sue dimensioni (si restringa) fino a far staccare gli angoli e i punti più esterni. Questo fenomeno è noto e temuto da tutti coloro che stampano, soprattutto con ABS e meno con PLA. Questo è il motivo per cui si usa il piatto riscaldato che mantiene il materiale a una temperatura in cui è ancora "morbido" e quindi non si deforma ritraendosi.

Il piatto riscaldato però può provocare ustioni se non trattato con attenzione, oltre a consumare altri ampere di alimentazione. Per chi stampa con il PLA l'alternativa è migliorare l'aderenza del piatto di stampa aggiungendo qualcosa che tenga più della sola vetronite resa porosa con la carta vetrata. Al proposito le soluzioni più pratiche ed economiche sono due: la prima prevede l'uso di un velo di colla vinilica diluita (Vinavil), steso con un batuffolo di cotone ogni due o tre stampe, rimosso periodicamente - con un po' di acqua per ammorbidirlo - con il taglierino a lama larga (o un raschietto) quando la superficie non è più uniforme.

La seconda soluzione invece è di recente individuazione e si basa sui fogli adesivi di pellicola per la stampa di etichette Inkjet: queste pellicole da stampa sono con un adesivo permanente applicato a una pellicola molto robusta e indeformabile su cui è steso uno strato di gelatina che serve a fissare l'inchiostro liquido in modo permanente. La superficie della gelatina è satinata e porosa, offrendo un ottimo appiglio al PLA. Oltre a questo, quando si

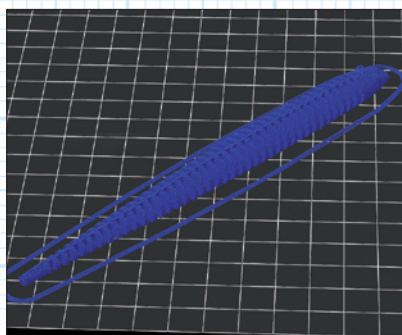
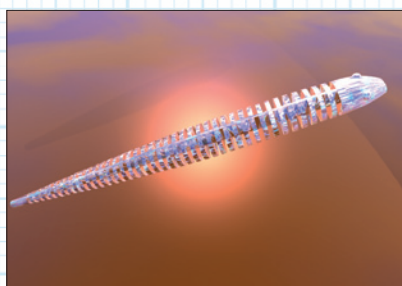
ha sui bordi l'instaurarsi della trazione per la tendenza a incurvarsi dell'oggetto in stampa, il foglio può staccarsi di poco dalla vetronite "accompagnando" la trazione del modello che viene così scaricata e contrastata, senza che l'oggetto si stacchi dal foglio (Fig. 4). Questo foglio adesivo tende a consumarsi con il tempo, lasciando la gelatina attaccata al PLA e perdendo quindi di adesione. Il costo è comunque ridotto e pensiamo che sia giustificato per i risultati che offre.

I FILE STL E G-CODE

Per passare dalle parole ai fatti, in questo "Focus Stampa 3D" abbiamo inserito dieci stampe, ciascuna delle quali mette in risalto una o più caratteristiche della stampa 3D. Si tratta anche di un'occasione di tipo didattico dato che condivideremo con voi degli aspetti tecnici per le impostazioni di ciascuna stampa. Per darvi la possibilità di reinterpretare il nostro lavoro, non forniamo – ovviamente sul sito nell'area dedicata a 3Drag – solo

il file G-Code pronto per la stampa, ma anche il collegamento al file STL; questo permette da un lato di avere con il G-Code la "ragionevole certezza" del risultato, mentre con STL siete anche voi in condizione di reinterpretare la stampa intervenendo sulle misure, sul numero di pezzi e sui parametri di slicing per ciascun modello.

In qualche caso troverete anche le indicazioni per andare sul sito Thingiverse di Makerbot Industries e generare i file con le



Questo è uno dei classici delle stampe 3D dimostrative e nella sua semplicità ha comunque degli elementi caratteristici. È un oggetto flessibile ed elastico, in grado di dimostrare

come sia possibile sfruttare le caratteristiche del materiale di stampa attraverso la scelta degli spessori. Nello specifico, il serpentello viene realizzato con un unico perimetro che crea la struttura sotto forma di costole e dorsale. Osservando i movimenti di stampa, si vede come testa e coda siano di fatto collegati da una specie di molla. Se il perimetro fosse composto da due pareti, la flessibilità sarebbe molto minore perché fino a mezzo millimetro di spessore il PLA è flessibile, a 1 mm diventa invece più rigido. Chi ha progettato il modello ha quindi fatto un ragionamento sul funzionamento del programma di slicing, così da obbligarlo a generare – con i normali parametri di stampa – solo un perimetro esterno e nessun riempimento nelle costole, un perimetro sovrapposto

Serpente flessibile

costole uniti per effetto dello schiacciamento dell'estrusione, ma se regolate opportunamente l'altezza dell'estrusore riuscirete a tenere le due facce delle costole separate ed il serpente potrà non solo essere flessibile a sinistra e a destra, ma anche arcuato sulla schiena, proprio perché le "costole" possono aprirsi nella parte bassa. Come esercizio, potete cercare i parametri corretti per stampare una versione flessibile, ma che occupa l'intera diagonale del nostro piano di stampa, raggiungendo la ragguardevole lunghezza di oltre 25 centimetri.

Nome File: **LongSnake.gcode**

PARAMETRI DI STAMPA:

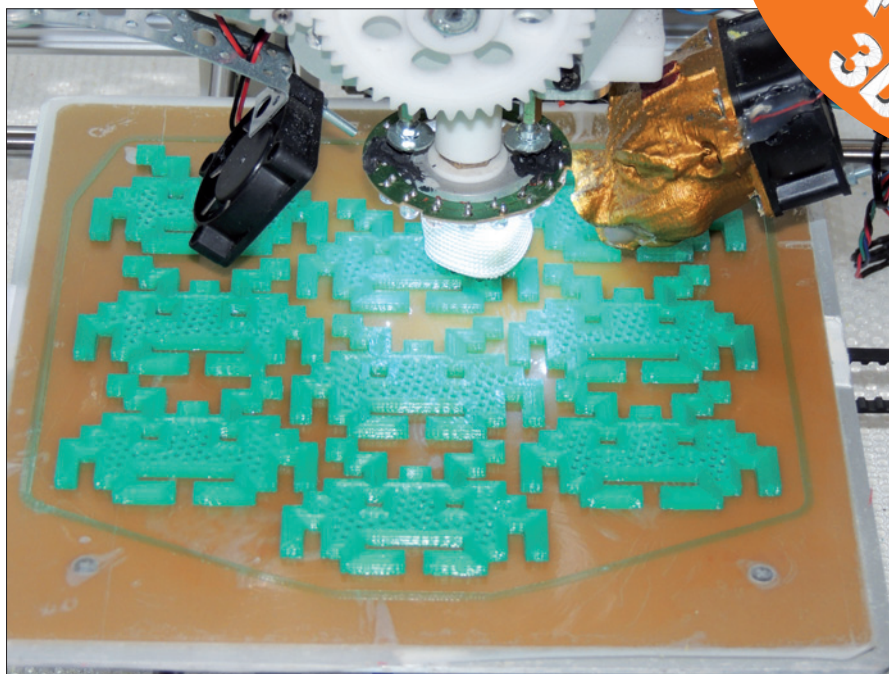
layer_height =	0.3
perimeters =	1
solid_layers =	3
fill_density =	0.3
nozzle_diameter =	0.5
filament_diameter =	3
extrusion_multiplier =	1.1
perimeter_speed =	40
infill_speed =	50
travel_speed =	100
extrusion_width_ratio =	1
scale =	0.8
single wall width =	0.70mm

per la spina dorsale e il corretto riempimento di testa e coda. Se ingrandiamo il modello, questo perde la caratteristica del singolo strato sulle costole e la flessibilità diminuisce. Da notare che nella maggior parte dei casi, il primo strato tiene i due perimetri delle



vostre personalizzazioni. Tutti i nostri G-Code, infine, sono per la stampa in PLA con un range di temperature medio. Intervenite - se necessario - con i comandi manuali per alzare o abbassare la temperatura. Ricordate che c'è sempre la reimpostazione della temperatura alla fine del primo layer.

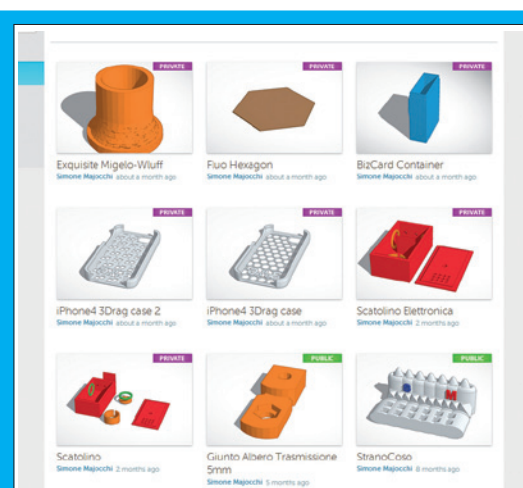
Fig. 4 - Questa stampa è fatta su un piatto in vetronite ricoperto dalla pellicola adesiva per le etichette da stampa su inkjet, codice Avery J8567 e disponibili da Futura Elettronica.



Tinkercad ce l'ha fatta!

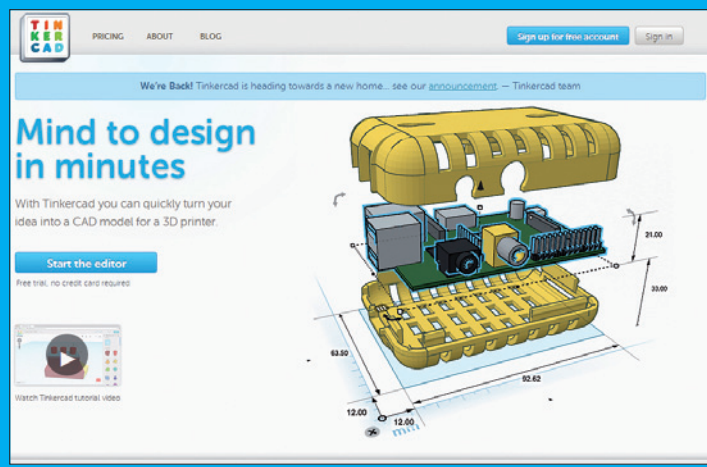
Il 26 marzo di quest'anno, i creatori di Tinkercad annunciavano a tutti i propri utenti registrati la dismissione del progetto che, dopo una fase progressiva di riduzione delle funzionalità e senza supporto, sarebbe stato scollegato dalla rete il 31 giugno 2014. Chi ha usato Tinkercad sa che si tratta di un ottimo prodotto, sviluppato in modo innovativo con un'interfaccia WebGL e basato su codice in grado di scalare ad applica-

zioni ancora più ricche come lo scripting e l'importazione di modelli STL e SVG; eppure in quei tre anni di vita gli utenti che avevano preso confidenza con questo sistema di modellazione sono stati migliaia. Le risorse sono però costose e non avendo trovato un modello di business efficace gli autori hanno pensato di spostare la loro attenzione su un altro progetto, annunciando la dismissione.



Il 19 maggio è invece successo "il miracolo" e Autodesk ha acquisito lo stile e le tecnologie di Tinkercad che - con effetto immediato dall'annuncio - ha ripreso ad accettare utenti e a permettere l'utilizzo senza limitazioni. Si tratta di una grande notizia per tutti coloro che non hanno particolare esperienza nella modellazione 3D e che - per l'estrema semplicità dell'interfaccia - possono comunque ottenere dei risultati impeccabili. Noi stessi ricorriamo a Tinkercad per molte realizzazioni di varia complessità e quindi eravamo

stati colpiti "nel vivo" dalla notizia. Ora vogliamo augurare ad Autodesk di riuscire a mantenere lo spirito innovativo e "semplice" di questo splendido servizio che ha permesso a tanti di entrare nel mondo 3D con tanta naturalezza e concretezza. Se ancora non lo conoscete ancora, basta un browser che supporta WebGL e digitare "tinkercad.com"; iniziate con le lezioni del tutorial e poi non ne farete più a meno per tutti quei progetti "da 5 minuti", ma anche per quelli molto più complessi.



MODIFICARE IL G-CODE

Dato che questo formato è testuale, è possibile intervenire sul file per modificare il suo contenuto.

Un classico intervento riguarda

l'impostazione della temperatura di stampa del primo strato e dei successivi.

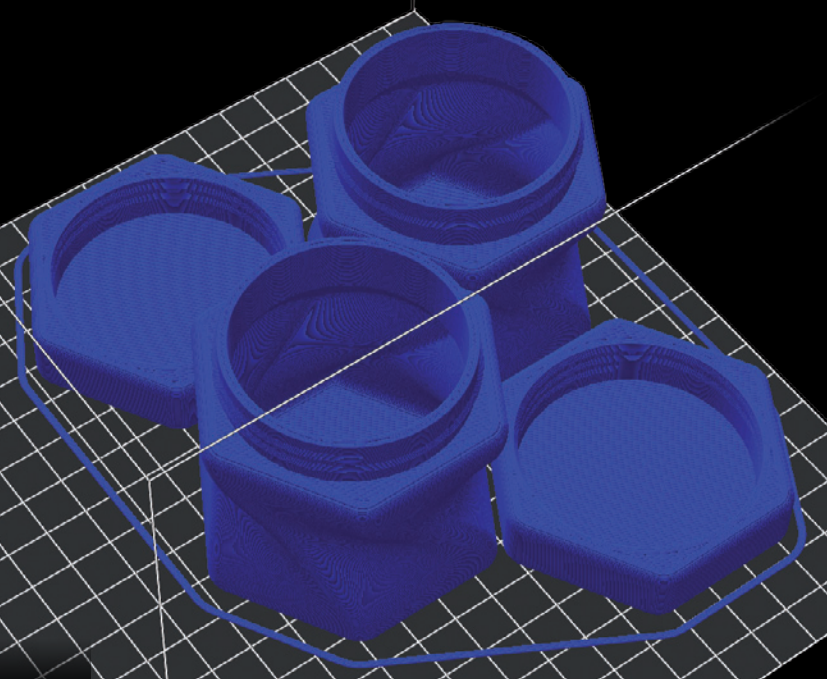
Apprendo con un editor di testo il file, troverete fra le prime righe queste istruzioni:

```
M104 S200 ; set temperature
G28 ; home all axes
M109 S200 ; wait for temperature to be reached
```

Mentre alla fine del primo strato troverete il cambio della temperatura a quella scelta per gli strati successivi:

```
M104 S192 ; set temperature
G1 F1800.000 E437.06446
G92 E0
G1 Z0.480 F7200.000
```

Da notare che a tre righe di distanza si trova l'istruzione per il movimento sull'asse Z che – nell'esempio – ci dice che l'estrusore si sposta a 0,48 millimetri dal piano di stampa. L'istruzione

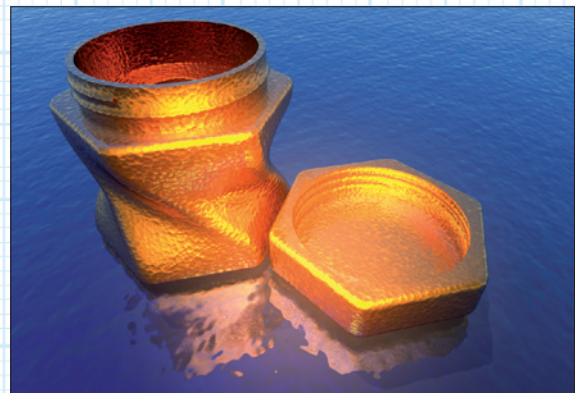


Twisted cap bottle multi size

Il modello originale fa parte della storia della stampa 3D in quanto è stato – ed è tutt'ora – il campione che Stratasys invia a chi ne fa richiesta per dimostrare la qualità delle proprie stampanti. Il modello è stato rilasciato nel pubblico dominio da tempo e quindi le versioni e reinterpretazioni sono facilmente reperibili. Noi abbiamo scelto l'originale e ne abbiamo solo modificato la scala

per renderlo più adatto alla creazione di un oggetto robusto e maneggiabile senza problemi. Quando avrete terminato la stampa, dovrete aver cura di ripulire un po' il filetto del tappo, per eliminare qualsiasi sbavatura o "pallino" di plastica. Il passo successivo è avvitare e svitare il tappo con rotazioni via via più complete per aiutare i due filetti a lavorare nel modo corretto; non

forzate troppo o bloccherete il tappo. Se il tappo fa molta fatica ad avvitarsi, vuol dire che avete stampato troppo "abbondante" e quindi le parti sono dimensionalmente più grandi del dovuto, mentre se il tappo "balla", la stampa è stata povera di plastica. In entrambi i casi avete il parametro relativo al numero di step per estrarre un millimetro di filo che è errato. Lo trovate in Repetier alla voce Configurazione -> Configurazione Firmware EEPROM. La finestra che si apre ha nella prima riga una serie di caselle e l'ultima è quella che determina quanto deve girare la ruota dell'estrusore per caricare nel sistema un millimetro di plastica. Se il vostro filo di PLA o ABS non ha il diametro previsto di 3mm, avrete stampe più "magre" o "grasse" e questo



PARAMETRI DI STAMPA:

layer_height =	0.18	travel_speed =	140
perimeters =	4	scale =	1
top_solid_layers =	4	nozzle_diameter =	0.5
bottom_solid_layers =	3	filament_diameter =	3
fill_density =	0.2	extrusion_multiplier =	1
perimeter_speed =	90	perimeters extrusion width =	0.35mm
infill_speed =	90	infill extrusion width =	0.35mm
		first layer extrusion width =	0.35mm

Dove scaricare tutti i file?

I file dei modelli presentati nei box "IDEA" sono presenti sul nostro sito dedicato alla stampa 3D: **3dprint.elettronica.in.it**.

Andate nell'area "Ready to Print" e scegliete il collegamento "Focus Stampa 3D" per accedere alla pagina dedicata dove troverete tutti i link ai file STL e G-CODE. I file sono compressi per ridurre il tempo di trasferimento; ricordate di espanderli prima di aprirli con Repetier o Slic3r per le lavorazioni successive.

che dovete cercare è "M104" che è seguita dal parametro "S<temperatura>".

Dopo le modifiche salvate il vostro file rinominandolo opportunamente.

Per fare un esempio dell'utilità di questo intervento, potete cambiare tutti i file da noi forniti per essere utilizzati con ABS. Basta intervenire sulla temperatura del primo strato cambiando gli M104 e M109 iniziali ai 250/265 gradi (usate i valori che sapete funzionare al meglio con i vostri ABS) e quindi cercate il M104 a fine del primo strato per impostare la temperatura di stampa per tutti gli altri strati. Ricordate infine che il valore di Z è assoluto e quindi basta fare due conti

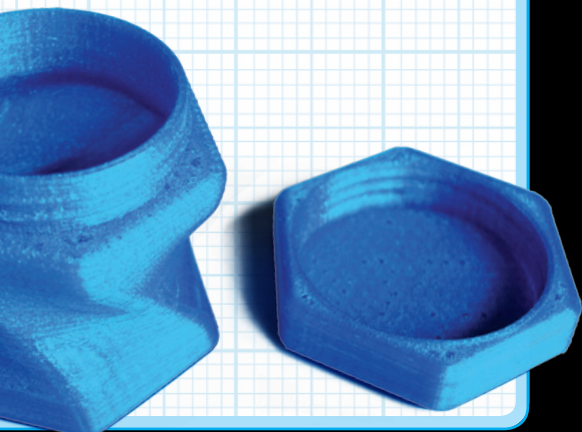
per sapere a quale valore di Z corrisponde un qualsiasi layer di stampa.

Con questa informazione potete ad esempio interrompere la stampa in un layer preciso per consentire un cambio di colore, oppure potete correggere dei problemi che capitano su uno specifico strato. Tutte le istruzioni G-Code utilizzabili con 3Drag/Marlin sono documentate sia sul sito di Marlin (<https://github.com/ErikZalm/Marlin>), sia sul wiki di RepRap. Repetier, nel G-Code Editor offre anche una finestra di aiuto che spiega i comandi standard.

parametro vi permette di compensare il problema. Ci sono anche altri modi per risolvere il problema, ma questo è il più rapido, anche a slicing effettuato. Diversamente potete agire sul profilo del Filament Setting di Slic3r e rendere la modifica specifica per ciascun tipo di PLA o ABS utilizzato. Vi invitiamo a stampare questo modello in varie scale per apprezzarne le caratteristiche. Noi siamo arrivati al 3x. Il file G-Code fornito stampa due set a dimensione doppia rispetto al modello STL: siamo certi che uno dei due vi sarà immediatamente sequestrato da un vostro amico e quindi tanto vale stamparne subito due.

Nome File:

Twisted_Bottle_mm_2_x2_3Drag0K.gcode



Stampa grassa o stampa magra?

Uno dei "grandi problemi" durante la stampa di un modello è la variazione nel diametro del filo e la differenza fra un colore e l'altro o da una partita e l'altra di materiale plastico. Per quanto ci si adoperi per trovare le impostazioni migliori, il numero di microstep che il motore deve effettuare per spingere nell'estrusore un millimetro di filo è conseguenza di parecchi parametri fisici - dall'effettivo diametro del bullone zigrinato al diametro del filo a quanto il filo viene inciso dai denti e altro ancora - che possono variare l'effettiva quantità di materiale che viene trasformata nel nostro oggetto. In pratica, ci sono stampe che con il medesimo file in G-Code risultano più "magre" ed altre che risultano più "grasse". Una superficie che dovrebbe essere piena risulta a volte con i singoli fili ben visibili, altre volte risulta ruvida per l'eccesso di materiale. La soluzione arriva dall'autore di Repetier-host che, sfruttando una caratteristica del firmware Marlin, ha aggiunto nella sezione dei controlli di velocità in tempo reale il cursore "FLOWRATE" sotto quello di "Feedrate". Con Flowrate viene inviata l'istruzione G-Code M221 che permette di definire un fattore di moltiplicazione (maggiore o minore di 1) per aumentare o diminuire la quantità di materiale realmente estruso rispetto a quanto previsto dal G-Code e, in precedenza, definito dai parametri di slicing. Grazie a questa nuova funzionalità potrete quindi compensare in tempo reale le stampe che a vista sono troppo scarse o troppo ricche di plastica a causa delle diverse caratteristiche del filo di stampa utilizzato.



Stampa GRASSA



Stampa OTTIMALE



Stampa MAGRA

Controlla i tuoi dispositivi da remoto tramite LAN e Internet.

cod. VM201

€ 189,00

SCHEDA ETHERNET CON 8 RELÉ E DISPLAY

- * È in grado di attivare o disattivare fino a 8 relè e verificare lo stato dell'ingresso;
- * Inviare rapporti tramite email;
- * Gestire uscite temporizzate mediante programmi giornalieri;
- * App gratuita per dispositivi Apple;
- * Fornita già montata e collaudata.

cod. ETH-RLY02
€ 68,00

SCHEDA ETHERNET CON 2 RELÉ 16A

- * 2 uscite a relè N.O. e N.C. con LED di stato;
- * Alimentazione 12Vdc;
- * Dimensioni 85 x 66 x 20mm;
- * Disponibile applicazione gratuita per Android;
- * Fornita già montata e collaudata.

cod. ETH-RLY16
€ 86,50

SCHEDA ETHERNET CON 8 RELÉ 16A

- * 8 uscite a relè N.O. e N.C. con LED di stato;
- * Alimentazione 12Vdc;
- * Dimensioni 143 x 85 x 20mm;
- * Fornita già montata e collaudata.



Schermata dell'App per VM201

La parola all'architetto
che ha rivoluzionato
il software di slicing
rendendolo migliore, più
veloce e user friendly.

A photograph of Alessandro Ranellucci, a man with dark hair and a beard, wearing a dark shirt and tie. He is standing in a workshop filled with various 3D printing materials, including spools of filament in various colors (yellow, orange, red, white, blue) and a 3D printer. He is holding a small white 3D printed object in his right hand. The background shows a cluttered workspace with tools and materials.

IL 3D DI ALESSANDRO RANELLUCCI

Parlano di modelli da stampare e di come sia oggi semplice e veloce raggiungere ottimi risultati, ci è sembrato importante completare l'articolo con un'intervista ad Alessandro Ranellucci, l'autore di Slic3r e quindi colui che permette a tutti noi - con il suo software Open Source - di raggiungere ogni giorno nuove mete nella stampa 3D. Alessandro è anche uno dei quattro fondatori di Unterwelt a Roma: un maker space dove periodicamente

i fondatori si incontrano per discutere e sviluppare nuove idee. Unterwelt nel tempo e con il passa parola ha però iniziato a catalizzare le attenzioni nell'ambiente makers ed è ora diventato un punto di aggregazione - a piccole dosi e compatibilmente con gli spazi - per il mondo dei maker romani. Poi, come ci ha detto Alessandro, "alla nostra porta bussano persone che hanno idee e progetti che vogliono condividere e spesso ci troviamo ad avere ospiti con cui passiamo



del tempo che avevamo invece programmato per portare avanti i nostri lavori.”

Come e perché hai iniziato a sviluppare Slic3r?

Da utente RepRap di un paio d'anni fa - un'epoca in cui si iniziavano a vedere stampe promettenti, ma la tecnologia era ancora farraginosa - soffrivo per la complessità della calibrazione che richiedeva un'alchimia di parametri da trovare sperimentalmente e da non toccare più. Il software era il collo di bottiglia per lo sviluppo di queste macchine e un grande ostacolo per la loro diffusione. Il 90% della community RepRap è composto da persone con competenze meccaniche, mentre il software è portato avanti da pochi utenti. Già dopo poche settimane si era creato molto interesse intorno a Slic3r per via della sua velocità di elaborazione. Arrivare alla stampa con spessore degli strati a 10 micron è stato possibile perché con Slic3r si impiegavano alcuni minuti anziché alcune ore. Skeinforge era arrivato ad un punto morto a causa della complessità del codice che rendeva pressoché

impossibile la sua evoluzione. L'unica strada era ricominciare da capo riscrivendo tutti gli algoritmi in modo pulito e ordinato.

Dov'è il cuore dello sviluppo 3D open?

L'Europa è stata la culla del progetto RepRap, che com'è noto nasce in Inghilterra, e tuttora ha in Olanda e Germania alcuni tra i suoi principali esponenti. Per non citare ovviamente Praga, dove opera Josef Prusa. Ma anche gli Stati Uniti ormai contribuiscono ampiamente, e anzi il primo festival RepRap si è recentemente tenuto proprio nel mid-west americano.

Come ha cambiato la tua vita essere l'autore di Slic3r?

La mia attività lavorativa è un'altra, anzi due: ho un piede nell'architettura e un piede nello sviluppo software. Il progetto Slic3r è nato con lo spirito Open Source e perché serviva a me per primo: cerco di usare tutto quello che entra - tramite le donazioni e le sponsorizzazioni - per il miglioramento del software e della documentazione. Gestire la community con le sue pro-

poste, il feedback e gli stimoli che offre richiede molto del mio tempo, che si aggiunge a quello necessario per lo sviluppo vero e proprio. Con la scelta dell'Open Source ho attivato dei meccanismi che solo la modalità Open offre: una vastissima rete di utenti che forniscono testing e feedback quasi in tempo reale. I contributi spontanei in termini di programmazione sono più scarsi vista la complessità del software.



PARAMETRI DI STAMPA:

layer_height =	0.18
perimeters =	4
top_solid_layers =	4
bottom_solid_layers =	3
fill_density =	0.2
perimeter_speed =	90
infill_speed =	90
travel_speed =	140
nozzle_diameter =	0.5
filament_diameter =	3
extrusion_multiplier =	1
perimeters extrusion width =	0.35mm
infill extrusion width =	0.35mm
solid infill extrusion width =	0.35mm
top infill extrusion width =	0.35mm
first layer extrusion width =	0.35mm

Da novembre un programmatore californiano mi affianca nello sviluppo con uno stipendio che posso offrirgli grazie ai fondi raccolti. Sempre grazie agli sponsor un professionista sta realizzando la documentazione e un manuale tecnico.

Cosa pensi della stampa 3D e di RepRap?

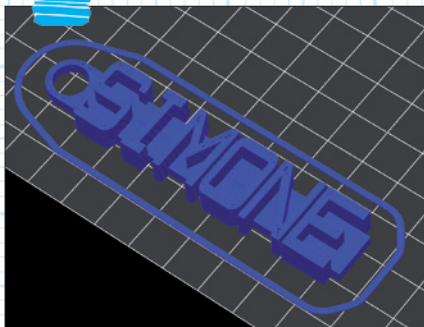
Ha portato e ha diffuso tante cose fra cui l'Open Hardware. RepRap

è una meta macchina che porta in sé molta elettronica, molto software e molta meccanica: questo le conferisce un forte valore educativo. Il progetto RepRap è oggi intenzionalmente privo di una linea unitaria e infatti più che un progetto è diventato un ecosistema, una community all'interno della quale si muovono tante

linee di ricerca individuali. Dal punto di vista della percezione da parte della gente comune sta finendo il sensazionalismo e si comincia ad assistere alle prime applicazioni utili in tanti settori: penso ai progetti nel settore educativo, o nel settore medico e dentistico soprattutto nei paesi in via di sviluppo dove l'accessi-



Portachiavi con nome



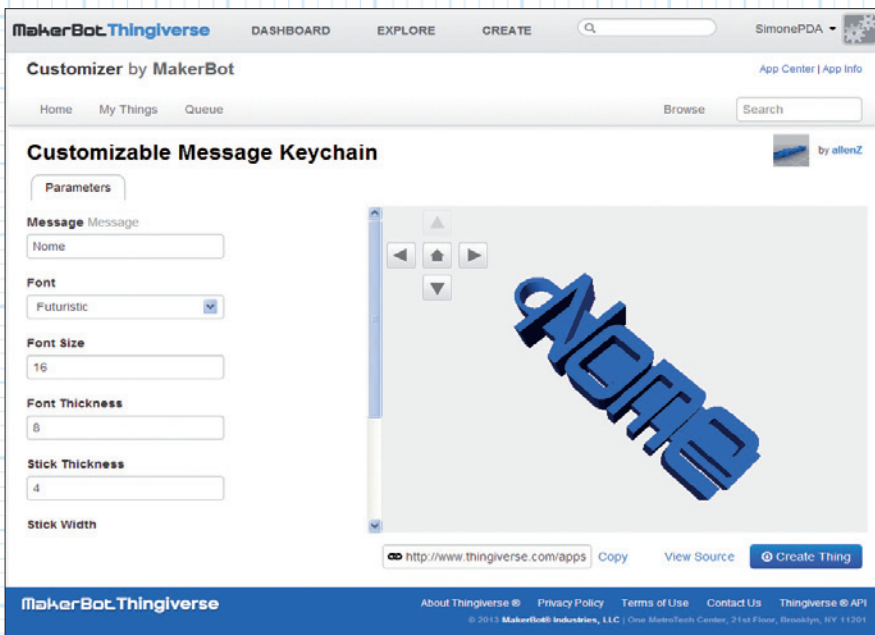
I portachiavi stampati in 3D sono un gadget di sicuro effetto, ma realizzare ogni volta un modello con un CAD per creare il nome del nostro amico o della nostra amica può essere noioso. Per questo un utente di Thingiverse ha realizzato un'applicazione (customizer, come le chiama il sito) in cui si possono immettere i parametri e il modello STL viene generato di conseguenza, pronto per la



stampa. Dovete partire dall'oggetto (<http://www.thingiverse.com/thing:52734>) e poi scegliere "Open in Customizer". Vi consigliamo di scegliere il parametro Flat Bottom per avere la barretta che sostiene le lettere a contatto con il piano di stampa, mentre se avete dei nomi con delle i, dovete decidere quale strategia adottare per evitare che il puntino non sia connesso al portachiavi (allargate la striscia di base oppure usate la "i" maiuscola). Al termine, con il pulsante "create thing" avvierete la generazione che porterà ad avere il vostro portachiavi negli oggetti del vostro profilo.

Scaricate e stampate ricordando che le lettere hanno bisogno di una certa definizione e quindi anche in questo caso è meglio scendere con il diametro del filo estruso. Il file che vi diamo già pronto è con il nome Simone e i parametri sono impostati per la stampa con PLA fosforescente, che richiede una temperatura più bassa del normale. Se volete stamparlo, modificate la temperatura come spiegato in precedenza.

Nome file:
Keychain_Simone_Glow192.gcode



bilità dei costi è un fattore chiave. Per fare un esempio, stampare in economia un'arcata dentale a partire da una risonanza magnetica consente ai dentisti di studiare l'intervento in anticipo e di addestrare i loro assistenti. Poi ovviamente le decine di migliaia di stampanti RepRap diffuse nel mondo vengono usate in modo futile per stampare fischietti, vasi, pupazzi, giocattoli. Ma non dimentichiamoci che tutto questo serve a migliorare la tecnologia e ad abbassarne i costi, e quindi a renderla disponibile per gli usi più "nobili" cui accennavo prima.

Cosa prevedi per i prossimi 2 anni? Quali evoluzioni nella fascia bassa per i materiali e le tecniche?

Il futuro che prevedo si basa anche sulla stereolitografia e la stampa a polveri, sempre Open Hardware

e a costi abbordabili; e non dimentichiamo che la digital fabrication non ha bisogno solo di stampanti 3D, ma anche di versioni open di tutte le altre tecniche di fabbricazione (fresatura, piegatura, taglio laser, tessitura e così via).

Open o Closed?

Se possibile, open. Non tutti i produttori cedono alle tentazioni di chiudere le loro macchine. La piccola economia nata intorno alla stampa 3D dimostra che anche chi sceglie convintamente la strada dell'open hardware può trovare il suo modello di business. Open vuol dire incentivare costi accessibili e tecnologie diffuse, che verranno raffinate dalle tante persone che vorranno dedicarci tempo. Per molti l'apprendimento e la sperimentazione sono un obiettivo ancor più importante rispetto all'uso effettivo della

tecnologia e questo è un valore prezioso da incentivare. Chi vuole approfittare dell'assemblaggio per imparare cose nuove e conoscere tutti i dettagli di funzionamento può trovare i progetti e gli schemi elettronici e meccanici in rete e procurarsi i pezzi, magari recuperandoli da vecchio hardware, ed entrare in contatto con la community. Chi invece desidera una macchina già assemblata può senz'altro comprarla da un'azienda, pagando un prezzo più alto composto dai costi dei componenti più l'assemblaggio e l'assistenza. Io tendo a supportare quelle imprese che adottano licenze open per restituire qualcosa alla community in cambio di quello che hanno preso. Il futuro della tecnologia dipende da come noi, utenti, e in fondo anche acquirenti, decideremo di influenzarla.

Perché stampare cibo?

Quando tempo fa si era diffusa la notizia di una Makerbot modificata per stampare la salsa sui tacos, probabilmente nessuno aveva provato ad immaginare cosa sarebbe potuto accadere dopo questo primo tentativo.

Beh, il tema è molto più serio di quanto si possa immaginare e addirittura la NASA sta finanziando la ricerca per arrivare a poter mettere le 3D food printers nelle prossime missioni spaziali. Ma cosa ci si deve aspettare da questa particolare branca della stampa 3D?

Innanzitutto le stampanti in via di progettazione sono "multi materiale", ovvero dispongono di serbatoi contenenti materie prime opportunamente ingegnerizzate per dare ciascuno un apporto specifico fra quelli previsti. Ci sarà quindi un componente fatto da sole proteine, uno da

carboidrati, uno da grassi, poi integratori per Omega3, calcio e altro ancora. La provenienza di queste materie prime sarà probabilmente la più varia, dalle alghe ad alcuni vegetali per arrivare agli insetti. Fin'ora queste "materie prime" non erano considerate appetibili per aspetto e gusto, ma grazie alla possibilità di combinare e stampare i materiali in forme interessanti, con texture e dimensioni gestibili via software, diventa plausibile che si possa mangiare qualcosa che deriva dai componenti appena citati - quindi poco attraenti - ma che ha un aspetto invitante e un gusto piacevole proprio perché stampato nel modo opportuno sia per aspetto, sia per gusto.

Quello che risulta ancora più interessante è la possibilità di personalizzare il "mix" in base alle esigenze di ciascuno, combinando le quantità di proteine, grassi e carboidrati (più gli integratori) su misura per ciascuno. Questo, soprattutto nello spazio, è un grosso vantaggio ed è il motivo per cui la NASA ha deciso di investirci. Si avvicina quindi il sintetizzatore di cibi che fa bella mostra di sé sin dai primi episodi di Star Trek? Che dire... probabilmente sì, anche se le modalità per fare il cibo non sono quelle immaginate nei telefilm.

Con la stampa dei cibi in 3D, basati su formulazioni personalizzate e componenti scelti per il loro contenuto è non per il loro aspetto, vedremo quindi una nuova "rivoluzione gastronomica" e - con queste premesse - ogni sviluppo del settore è plausibile.





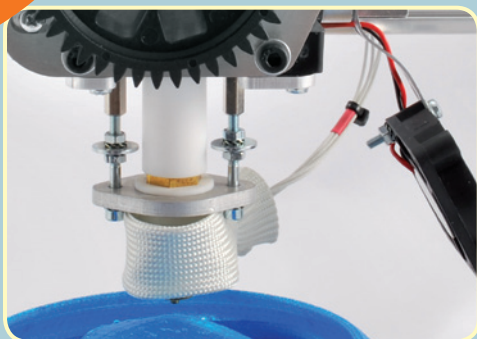
3DRAG

COMPIE 1 ANNO E SI RINNOVA

Sono passati 12 mesi dall'entrata in funzione della nostra prima 3Drag e ora il progetto approda alla versione 1.1 con alcuni significativi miglioramenti.

Due numeri fa abbiamo presentato la scheda basata su ATmega 2560 e quindi più simile a una RAMPS che all'iniziale Sanguinololu. Ora quella scheda è diventata lo standard montato sulle nostre stampanti in produzione, permettendo maggiore affidabilità e anche nuove funzioni quali il controllo

della ventola tramite Mosfet e PWM, oltre che il collegamento di un sistema di stampa autonoma come quello descritto in questo numero. Un altro significativo miglioramento della nostra 3Drag è il rinnovato blocco estrusore: al posto del cilindro in ottone monoblocco, collegato al distanziale cilindrico in



1.

Nuovo Estrusore

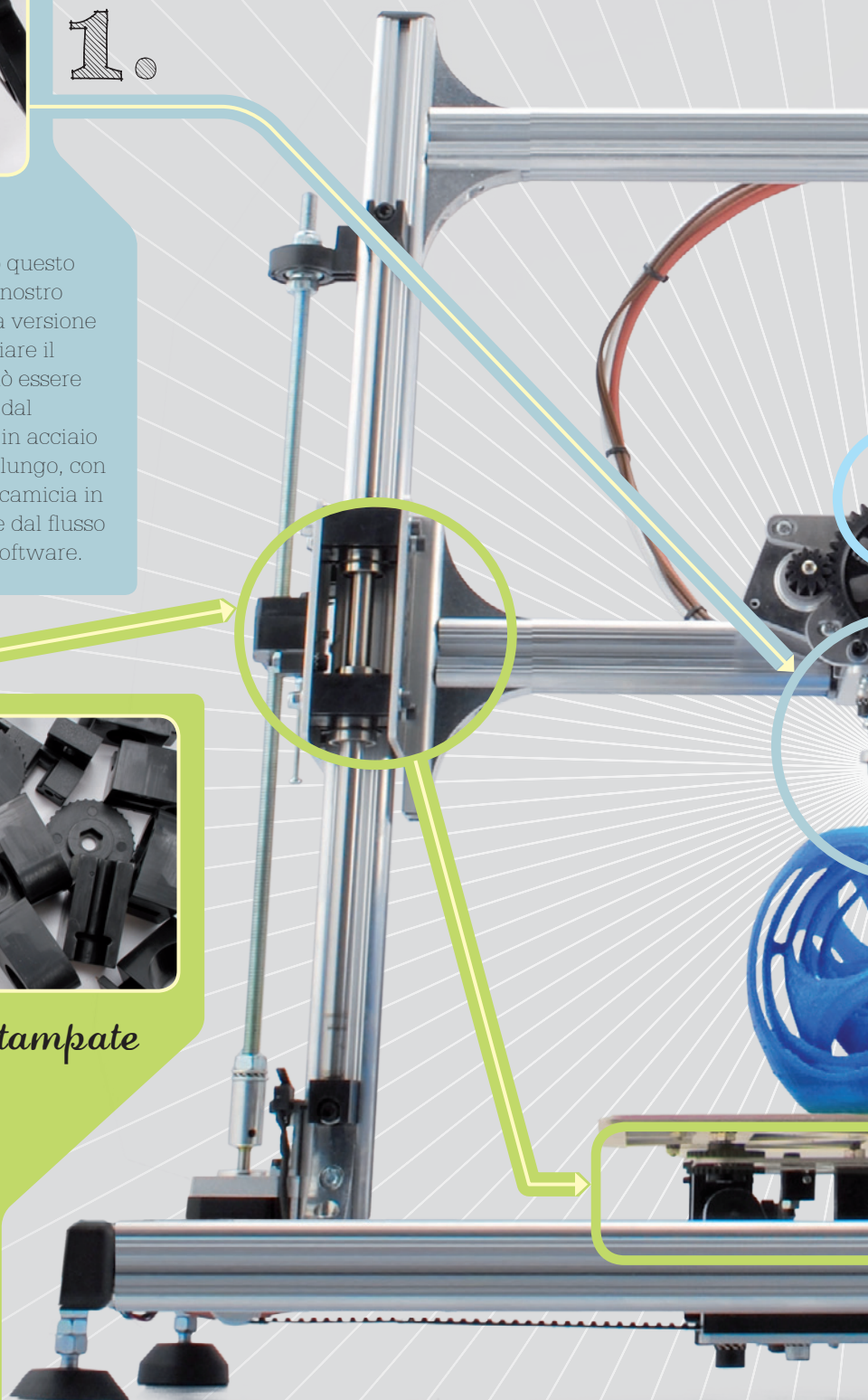
Dopo svariate prove, abbiamo realizzato questo nuovo estrusore risolvendo quelli che, a nostro avviso, erano i limiti più importanti della versione precedente. Ora si può facilmente cambiare il diametro dell'ugello dato che la testa può essere rimossa indipendentemente dal corpo e dal blocchetto. L'elemento riscaldante è ora in acciaio ed ha il corretto wattaggio per durare a lungo, con il giusto "spunto" nel riscaldamento. La camicia in fibra di vetro protegge il blocco in ottone dal flusso d'aria della ventola, ora controllata via software.

2.



Parti in plastica stampate

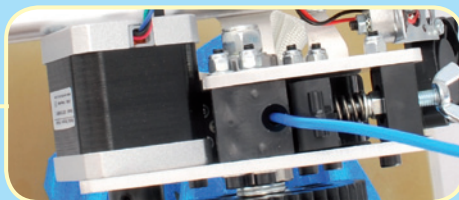
Al posto degli elementi in plastica lavorati al pantografo, ora vengono forniti degli elementi in plastica stampata a iniezione. Le caratteristiche meccaniche sono tali da permetterci una maggiore uniformità nella produzione, garantendo allo stesso tempo facili sostituzioni e montaggio semplificato. Tutto questo senza rinunciare alla precisione e alle soluzioni realizzate con la soluzione dei pezzi "fatti a mano".



Sistema di cambio filo semplificato

Cambiare filo di stampa richiedeva un certo sforzo perché non era facile allentare la pressione del cuscinetto sul filo; ora c'è una leva che permette di spostare facilmente indietro il blocco del cuscinetto ed ottenere quindi una rapida estrazione del filo senza agire sulla rotazione del bullone zigrinato. Un piccolo dettaglio per un significativo miglioramento funzionale.

3.



4.



Scheda ATmega 2560

Oltre a montare il chip ATmega2560, - quello di Arduino Mega - la nuova board 1.1 dispone del Mosfet di potenza per il controllo della ventola tramite Soft PWM, così come gestito dalle attuali versioni di Slic3r e Repetier-Host. La scheda, inoltre, è stata dimensionata per gestire correttamente anche il piatto riscaldato con il suo sensore di temperatura (disponibili come opzione da Futura Elettronica). La nostra 3Drag v 1.1 è supportata da Marlin con il codice "77".

Tabella 1 - Modifiche Firmware Marlin per 3Drag V1.1

File Configuration.h		
Riga	Originale	Modificare in
55	#define MOTHERBOARD 7	#define MOTHERBOARD 77
75	#define DELTA	//#define DELTA
134	#define TEMP_SENSOR_0 -1	#define TEMP_SENSOR_0 5
185	#define DEFAULT_Kp 22.2	//#define DEFAULT_Kp 22.2
186	#define DEFAULT_Ki 1.08	//#define DEFAULT_Ki 1.08
187	#define DEFAULT_Kd 114	//#define DEFAULT_Kd 114
195	// #define DEFAULT_Kp 63.0	#define DEFAULT_Kp 63.0
196	// #define DEFAULT_Ki 2.25	#define DEFAULT_Ki 2.25
197	// #define DEFAULT_Kd 440	#define DEFAULT_Kd 440
268	#define ENDSTOPPULLUP_XMAX	//#define ENDSTOPPULLUP_XMAX
269	#define ENDSTOPPULLUP_YMAX	//#define ENDSTOPPULLUP_YMAX
270	#define ENDSTOPPULLUP_ZMAX	//#define ENDSTOPPULLUP_ZMAX
277	const bool X_ENDSTOPS_INVERTING = true;	const bool X_ENDSTOPS_INVERTING = false;
278	const bool Y_ENDSTOPS_INVERTING = true;	const bool Y_ENDSTOPS_INVERTING = false;
279	const bool Z_ENDSTOPS_INVERTING = true;	const bool Z_ENDSTOPS_INVERTING = false;
296	#define DISABLE_Z false	#define DISABLE_Z true
299	#define INVERT_X_DIR true	#define INVERT_X_DIR false
301	#define INVERT_Z_DIR true	#define INVERT_Z_DIR false
302	#define INVERT_E0_DIR false	#define INVERT_E0_DIR true
315	#define X_MAX_POS 205	#define X_MAX_POS 200
317	#define Y_MAX_POS 205	#define Y_MAX_POS 200
319	#define Z_MAX_POS 200	#define Z_MAX_POS 220
343	#define DEFAULT_AXIS_STEPS_PER_UNIT {78.7402,78.7402,200.0*8/3,760*1.1}	#define DEFAULT_AXIS_STEPS_PER_UNIT {64.25,64.25,2560,600}
371	//#define EEPROM_SETTINGS	#define EEPROM_SETTINGS
374	//#define EEPROM_CHITCHAT	#define EEPROM_CHITCHAT
		SOLO PER ATTIVARE IL PIATTO RISCALDATO
137	#define TEMP_SENSOR_BED 0	#define TEMP_SENSOR_BED 5
		SOLO PER ATTIVARE LA STAMPA AUTONOMA
391	//#define ULTIMAKERCONTROLLER	#define ULTIMAKERCONTROLLER

Per avere i messaggi su LCD e monitor seriale in italiano, fate la seguente modifica. Noi abbiamo preferito non attivare la traduzione per mantenere coerenza fra il programma di slicing, alcune voci di Repetier-Host e le indicazioni sul display.

File language.h		
21	#define LANGUAGE_CHOICE 1	#define LANGUAGE_CHOICE 7

teflon, abbiamo riprogettato il sistema per consentire due cose: l'eventuale cambio nelle dimensioni del foro di estrusione e una maggiore affidabilità del riscaldatore.

Come avrete notato nella pagina precedente con i dettagli delle modifiche, abbiamo anche abbandonato il silicone per alte temperature e abbiamo aggiunto una camicia in fibra di vetro per proteggere il blocchetto di ottone dall'aria soffiata dalla ventola, ora controllata via software.

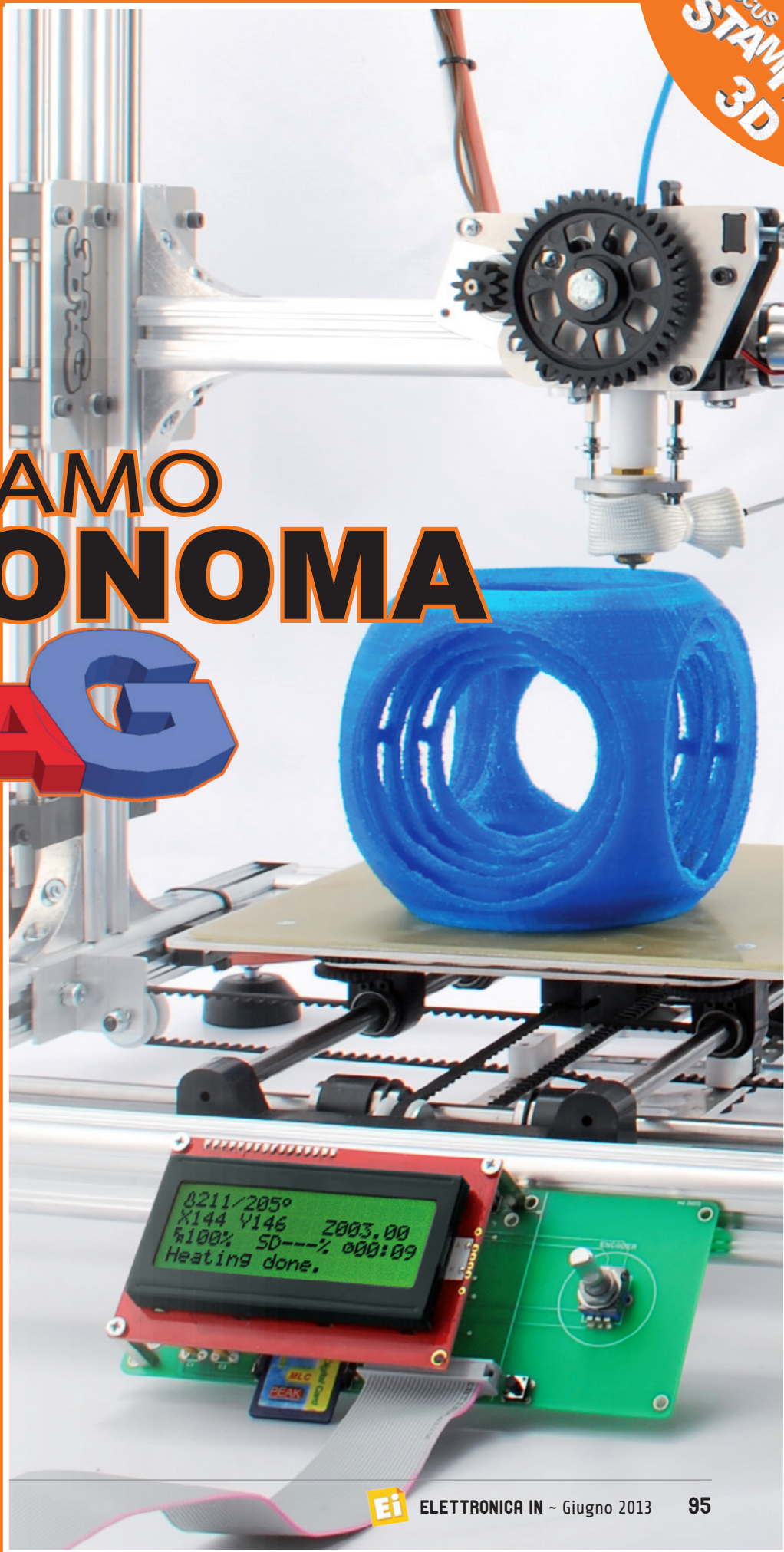
Infine 3Drag V 1.1 può contare su un set di parti, stampate a iniezione in materiale plastico, opportunamente progettato e che sostituisce tutti gli elementi precedentemente realizzati singolarmente con sistemi a controllo numerico.

Con queste modifiche, 3Drag è ancora più affidabile nel funzionamento e semplice da montare, restando aperta a tutte le modifiche software e hardware che vorrete applicare. La scheda controller è stata aggiunta a quelle supportate direttamente dal firmware Marlin ed è connotata come “// 77 = 3Drag Controller”; riportiamo le modifiche da apportare al firmware - in termini di impostazioni delle parti meccaniche ed elettromeccaniche - dato che in ogni caso queste vanno inserite manualmente in ogni nuova release. Le righe indicate sono relative alla versione del 10/06/2013 e con molta probabilità il loro numero (quello che si legge in basso a sinistra nella finestra dell'IDE di Arduino) cambierà via via che l'autore modifica e implementa funzionalità nuove al suo firmware. Sono comunque in sequenza e se il numero riga non corrisponde esattamente, cercate nelle vicinanze.

Un accessorio tanto semplice quanto indispensabile per poter controllare 3Drag e stampare senza un computer collegato grazie a un cursore, a uno slot per schede SD e a un display LCD.

RENDIAMO AUTONOMA 3DRAG

Una delle caratteristiche delle stampanti 3D è la relativa lentezza nella stampa: per quanto la meccanica di 3Drag sia in grado di raggiungere e superare la velocità 100 millimetri al secondo per la stampa, un oggetto richiede comunque decine di minuti se non ore per essere portato a termine. Basta un fill un po' impegnativo o un pezzo massiccio per entrare nell'ambito delle ore, mentre chi è ormai abituato a stampare più parti in un sol colpo, organizzandole



sul piatto di stampa, sa che può essere necessaria la notte per stampare tutto.

In tutto questo tempo è necessario inviare alla stampante le istruzioni in G-Code per controllarne i vari movimenti fino al termine della costruzione del modello in stampa. In pratica un notebook, un PC o un altro computer deve restare acceso e collegato, con il programma di stampa in funzio-

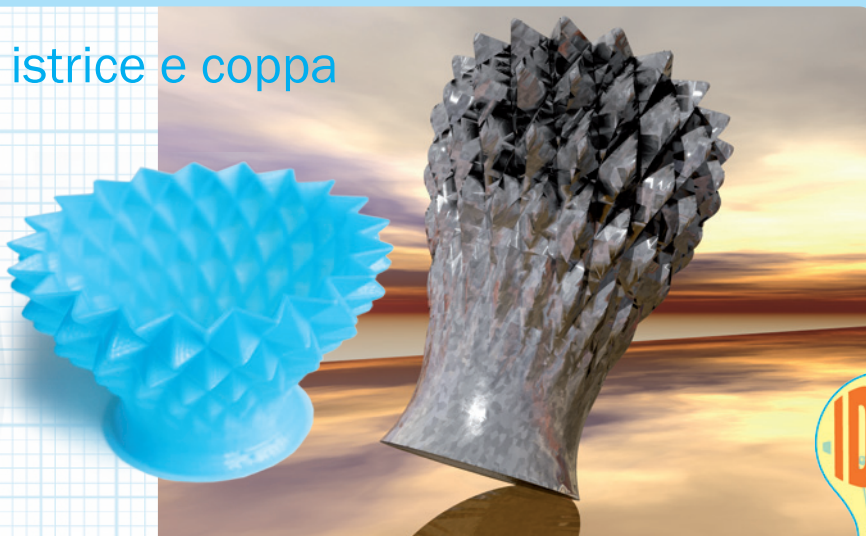
ne. E se uno ha a disposizione solo un notebook e vuole poterlo spostare in un'altra stanza per continuare a lavorare? Oppure deve uscire di casa, sempre con il notebook? Con la stampante in stampa non si può far altro che aspettare il termine del processo, oppure si può rinunciare alla stampa in corso.

La soluzione al problema è l'aggiunta delle funzionalità per

la stampa autonoma, ovvero l'aggiunta di una scheda SD contenente i file da stampare, un pannello LCD con cui visualizzare stato, menu e comandi per la stampante e infine un sistema di selezione e input. Tutti dispositivi che normalmente si collegano a un Arduino e la nostra stampante, come molte altre RepRap, ha proprio un Arduino come cuore.

Vaso alto/basso istrice e coppa

Questa è una stampa di bassa difficoltà, tempi ridotti, ma di un certo effetto. Appartiene alla categoria dei vasi e ha la particolarità di adattarsi a rielaborazioni tramite il processo di scalatura su un solo asse offerto da Repetier. La genesi di questo modello è semplice: abbiamo trovato una coppia di vasi chiamata "Double Twisted Vase Inside Out" e dopo aver stampato quello più tondo, abbiamo pensato di rielaborare il tutto tagliando il vaso a metà. La parte inferiore è diventata la ciotola, mentre quella superiore ruotata di 180 gradi sull'asse X, normale e allungata è diventata un calice basso o alto. In tutti i casi, per stampare i vasi come questo, nati come modello 3D pieno, è necessario utilizzare delle impostazioni particolari, con i "top solid layers" impostati a 0 e il riempimento (fill) ugualmente a 0. In questo modo lo slicer produce solo la "pelle" dell'oggetto con il numero di perimetri indicato (noi abbiamo scelto 2 per dare un po' di robustezza), la base d'appoggio (bottom solid layer) e nient'altro. Questa serie di tre vasi, che abbiamo raccolto sotto un unico gruppo di cui avete tre STL



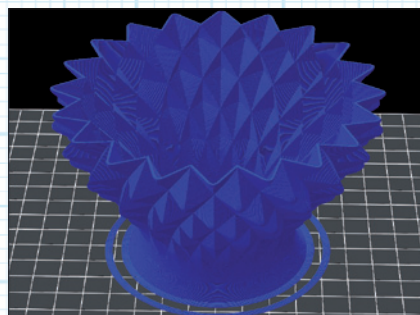
e un G-Code, si presta a numerose sperimentazioni su parametri quali perimetri, velocità di stampa, spessore (z) degli strati e temperature. Come nota pratica, vi segnaliamo che nel calice basso è critica la serie dei primi strati a causa delle temperature troppo elevate (accendete la ventola alla fine del 1° layer) o velocità di stampa sostenute, mentre dopo i primi 30 strati la difficoltà della stampa rientra

nella norma. Il problema sta nel fatto che in questi primi strati il diametro si stringe molto rapidamente, arrivando ad un angolo delle pareti tale da essere critico con Z superiori a 0.18mm; se poi il materiale è troppo caldo e morbido, lo strato si affloscia e la stampa viene compromessa.

Nome file:
DoubleInsideout100 CaliceAlto.gcode

PARAMETRI DI STAMPA:

layer_height =	0.2	travel_speed =	100
perimeters =	2	nozzle_diameter =	0.5
top_solid_layers =	0	filament_diameter =	3
bottom_solid_layers =	4	extrusion_multiplier =	1
fill_density =	0	perimeters extrusion width =	0.85mm
perimeter_speed =	80	infill extrusion width =	1.02mm
infill_speed =	80	solid infill extrusion width =	0.85mm
		top infill extrusion width =	0.85mm
		first layer extrusion width =	0.60mm



PREDISPOSIZIONE SOFTWARE

Uno dei motivi che ci ha portato quasi un anno fa a preferire Marlin come firmware agli altri, tutto sommato equivalenti all'epoca, è stato l'approccio innovativo che l'autore ha dimostrato sin dalle prime versioni. La scelta si è rivelata vincente e molte altre stampanti 3D sfruttano con profitto questo firmware, così come sono molti gli utenti che hanno

scelto di passare da Pronterface a Repetier-Host.

Ora Marlin ci permette di gestire le funzionalità per la stampa autonoma disponendo già da varie release di tutto il codice necessario per questa modalità funzionale. Chi ha provato ad aggiornare la scheda avrà notato che nel firmware sono presenti dei moduli dedicati sia al pilotaggio degli LCD, sia alla gestione

della scheda di memoria SD.

Basta quindi intervenire su pochi file per avere la stampante perfettamente operativa in modalità autonoma.

CON PC E SENZA PC

Nel realizzare il sistema di

Verme snodato



La prima volta che abbiamo stampato questo modello, siamo rimasti stregati dalla "magia" dei modelli con snodi auto assemblanti. In pratica, un pezzo unico diventava un vermicello con quattro parti snodate fra loro, ma indissolubilmente collegate. Il trucco sta in un sistema di perni e cavità che si creano e si chiudono durante il processo di stampa a strati, con ciascun segmento che viene quindi agganciato al precedente con il perno,

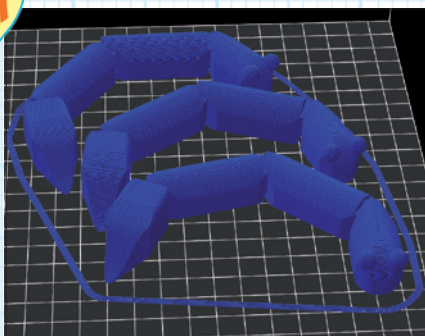
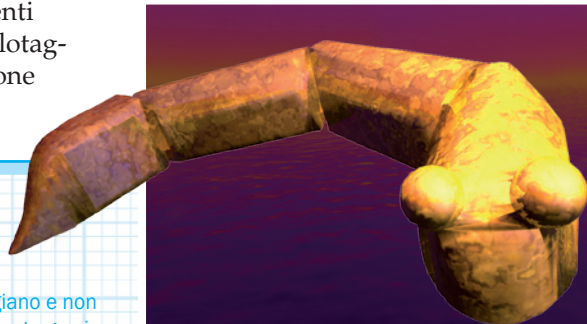
ma con le cavità che si appoggiano e non si saldano al segmento precedente grazie a qualche strato di separazione. Terminata la stampa, si staccano i pezzi e con un po' di forza si sbloccano le giunture. Va fatto un movimento di rotazione senza strappi o senza colpi secchi: il PLA cederà nei punti dove gli strati sono a contatto e lo snodo sarà libero.

Con qualche movimento ripetuto, la situazione migliora fino ad avere una completa libertà di movimento. Il file di stampa che vi abbiamo preparato ha tre vermi ingranditi a 1.2 rispetto all'originale, con un maggior dettaglio nella stampa grazie all'impostazione a 0.4 mm del diametro di estrusione per perimetri e fill. Quando si stampa "truccando" il diametro del filo estruso, la temperatura va impostata con precisione e quindi a seconda del PLA usato potrete dover alzare o abbassare i valori in corso di stampa.

Per prendere confidenza con la stampa di questo oggetto, potete portare il curso-

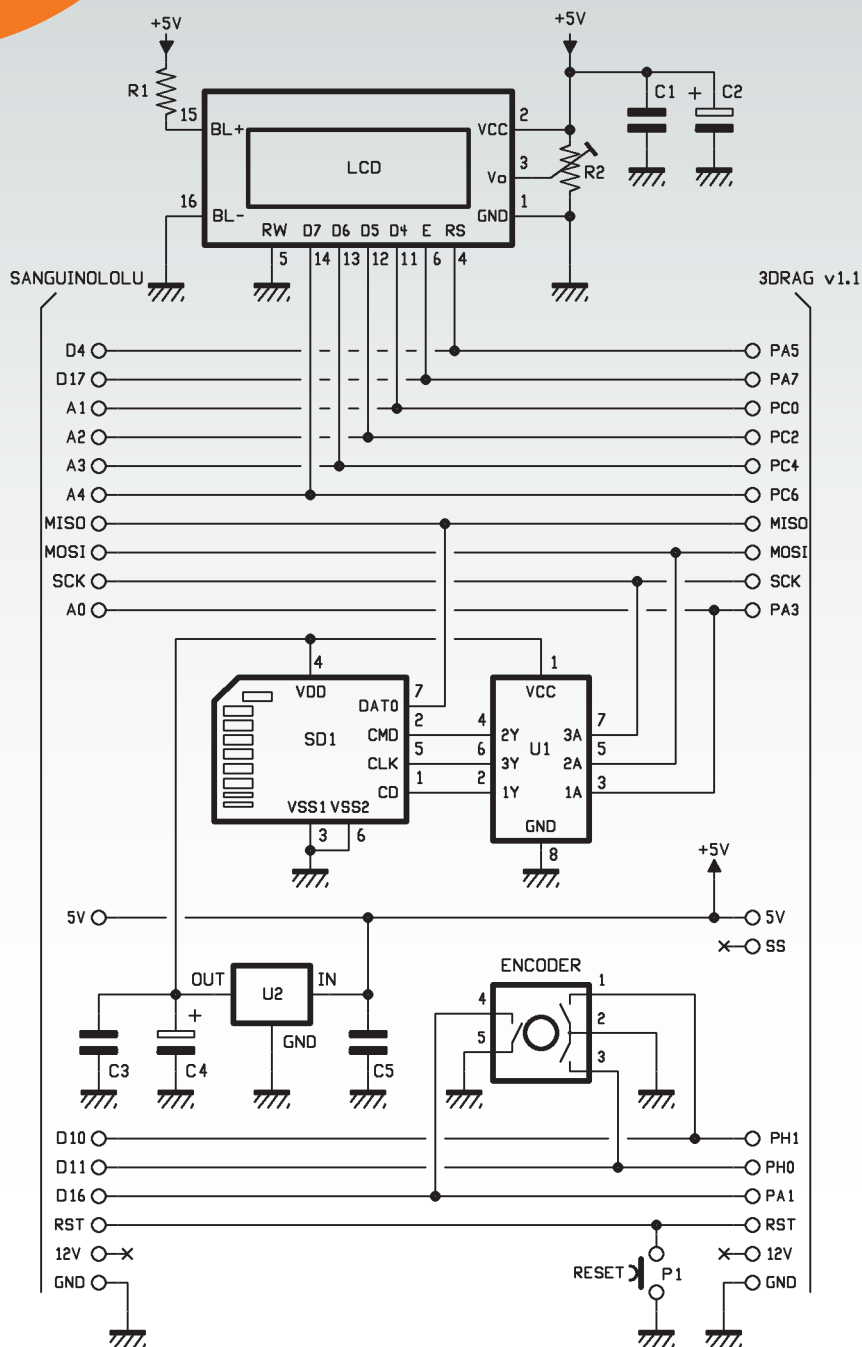
re "feedrate" a 60 per far rallentare la stampa dei primi strati, consentendovi di trovare le temperature ottimali, quindi aumentate a passi di 10 fino a tornare al 100% della velocità (90 mm/sec) che permette di avere il risultato perfetto solo se le temperature sono a posto. Tre vermi, ingranditi e stampati a 0.4 mm di diametro del filo estruso richiedono almeno 4 ore per essere stampati, mentre se stampate un solo elemento, a dimensioni naturali, avete il risultato in circa 50 minuti. Noi vi forniamo il G-Code per i 3 vermi ingranditi, mentre quello con un solo verme potete provare a realizzarlo voi scegliendo i parametri che ritenete adatti.

Nome File: **worm_snodato_x3_pla.gcode**



PARAMETRI DI STAMPA:

layer_height =	0.18	nozzle_diameter =	0.5
perimeters =	1	filament_diameter =	3
top_solid_layers =	4	extrusion_multiplier =	1
bottom_solid_layers =	4	perimeters extrusion width =	0.40mm
fill_density =	0.3	infill extrusion width =	0.40mm
perimeter_speed =	90	solid infill extrusion width =	0.40mm
infill_speed =	90	top infill extrusion width =	0.40mm
travel_speed =	140	first layer extrusion width =	0.80mm



con la stampante collegata a un pc, grazie alla serie di messaggi e comandi che il firmware ricetrasmette via seriale con il computer ospite si ha un controllo del processo di stampa ancora completo di tutte le opzioni che offrono i migliori pacchetti di stampa da PC. Senza PC, invece, abbiamo il controllo da display ed encoder di temperatura in tempo reale, accelerazione, velocità, flow rate, preriscaldamento, blocco e rilascio dei motori, spostamenti e tutto il controllo del posizionamento del piano o dell'estrusore, più l'avvio della stampa in manuale o in automatico tramite autostart.

LO SCHEMA ELETTRICO

Il circuito utilizza i due connettori della scheda Sanguinololu 1.3a - posti vicino al pulsante di reset - o il connettore della nostra scheda 3Drag v 1.1 per portare alcune linee dati al display LCD, i pin di SPI per gestire la scheda SD, tre ingressi digitali per il digital encoder e il click di selezione e la linea di Reset per consentire di intervenire in caso di necessità. Il fatto che si riesca a collegare tutti questi nuovi elementi alla scheda deriva dal fatto che il circuito utilizza sì parecchie porte di I/O digitali e due analogiche per azionare motori, riscaldatori e leggere i vari microswitch, ma resta ancora quel tanto che basta di I/O per i nostri scopi. Quello che invece non basta sono i 64K di memoria

stampa autonoma, Erik van der Zalm ha studiato una soluzione completa e di grande comodità: l'utente può gestire un approccio di tipo "carica, bufferizza e stampa" che libera il PC dopo aver trasferito su SD il file da stampare, oppure "stampa da SD" senza che il PC entri a far parte del processo, interamente gestito dal display LCD aggiunto

alla stampante. Come vedremo in fase di messa in opera e collaudo, l'obiettivo di rendere 3Drag e le stampanti dotate di questo firmware delle vere stampanti autonome è stato pienamente raggiunto; nella stampa di file presenti su SD, ma



dell'ATmega 664P, montato su Sanguinololu. La nostra scheda invece già dispone di un ATmega 2560 con 256K di memoria Flash per il firmware e quindi non ha problemi a caricare e far girare Marlin compilato con le opzioni SD e LCD attivate. Il display utilizza solo sette collegamenti:

LCD-RS: PB4/PWM
(Sanguinololu riporta erroneamente D12)
LCD-RW: GND
LCD-E: PC1/SDA
LCD-DB4: A1
LCD-DB5: A2
LCD-DB6: A3
LCD-DB7: A4

La modalità di colloquio con il controller di tipo Hitachi HD44780 è su quattro linee dati; vengono mandati due nibble per ricreare un byte. La libreria per usare questo tipo di display con il protocollo necessario fa parte del "core" ed è quindi presente da tempo nella distribuzione base dell'IDE di Arduino. La scheda SD si collega invece ai pin per il colloquio SPI, presente su Sanguinololu per consentire la programmazione di bootloader e firmware anche con il microcontrollore nuovo (ovvero privo del bootloader Arduino).

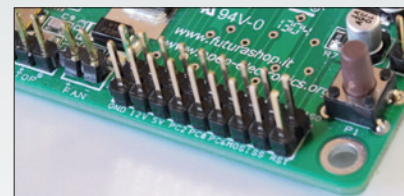
SD-CS: A0
SD-DI: MOSI
SD-SCLK: SCK
SD-DO: MISO

Come richiede la libreria relativa all'uso delle SD e di SPI, oltre ai pin nativi della porta SPI dell'ATmega, un pin va destinato al chip select del dispositivo e nel nostro caso è stato utilizzato A0 riconfigurato come I/O digitale in uscita (PA3 nel caso della scheda 3Drag V.1.1).

Per adattare tensioni e livelli agli standard delle schede SD e microSD è presente sia il regolatore di tensione MCP1702-3302E, sia l'adattatore di livelli 74HC4050. Il primo dei due circuiti integrati ci permette di portare a 3,3V l'alimentazione della scheda di memoria, rendendo così il circuito conforme agli standard delle memorie di questo tipo: lo standard prevede che l'alimentazione possa scendere anche a 2,7V, ma non deve superare i 3,6V. Per la comunicazione, noterete come i segnali che vanno dal microcontrollore della stampante, ovvero MOSI, SCK e il pin per selezionare il dispositivo (CO nello schema), allo slot in cui inserirete la memoria transitano da U1. Questo adattatore di livelli ha dei buffer non invertenti che accettano sugli ingressi - contrassegnati con A - ingressi fino a 15V per il valore logico 1, mentre questo livello sulle uscite - contrassegnate con Y - è limitato alla tensione di alimentazione dell'integrato che, nel nostro caso, arriva dal regolatore da 3,3V.

Il pin (DAT0) su cui la scheda SD invia i dati al microcontrollore secondo il protocollo SPI, (MISO) non ha alcun adattatore di livello perché un livello di 3,3V viene già interpretato dall'elettronica del microcontrollore come un livello logico "1" (sopra la soglia d'incertezza dei 2,5V).

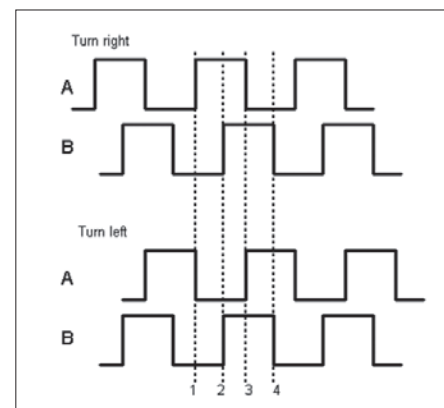
Il componente indicato come Encoder, sullo schema, è nella sua semplicità un dispositivo che ci permette di scorrere il cursore sui menu nelle due direzioni "su" e "giù"; in realtà i due interruttori collegati ai pin 1, 2, e 3 generano impulsi con i fronti di salita sfasati per consentire di capire se la rotazione è oraria o antioraria. La chiusura del contatto fra i pin 4 e 5, generata dalla pressione sull'alberino, è raccolta e



interpretata dal software come il comando di selezione. In pratica questo componente si comporta come la rotella d'interazione montata sulle autovetture di alcune marche tedesche che hanno trovato il sistema "turn and click" molto efficace per scorrere dei menu composti da liste su più livelli.

ENCODER-PUSH: PC0/SCL
ENCODER-RIGHT: PD2/RX1
ENCODER-LEFT: PD3/TX1

Il pulsante di reset è infine collegato al pin di Reset del micro, presente sul connettore a sei pin dell'ISP e necessario in fase di programmazione. Per noi diventa importante al fine di



Lo sfasamento dei segnali di un encoder digitale.

I file di configurazione

Per poter far funzionare il sistema di stampa autonoma è necessario partire dai sorgenti di Marlin, quindi fare le modifiche al file Configuration.h e poi verificare che sia tutto a posto anche in PINS.H (anche se è solo una precauzione). Iniziamo da chi ha una scheda controller di tipo **Sanguinololu**; questa scheda viene indicata in Configuration.h come 62 e dispone già di una serie di impostazioni definite da tempo. Ovviamente i parametri hardware specifici di 3Drag, relativi agli step dei motori, le accelerazioni e il funzionamento dei fine corsa va impostato come descritto nell'articolo di presentazione di 3Drag, mentre per aggiungere questo pannello dovete intervenire su altri parametri che trovate nella sezione "//LCD and SD support".

Fate riferimento agli estratti che trovate in questo box per capire dove intervenire: sono le righe evidenziate in neretto. Se vi sentite "persi", potete sempre recuperare dal nostro sito - 3dprint.elettronica.in - il firmware correttamente impostato per 3Drag con Sanguinololu e attivare le poche righe. Se invece imparate ad apportare tutte le varie modifiche manualmente, potrete cambiare firmware tutte e volte che vorrete in piena autonomia. Riportiamo anche PINS.H per darvi riscontro di come siano mappati i pin di LCD ed Encoder. Dato che il processore sulla board Sanguinololu va cambiato con un ATmega 1284P (dotato della memoria necessaria ad ospitare il codice), abbiamo predisposto il kit con il chip ATmega sostitutivo già programmato, mentre chi vuole fare tutto da sé può far riferimento alle istruzioni che si trovano in rete per programmare il 1284P con un bootloader Arduino per poi poterlo programmare normalmente con il firmware Marlin opportunamente modificato. Se avete una 3Drag V1.1 allora la scheda dispone di un ATmega 2560 e non va sostituito; dovete solo riprogrammarlo con la versione di Marlin modificata per avere SD e display LCD attivati. La nuova scheda di 3Drag V 1.1 è ora supportata direttamente da Marlin. Per essere certi che il sistema di stampa autonoma funzioni anche sulla nuova scheda, indicata nel firmware come "// 77 = 3Drag Controller", dovete necessariamente usare un firmware Marlin successivo al 10 giugno 2013, le versioni precedenti non riportano ancora la nostra scheda e le modifiche sarebbero impegnative. Ricordate che nel firmware vanno sempre impostati in Configuration.h i valori relativi ai microswitch e ai motori; trovate queste informazioni anche su questo numero, in fondo all'articolo di presentazione della nuova 3Drag v 1.1, a pag. 94. L'autore del firmware non ha creato degli automa-

tismi per impostare questi parametri anche perché sono molto "personali", ovvero ogni utente può ad esempio decidere di posizionare un microswitch a inizio o fine corsa, scegliendo la modalità normalmente aperto o normalmente chiuso; il numero di step per i motori dipende sia dai ponticelli scelti per la gestione dei microstep, sia dalle dimensioni fisiche di ingranaggi, pulegge ecc.

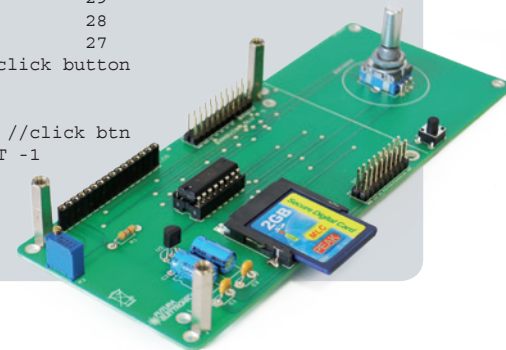
Quanto abbiamo riportato a pag. 94 è quindi la "carta d'identità meccanica ed elettrica" della nostra 3Drag, montata seguendo le nostre indicazioni, da integrare nel firmware per consentirne il corretto funzionamento. Sempre a pag. 94 trovate come attivare la stampa autonoma su 3Drag v 1.1.

Da CONFIGURATION.H (per Sanguinololu)

```
// 62 = Sanguinololu 1.2 and above
...
#define MOTHERBOARD 62
...
// EEPROM
...
#define EEPROM_SETTINGS
...
#define EEPROM_CHITCHAT
...
//LCD and SD support
...
#define SDSUPPORT // Enable SD Card Support in Hardware Console
...
#define ULTIPANEL //the ultipanel as on thingiverse
```

Da PINS.H (per Sanguinololu)

```
#ifndef ULTRA_LCD
#define NEWPANEL
//we have no buzzer installed
#define BEEPER -1
...
#define LCD_PINS_RS 4
#define LCD_PINS_ENABLE 17
#define LCD_PINS_D4 30
#define LCD_PINS_D5 29
#define LCD_PINS_D6 28
#define LCD_PINS_D7 27
//The encoder and click button
#define BTN_EN1 11
#define BTN_EN2 10
#define BTN_ENC 16 //click btn
#define SDCARDDTECT -1
#endif //Newpanel
#endif //Ultipanel
```



"sbloccare" la stampante in caso di crash, errore o altre situazioni in cui un Reset è la via più breve per riprendere il controllo della situazione.

RESET: RESET

Tutti questi collegamenti sono effettuati su pin diversi nel caso si utilizzi la nostra nuova scheda di controllo per 3Drag, versione 1.1.

I collegamenti sono i seguenti:

LCD-RS: PA5
LCD-RW: GND
LCD-E: PA7
LCD-DB4: PC0
LCD-DB5: PC2
LCD-DB6: PC4
LCD-DB7: PC6
SD-CS: PA3
SD-DI: MOSI
SD-SCLK: SCK
SD-DO: MISO

ENCODER-PUSH: PA1
ENCODER-RIGHT: PH1
ENCODER-LEFT: PH0
RESET: RESET

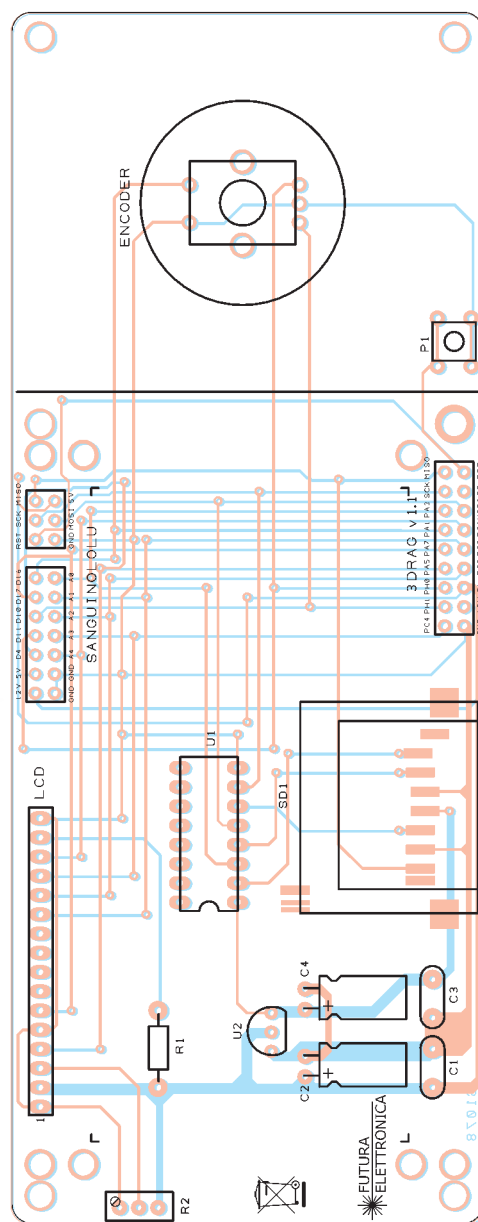
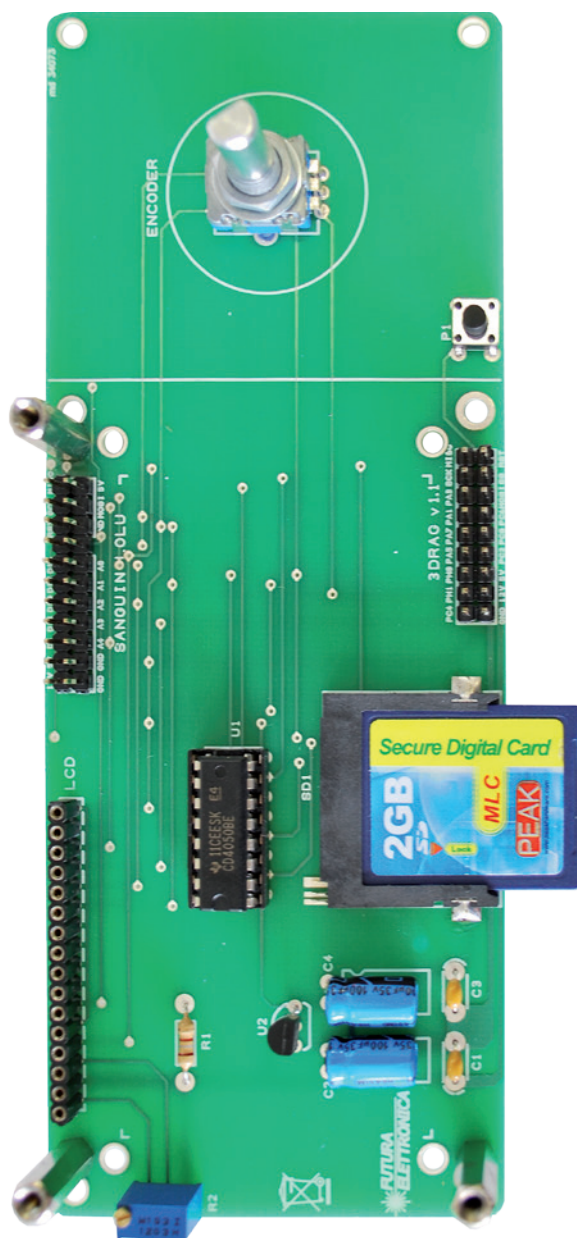
La flessibilità del firmware Marlin permette comunque di rimappare tutti i vari pin nel modo opportuno, modificando il file PINS.H; per vostra comodità,

Elenco Componenti:

R1: 33 ohm
R2: Trimmer multigiri
10 kohm
C1, C3: 100 nF
multistrato
C2: 100 μ F 16 VL
elettrolitico
C4: 100 μ F 16 VL
elettrolitico
P1: Microswitch
SD1: Connettore
SD-Card
U1: 74HC4050
U2: MCP1702-3302E/TO
ENCODER:
Encoder rotativo
(cod. ENCODER12MM)
LCD: Display LCD 20x4
(cod. LCD20X4)

Varie:

- Zoccolo 8+8
- Strip maschio 3 poli
11mm (2 pz.)
- Strip maschio 7 poli
11mm (2 pz.)
- Strip maschio 9 poli
11mm (2 pz.)
- Strip maschio 16 poli
25mm (1 pz.)
- Strip femmina 16 poli
(1 pz.)
- Torretta F/F 20mm
(3 pz.)
- Vite 3MA/8mm testa
conica (6 pz.)
- Flat cable
con connettore universale
- Circuito stampato



sul nostro sito è disponibile il file per la scheda di nostra produzione con le definizioni dei pin gestite attraverso l'aggiunta della definizione di una nuova scheda – la nostra - fra quelle disponibili in Configuration.h.

I pochi componenti passivi che trovate sulla basetta sono condensatori che servono per filtrare l'alimentazione che, pur arrivando dallo stabilizzatore della sche-

da della stampante, passando su un cavo flat potrebbe raccogliere interferenze; il piccolo trimmer R2 che serve per la regolazione del contrasto del pannello LCD, mentre R1 ha il compito di limitare la corrente assorbita dalla retroilluminazione LED.

La scheda SD viene letta tramite lo Slot SD1 da stampato; abbiamo scelto il formato SD e non microSD perché lo riteniamo molto

più robusto fisicamente per un impiego con ripetuti inserimenti ed estrazioni.

Poi, abbiamo tutti in giro per casa una o più schede SD che avanzano perché sostituite dalle ora più diffuse microSD.

Infine, se acquistate una microSD, solitamente viene fornita con un adattatore di formato che ne permette comunque l'uso negli slot per SD.

IL MONTAGGIO

Il circuito stampato che abbiamo predisposto è di tipo a doppia faccia e monta componenti di tipo tradizionale. Potete scaricare come sempre i master dal nostro sito per produrre la basetta autonomamente, oppure potete approfittare delle nostre versioni in kit.

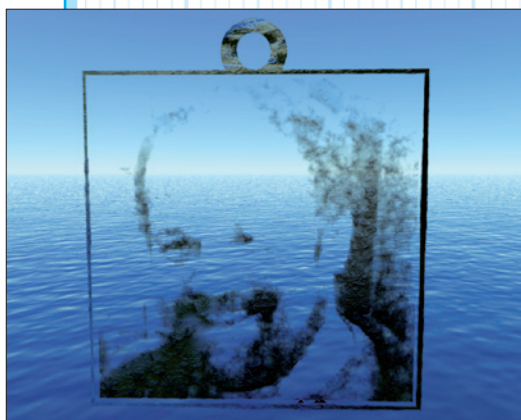
I componenti sono pochi e non ci sono particolari difficoltà nel montaggio, ma come sempre è meglio partire dai componenti passivi più a basso profilo per terminare con i componenti attivi

e i vari strip di collegamento.

La scheda che abbiamo realizzato è compatibile con l'elettronica della nostra prima 3Drag (e con tutte le stampanti che utilizzano la scheda controller "Sanguinololu" fino alla release 1.3a) tramite il connettore dedicato, mentre chi ha una 3Drag con la scheda di nostra progettazione utilizza il connettore indicato con 3Drag V 1.1. In teoria potreste quindi saldare i pinstrip maschio solo del connettore che vi interessa utilizzare, senza montare l'altro. Quando montate il regolatore

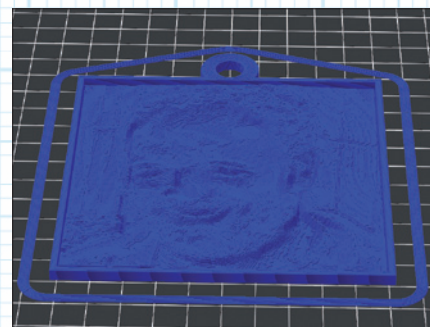
U2, fate attenzione a come lo posizionate e allo stesso modo il circuito integrato U1 a sedici pin, va orientato rispettando la tacca di riferimento che permette di individuare il pin 1. Anche C2 e C4 hanno una polarità e pertanto questa va rispettata. Vi ricordiamo che il reoforo più lungo corrisponde al positivo, mentre la banda continua stampigliata sull'involucro individua il reoforo del negativo. Lo slot per scheda SD è stato scelto fra quelli che hanno il montaggio più semplice, con saldature e terminali che non

Foto a strati - Lithopane



Per finire in bellezza, vi proponiamo una stampa 3D molto particolare. A seconda del materiale utilizzato, si ha l'attenuazione della luce trasmessa molto variabile. Il PLA offre varie gra-

dazioni di attenuazione, iniziando con il neutro, passando dai trasparenti (giallo, rosso, verde, argento, oro, blu) e finendo con quelli opachi (verde nucleare, arancio, rosa, grigio, bianco e nero), con il fosforescente che rappresenta una categoria a sé stante. Questa introduzione serve a capire come funziona la stampa delle fotografie in 3D: stratificando l'immagine con il numero di strati che corrisponde all'intensità del nero in ciascun punto. Il risultato è una stampa 3D che vista direttamente dice poco, ma che illuminata in trasparenza acquista la dinamica della fotografia da cui è stata creata. L'effetto è più facile da mostrare con un'immagine che da spiegare e il risultato è decisamente interessante. Per realizzare le vostre foto 3D (o lithopanes - modalità molto diffusa a fine 19° secolo per "nascondere" immagini di fanciulle poco vestite nei fondi delle tazze da the in porcellana) useremo la sintesi additiva con



una soluzione resa disponibile su Thingiverse (<http://www.thingiverse.com/thing:74322>), mentre chi ha un sistema ad ablazione con fresa (un CNC), probabilmente già conosce questa tecnica; se volete una carrellata di esempi, cercate in Google immagini Lithopane. Con Thingiverse, l'immagine da trasformare in stampa 3D ha 100x100 pixel di risoluzione



PARAMETRI DI STAMPA:

layer_height =	0.18	nozzle_diameter =	0.5
perimeters =	2	filament_diameter =	3
top_solid_layers =	4	extrusion_multiplier =	1
bottom_solid_layers =	4	perimeters extrusion width =	0.40mm
fill_density =	0.4	infill extrusion width =	0.40mm
perimeter_speed =	50	solid infill extrusion width =	0.40mm
infill_speed =	50	top infill extrusion width =	0.40mm
travel_speed =	120	first layer extrusion width =	0.60mm

richiedono attrezzi specifici o per SMD e il tutto dovrebbe essere pronto per l'uso in poco tempo.

PREPARAZIONE DEL SOFTWARE

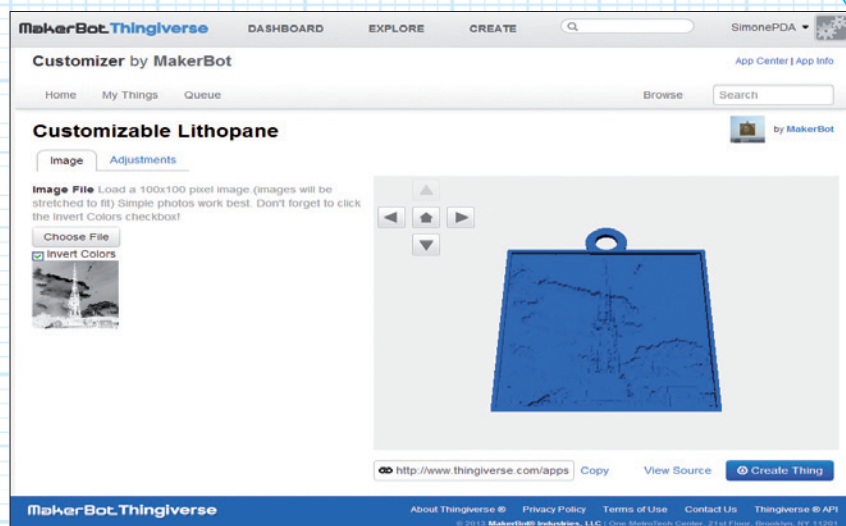
Questo pannello di controllo per la stampa autonoma non contiene di per sé le funzionalità che servono allo scopo: la memoria contiene i file, ma deve essere letta e il display mostra quello che gli viene inviato. Le reali funzionalità sono contenute dal firmware della scheda e nello specifico dal firmware Marlin. Tra librerie e codice aggiuntivo,

però, si supera il limite dei 64K di memoria disponibile per il codice sugli ATmega 664P e quindi il processore montato dalle schede Sanguinololu – incluse quelle montate sulla prima serie delle nostre 3Drag – non è adatto a far funzionare oltre a tutto il resto anche la parte di SD e LCD. Tutto questo è ampiamente documentato dall'autore del firmware e in effetti il problema è facilmente risolto passando a una versione di ATmega più capiente, ma che per nostra fortuna resta "pin to pin" compatibile. Il mi-

crocontrollore 1284P è quello più indicato per la sostituzione sulle schede Sanguinololu, mentre la nostra board 3Drag v 1.1 già dispone di un ATmega 2560 che ha addirittura 256 KB di memoria disponibile per il firmware. Come sapete, le RepRap sono nate partendo da un Arduino MEGA e poi si sono evolute con schede dedicate che, però, conservano la compatibilità con Arduino ed il suo IDE. Grazie a questo è possibile riprogrammarle con grande facilità purché il micro abbia già a bordo il bootloader. Diversamente bisogna utilizzare la programmazione tramite i pin ISP (In System Programming) del micro, a cui vanno collegati quelli di un Arduino come documentato in questo tutorial (<http://arduino.cc/en/Tutorial/ArduinoISP>).

In generale, Marlin non è comunque compilato con la parte di LCD e controllo tramite encoder attivi e per questo è necessario scaricare dal sito dell'autore l'ultima versione e procedere alla sua personalizzazione con i parametri di 3Drag nel file "Configuration.h" e attivando la parte di LCD e scheda SD tramite il relativo "#define". Trovate tutte le modifiche e le impostazioni nel box dedicato e comunque abbiamo predisposto sul nostro sito le due versioni del firmware per Sanguinololu con 1284P e per 3Drag V. 1.1 pronte all'uso. Colleghiamo il cavo.

Alimentazione e linee dati passano su un unico cavo di tipo flat con due connettori agli estremi. Per chi ha una scheda Sanguinololu, serve quello con il connettore da 22 poli (due file da 11), mentre per chi ha la scheda 3Drag v 1.1 è necessario il cavo flat con connettore a 18 poli (due file da 9). A stampante spenta potete collegare il cavo da entrambe

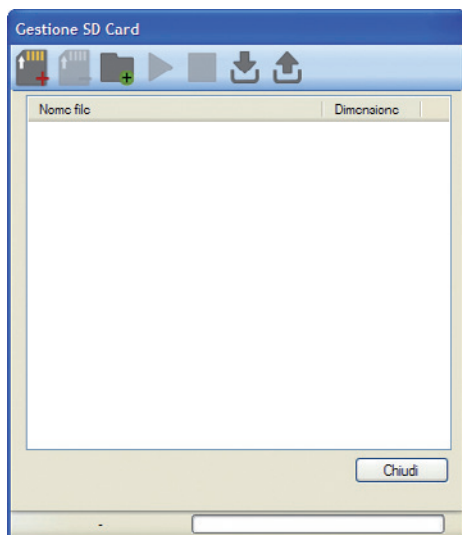


e la procedura provvede a ridimensionare quello che caricherete a questa dimensione (attenzione alle proporzioni!). Il file generato è un STL per la stampa di un pannello da 10,3 x 10,3 cm, spesso 3,8 mm. Per consentirvi di avere subito una prova abbiamo prodotto un file scegliendo l'immagine di Massimo Banzì, scattata in occasione del tour per la presentazione della Maker Faire che si terrà a Roma il prossimo ottobre. La stampa in assoluto migliore l'abbiamo ottenuta utilizzando il PLA fosforescente: grande omogeneità di estrusione – a circa 192 gradi – e soprattutto

una relativa opacità degli strati che ci ha permesso di avere un risultato con un'ottima dinamica. Voi sperimentate con i vari PLA che avete a disposizione per trovare quello che rende meglio. Lo spessore di stampa a 0.18 è quello che ci ha dato risultati migliori, mentre la densità di fill e il numero di perimetri offre risultati diversi. Se userete il nostro file in G-Code, modificate a mano o nel file le temperature per il PLA che userete.

Nome File:
Massimo_Glow_Lithoplate.gcode

*Repetier Host: il pannello
dell'SD card Manager.*



le parti facendo in modo che ci sia corrispondenza fra i pin. In pratica seguite il filo contrassegnato su un lato del cavo flat e fate in modo che se questo è dal lato di MISO e 5V (Sanguinololu), sulla scheda di stampa autonoma sia dal medesimo lato. Il medesimo ragionamento vale per la scheda 3Drag v 1.1.

COSA CAMBIA?

Con l'attivazione della parte relativa alla scheda SD, Marlin aggiunge comandi e funzioni al suo già ricco arsenale. Queste nuove funzioni hanno lo scopo di consentire la gestione dei file anche da computer, senza dover estrarre la scheda per aggiungere o modificare i file che poi saranno stampati. Con la memoria a bordo, il firmware diventa poi in grado di accodare vari lavori, comunicando sempre con il software di stampa attraverso specifici messaggi. Repetier-Host – il programma che noi utilizziamo e consigliamo per la gestione della stampante – dispone di un pannello di controllo specifico, che si trova sotto la voce "Stampante -> SD Card Manager". Con questo pannello è possibile interagire in

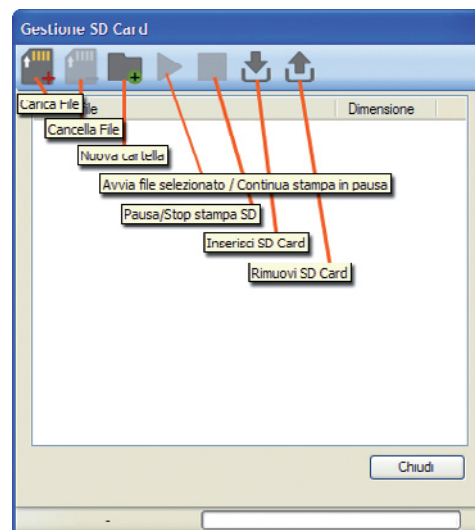
modo specifico con la stampante che, con una SD card inserita, dispone della memoria locale necessaria a "bufferizzare" uno o più file. Proprio per questa possibilità di scaricare il file, lanciare la stampa e poi scollegare il computer, Repetier-Host offre la possibilità di aggiungere comandi G-Code all'inizio e alla fine del file da stampare; con questa funzionalità diventa quindi semplice prevedere lo spegnimento dei vari riscaldatori, il posizionamento in un punto preciso e poi la disattivazione dei motori, riducendo a fine stampa l'assorbimento della stampante. Si tratta solo di un esempio, basato sulle nostre esperienze pratiche, ma la serie di comandi disponibile con il firmware Marlin potrà ispirarvi altre soluzioni d'utilizzo.

Il pannello mostra, se è presente una scheda SD, tutti i file che è possibile stampare (con ".g" quale estensione) e nasconde invece i file non stampabili e di sistema. Il file system gestito dalla libreria consente l'impiego di nomi nello standard 8.3 con sub directory nelle quali è possibile entrare e uscire cliccando opportunamente sulle etichette mostrate nell'area della lista dei file e delle cartelle. I file già presenti possono essere cancellati o avviati come stampa, mentre dal pannello è possibile creare una cartella o caricare un file da stampare successivamente. Sono presenti i due pulsanti a triangolo (PLAY) e a quadrato (STOP) con i quali si può avviare o riavviare la stampa e mettere in pausa o fermare la stampa.

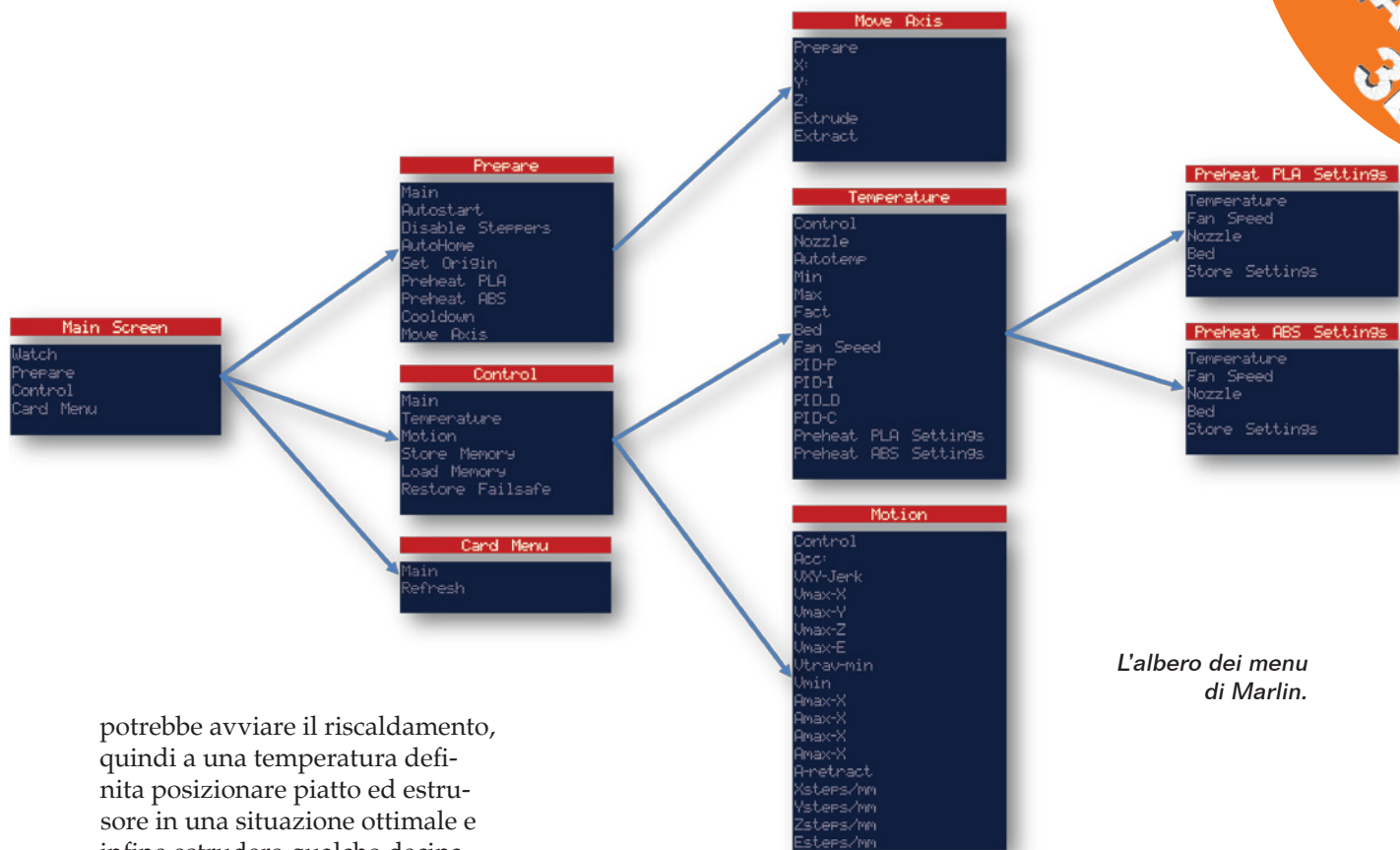
Le ultime due icone servono per chiudere i file e permettere la rimozione della SD card o per far leggere una card appena inserita. Provare a utilizzare le funzioni di questo pannello software è un buon collaudo dell'effettivo funzionamento del firmware, ma

siamo abbastanza certi che siate più ansiosi di passare all'utilizzo della stampante direttamente senza alcun collegamento. Se avete problemi, saltate al capitolo "Problemi?" e poi ritornate qui. Prima di fare stampe dalla scheda SD, dovete comunque preparare uno o più file su di essa e solo allora potrete lanciare le stampe.

Nel caricare i file, ricordate di osservare la struttura 8.3 ovvero otto caratteri al massimo per il nome, un "." e poi al massimo tre caratteri per l'estensione, ma nel nostro caso questa deve essere ".g" per i file che vorremo stampare (ad esempio "serpente.g"). Se poi il nome del file è di tipo "auto<0-9>.g" allora potrete utilizzare la stampa con l'avvio automatico in sequenza ordinata (se ci sono più file, si parte da quello con il numero più basso e poi gli altri a seguire). Gli utenti più "esperti", sapendo che all'accensione la stampante esegue questi file, può predisporre uno solo che effettua delle operazioni tramite G-Code scritto a mano e che prepara la stampante alle sue normali attività (ad esempio si



Le funzioni delle varie icone dell'interfaccia SD Card Manager di Repetier Host.



L'albero dei menu di Marlin.

potrebbe avviare il riscaldamento, quindi a una temperatura definita posizionare piatto ed estrudere in una situazione ottimale e infine estrudere qualche decina di millimetri per mettere in piena efficienza il blocco riscaldante). Come vedremo più avanti, senza rifare il boot della stampante (possibile anche premendo il pulsante di Reset del pannello), è possibile avviare l'esecuzione dei file "auto" scegliendo l'opzione Autostart dal menu su LCD.

PROBLEMI?

Se avete eseguito tutto il montaggio come indicato e avete aggiornato il software sul controller della stampante (o avete montato il nuovo ATmega fornito nel kit per schede Sanguinololu), avrete sullo schermo a LCD delle scritte. Se non appare nulla o ci sono caratteri privi di significato, c'è un problema. Potete capire se si tratta di qualche banalità nei collegamenti se tramite seriale potete gestire ancora la stampante da computer oppure avete lavorato sulla scheda SD ed ha funzionato tutto; in questo caso il problema è probabilmente legato al trimmer per regolare il contrasto: agite su di esso ed è molto

probabile che compaia il testo. Se invece avete la stampante che non risponde al pc, il firmware ha dei problemi e va riscritto facendo attenzione ad eventuali messaggi che appaiono nella finestra in basso nell'IDE. Altra possibilità, se la stampante risponde al PC, ma non vede la scheda SD e l'LCD è spento (senza neppure retroilluminazione), allora il cavo di collegamento è stato utilizzato in modo errato: connettore sbagliato sulla nuova scheda? Connettore nel verso sbagliato fra controller e scheda? Verificate e poi ridate tensione.

COSA FARE DA LCD

Con il display collegato ed attivato, il firmware presenta diverse informazioni, sia di tipo statico, sia dinamico. Nell'apposita immagine realizzata direttamente dall'autore c'è tutto l'albero dei menu, mentre durante la stampa o il funzionamento si può passare alla schermata "Watch"

("Guarda" se avete scelto l'interfaccia in lingua italiana) che mostra in tempo reale tutti i parametri funzionali come temperatura e posizione dei vari assi.

La struttura si basa su quattro voci principali – Watch, Prepare, Control, Card Menu – dai quali si può gestire la stampante vedendo cosa sta facendo (watch), oppure impostando i vari parametri funzionali e attivando i vari riscaldatori, quindi c'è la parte di controllo dove si gestiscono movimenti e parametri permanenti con ben due ulteriori livelli di menu, oppure si accede alla scheda di memoria per scegliere il file da lanciare in stampa. Da un menu al successivo si passa selezionando con il click dell'encoder (si preme la manopola), mentre per tornare al livello precedente ci si sposta in cima alla lista delle voci dove c'è sempre il titolo del menu precedente. Per impostare un parametro, ci si sposta su di esso ruotan-

do l'encoder, si clicca, si ruota la manopola per variare il valore, quindi si clicca nuovamente per uscire dal cambio del valore e tornare alla modalità di selezione delle voci della lista. I cambi di valore sono in tempo reale: come si cambia il valore di un asse, ad esempio, si aziona istantaneamente il motore.

Non mancano le funzioni per mettere in pausa la stampa ed accedere ai comandi di spostamento – utili ad esempio per cambiare il colore del filo di stampa.

SD CARICA? VIA ALLA STAMPA!

Se a questo punto avete preparato un file adatto alla vostra prima stampa di prova ed è sulla scheda SD presente nello slot del pannello aggiuntivo, potete procedere alla vostra prima vera stampa autonoma. Iniziate con l'accensione della stampante senza collegamento al PC, quindi quando apparirà la scritta "Printer ready..." potete inserire la scheda SD e vedere se sul display appare "Card inser-

ted". A questo punto con un clic entrate nel menu principale, dal quale scegliete "card menu" e successivamente il vostro file da stampare. Prima di iniziare la stampa, potreste voler sperimentare un po' con i comandi di controllo (temperatura, spostamento ecc.) e questo non può che aumentare la familiarità con le funzioni disponibili.

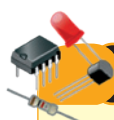
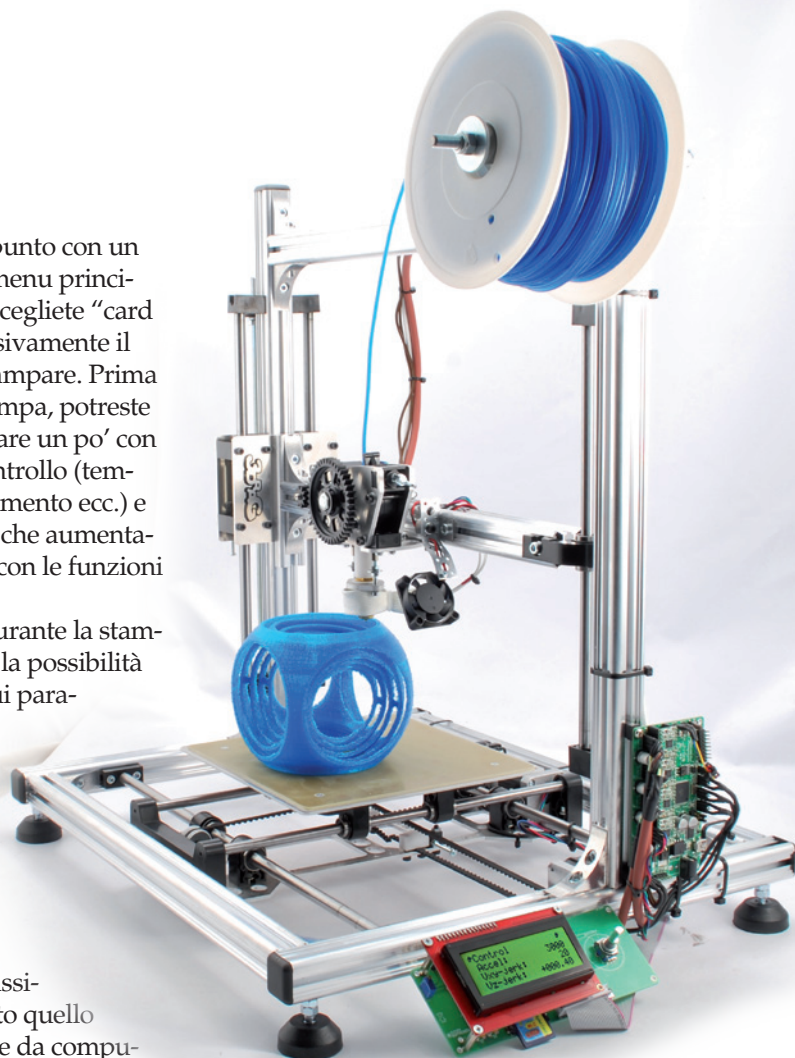
Ricordate che durante la stampa avete ancora la possibilità di intervenire sui parametri di stampa così come potete fare da Repetier-Host. Per motivi pratici e di com'è costruita l'interfaccia, vi assicuriamo che tutto quello che potevate fare da computer per controllare la stampante è ancora possibile tramite LCD, ma la posizione dei parametri e dei comandi segue una logica specifica. Fate riferimento alla mappa dei menu e confrontatela sia con la scheda dei parametri della EEPROM (su Repetier-Host), sia con le varie funzioni del "Controllo Manuale", sempre di Repetier.

OPEN SOFTWARE & HARDWARE!

Vogliamo chiudere l'articolo evidenziando che sia la parte software, sia la parte elettronica sono stati sviluppati da vari autori che hanno partecipato alla community di RepRap con il loro lavoro. Vogliamo quindi manifestare il nostro apprezzamento per la qualità raggiunta che è oggi anche superiore ad alcune soluzioni commerciali; questo dimostra che rimettere in circolo le proprie modifiche e migliorie, discutendo con la community e senza segreti o brevetti porta –

nelle situazioni giuste – ad una crescita reale e consistente del materiale. Se non avete ancora letto l'intervista all'autore di Slic3r, adesso è proprio il momento di farlo per capire meglio cosa intendiamo.

Come abbiamo indicato nell'articolo di apertura di questo "focus" sulla stampa 3D, stiamo utilizzando le versioni più recenti, ma è molto probabile – e anche desiderabile – che queste vengano presto superate da release successive. È ragionevole supporre che l'hardware non sarà da modificare, mentre i firmware e i software quasi certamente acquisiranno funzionalità nuove e diverse che non abbiamo potuto ovviamente documentare. Controllate quindi i siti degli autori e leggete le note di rilascio per decidere se è il caso di aggiornare o restare a quello che avete imparato ad utilizzare.



per il MATERIALE

Il kit completo per la stampa autonoma da collegare alla versione 1.1 della stampante 3Drag è disponibile al prezzo di 52,00 Euro (cod. FT1078K). La scatola di montaggio comprende la basetta, tutti i componenti, il display e il cavo flat; non è compresa la scheda SD. Il kit per la 3Drag versione 1.0 (che utilizza la scheda Sanguinololu) costa invece 61,00 Euro (cod. FT1078SK) e contiene in più un chip ATmega 1284PU completo di bootloader. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:

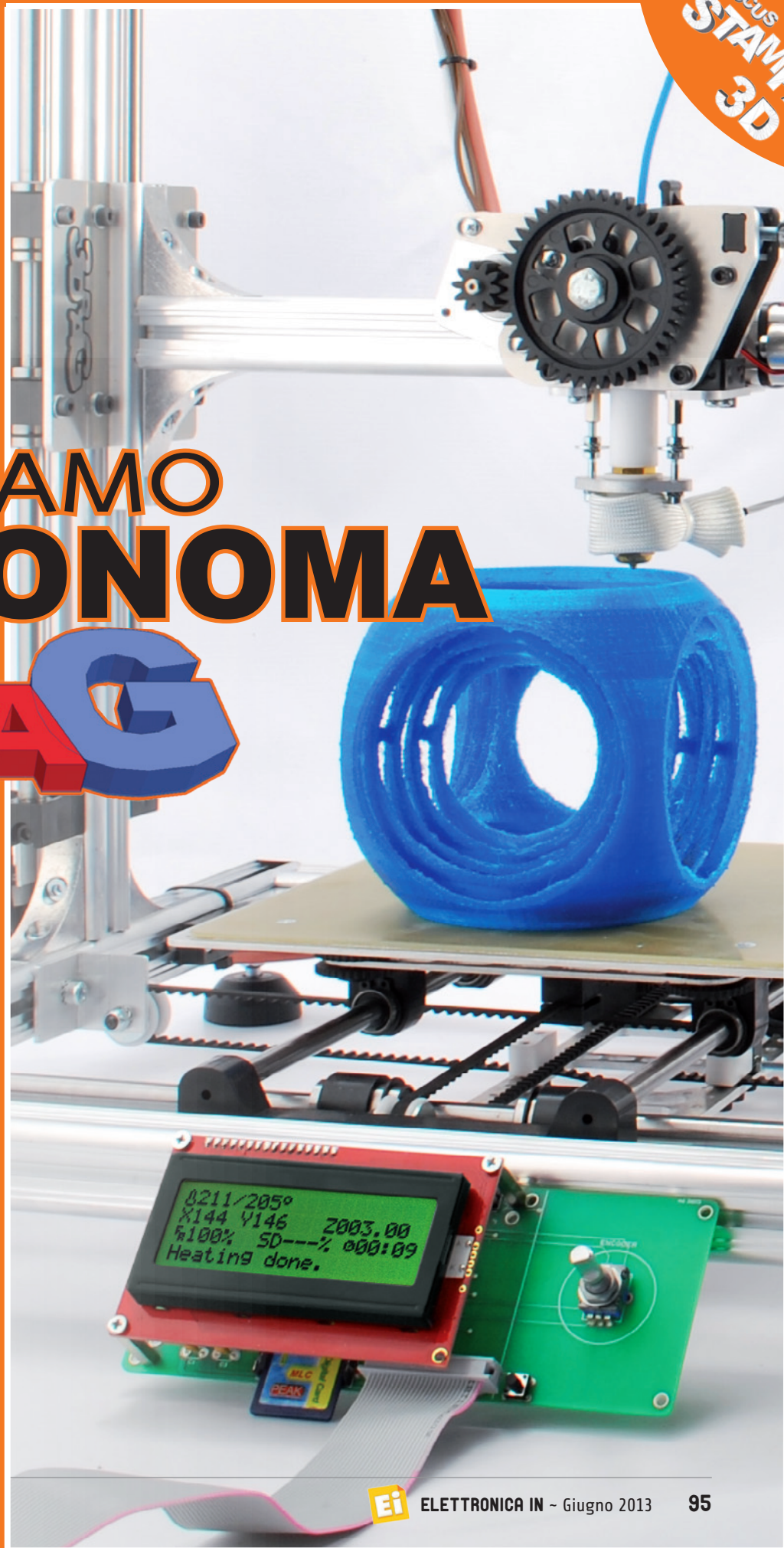
Futura Elettronica, Via Adige 11,
21013 Gallarate (VA)

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

Un accessorio tanto semplice quanto indispensabile per poter controllare 3Drag e stampare senza un computer collegato grazie a un cursore, a uno slot per schede SD e a un display LCD.

RENDIAMO AUTONOMA 3DRAG

Una delle caratteristiche delle stampanti 3D è la relativa lentezza nella stampa: per quanto la meccanica di 3Drag sia in grado di raggiungere e superare la velocità 100 millimetri al secondo per la stampa, un oggetto richiede comunque decine di minuti se non ore per essere portato a termine. Basta un fill un po' impegnativo o un pezzo massiccio per entrare nell'ambito delle ore, mentre chi è ormai abituato a stampare più parti in un sol colpo, organizzandole



sul piatto di stampa, sa che può essere necessaria la nottata per stampare tutto.

In tutto questo tempo è necessario inviare alla stampante le istruzioni in G-Code per controllarne i vari movimenti fino al termine della costruzione del modello in stampa. In pratica un notebook, un PC o un altro computer deve restare acceso e collegato, con il programma di stampa in funzio-

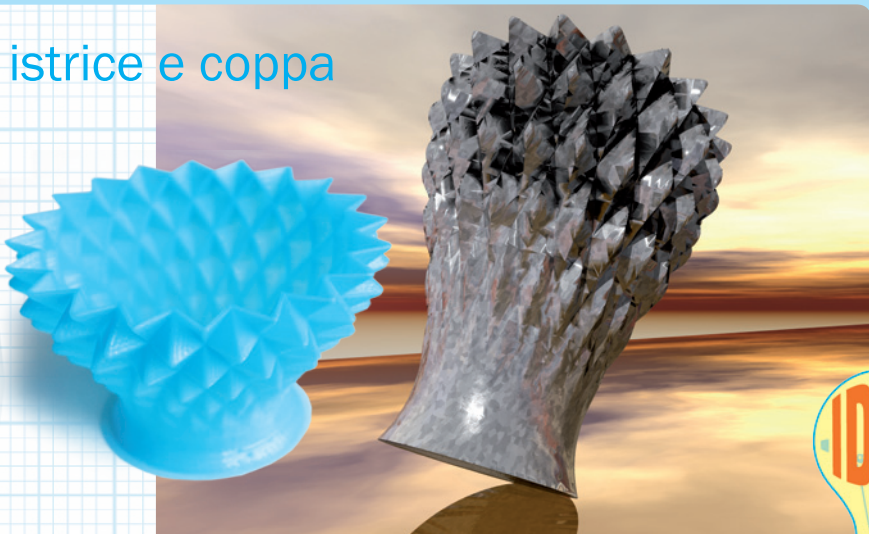
ne. E se uno ha a disposizione solo un notebook e vuole poterlo spostare in un'altra stanza per continuare a lavorare? Oppure deve uscire di casa, sempre con il notebook? Con la stampante in stampa non si può far altro che aspettare il termine del processo, oppure si può rinunciare alla stampa in corso.

La soluzione al problema è l'aggiunta delle funzionalità per

la stampa autonoma, ovvero l'aggiunta di una scheda SD contenente i file da stampare, un pannello LCD con cui visualizzare stato, menu e comandi per la stampante e infine un sistema di selezione e input. Tutti dispositivi che normalmente si collegano a un Arduino e la nostra stampante, come molte altre RepRap, ha proprio un Arduino come cuore.

Vaso alto/basso istrice e coppa

Questa è una stampa di bassa difficoltà, tempi ridotti, ma di un certo effetto. Appartiene alla categoria dei vasi e ha la particolarità di adattarsi a rielaborazioni tramite il processo di scalatura su un solo asse offerto da Repetier. La genesi di questo modello è semplice: abbiamo trovato una coppia di vasi chiamata "Double Twisted Vase Inside Out" e dopo aver stampato quello più tondo, abbiamo pensato di rielaborare il tutto tagliando il vaso a metà. La parte inferiore è diventata la ciotola, mentre quella superiore ruotata di 180 gradi sull'asse X, normale e allungata è diventata un calice basso o alto. In tutti i casi, per stampare i vasi come questo, nati come modello 3D pieno, è necessario utilizzare delle impostazioni particolari, con i "top solid layers" impostati a 0 e il riempimento (fill) ugualmente a 0. In questo modo lo slicer produce solo la "pelle" dell'oggetto con il numero di perimetri indicato (noi abbiamo scelto 2 per dare un po' di robustezza), la base d'appoggio (bottom solid layer) e nient'altro. Questa serie di tre vasi, che abbiamo raccolto sotto un unico gruppo di cui avete tre STL



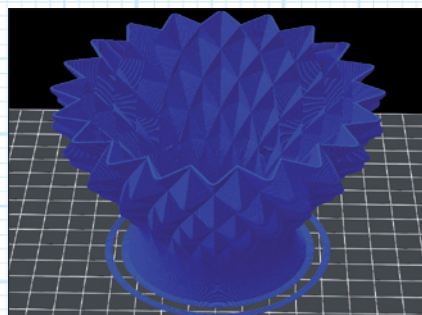
e un G-Code, si presta a numerose sperimentazioni su parametri quali perimetri, velocità di stampa, spessore (z) degli strati e temperature. Come nota pratica, vi segnaliamo che nel calice basso è critica la serie dei primi strati a causa delle temperature troppo elevate (accendete la ventola alla fine del 1° layer) o velocità di stampa sostenute, mentre dopo i primi 30 strati la difficoltà della stampa rientra

nella norma. Il problema sta nel fatto che in questi primi strati il diametro si stringe molto rapidamente, arrivando ad un angolo delle pareti tale da essere critico con Z superiori a 0.18mm; se poi il materiale è troppo caldo e morbido, lo strato si affloscia e la stampa viene compromessa.

Nome file:
DoubleInsideout100 CaliceAlto.gcode

PARAMETRI DI STAMPA:

layer_height =	0.2	travel_speed =	100
perimeters =	2	nozzle_diameter =	0.5
top_solid_layers =	0	filament_diameter =	3
bottom_solid_layers =	4	extrusion_multiplier =	1
fill_density =	0	perimeters extrusion width =	0.85mm
perimeter_speed =	80	infill extrusion width =	1.02mm
infill_speed =	80	solid infill extrusion width =	0.85mm
		top infill extrusion width =	0.85mm
		first layer extrusion width =	0.60mm



PREDISPOSIZIONE SOFTWARE

Uno dei motivi che ci ha portato quasi un anno fa a preferire Marlin come firmware agli altri, tutto sommato equivalenti all'epoca, è stato l'approccio innovativo che l'autore ha dimostrato sin dalle prime versioni. La scelta si è rivelata vincente e molte altre stampanti 3D sfruttano con profitto questo firmware, così come sono molti gli utenti che hanno

scelto di passare da Pronterface a Repetier-Host.

Ora Marlin ci permette di gestire le funzionalità per la stampa autonoma disponendo già da varie release di tutto il codice necessario per questa modalità funzionale. Chi ha provato ad aggiornare la scheda avrà notato che nel firmware sono presenti dei moduli dedicati sia al pilotaggio degli LCD, sia alla gestione

della scheda di memoria SD.

Basta quindi intervenire su pochi file per avere la stampante perfettamente operativa in modalità autonoma.

CON PC E SENZA PC

Nel realizzare il sistema di

Verme snodato



La prima volta che abbiamo stampato questo modello, siamo rimasti stregati dalla "magia" dei modelli con snodi auto assemblanti. In pratica, un pezzo unico diventava un vermicello con quattro parti snodate fra loro, ma indissolubilmente collegate. Il trucco sta in un sistema di perni e cavità che si creano e si chiudono durante il processo di stampa a strati, con ciascun segmento che viene quindi agganciato al precedente con il perno,

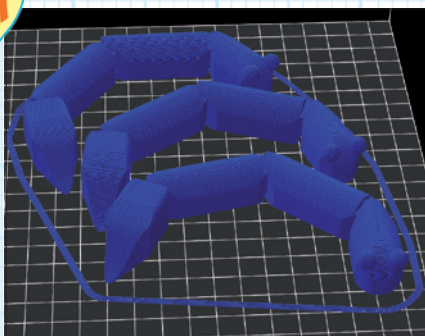
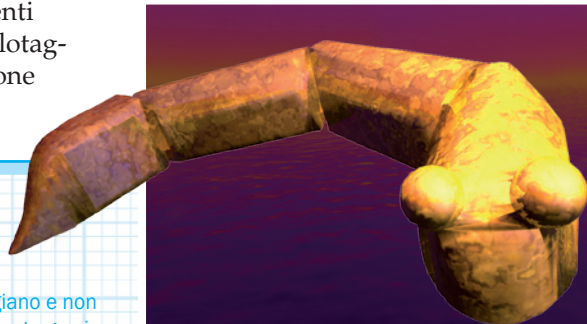
ma con le cavità che si appoggiano e non si saldano al segmento precedente grazie a qualche strato di separazione. Terminata la stampa, si staccano i pezzi e con un po' di forza si sbloccano le giunture. Va fatto un movimento di rotazione senza strappi o senza colpi secchi: il PLA cederà nei punti dove gli strati sono a contatto e lo snodo sarà libero.

Con qualche movimento ripetuto, la situazione migliora fino ad avere una completa libertà di movimento. Il file di stampa che vi abbiamo preparato ha tre vermi ingranditi a 1.2 rispetto all'originale, con un maggior dettaglio nella stampa grazie all'impostazione a 0.4 mm del diametro di estrusione per perimetri e fill. Quando si stampa "truccando" il diametro del filo estruso, la temperatura va impostata con precisione e quindi a seconda del PLA usato potrete dover alzare o abbassare i valori in corso di stampa.

Per prendere confidenza con la stampa di questo oggetto, potete portare il curso-

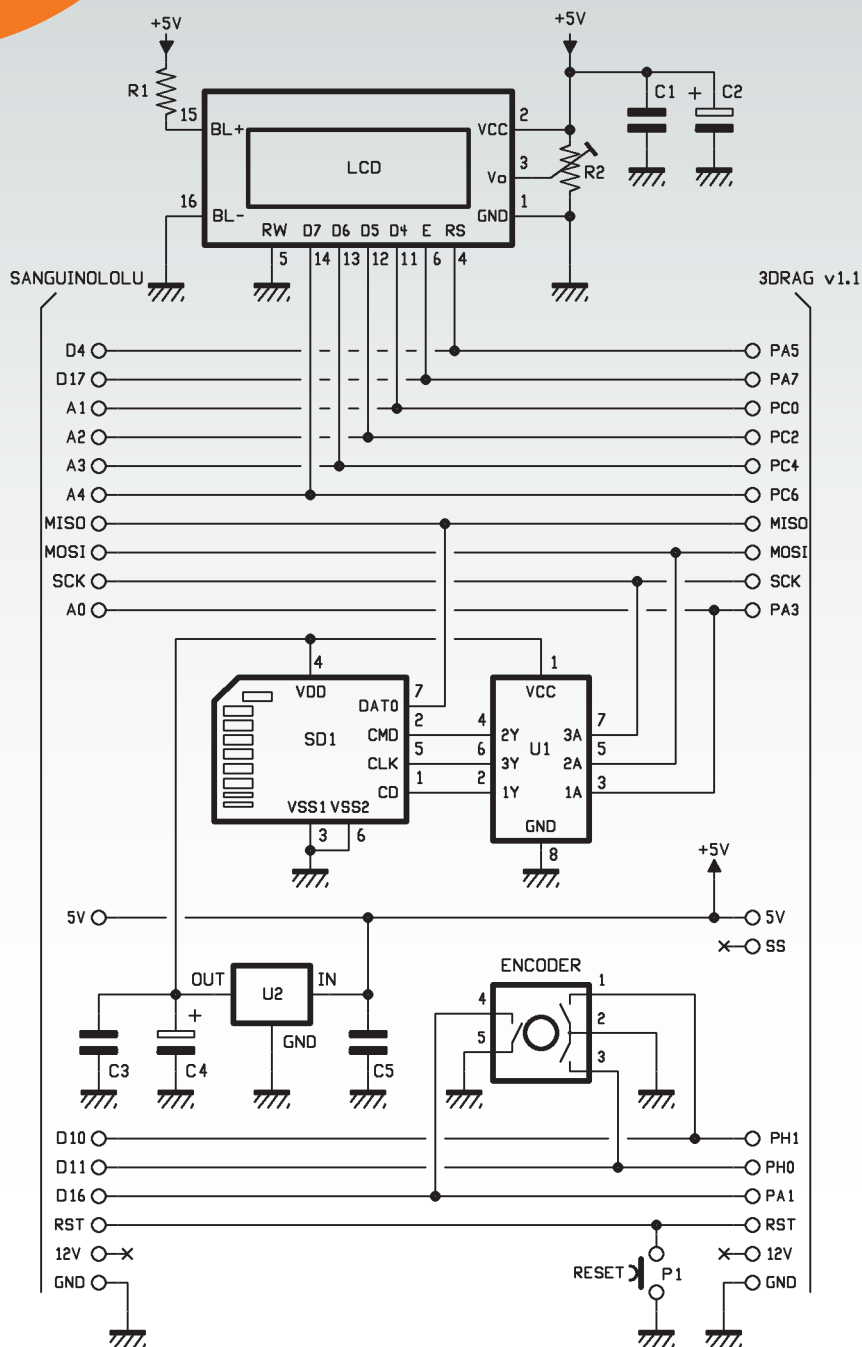
re "feedrate" a 60 per far rallentare la stampa dei primi strati, consentendovi di trovare le temperature ottimali, quindi aumentate a passi di 10 fino a tornare al 100% della velocità (90 mm/sec) che permette di avere il risultato perfetto solo se le temperature sono a posto. Tre vermi, ingranditi e stampati a 0.4 mm di diametro del filo estruso richiedono almeno 4 ore per essere stampati, mentre se stampate un solo elemento, a dimensioni naturali, avete il risultato in circa 50 minuti. Noi vi forniamo il G-Code per i 3 vermi ingranditi, mentre quello con un solo verme potete provare a realizzarlo voi scegliendo i parametri che ritenete adatti.

Nome File: **worm_snodato_x3_pla.gcode**



PARAMETRI DI STAMPA:

layer_height =	0.18	nozzle_diameter =	0.5
perimeters =	1	filament_diameter =	3
top_solid_layers =	4	extrusion_multiplier =	1
bottom_solid_layers =	4	perimeters extrusion width =	0.40mm
fill_density =	0.3	infill extrusion width =	0.40mm
perimeter_speed =	90	solid infill extrusion width =	0.40mm
infill_speed =	90	top infill extrusion width =	0.40mm
travel_speed =	140	first layer extrusion width =	0.80mm



con la stampante collegata a un pc, grazie alla serie di messaggi e comandi che il firmware ricetrasmette via seriale con il computer ospite si ha un controllo del processo di stampa ancora completo di tutte le opzioni che offrono i migliori pacchetti di stampa da PC. Senza PC, invece, abbiamo il controllo da display ed encoder di temperatura in tempo reale, accelerazione, velocità, flow rate, preriscaldamento, blocco e rilascio dei motori, spostamenti e tutto il controllo del posizionamento del piano o dell'estrusore, più l'avvio della stampa in manuale o in automatico tramite autostart.

LO SCHEMA ELETTRICO

Il circuito utilizza i due connettori della scheda Sanguinololu 1.3a - posti vicino al pulsante di reset - o il connettore della nostra scheda 3Drag v 1.1 per portare alcune linee dati al display LCD, i pin di SPI per gestire la scheda SD, tre ingressi digitali per il digital encoder e il click di selezione e la linea di Reset per consentire di intervenire in caso di necessità. Il fatto che si riesca a collegare tutti questi nuovi elementi alla scheda deriva dal fatto che il circuito utilizza sì parecchie porte di I/O digitali e due analogiche per azionare motori, riscaldatori e leggere i vari microswitch, ma resta ancora quel tanto che basta di I/O per i nostri scopi. Quello che invece non basta sono i 64K di memoria

stampa autonoma, Erik van der Zalm ha studiato una soluzione completa e di grande comodità: l'utente può gestire un approccio di tipo "carica, bufferizza e stampa" che libera il PC dopo aver trasferito su SD il file da stampare, oppure "stampa da SD" senza che il PC entri a far parte del processo, interamente gestito dal display LCD aggiunto

alla stampante. Come vedremo in fase di messa in opera e collaudo, l'obiettivo di rendere 3Drag e le stampanti dotate di questo firmware delle vere stampanti autonome è stato pienamente raggiunto; nella stampa di file presenti su SD, ma



dell'ATmega 664P, montato su Sanguinololu. La nostra scheda invece già dispone di un ATmega 2560 con 256K di memoria Flash per il firmware e quindi non ha problemi a caricare e far girare Marlin compilato con le opzioni SD e LCD attivate. Il display utilizza solo sette collegamenti:

LCD-RS: PB4/PWM
(Sanguinololu riporta erroneamente D12)
LCD-RW: GND
LCD-E: PC1/SDA
LCD-DB4: A1
LCD-DB5: A2
LCD-DB6: A3
LCD-DB7: A4

La modalità di colloquio con il controller di tipo Hitachi HD44780 è su quattro linee dati; vengono mandati due nibble per ricreare un byte. La libreria per usare questo tipo di display con il protocollo necessario fa parte del "core" ed è quindi presente da tempo nella distribuzione base dell'IDE di Arduino. La scheda SD si collega invece ai pin per il colloquio SPI, presente su Sanguinololu per consentire la programmazione di bootloader e firmware anche con il microcontrollore nuovo (ovvero privo del bootloader Arduino).

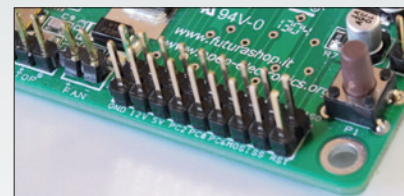
SD-CS: A0
SD-DI: MOSI
SD-SCLK: SCK
SD-DO: MISO

Come richiede la libreria relativa all'uso delle SD e di SPI, oltre ai pin nativi della porta SPI dell'ATmega, un pin va destinato al chip select del dispositivo e nel nostro caso è stato utilizzato A0 riconfigurato come I/O digitale in uscita (PA3 nel caso della scheda 3Drag V.1.1).

Per adattare tensioni e livelli agli standard delle schede SD e microSD è presente sia il regolatore di tensione MCP1702-3302E, sia l'adattatore di livelli 74HC4050. Il primo dei due circuiti integrati ci permette di portare a 3,3V l'alimentazione della scheda di memoria, rendendo così il circuito conforme agli standard delle memorie di questo tipo: lo standard prevede che l'alimentazione possa scendere anche a 2,7V, ma non deve superare i 3,6V. Per la comunicazione, noterete come i segnali che vanno dal microcontrollore della stampante, ovvero MOSI, SCK e il pin per selezionare il dispositivo (CO nello schema), allo slot in cui inserirete la memoria transitano da U1. Questo adattatore di livelli ha dei buffer non invertenti che accettano sugli ingressi - contrassegnati con A - ingressi fino a 15V per il valore logico 1, mentre questo livello sulle uscite - contrassegnate con Y - è limitato alla tensione di alimentazione dell'integrato che, nel nostro caso, arriva dal regolatore da 3,3V.

Il pin (DAT0) su cui la scheda SD invia i dati al microcontrollore secondo il protocollo SPI, (MISO) non ha alcun adattatore di livello perché un livello di 3,3V viene già interpretato dall'elettronica del microcontrollore come un livello logico "1" (sopra la soglia d'incertezza dei 2,5V).

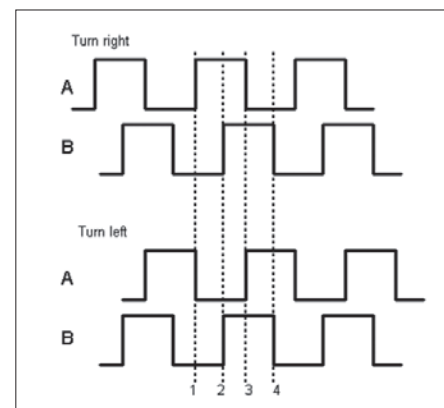
Il componente indicato come Encoder, sullo schema, è nella sua semplicità un dispositivo che ci permette di scorrere il cursore sui menu nelle due direzioni "su" e "giù"; in realtà i due interruttori collegati ai pin 1, 2, e 3 generano impulsi con i fronti di salita sfasati per consentire di capire se la rotazione è oraria o antioraria. La chiusura del contatto fra i pin 4 e 5, generata dalla pressione sull'alberino, è raccolta e



interpretata dal software come il comando di selezione. In pratica questo componente si comporta come la rotella d'interazione montata sulle autovetture di alcune marche tedesche che hanno trovato il sistema "turn and click" molto efficace per scorrere dei menu composti da liste su più livelli.

ENCODER-PUSH: PC0/SCL
ENCODER-RIGHT: PD2/RX1
ENCODER-LEFT: PD3/TX1

Il pulsante di reset è infine collegato al pin di Reset del micro, presente sul connettore a sei pin dell'ISP e necessario in fase di programmazione. Per noi diventa importante al fine di



Lo sfasamento dei segnali di un encoder digitale.

I file di configurazione

Per poter far funzionare il sistema di stampa autonoma è necessario partire dai sorgenti di Marlin, quindi fare le modifiche al file Configuration.h e poi verificare che sia tutto a posto anche in PINS.H (anche se è solo una precauzione). Iniziamo da chi ha una scheda controller di tipo **Sanguinololu**; questa scheda viene indicata in Configuration.h come 62 e dispone già di una serie di impostazioni definite da tempo. Ovviamente i parametri hardware specifici di 3Drag, relativi agli step dei motori, le accelerazioni e il funzionamento dei fine corsa va impostato come descritto nell'articolo di presentazione di 3Drag, mentre per aggiungere questo pannello dovete intervenire su altri parametri che trovate nella sezione "//LCD and SD support".

Fate riferimento agli estratti che trovate in questo box per capire dove intervenire: sono le righe evidenziate in neretto. Se vi sentite "persi", potete sempre recuperare dal nostro sito - 3dprint.elettronica.in - il firmware correttamente impostato per 3Drag con Sanguinololu e attivare le poche righe. Se invece imparate ad apportare tutte le varie modifiche manualmente, potrete cambiare firmware tutte e volte che vorrete in piena autonomia. Riportiamo anche PINS.H per darvi riscontro di come siano mappati i pin di LCD ed Encoder. Dato che il processore sulla board Sanguinololu va cambiato con un ATmega 1284P (dotato della memoria necessaria ad ospitare il codice), abbiamo predisposto il kit con il chip ATmega sostitutivo già programmato, mentre chi vuole fare tutto da sé può far riferimento alle istruzioni che si trovano in rete per programmare il 1284P con un bootloader Arduino per poi poterlo programmare normalmente con il firmware Marlin opportunamente modificato. Se avete una 3Drag V1.1 allora la scheda dispone di un ATmega 2560 e non va sostituito; dovete solo riprogrammarlo con la versione di Marlin modificata per avere SD e display LCD attivati. La nuova scheda di 3Drag V 1.1 è ora supportata direttamente da Marlin. Per essere certi che il sistema di stampa autonoma funzioni anche sulla nuova scheda, indicata nel firmware come "// 77 = 3Drag Controller", dovete necessariamente usare un firmware Marlin successivo al 10 giugno 2013, le versioni precedenti non riportano ancora la nostra scheda e le modifiche sarebbero impegnative. Ricordate che nel firmware vanno sempre impostati in Configuration.h i valori relativi ai microswitch e ai motori; trovate queste informazioni anche su questo numero, in fondo all'articolo di presentazione della nuova 3Drag v 1.1, a pag. 94. L'autore del firmware non ha creato degli automa-

tismi per impostare questi parametri anche perché sono molto "personali", ovvero ogni utente può ad esempio decidere di posizionare un microswitch a inizio o fine corsa, scegliendo la modalità normalmente aperto o normalmente chiuso; il numero di step per i motori dipende sia dai ponticelli scelti per la gestione dei microstep, sia dalle dimensioni fisiche di ingranaggi, pulegge ecc.

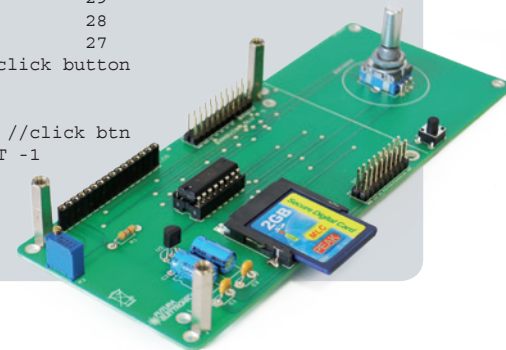
Quanto abbiamo riportato a pag. 94 è quindi la "carta d'identità meccanica ed elettrica" della nostra 3Drag, montata seguendo le nostre indicazioni, da integrare nel firmware per consentirne il corretto funzionamento. Sempre a pag. 94 trovate come attivare la stampa autonoma su 3Drag v 1.1.

Da CONFIGURATION.H (per Sanguinololu)

```
// 62 = Sanguinololu 1.2 and above
...
#define MOTHERBOARD 62
...
// EEPROM
...
#define EEPROM_SETTINGS
...
#define EEPROM_CHITCHAT
...
//LCD and SD support
...
#define SDSUPPORT // Enable SD Card Support in Hardware Console
...
#define ULTIPANEL //the ultipanel as on thingiverse
```

Da PINS.H (per Sanguinololu)

```
#ifndef ULTRA_LCD
#define NEWPANEL
//we have no buzzer installed
#define BEEPER -1
...
#define LCD_PINS_RS 4
#define LCD_PINS_ENABLE 17
#define LCD_PINS_D4 30
#define LCD_PINS_D5 29
#define LCD_PINS_D6 28
#define LCD_PINS_D7 27
//The encoder and click button
#define BTN_EN1 11
#define BTN_EN2 10
#define BTN_ENC 16 //click btn
#define SDCARDDTECT -1
#endif //Newpanel
#endif //Ultipanel
```



"sbloccare" la stampante in caso di crash, errore o altre situazioni in cui un Reset è la via più breve per riprendere il controllo della situazione.

RESET: RESET

Tutti questi collegamenti sono effettuati su pin diversi nel caso si utilizzi la nostra nuova scheda di controllo per 3Drag, versione 1.1.

I collegamenti sono i seguenti:

LCD-RS: PA5
LCD-RW: GND
LCD-E: PA7
LCD-DB4: PC0
LCD-DB5: PC2
LCD-DB6: PC4
LCD-DB7: PC6
SD-CS: PA3
SD-DI: MOSI
SD-SCLK: SCK
SD-DO: MISO

ENCODER-PUSH: PA1
ENCODER-RIGHT: PH1
ENCODER-LEFT: PH0
RESET: RESET

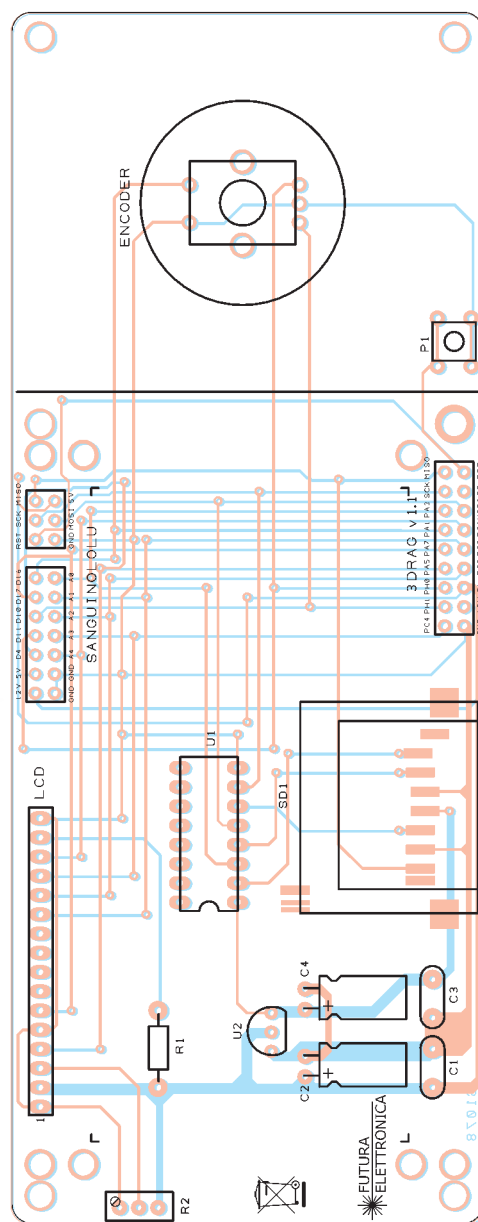
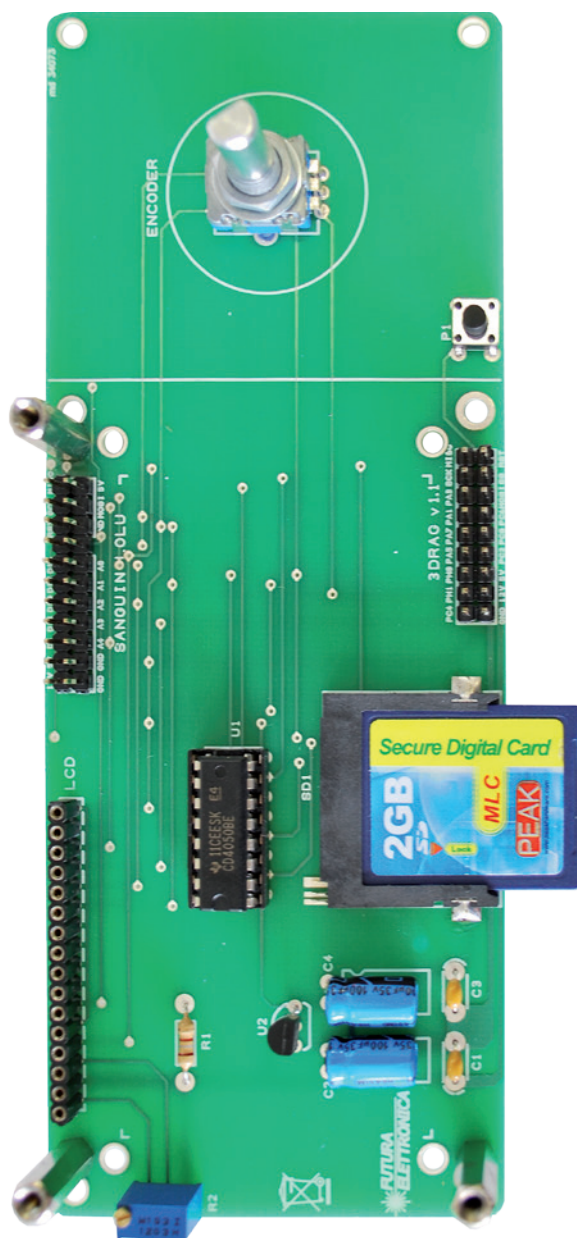
La flessibilità del firmware Marlin permette comunque di rimappare tutti i vari pin nel modo opportuno, modificando il file PINS.H; per vostra comodità,

Elenco Componenti:

R1: 33 ohm
R2: Trimmer multigiri
10 kohm
C1, C3: 100 nF
multistrato
C2: 100 μ F 16 VL
elettrolitico
C4: 100 μ F 16 VL
elettrolitico
P1: Microswitch
SD1: Connettore
SD-Card
U1: 74HC4050
U2: MCP1702-3302E/TO
ENCODER:
Encoder rotativo
(cod. ENCODER12MM)
LCD: Display LCD 20x4
(cod. LCD20X4)

Varie:

- Zoccolo 8+8
- Strip maschio 3 poli
11mm (2 pz.)
- Strip maschio 7 poli
11mm (2 pz.)
- Strip maschio 9 poli
11mm (2 pz.)
- Strip maschio 16 poli
25mm (1 pz.)
- Strip femmina 16 poli
(1 pz.)
- Torretta F/F 20mm
(3 pz.)
- Vite 3MA/8mm testa
conica (6 pz.)
- Flat cable
con connettore universale
- Circuito stampato



sul nostro sito è disponibile il file per la scheda di nostra produzione con le definizioni dei pin gestite attraverso l'aggiunta della definizione di una nuova scheda – la nostra - fra quelle disponibili in Configuration.h.

I pochi componenti passivi che trovate sulla basetta sono condensatori che servono per filtrare l'alimentazione che, pur arrivando dallo stabilizzatore della sche-

da della stampante, passando su un cavo flat potrebbe raccogliere interferenze; il piccolo trimmer R2 che serve per la regolazione del contrasto del pannello LCD, mentre R1 ha il compito di limitare la corrente assorbita dalla retroilluminazione LED.

La scheda SD viene letta tramite lo Slot SD1 da stampato; abbiamo scelto il formato SD e non microSD perché lo riteniamo molto

più robusto fisicamente per un impiego con ripetuti inserimenti ed estrazioni.

Poi, abbiamo tutti in giro per casa una o più schede SD che avanzano perché sostituite dalle ora più diffuse microSD.

Infine, se acquistate una microSD, solitamente viene fornita con un adattatore di formato che ne permette comunque l'uso negli slot per SD.

IL MONTAGGIO

Il circuito stampato che abbiamo predisposto è di tipo a doppia faccia e monta componenti di tipo tradizionale. Potete scaricare come sempre i master dal nostro sito per produrre la basetta autonomamente, oppure potete approfittare delle nostre versioni in kit.

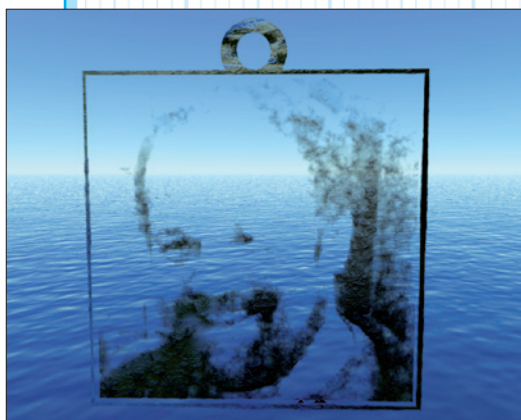
I componenti sono pochi e non ci sono particolari difficoltà nel montaggio, ma come sempre è meglio partire dai componenti passivi più a basso profilo per terminare con i componenti attivi

e i vari strip di collegamento.

La scheda che abbiamo realizzato è compatibile con l'elettronica della nostra prima 3Drag (e con tutte le stampanti che utilizzano la scheda controller "Sanguinololu" fino alla release 1.3a) tramite il connettore dedicato, mentre chi ha una 3Drag con la scheda di nostra progettazione utilizza il connettore indicato con 3Drag V 1.1. In teoria potreste quindi saldare i pinstrip maschio solo del connettore che vi interessa utilizzare, senza montare l'altro. Quando montate il regolatore

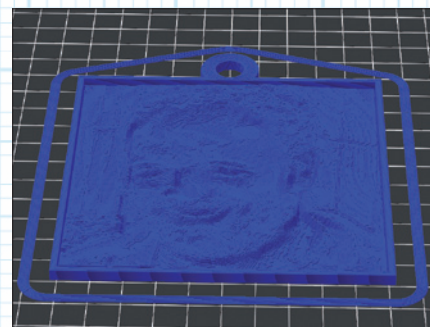
U2, fate attenzione a come lo posizionate e allo stesso modo il circuito integrato U1 a sedici pin, va orientato rispettando la tacca di riferimento che permette di individuare il pin 1. Anche C2 e C4 hanno una polarità e pertanto questa va rispettata. Vi ricordiamo che il reoforo più lungo corrisponde al positivo, mentre la banda continua stampigliata sull'involucro individua il reoforo del negativo. Lo slot per scheda SD è stato scelto fra quelli che hanno il montaggio più semplice, con saldature e terminali che non

Foto a strati - Lithopane



Per finire in bellezza, vi proponiamo una stampa 3D molto particolare. A seconda del materiale utilizzato, si ha l'attenuazione della luce trasmessa molto variabile. Il PLA offre varie gra-

dazioni di attenuazione, iniziando con il neutro, passando dai trasparenti (giallo, rosso, verde, argento, oro, blu) e finendo con quelli opachi (verde nucleare, arancio, rosa, grigio, bianco e nero), con il fosforescente che rappresenta una categoria a sé stante. Questa introduzione serve a capire come funziona la stampa delle fotografie in 3D: stratificando l'immagine con il numero di strati che corrisponde all'intensità del nero in ciascun punto. Il risultato è una stampa 3D che vista direttamente dice poco, ma che illuminata in trasparenza acquista la dinamica della fotografia da cui è stata creata. L'effetto è più facile da mostrare con un'immagine che da spiegare e il risultato è decisamente interessante. Per realizzare le vostre foto 3D (o lithopanes - modalità molto diffusa a fine 19° secolo per "nascondere" immagini di fanciulle poco vestite nei fondi delle tazze da the in porcellana) useremo la sintesi additiva con



una soluzione resa disponibile su Thingiverse (<http://www.thingiverse.com/thing:74322>), mentre chi ha un sistema ad ablazione con fresa (un CNC), probabilmente già conosce questa tecnica; se volete una carrellata di esempi, cercate in Google immagini Lithopane. Con Thingiverse, l'immagine da trasformare in stampa 3D ha 100x100 pixel di risoluzione



PARAMETRI DI STAMPA:

layer_height =	0.18	nozzle_diameter =	0.5
perimeters =	2	filament_diameter =	3
top_solid_layers =	4	extrusion_multiplier =	1
bottom_solid_layers =	4	perimeters extrusion width =	0.40mm
fill_density =	0.4	infill extrusion width =	0.40mm
perimeter_speed =	50	solid infill extrusion width =	0.40mm
infill_speed =	50	top infill extrusion width =	0.40mm
travel_speed =	120	first layer extrusion width =	0.60mm

richiedono attrezzi specifici o per SMD e il tutto dovrebbe essere pronto per l'uso in poco tempo.

PREPARAZIONE DEL SOFTWARE

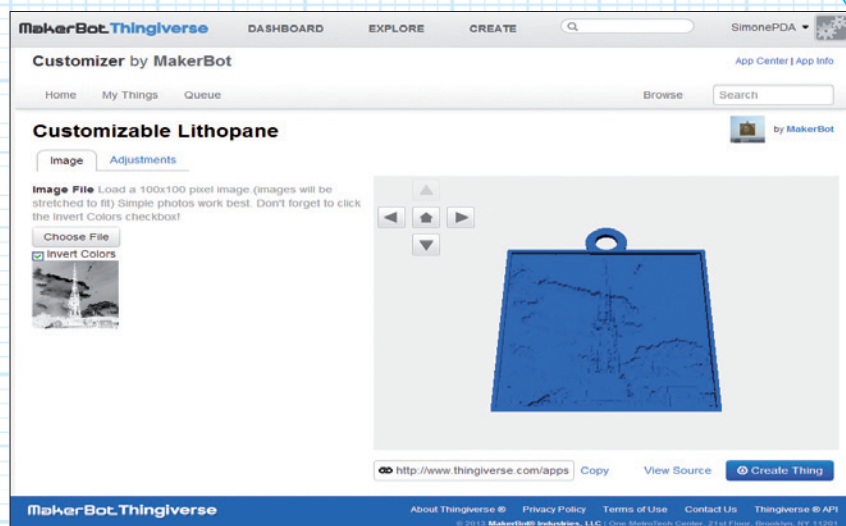
Questo pannello di controllo per la stampa autonoma non contiene di per sé le funzionalità che servono allo scopo: la memoria contiene i file, ma deve essere letta e il display mostra quello che gli viene inviato. Le reali funzionalità sono contenute dal firmware della scheda e nello specifico dal firmware Marlin. Tra librerie e codice aggiuntivo,

però, si supera il limite dei 64K di memoria disponibile per il codice sugli ATmega 664P e quindi il processore montato dalle schede Sanguinololu – incluse quelle montate sulla prima serie delle nostre 3Drag – non è adatto a far funzionare oltre a tutto il resto anche la parte di SD e LCD. Tutto questo è ampiamente documentato dall'autore del firmware e in effetti il problema è facilmente risolto passando a una versione di ATmega più capiente, ma che per nostra fortuna resta "pin to pin" compatibile. Il mi-

crocontrollore 1284P è quello più indicato per la sostituzione sulle schede Sanguinololu, mentre la nostra board 3Drag v 1.1 già dispone di un ATmega 2560 che ha addirittura 256 KB di memoria disponibile per il firmware. Come sapete, le RepRap sono nate partendo da un Arduino MEGA e poi si sono evolute con schede dedicate che, però, conservano la compatibilità con Arduino ed il suo IDE. Grazie a questo è possibile riprogrammarle con grande facilità purché il micro abbia già a bordo il bootloader. Diversamente bisogna utilizzare la programmazione tramite i pin ISP (In System Programming) del micro, a cui vanno collegati quelli di un Arduino come documentato in questo tutorial (<http://arduino.cc/en/Tutorial/ArduinoISP>).

In generale, Marlin non è comunque compilato con la parte di LCD e controllo tramite encoder attivi e per questo è necessario scaricare dal sito dell'autore l'ultima versione e procedere alla sua personalizzazione con i parametri di 3Drag nel file "Configuration.h" e attivando la parte di LCD e scheda SD tramite il relativo "#define". Trovate tutte le modifiche e le impostazioni nel box dedicato e comunque abbiamo predisposto sul nostro sito le due versioni del firmware per Sanguinololu con 1284P e per 3Drag V. 1.1 pronte all'uso. Collegiamo il cavo.

Alimentazione e linee dati passano su un unico cavo di tipo flat con due connettori agli estremi. Per chi ha una scheda Sanguinololu, serve quello con il connettore da 22 poli (due file da 11), mentre per chi ha la scheda 3Drag v 1.1 è necessario il cavo flat con connettore a 18 poli (due file da 9). A stampante spenta potete collegare il cavo da entrambe

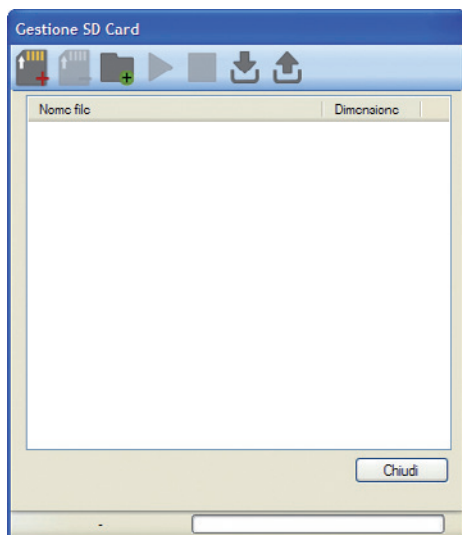


e la procedura provvede a ridimensionare quello che caricherete a questa dimensione (attenzione alle proporzioni!). Il file generato è un STL per la stampa di un pannello da 10,3 x 10,3 cm, spesso 3,8 mm. Per consentirvi di avere subito una prova abbiamo prodotto un file scegliendo l'immagine di Massimo Banzì, scattata in occasione del tour per la presentazione della Maker Faire che si terrà a Roma il prossimo ottobre. La stampa in assoluto migliore l'abbiamo ottenuta utilizzando il PLA fosforescente: grande omogeneità di estrusione – a circa 192 gradi – e soprattutto

una relativa opacità degli strati che ci ha permesso di avere un risultato con un'ottima dinamica. Voi sperimentate con i vari PLA che avete a disposizione per trovare quello che rende meglio. Lo spessore di stampa a 0.18 è quello che ci ha dato risultati migliori, mentre la densità di fill e il numero di perimetri offre risultati diversi. Se userete il nostro file in G-Code, modificate a mano o nel file le temperature per il PLA che userete.

Nome File:
Massimo_Glow_Lithoplate.gcode

*Repetier Host: il pannello
dell'SD card Manager.*



le parti facendo in modo che ci sia corrispondenza fra i pin. In pratica seguite il filo contrassegnato su un lato del cavo flat e fate in modo che se questo è dal lato di MISO e 5V (Sanguinololu), sulla scheda di stampa autonoma sia dal medesimo lato. Il medesimo ragionamento vale per la scheda 3Drag v 1.1.

COSA CAMBIA?

Con l'attivazione della parte relativa alla scheda SD, Marlin aggiunge comandi e funzioni al suo già ricco arsenale. Queste nuove funzioni hanno lo scopo di consentire la gestione dei file anche da computer, senza dover estrarre la scheda per aggiungere o modificare i file che poi saranno stampati. Con la memoria a bordo, il firmware diventa poi in grado di accodare vari lavori, comunicando sempre con il software di stampa attraverso specifici messaggi. Repetier-Host – il programma che noi utilizziamo e consigliamo per la gestione della stampante – dispone di un pannello di controllo specifico, che si trova sotto la voce "Stampante -> SD Card Manager". Con questo pannello è possibile interagire in

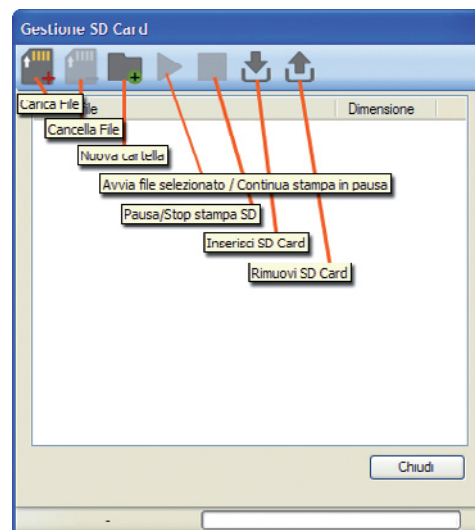
modo specifico con la stampante che, con una SD card inserita, dispone della memoria locale necessaria a "bufferizzare" uno o più file. Proprio per questa possibilità di scaricare il file, lanciare la stampa e poi scollegare il computer, Repetier-Host offre la possibilità di aggiungere comandi G-Code all'inizio e alla fine del file da stampare; con questa funzionalità diventa quindi semplice prevedere lo spegnimento dei vari riscaldatori, il posizionamento in un punto preciso e poi la disattivazione dei motori, riducendo a fine stampa l'assorbimento della stampante. Si tratta solo di un esempio, basato sulle nostre esperienze pratiche, ma la serie di comandi disponibile con il firmware Marlin potrà ispirarvi altre soluzioni d'utilizzo.

Il pannello mostra, se è presente una scheda SD, tutti i file che è possibile stampare (con ".g" quale estensione) e nasconde invece i file non stampabili e di sistema. Il file system gestito dalla libreria consente l'impiego di nomi nello standard 8.3 con sub directory nelle quali è possibile entrare e uscire cliccando opportunamente sulle etichette mostrate nell'area della lista dei file e delle cartelle. I file già presenti possono essere cancellati o avviati come stampa, mentre dal pannello è possibile creare una cartella o caricare un file da stampare successivamente. Sono presenti i due pulsanti a triangolo (PLAY) e a quadrato (STOP) con i quali si può avviare o riavviare la stampa e mettere in pausa o fermare la stampa.

Le ultime due icone servono per chiudere i file e permettere la rimozione della SD card o per far leggere una card appena inserita. Provare a utilizzare le funzioni di questo pannello software è un buon collaudo dell'effettivo funzionamento del firmware, ma

siamo abbastanza certi che siate più ansiosi di passare all'utilizzo della stampante direttamente senza alcun collegamento. Se avete problemi, saltate al capitolo "Problemi?" e poi ritornate qui. Prima di fare stampe dalla scheda SD, dovete comunque preparare uno o più file su di essa e solo allora potrete lanciare le stampe.

Nel caricare i file, ricordate di osservare la struttura 8.3 ovvero otto caratteri al massimo per il nome, un "." e poi al massimo tre caratteri per l'estensione, ma nel nostro caso questa deve essere ".g" per i file che vorremo stampare (ad esempio "serpente.g"). Se poi il nome del file è di tipo "auto<0-9>.g" allora potrete utilizzare la stampa con l'avvio automatico in sequenza ordinata (se ci sono più file, si parte da quello con il numero più basso e poi gli altri a seguire). Gli utenti più "esperti", sapendo che all'accensione la stampante esegue questi file, può predisporre uno solo che effettua delle operazioni tramite G-Code scritto a mano e che prepara la stampante alle sue normali attività (ad esempio si



Le funzioni delle varie icone dell'interfaccia SD Card Manager di Repetier Host.

do l'encoder, si clicca, si ruota la manopola per variare il valore, quindi si clicca nuovamente per uscire dal cambio del valore e tornare alla modalità di selezione delle voci della lista. I cambi di valore sono in tempo reale: come si cambia il valore di un asse, ad esempio, si aziona istantaneamente il motore.

Non mancano le funzioni per mettere in pausa la stampa ed accedere ai comandi di spostamento – utili ad esempio per cambiare il colore del filo di stampa.

SD CARICA? VIA ALLA STAMPA!

Se a questo punto avete preparato un file adatto alla vostra prima stampa di prova ed è sulla scheda SD presente nello slot del pannello aggiuntivo, potete procedere alla vostra prima vera stampa autonoma. Iniziate con l'accensione della stampante senza collegamento al PC, quindi quando apparirà la scritta "Printer ready..." potete inserire la scheda SD e vedere se sul display appare "Card inser-

ted". A questo punto con un clic entrate nel menu principale, dal quale scegliete "card menu" e successivamente il vostro file da stampare. Prima di iniziare la stampa, potreste voler sperimentare un po' con i comandi di controllo (temperatura, spostamento ecc.) e questo non può che aumentare la familiarità con le funzioni disponibili.

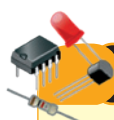
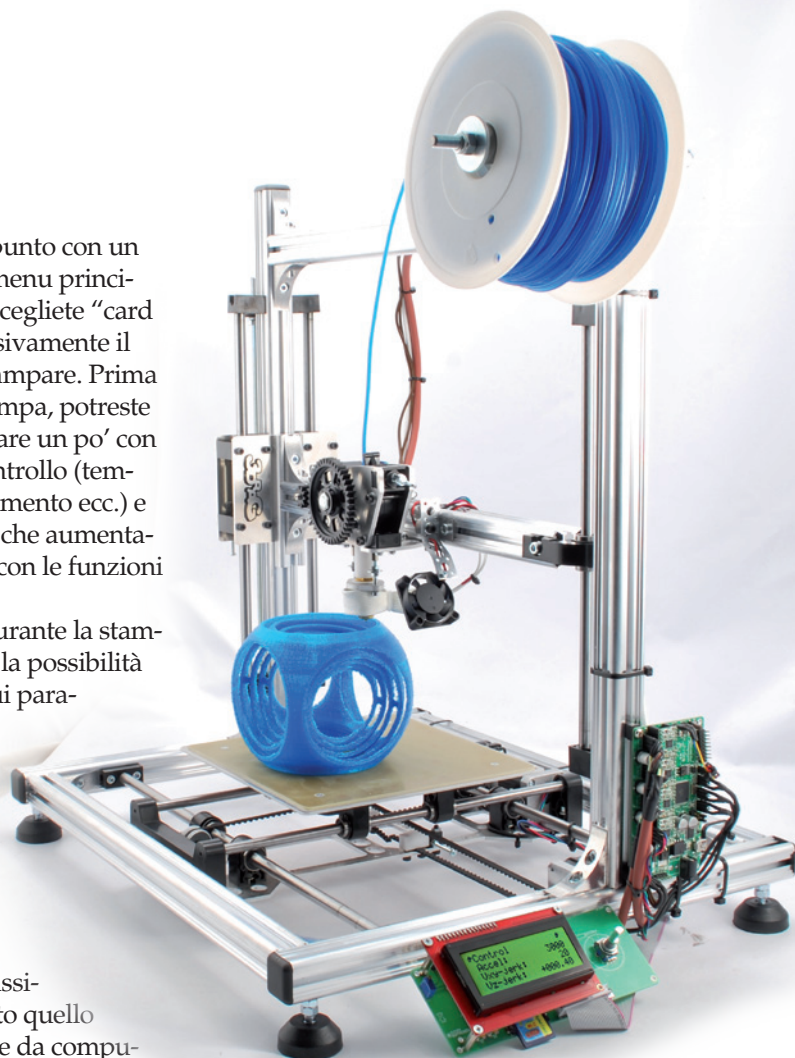
Ricordate che durante la stampa avete ancora la possibilità di intervenire sui parametri di stampa così come potete fare da Repetier-Host. Per motivi pratici e di com'è costruita l'interfaccia, vi assicuriamo che tutto quello che potevate fare da computer per controllare la stampante è ancora possibile tramite LCD, ma la posizione dei parametri e dei comandi segue una logica specifica. Fate riferimento alla mappa dei menu e confrontatela sia con la scheda dei parametri della EEPROM (su Repetier-Host), sia con le varie funzioni del "Controllo Manuale", sempre di Repetier.

OPEN SOFTWARE & HARDWARE!

Vogliamo chiudere l'articolo evidenziando che sia la parte software, sia la parte elettronica sono stati sviluppati da vari autori che hanno partecipato alla community di RepRap con il loro lavoro. Vogliamo quindi manifestare il nostro apprezzamento per la qualità raggiunta che è oggi anche superiore ad alcune soluzioni commerciali; questo dimostra che rimettere in circolo le proprie modifiche e migliorie, discutendo con la community e senza segreti o brevetti porta –

nelle situazioni giuste – ad una crescita reale e consistente del materiale. Se non avete ancora letto l'intervista all'autore di Slic3r, adesso è proprio il momento di farlo per capire meglio cosa intendiamo.

Come abbiamo indicato nell'articolo di apertura di questo "focus" sulla stampa 3D, stiamo utilizzando le versioni più recenti, ma è molto probabile – e anche desiderabile – che queste vengano presto superate da release successive. È ragionevole supporre che l'hardware non sarà da modificare, mentre i firmware e i software quasi certamente acquisiranno funzionalità nuove e diverse che non abbiamo potuto ovviamente documentare. Controllate quindi i siti degli autori e leggete le note di rilascio per decidere se è il caso di aggiornare o restare a quello che avete imparato ad utilizzare. ■



per il MATERIALE

Il kit completo per la stampa autonoma da collegare alla versione 1.1 della stampante 3Drag è disponibile al prezzo di 52,00 Euro (cod. FT1078K). La scatola di montaggio comprende la basetta, tutti i componenti, il display e il cavo flat; non è compresa la scheda SD. Il kit per la 3Drag versione 1.0 (che utilizza la scheda Sanguinololu) costa invece 61,00 Euro (cod. FT1078SK) e contiene in più un chip ATmega 1284PU completo di bootloader. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:

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

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

OpenWheels

IL FIRMWARE

di MIRCO SEGATELLO

Ultima puntata dedicata ad alcuni approfondimenti per capire come sono state implementate le funzioni nello sketch di OpenWheels.



Gia prima di iniziare a sviluppare il progetto del nostro veicolo personale sorsero dei seri dubbi sull'effettiva capacità che il piccolo microcontrollore Atmel programmato con l'IDE di Arduino fosse in grado di reggere il gravoso compito della stabilizzazione. Dubbi che furono fugati dopo le prime prove che dimostrarono come, per la stabilizzazione, in realtà non sono necessarie molte righe di programma; casomai le difficoltà arrivano successivamente quando, oltre alla stabilizzazione, è necessario gestire contemporaneamente l'acquisizione dei segnali dai vari sensori, i dati della telemetria, il display nonché implementare tutta una serie di sistemi di sicurezza. In questo arti-

colo spieghiamo in dettaglio alcuni accorgimenti che abbiamo adottato per implementare tutte le funzioni necessarie nello sketch di OpenWheels. La stabilizzazione viene determinata tramite un controllo PID in cui il parametro proporzionale prende in considerazione il valore dell'angolo, il parametro derivativo tiene conto della misura del giroscopio ed il parametro integrativo viene determinato tramite un accumulatore matematico; il tutto secondo l'equazione base:

$$drive = P \cdot Angle + I \cdot Accumulator + D \cdot Gyro$$

Complessivamente il controllo PID è formato da poche righe di codice:

```
void PID()
{
  //Implement PID
  Pval = float(KP) * Angle;
  Accumulator = Accumulator + Angle*0.01;
  Accumulator = constrain(Accumulator, -100, 100);
  Ival = float(KI) * Accumulator;
  Dval = float(KD) * Gyro_Rate;
  drive = (Pval + Ival + Dval)/10;
}
```

La variabile Gyro_Rate è determinata direttamente dalla lettura del giroscopio mentre la stima dell'angolo di inclinazione viene calcolata tramite il filtro complementare basato sulle misure Angle_Rate e Gyro_Rate, come già descritto nel precedente articolo.

```
void AccAngleMeasure()
{
    Acc_RAW=0;
    for (i=0; i<10; i++)
    {
        Acc_RAW += analogRead(Acc_ADC_pin);
        delayMicroseconds(500);
    }
    Acc_RAW /= i;
    Acc_Angle = float(Acc_Zero-Acc_RAW)*Acc_Scale;
}

void GyroRateMeasure()
{
    Gyro_RAW=0;
    for (i=0; i<10; i++)
    {
        Gyro_RAW += analogRead(Gyro_ADC_pin);
    }
    Gyro_RAW /= i;
    Gyro_Rate = float(Gyro_Zero-Gyro_RAW)*Gyro_Scale;
}

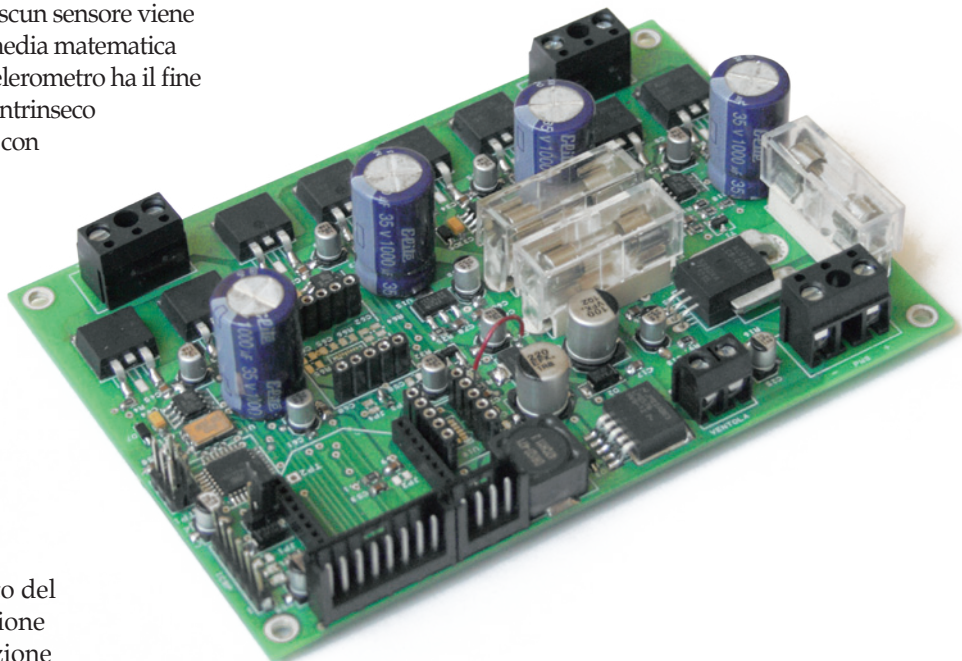
void EstimateAngle()
{
    // Use complementary filter to fusion acc e gyro data
    float Comp_Filter_Constant = 0.03;
    float dt=1/Cycle_Freq; // time for cycle
    Angle = (1-Comp_Filter_Constant)*(Angle+Gyro_Rate*dt) +
    (Comp_Filter_Constant * Acc_Angle);
    Angle = constrain(Angle, -30, 30);
}
```

Notate dai listati che la lettura di ciascun sensore viene eseguita tramite un'operazione di media matematica molto marcata che nel caso dell'accelerometro ha il fine di ridurre il più possibile il rumore intrinseco del sensore. Nella stima dell'angolo con l'algoritmo del filtro complementare, di particolare rilevanza riveste il valore *Comp_Filter_Constant* che imposta il punto di intervento dei due filtri. Un valore più basso rende la misura più pulita ma molto lenta mentre un valore più alto rende la misura più veloce ma più sensibile al rumore dell'accelerometro. La variabile *drive* non viene usata direttamente nella gestione dei motori perché occorre tenere conto del livello della batteria, dell'inclinazione del manubrio (sterzo) e della funzione

soft-start che evita bruschi movimenti al mezzo.

Il controllo del driver dei motori avviene tramite due segnali PWM, più un controllo di ShutDown; quest'ultimo pone i drive in modalità stand-by a ridotto consumo di energia. La classica funzione di Arduino chiamata *AnalogWrite()* non può essere usata a causa di due rilevanti limitazioni: la prima è dovuta al fatto che la frequenza del PWM è fissa e non modificabile; la seconda riguarda la risoluzione che è di soli otto bit. Per un preciso controllo dei nostri motori è opportuno che venga impostata la corretta frequenza del PWM e che la risoluzione sia di almeno nove bit. Come già spiegato nel precedente articolo, dovremmo far lavorare i motori con una segnale PWM della frequenza di 16 kHz, valore ottenuto anche da considerazioni sperimentali a seguito di numerose prove.

A questa frequenza il drive presenta il migliore rendimento possibile, i motori risultano molto reattivi e non vengono emessi fastidiosi sibili durante il funzionamento. L'efficienza è così elevata che i MOSFET finali del driver in formato SMD non hanno dissipatore e durante il funzionamento risultano appena tiepidi. Abbiamo quindi bisogno di una elevata flessibilità nella gestione del segnale PWM, ottenibile solo agendo direttamente sui registri interni del micro che comandano il modulo PWM. Richiamiamo quindi la procedura *initMotor()* per impostare i registri interni dell'Atmel denominati ICR1, TCCR1A e TCCR1B facenti parte del TIMER1 di sistema in modo venga attivata la modalità *phase and frequency correct mode* con *prescaler* a 1 e *mode select* a 8. La frequenza del PWM può essere modificata agendo sul valore



Listato 1

```
void initMotor()
{
    pinMode(ShutDown_pin, OUTPUT);
    digitalWrite(ShutDown_pin, LOW);    //ShutDown driver at startUP

    pinMode(PWMLeft_pin, OUTPUT);
    pinMode(PWMRight_pin, OUTPUT);

    // Phase and Frequency correct PWM on TIMER1
    //PWM_frequency = (16000000)/(2*Prescale_factor*ICR1)
    //ICR1=(16000000)/(2*Prescale_factor*PWM_frequency)

    TCCR1A = _BV(COM1A1) | _BV(COM1B1) ; // phase and frequency correct mode. NON-inverted mode
    // TCCR1A = _BV(COM1A1) | _BV(COM1B1) | _BV(COM1A0) | _BV(COM1B0) ; //phase/frequency correct mode.
    TCCR1B = _BV(WGM13) | _BV(CS10); // Select mode 8 and select divide by 1 on prescaler
    ICR1=500;

    // limit for driver compatibiliti (5% di ICR1A) < drivePWM < (95% di ICR1A)
    // stop motor driver=stallMotor=ICR1/2;

    maxPWM = float(ICR1)*0.95/2;
    stallMotor=ICR1/2;    //250
    OCR1A = stallMotor;    //0=0% ICR1/2=50% ICR1=100%
    OCR1B = stallMotor;
}
```

Listato 2

```
void SetMotor()
{
    drivePWM = int(drive * goStart);    // 0 in fase di standby
    drivePWM = constrain(drivePWM, -maxPWM, maxPWM);
    dutyCycleLeft = stallMotor + drivePWM + Steer;
    dutyCycleRight = stallMotor + drivePWM - Steer;
    dutyCycleLeft = constrain(dutyCycleLeft, stallMotor-maxPWM, stallMotor+maxPWM);
    dutyCycleRight = constrain(dutyCycleRight, stallMotor-maxPWM, stallMotor+maxPWM);
    OCR1B = dutyCycleLeft;    //set PWM SX 250+-250
    OCR1A = dutyCycleRight;    //set PWM DX 250+-250
}
```

del registro ICR1 (16 bit counter) secondo la formula riportata nel listato. Impostando questo registro con il valore a 500, la frequenza del PWM è esattamente di 16 kHz assicurando una risoluzione di circa nove bit. È possibile modificare questo valore agendo sul registro ICR1 ma di fatto ciò varierebbe anche la risoluzione con la quale viene gestito il segnale PWM; ad esempio, un valore di ICR1 pari a 250 porterebbe la frequenza del PWM a 32 kHz però con una risoluzione di soli otto bit; per contro frequenze più basse permetterebbero risoluzioni maggiori. Per impostare il *duty-cycle* del PWM è necessario agire sui registri OCR1A e OCR1B; notate come il valore del *duty-cycle* viene limitato tra un minimo del 5% ed un massimo del 95% in accordo con le funzionalità del driver di potenza basato sulla tecnologia a pompa di carica (*bootstrap operation*) (**Listato 1**). La funzione *SetMotor()* provvede infine ad impostare il corretto valore del *duty-cycle* del PWM considerando la funzione *softstart* ed il valore dello sterzo derivato dalla lettura del potenziometro sul piantone (**Listato 2**). Tutte le funzioni fino ad ora descritte devono essere eseguite ad intervalli regolari ed hanno la priorità su qua-

lunque altra operazione: è quindi necessario impostare un *interrupt* interno agganciato al *clock* di sistema. Per questa funzione viene impostato il TIMER2 affinché richiami ad intervalli regolari la funzione *ISR(TIMER2_COMPA_vect)* che riporta le istruzioni da eseguire. Ecco quindi il loop principale che scandisce tutte le operazioni del firmware:

```
ISR(TIMER2_COMPA_vect)
{
    bitSet(PORTB, 5);
    SoftStart();
    AccAngleMeasure();
    GyroRateMeasure();
    EstimateAngle();
    Steering();
    Status();
    PID();
    SetMotor();
    SetZero();
    Telemetry();
    bitClear(PORTB, 5);
}
```

L'attivazione del pin 5 del registro PORTB corrisponde al LED13 di Arduino e permette, con un oscilloscopio,

Listato 3

```
void Init_Timer()
{
    //freq_INT = 16000000/(Prescaler * OCR2A)
    //OCR2A = 16000000/(Prescaler * freq_INT)

    int Count = 15625/Cycle_Freq;

    cli(); // disable global interrupts
    TCCR2A = 0; // set TCCR2A control register to 0
    TCCR2B = 0; // set TCCR2B control register to 0
    TCCR2A = _BV(WGM21);
    TCCR2B = _BV(CS20) | _BV(CS21) | _BV(CS22); //set prescaler to 1024
    OCR2A = Count; // Set compared value
    ASSR &= ~(1 << AS2); // Use system clock for Timer/Counter2 AS2=1
    TIMSK2 = 0; // Reset Timer/Counter2 Interrupt Mask Register
    TIMSK2 = _BV(OCIE2A); // Enable Output Compare Match A Interrupt
    sei(); // enable global interrupts
}
```

di misurare accuratamente il tempo di ciclo e verificare che non ci siano operazioni concomitanti.

Il TIMER2 è il timer di sistema ad otto bit configurabile agendo sui registri di nome TCCR2A, TCCR2B, OCR2A. Il suo funzionamento è molto semplice e si basa sull'incremento del registro OCR2A ad intervalli regolari pre-impostabili tramite il valore del *prescaler*; quando il registro eguaglia il valore della variabile *count* viene generato automaticamente un'interruzione al programma principale e richiamata la funzione

ISR(TIMER2_COMPA_vect) (Listato 3).

La frequenza di ciclo viene indicata con il valore *Cycle_Freq* (default=65.0 Hz) dalla quale è ricavato il valore *Count=15625/Cycle_Freq* usato per l'interrupt. Dalle misurazioni eseguite sappiamo che il tempo necessario per eseguire tutte le operazioni è di 9,4 msec e quindi la massima frequenza di ciclo potrebbe essere di 106 Hz. Dobbiamo però considerare il tempo necessario per la gestione della telemetria e del display LCD ed i 65Hz da noi scelti rappresentano un ottimo compromesso tra la velocità di gestione delle operazioni per il bilanciamento e le risorse disponibili.

A complemento alle funzioni principali ne troviamo altre quali, ad esempio, la gestione della memoria EEPROM, usata per salvare le impostazioni di configurazione, compresi gli offset dei sensori ed i valori del PID. Il compilatore di Arduino mette a disposizione le funzioni *EEPROM.read* e *EEPROM.write* che però funzionano solo con valori di tipo byte e dovendo noi salvare e leggere anche variabili a sedici bit è necessario ridefinire nuove funzioni:

```
int eepromReadInt(int address){
    int value = 0x0000;
    value = value | (EEPROM.read(address) << 8);
    value = value | EEPROM.read(address+1);
    return value;
}
```

```
void eepromWriteInt(int value, int address){
    EEPROM.write(address, (value >> 8) & 0xFF);
    EEPROM.write(address+1, value & 0xFF);
}
```

Per quanto riguarda la telemetria abbiamo previsto alcune funzioni che si attivano quando viene ricevuto uno specifico carattere; in questo modo è l'utente esterno che chiede i dati di OpenWheels definendo di fatto la cadenza di ricezione.

```
void Telemetry()
{
    if (Serial.available())
    {
        switch (Serial.read())
        {
            case 'A':
                // Arduino to GUI Telemetry data
                Serial.write('A');
                sendSerialInt(Acc_RAW);
                sendSerialInt(Gyro_RAW);
                sendSerialInt(int(Angle*10));
                sendSerialInt(int(IBatLevelSmooth*10));
                sendSerialInt(int(drive));
                sendSerialInt(statusFlag);
                sendSerialInt(int(VBatLevel*10));
                sendSerialInt(Steer);
                break;

            case 'E':
                Serial.write('E');
                sendSerialInt(KP);
                sendSerialInt(KI);
                sendSerialInt(KD);
                break;

            case 'R':
                // Arduino to PC PID parameters
                Serial.print("KP=");
                Serial.print(KP);
                Serial.print(" KI=");
                Serial.print(KI);
                Serial.print(" KD=");
                Serial.print(KD);
                Serial.println();
                break;

            case 'W':
```



```

// GUI to Arduino PID parameters
index=0;
while (Serial.available()>0)
{
    inByte[index] = Serial.read();
    index++;
}
KP=inByte[0];
KI=inByte[1];
KD=inByte[2];
break;

case 'S':
    // Save PID parameter on Arduino EEPROM
    EEPROM.write(KP_ADR, KP);
    EEPROM.write(KI_ADR, KI);
    EEPROM.write(KD_ADR, KD);
    break;

case 'Z':
    // Set Zero Acc & Gyro
    setZeroCount = 50;
    AccZeroSum = 0;
    GyroZeroSum = 0;
    break;
}
}
}

```

Notate come tutte le variabili sono trattate in formato intero ed i valori float sono convertiti in interi dopo essere stati moltiplicati per 10. Questo permette di ottimizzare il software e di rendere le operazioni molto più veloci; ricordiamo che Arduino usa un microcontrollore ad 8 bit privo di processore matematico specifico per le operazioni in virgola mobile (**Tabella 1**).

La telemetria è stata pensata per funzionare assieme al software per PC denominato OpenWheelsGUI con il quale è possibile gestire anche grafici, mentre un sistema di telemetria basato su SerialMonitor di Arduino sarebbe stato troppo limitato.

Un'altra particolarità del firmware è l'utilizzo delle funzioni bitSet() e bitClear() - al posto della classica digitalWrite() - in quanto decisamente più veloci. Per rendersene conto è sufficiente andare ad analizzare il

Tabella 1 - Le funzioni implementate.

Carattere ricevuto	Funzione svolta
A	Invio dati di telemetria (formato HEX)
E	Invio parametri PID (formato HEX)
R	Invio a serial monitor dei parametri PID (formato stringa)
W	Ricezione configurazione PID (formato HEX)
S	Salva parametri PID in memoria
Z	Avvia procedura rilevamento offset sensori

file wiring_digital.c disponibile nella cartella di Arduino e contenenti le istruzioni del compilatore per ciascuna funzione implementata. Per quanto riguarda la gestione del display LCD diciamo che non ci sono particolarità significative ed essendo il codice molto lungo non lo riportiamo. Non avendo a disposizione molte linee digitali, il display è necessariamente di tipo seriale e gestito con una sola linea dati tramite la libreria di sistema denominata *SoftwareSerial*. Non vi è neppure la necessità di ottimizzare troppo il codice o renderlo troppo veloce perché è continuamente interrotto dall'interrupt che controlla la stabilizzazione. In questa descrizione sono state omesse alcune parti complementari come ad esempio la gestione degli errori anch'essa richiamata ad intervalli regolari dall'interrupt principale, cioè 65 volte al secondo. Questa procedura analizza i valori ottenuti da tutti i sensori e se questi non rientrano entro determinati valori viene avviata immediatamente la procedura di allarme, con conseguente spegnimento del driver dei motori; possiamo quindi contare su un tempo di intervento non superiore ai 15 msec per la determinazione di un'anomalia. Tra gli stati di allarme quello che si è verificato più frequentemente durante le prove è sicuramente il superamento dei limiti di assorbimento dei motori che si riscontra durante una salita e nel superamento di un piccolo ostacolo.

Si conclude qui la descrizione del progetto OpenWheels: rimandiamo al sito dedicato (openwheels.elettronica.in.it) per ulteriori aggiornamenti e sviluppi futuri. ■

RM ELETTRONICA

Forniture per hobbisti, scuole e industria

Distributore autorizzato:

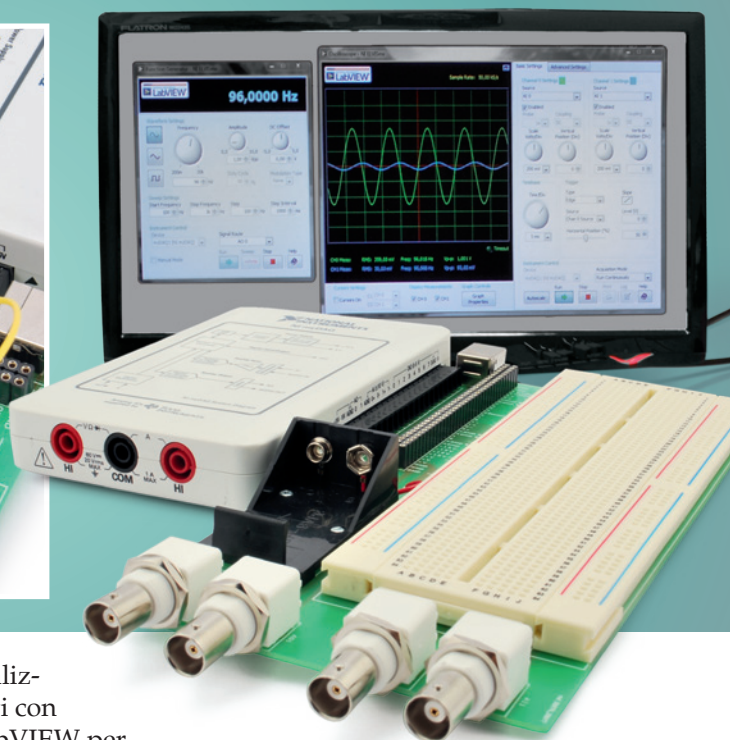
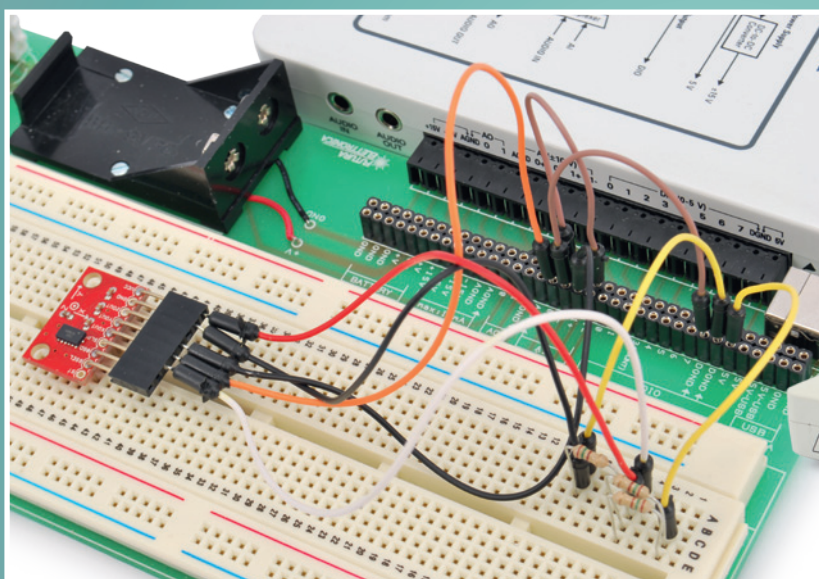
Il tuo punto
di riferimento
per l'elettronica
a ROMA

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

myKIT, MISURA DI ACCELERAZIONE CON SENSORE TRIASSALE

È uno dei numerosi esperimenti per NI myDAQ presenti nel set myKIT; nello specifico questo esperimento si propone di misurare l'accelerazione mediante un dispositivo MEMS a tre assi.

di DAVIDE SCULLINO



Quando (fascicolo n. 171) abbiamo introdotto la protoboard e spiegato come utilizzarla insieme a NI myDAQ per compiere vari tipi di misure e simulazioni su circuiti analogici e digitali, avevamo promesso che avremmo proposto, nei fascicoli successivi della rivista, un esperimento tra quelli facenti parte del programma didattico di myKIT, il set per esperimenti che, utilizzando la protoboard e NI myDAQ, e sotto la guida di complete dispense didattiche, consente di testare in pratica programmi

realizzati con LabVIEW per acquisire dati da svariati tipi di sensore, gestire trasduttori, analizzare il funzionamento di circuiti come i filtri elettronici e tanti altri ancora. Ebbene, ecco questo mese un esperimento decisamente interessante, la misura di accelerazione tramite NI myDAQ, la protoboard ed un accelerometro a tre assi (x, y, z) low cost. I dati

rilevati vengono poi acquisiti con il DAQ Assistant che viene installato in LabVIEW mediante il driver NI DAQmx.

L'accelerometro che utilizzeremo è un Freescale MMA7361L collegato ai terminali d'ingresso analogici di NI myDAQ, gestito mediante LabVIEW. Naturalmente per seguire con noi l'esperimento dovreste essere in possesso del kit didattico myKIT.

L'ACCELEROMETRO

Prima di iniziare, è utile, a beneficio di chi non conosce gli accelerometri, spiegare rapidamente cosa sono e che tipo di segnale forniscono questi componenti elettronici; ebbene, un moderno accelerometro è un dispositivo MEMS (Micro ElectroMechanical System) che combina su un chip a semiconduttore sia componenti elettronici, sia dispositivi elettromeccanici dotati, ad esempio, di piccole masse la cui inerzia determina segnali elettrici misurabili all'interno dei chip. Gli accelerometri possono rilevare movimenti su due o tre assi e tipicamente si impiegano per misurare l'accelerazione e, tramite essa, la forza; rilevano anche l'accelerazione di gravità. Quelli a tre assi vengono utilizzati, ad esempio, per misurare nelle automobili le forze G (rilevano la decelerazione a seguito dell'urto in un incidente o l'accelerazione in modo da passare i segnali ai sistemi ESP e TCS che ottimizzano la trazione in funzione dell'accelerazione richiesta dal conducente); negli aerei vengono utilizzati nelle unità di controllo del volo -ADIRU- per registrare l'assetto (rollata, picchiata/cabrata, virata). Gli accelerometri vengono utilizzati anche nei pedometri, nei controller per videogiochi (rilevano i movimenti del giocatore), nelle applicazioni robotiche (verificano l'assetto del robot e il movimento degli arti nei robot umanoidi) e in molti altri ambiti.

L'accelerometro usato in questo esperimento è il circuito integrato MMA7361L (Fig. 1), che si trova già saldato su una breakout board che ne rende più facile l'utilizzo; infatti il chip è incapsulato in un case per SMD e la basetta su cui viene montato

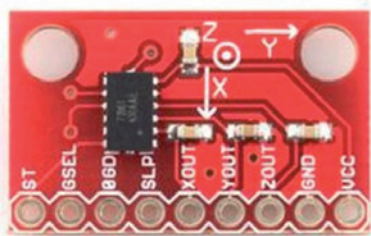


Fig. 1 - L'accelerometro impiegato nell'esperimento.

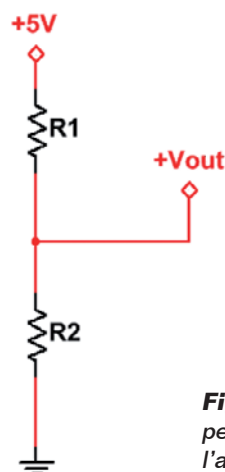


Fig. 2 - Il partitore per alimentare l'accelerometro.

consente di avere a disposizione i contatti su un più comodo layout con pad a passo standard di 2,54 mm. Il piccolo circuito stampato che ospita il sensore include i pochi componenti necessari al corretto funzionamento dell'accelerometro.

L'integrato della Freescale fornisce le informazioni sulle accelerazioni rilevate lungo i tre assi mediante tre tensioni analogiche, prelevabili dai contatti XOUT (accelerazione sull'asse X) YOUT (accelerazione sull'asse Y) e ZOUT (accelerazione sull'asse Z); ciascun potenziale va da quello di massa (0 V) ad un valore pari a circa metà tensione di alimentazione.

ABBINARE L'ACCELEROMETRO A PROTOBOARD E NI myDAQ

L'accelerometro triassiale usato nell'esperimento deve essere alimentato con una tensione di 3,3 V, che potete ricavare mediante un partitore di tensione collegato tra la porta +5V e GND di NI myDAQ, secondo lo schema mostrato nella Fig. 2.

Le resistenze componenti il partitore devono essere in rapporto 1:2, il che permette di avere una tensione di uscita (rilevata ai capi di R2) di 3,3 V. Nel nostro caso vanno bene una R1 da 75 ohm ed una R2 da 150 ohm.

Essendo alimentato a 3,3 volt, l'accelerometro fornisce per ciascun asse una tensione analogica che può spaziare tra 0 e 3,3 V. Nell'esperimento i segnali corrispondenti alle accelerazioni vengono acquisiti con i terminali d'ingresso analogici di NI myDAQ; tuttavia, siccome quest'ultimo ha solo due ingressi analogici, sarà possibile sfruttare solo due assi dell'accelerometro.

Il componente Freescale permette di impostare la sensibilità della misura (ovvero in mV/g) e l'intervallo dei valori di input (ciò viene meglio descritto nel datasheet dell'accelerometro, fornito dal costruttore, e scaricabile da www.freescale.com) ossia la scala di misura; infatti avendo un range di misurazione

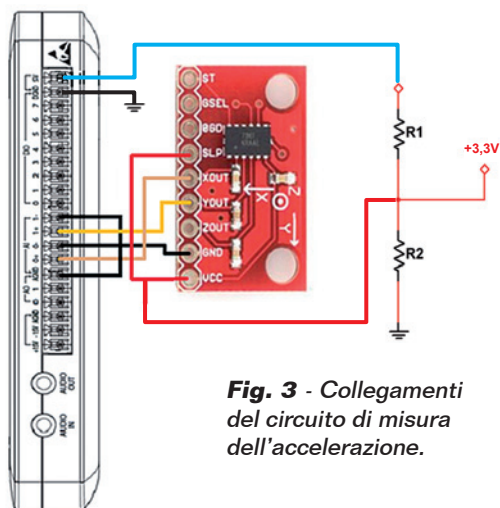


Fig. 3 - Collegamenti del circuito di misura dell'accelerazione.

notevolmente esteso, per garantire un'elevata risoluzione non è possibile esprimere l'intera gamma in una sola "portata". Ecco quindi che ripartendo la gamma di misura in più portate si riescono ad ottenere misurazioni più precise.

Impostate la scala e la sensibilità, per ottenere il valore di accelerazione basterà dividere la tensione in uscita dall'accelerometro per la sensibilità.

Altra particolarità dell'MMA7361L è che dispone di una modalità "sleep" (spento) grazie alla quale è possibile mantenerlo alimentato ma praticamente inerte quando non è necessario compiere misure; questa funzione è apprezzabile nei dispositivi a batterie e comunque dove serve ottimizzare al massimo il consumo di elettricità. Per "risvegliare" il componente bisogna applicare un potenziale di 3,3 V al pin SLP.

Detto ciò, vediamo come realizzare il cablaggio tramite la protoboard: collegate le uscite degli assi X e Y del sensore alle porte AI0+ AI1+, e il cavo di ground all'ingresso AGND; questo va posto in comune con i terminali negativi degli ingressi analogici AI0- e AI1-. Collegate quindi le resistenze che formano il partitore a +5V e GND di NI myDAQ e portate il punto comune delle resistenze al contatto Vcc dell'accelerometro. A questo punto dovete collegare anche l'ingresso SLP dell'accelerometro per mantenere l'integrato nella modalità attiva (altrimenti rimane in sleep e non si può compiere alcuna misura), come indicato in Fig. 3.

CODIFICARE LA STRATEGIA

Preparato il circuito di test è il momento di passare all'elaborazione in LabVIEW, che presuppone la



Fig. 4 - Schema a blocchi di codifica.

stesura di una strategia e di un diagramma a blocchi dell'elaborazione, quindi la realizzazione dell'interfaccia utente, costituita dal Pannello Frontale. In LabVIEW avete bisogno di acquisire il segnale di tensione dell'accelerometro con NI myDAQ tramite il DAQ Assistant. Successivamente dovrete convertire la tensione in un valore di accelerazione basandovi sulla sensibilità impostata dell'accelerometro. Infine verrà prodotto un grafico sul Pannello Frontale contenente la forma d'onda derivante dai valori di accelerazione degli assi X e Y.

La Fig. 4 mostra lo schema a blocchi di codifica del processo, che è molto simile al diagramma a blocchi in LabVIEW.

Per comprendere come avviene l'elaborazione e la visualizzazione nel Pannello di controllo dei risultati, si deve prima descrivere la configurazione dell'accelerometro: la sensibilità predefinita del sensore è 800 mV/g, che corrisponde a un intervallo di $\pm 1,5g$ di accelerazione. Agendo sull'apposito pin d'ingresso digitale dell'accelerometro (GSEL), però, è possibile cambiare tale sensibilità, riducendola a 200 mV/g, che corrisponde a un intervallo di $\pm 6g$. Queste informazioni possono essere reperite sul datasheet dell'accelerometro, fornito dal costruttore, da dove si vede che lasciando aperto (0 logico) tale pin (che corrisponde al contatto 10, G-Select del chip Freescale) si ottiene la portata di 1,5 g, mentre portandolo a livello logico alto si arriva a 6 g. Nell'esperimento qui descritto lasciamo l'ingresso di impostazione della sensibilità scollegato, mantenendo la sensibilità al valore predefinito di 800 mV/g.

Nel diagramma a blocchi, acquisirete i valori di tensione tramite DAQ Assistant. Dovrete poi sottrarre il valore nominale della tensione di offset a

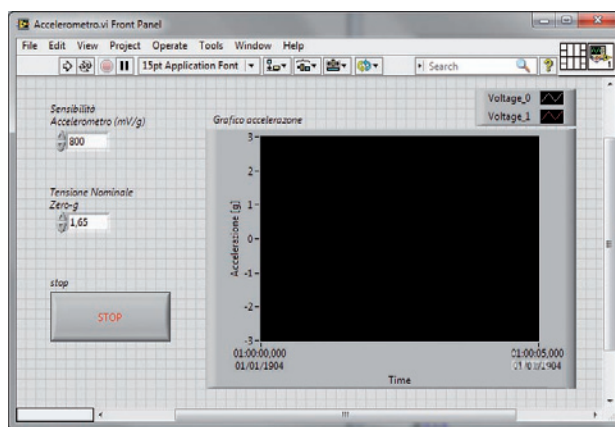
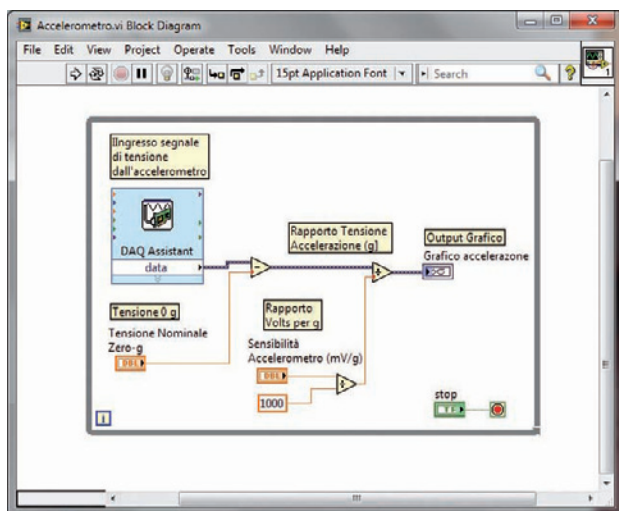


Fig. 5 - Pannello Frontale LabVIEW della misura dell'accelerazione.



zero-g specificato nel datasheet dell'accelerometro. Il valore nominale di tensione a zero-g può essere impostato dal Pannello Frontale, perché è un valore che dipende dalla temperatura.

Successivamente dovete convertire il valore di tensione in un valore di accelerazione dividendolo per la sensibilità in V/g. Questa sensibilità può essere inserita sul Pannello Frontale tramite un controllo numerico. Il valore risultante sarà l'accelerazione in g, da -1,5g a 1,5g, che sarà riportato in un grafico di forme d'onda per la visualizzazione dei valori sul Pannello Frontale.

PANNELLO FRONTALE LabVIEW

Adesso è giunto il momento di vedere come realizzare il Pannello Frontale: questo dovrà avere un controllo numerico per l'impostazione della sensibilità dell'accelerometro, un secondo controllo numerico per la tensione di uscita nominale a zero-g di accelerazione, nonché un indicatore grafico delle due forme d'onda per visualizzare i valori di accelerazione dei due assi X e Y.

Create il VI e subito vi apparirà il Pannello Frontale, cliccando all'interno del quale con il pulsante destro del mouse farete aprire la palette *Controls*, dove sono elencati tutti i tipi di controllo e di indicatore, ma anche decorazioni, tabelle, eccetera. Bloccate la palette nella posizione meno ingombrante facendo clic sulla "puntina" in alto a destra, così da non doverla richiamare ogni volta. Ora, sulla palette alla voce *Modern*»*Numeric* selezionate un *Numeric Control* e trascinatelo sul pannello frontale. Ripetete questa operazione e trascinate un secondo controllo numerico sul pannello frontale. Con lo strumento *Labeling* rinominate il primo controllo numerico scrivendo "Sensibilità Accelerometro (mV/g)" e il secondo, scrivendo "Tensione Nominale Zero-g" (Fig. 5). Ora sulla palette *Controls* alla voce *Modern*»*Boolean* selezionate

Fig. 6 - Diagramma a blocchi.

Stop Button e posizionalo sul pannello frontale. Adesso inserirete il grafico che dovrà visualizzare i valori di accelerazione dei due assi X e Y nel tempo. Per fare ciò alla voce *Modern*»*Graph* selezionate *Waveform Chart* e sistemato sul pannello frontale. Potete ingrandire questo grafico a vostro piacimento utilizzando lo strumento *Positioning*. Successivamente, con lo strumento *Labeling* rinominate l'asse Y del grafico scrivendo *Accelerazione [g]* al posto di *Amplitude* e, cliccando con il tasto destro del mouse sull'etichetta appena inserita, deselectionate la voce *AutoScale Y*. Adesso, con lo strumento *Labeling* rinominate i valori estremi dell'asse Y inserendo +3 e -3, perché il nostro range di misura è noto ed è $\pm 1,5$ g (Fig. 5). Per rendere un po' più chiaro il grafico potete estendere il riquadro dove è indicato *Voltage_0* e far comparire la legenda del secondo grafico *Voltage_1*; i due corrisponderanno all'accelerazione dell'asse X e a quella dell'asse Y. Ora, per completare il Pannello Frontale inserite i valori sui due controlli numerici: scrivete il valore "800" su *Sensibilità Accelerometro* e 1,65 su *Tensione Nominale Zero-g*. Poiché questi due valori saranno quelli predefiniti, conviene impartire il comando di menu *Edit*»*Make Current Values Default*, così, ogni qual volta utilizzerete questo VI, non dovrete inserire nuovamente tali valori. Ora salvate momentaneamente il vostro VI con il nome e la destinazione che preferite.

COSTRUZIONE DEL DIAGRAMMA A BLOCCHI

Implementerete ora 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, nel diagramma a blocchi, come terminali. I terminali sono porte di ingresso e di uscita che scambiano informazioni tra il Pannello Frontale e il diagramma a blocchi. Qui dovrete visualizzare i controlli numerici e il pulsante *Stop* con le stesse etichette che avevate messo sul Pannello Frontale. I terminali dei controlli numerici appaiono in questo modo:



Però è preferibile poterli visualizzare così:



Ciò permette di ridurre lo spazio che occupano sul diagramma a blocchi. Per fare ciò, 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*. Ora, prima di iniziare ad implementare il diagramma a blocchi osservate che i rettangolini che rappresentano i controlli possono essere impostati

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

sia come indicatori (cioè ricevono dati) che come controlli (cioè producono dati). I controlli si distinguono dagli indicatori per due caratteristiche: la prima è una freccia sul terminale che indica la direzione del flusso dei dati (i controlli hanno frecce che mostrano il dato partire dal terminale, mentre gli indicatori hanno una freccia che mostra il dato entrare nel terminale); la seconda è il bordo intorno al terminale (i controlli hanno un bordo più spesso e gli indicatori più sottile). Per trasformare un elemento da indicatore a controllo, dovete cliccare col pulsante destro del mouse sopra il rettangolino che lo rappresenta e impartire il comando *Change to Control*; questa operazione può essere effettuata anche dal Pannello Frontale. Notate che il pulsante *Stop* è già impostato come controllo.

Prestate attenzione ai collegamenti che effettuerete sul diagramma a blocchi: essi permettono di trasferire dati tra oggetti del diagramma a blocchi; ogni collegamento parte da una sorgente di dati e la collega ad uno o più VI e funzioni che leggono i dati stessi. I collegamenti hanno differenti colori, stili e spessori, in funzione del tipo di dati. 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. In questo VI, il blocco *DAQ Assistant* è configurato per il campionamento del canale d'ingresso analogico. Le istruzioni seguenti guidano passo-passo nella configurazione di *DAQ Assistant*.

1. Assicuratevi che *myDAQ* sia collegato al PC;
2. Premete *Ctrl-Spazio* per aprire la finestra *Quick Drop* (al primo utilizzo, l'apertura impiegherà circa un minuto);
3. Cercate la voce *DAQ Assistant* e fate doppio clic su di essa quando appare nella lista;

Tabella 1

4. Trascinate sul diagramma a blocchi;
5. Quando appare un nuovo riquadro per le attività di configurazione rapida, selezionate:
 - a. *Acquire Signals*;
 - b. *Analog Input*;
 - c. *Voltage*;
 - d. *Dev 1 (NI myDAQ)*. Se si dispone di un altro hardware NI installato, il *myDAQ* non sarà *Dev1*;
 - e. *ai0* e *ai1* (Utilizzare *Ctrl + clic* per elementi multipli);
 - f. *Finish*;
6. Impostate il corretto range di tensione su *Signal Input Range* per entrambi gli ingressi:
 - a. *Max: 3.3V*;
 - b. *Min: 0V*;
7. Cambiate il *Timing Settings*:
 - a. *Continuous Samples*;
 - b. *Samples to Read 10k*;
 - c. *Rate (Hz) 50k*;
8. Fate clic su *OK*.

Adesso, nella palette *Functions* andate su *Programming »Numeric* e trascinate sul diagramma a blocchi un elemento *Subtract* e due elementi *Divide*. Sistematele sul diagramma a blocchi come si vede in **Fig. 8**. Ora, con lo strumento *Wiring* collegate l'uscita *Data* del *DAQ Assistant* al primo ingresso del blocco *Subtract* e il controllo numerico *Tensione Nominale Zero-g* al secondo ingresso dello stesso. Proseguite collegando il controllo numerico *Sensibilità Accelerometro* al primo ingresso del blocco *Divide* e, cliccando con il tasto destro del mouse

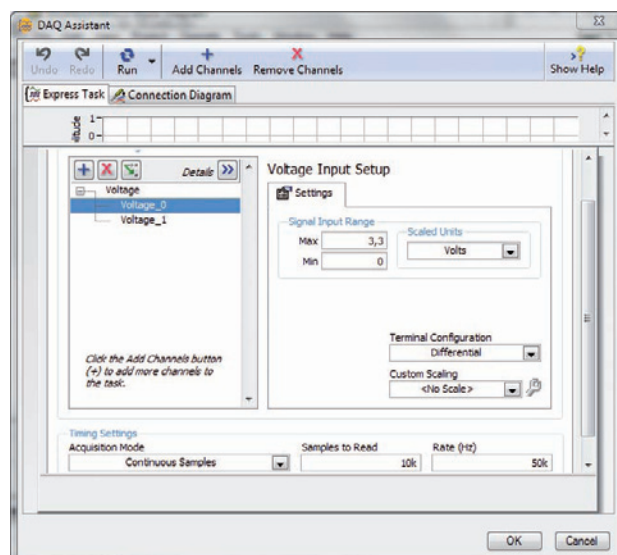


Fig. 7 - Configurazione DAQ Assistant.

in corrispondenza del secondo ingresso, selezionate Create»Constant e assegnategli il valore 1000. Questa operazione cambierà l'unità di misura della sensibilità dell'accelerometro in V/g. Ora, sempre con lo strumento Wiring, collegate l'uscita del blocco Subtract al primo ingresso dell'altro blocco Divide; al secondo ingresso, collegate l'uscita del primo blocco Divide utilizzato (Fig. 8). A questo punto collegate l'uscita finale all'indicatore Waveform Chart che riceverà le forme d'onda dell'accelerazione degli assi X e Y sotto forma di unità g. Adesso inserite tutto il codice a blocchi dentro un ciclo While in modo da eseguire il codice in modo continuo quando verrà premuto il pulsante Run. Nella palette Functions, alla voce Programming, selezionate Structures e cliccate su While Loop; disponete questo elemento nel diagramma a blocchi inglobando tutto ciò che vi è presente. Ora, usando lo strumento Wiring collegate il controllo

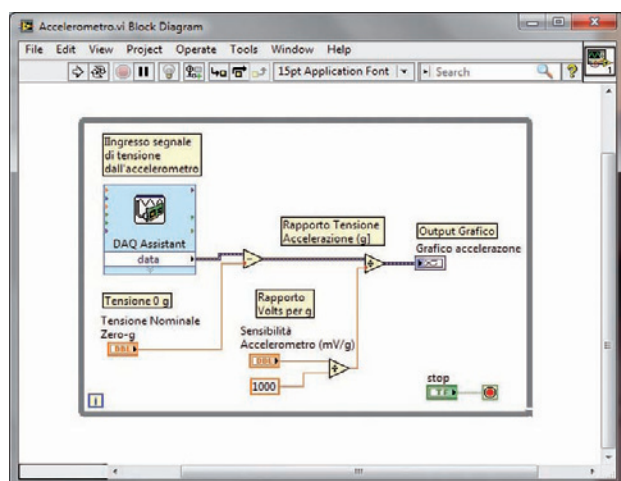


Fig. 8 - Il diagramma a blocchi completato.

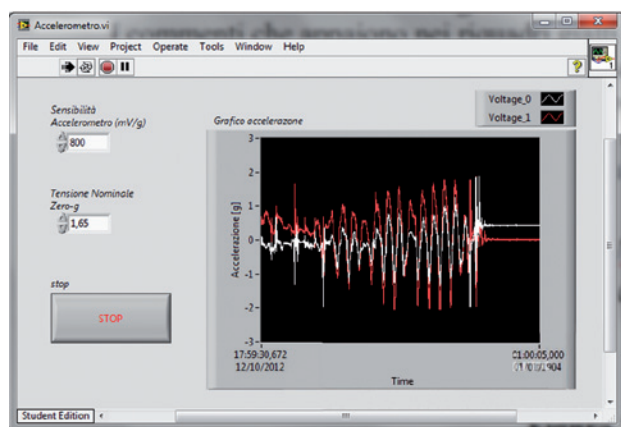
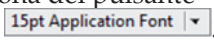
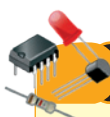


Fig. 9 - Il VI in esecuzione.

Stop al quadratino rosso (Conditional Terminal) in basso a destra all'interno del ciclo While (Fig. 8); ciò consentirà, dopo aver mandato in esecuzione il programma, di uscire dal ciclo While, quindi terminare l'esecuzione del programma, premendo il pulsante Stop. Se avete agito come appena descritto, il diagramma a blocchi dovrebbe presentarsi come in Fig. 8.

Potete inserire a piacimento i commenti che in figura appaiono nei riquadri; l'inserimento si fa con lo strumento Labeling. I commenti non interferiscono con il codice grafico ma ne aumentano la comprensione. Se preferite cambiare le impostazioni di scrittura (font, dimensione, colore, ecc.) sia nel pannello frontale che nel diagramma a blocchi, nella barra del menu in alto cliccate su Text Settings (è il riquadro che riporta scritti la dimensione e il font che state utilizzando). Quando avrete finito di costruire il vostro VI andate su File»Save as e scegliete un nome e una destinazione per salvare il vostro VI. 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.



per il MATERIALE

Questo progetto fa parte del set di esperimenti didattici realizzato da IRS-Elettronica per la board NI myDAQ con il supporto del software LabVIEW. Il set (cod. SETmyKIT, Euro 232,00) comprende tutti i componenti per realizzare i 22 esperimenti, i VI relativi 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

Flowcode permette di apprendere in maniera molto intuitiva lo sviluppo di software per applicazioni embedded, in quanto il codice viene scritto facendo uso di oggetti grafici, in luogo dei classici linguaggi di programmazione come il C e l'Assembler. Prima puntata.

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

Dal giorno dell'invenzione del primo transistor, avvenuta ormai nel 1947, l'elettronica è diventata parte integrante della nostra vita, e sarebbe impensabile ormai, credere di poter fare a meno di questo tassello fondamentale della storia dell'evoluzione umana. Una delle derivazioni più comuni di questa disciplina è sicuramente rappresentata dal mondo dei dispositivi embedded. Quando facciamo una telefonata, o navighiamo su internet usando il nostro smartphone, quando guidiamo la nostra auto, quando facciamo la spesa pagando con la nostra carta di credito, in ognuna di queste circostanze, diversi dispositivi elettronici

dotati di intelligenza elaborano milioni di informazioni al secondo, semplificandoci la vita. Dietro tali dispositivi, veri e propri instancabili ed invisibili servitori dei nostri giorni, si cela comunque la mano dell'uomo: qualcuno ne ha ideato le funzionalità, li ha progettati elettronicamente e ne ha sviluppato la logica. Anche lo sviluppo dei sistemi embedded ha subito una sua evoluzione nel corso degli anni, e si è passati dai dispositivi totalmente analogici ai sistemi a microcontrollore che, per svolgere il compito specifico, devono essere opportunamente programmati, usando appositi linguaggi di programmazione. La programmazione, vero e proprio cuore dello sviluppo, è stata

per anni appannaggio di poche aziende nel mondo, essendo di fatto inaccessibile alle aziende non specializzate ed ai privati, per via degli altissimi costi dei sistemi di sviluppo. Negli ultimi vent'anni, con l'avvento dei primi controllori basati su memorie FLASH (riprogrammabili elettricamente), il costo dei sistemi di sviluppo è calato vertiginosamente, rendendo questo tipo di tecnologie accessibili anche agli sviluppatori privati ed al mondo hobbistico. Sono così comparsi i primi ambienti di sviluppo software e compilatori (generalmente basati su linguaggi come il C e l'assembler) a basso costo. Tali ambienti hanno, comunque, continuato a mantenere caratteristiche di complessità d'uso ed i linguaggi utilizzati richiedono, per essere impiegati efficacemente, delle basi di programmazione non indifferenti. Così lo sviluppo su sistemi embedded è diventato più accessibile dal punto di vista economico, ma le difficoltà tecniche nell'uso dei *tools* sono rimaste comunque uno scoglio notevole da superare per i neofiti della programmazione software. Per ovviare a problematiche di questo tipo sta nascendo, in questi anni, una nuova generazione di ambienti di sviluppo, che ha lo scopo di astrarre le difficoltà intrinseche di un linguaggio di programmazione text-based, facendo uso di formalismi più comuni. Uno degli esponenti di spicco di questa nuova generazione di ambienti di sviluppo è Flowcode, prodotto da Matrix Multimedia, azienda leader nel settore della produzione di sistemi di sviluppo hardware/software per microcontrollori.

Programmazione grafica con Flowcode

L'ambiente di sviluppo Flowcode, giunto ormai alla versione 5, nasce per ovviare al problema della complessità degli ambienti di sviluppo tradizionali e per consentire lo sviluppo di applicazioni embedded complesse anche a chi ha una minima (se non nulla) conoscenza di linguaggi di programmazione. Flowcode è un completo IDE (Integrated Development Environment) per microcontrollori ad 8, 16 e 32-bit. Gli attuali target supportati sono:

- Microchip PICmicro della serie 10, 12, 16

e 18 (8-bit),

- Microchip dsPIC30F, dsPIC33F, dsPIC33E e PIC24 (E, H ed F),
- Atmel AVR e piattaforma Arduino,
- Atmel ARM.

Il concetto su cui si basa Flowcode è l'uso di una rappresentazione grafica basata su diagrammi di flusso (o flowchart, da cui il nome Flowcode) per lo sviluppo di un programma per target embedded. Questo approccio permette di rimpiazzare l'implementazione basata sulle *keywords*, tipica dei linguaggi di programmazione tradizionali. Un esempio di frammento di codice grafico scritto in Flowcode è riportato in Fig. 2. L'ambiente di sviluppo, oltre a generare l'eseguibile per la programmazione del chip (ed eventualmente anche la traduzione in codice sorgente C), integra un simulatore ed un ICD (*in-circuit debugger*) per eseguire il *debugging* del codice direttamente sul target. Una delle caratteristiche più interessanti di Flowcode è, come vedremo in dettaglio nelle prossime puntate, la disponibilità di una ricca libreria di componenti per l'utilizzo delle periferiche normalmente integrate nei microcontrollori. L'uso dei componenti semplifica notevolmente l'accesso alle periferiche, anche nel caso delle periferiche più complesse, come

Fig. 2
Esempio
di codice
Flowcode.

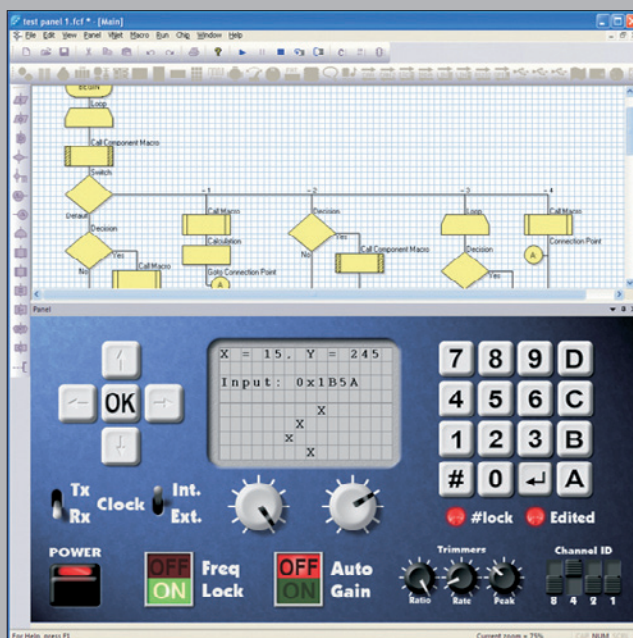


Fig. 3
Sito Matrix multimedia per download di FlowCode.



ad esempio la porta USB, la porta Ethernet o altri sistemi di comunicazione e/o controllo. Una descrizione più dettagliata dei componenti sarà fatta in seguito, ma è importante sottolineare che la disponibilità di queste librerie consente di sviluppare applicazioni anche molto complesse con l'interconnessione di pochi semplici blocchi funzionali.

Installazione e configurazione di Flowcode V5

Come primo argomento del corso illustriamo come installare e configurare l'IDE. Il pacchetto di installazione di Flowcode è scaricabile dal sito web della Matrix (www.matrixmultimedia.com) e, come già detto precedentemente, attualmente sono supportati diversi target, a cui corrispondono diverse versioni del software. Come versione base del corso è stata scelta la versione per i microcontrollori PIC della Microchip Technology. Dopo esserci collegati al sito web della Matrix, accediamo alla sezione download e scarichiamo la versione per PICmicro.

Al termine del download eseguiamo il file FlowcodeV5 - PICMicro.exe e avviamo l'installazione del programma. Comparirà il wizard di installazione, la cui prima schermata è riportata in Fig. 4. Premiamo il pulsante *Next* per accedere alla schermata successiva, confermiamo di accettare la licenza d'uso e proseguiamo.



Fig. 4 - Finestra iniziale di installazione di FlowCode.

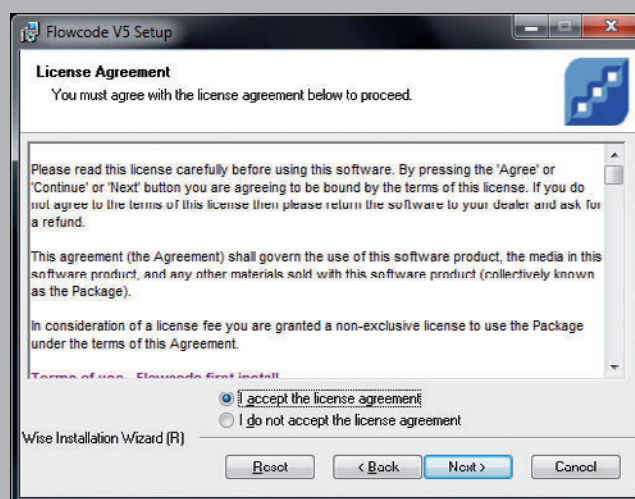


Fig. 5 - Accettazione della licenza d'uso.

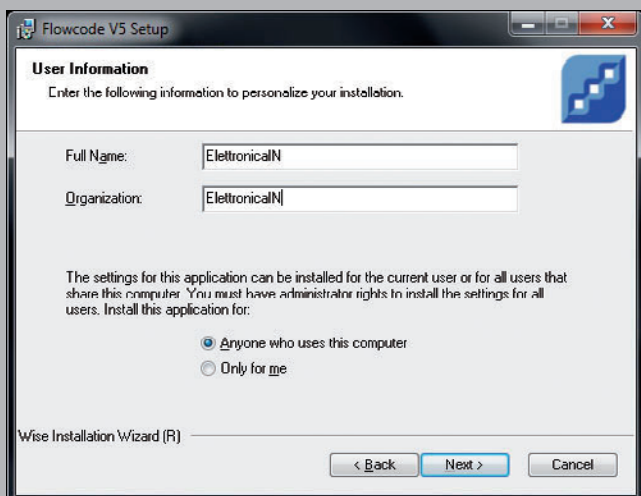


Fig. 6 - Inserimento dati di installazione.

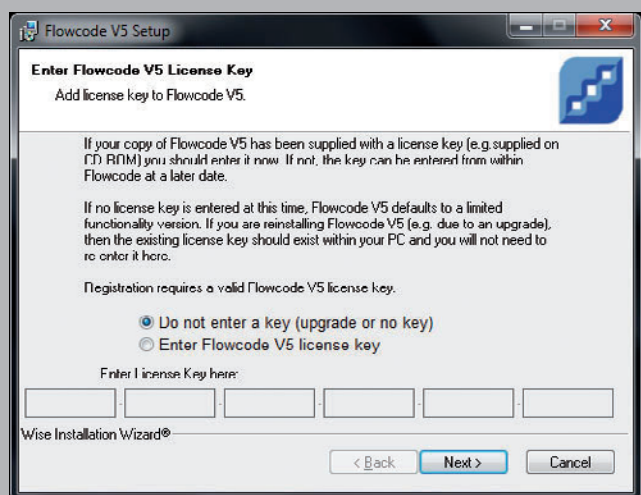


Fig. 7 - Selezione codice attivazione prodotto.

Inseriamo ora il nostro nome ed il nome nell'organizzazione (che in questo caso abbiamo indicato come "ElettronicaIn"), come illustrato in Fig. 6, e premiamo ancora una volta il pulsante *Next*. Selezioniamo la cartella di destinazione e procediamo alla schermata successiva.

A questo punto il *wizard* ci chiede conferma di proseguire con l'installazione e dopo averla ricevuta inizierà il processo di copia dei file su disco. Al termine dell'operazione viene richiesto di attivare il prodotto mediante una chiave di attivazione, o di attivare una licenza free, come possiamo vedere in Fig. 7. La licenza free non ha limitazioni in termini di tempo, ma è attivato un set ridotto di componenti all'interno dell'ambiente.

Installazione scheda di sviluppo EB006

A questo punto il nostro ambiente di

sviluppo è correttamente installato e possiamo procedere alla realizzazione delle nostre applicazioni. Come supporto hardware al corso verrà utilizzato un sistema modulare di schede di sviluppo, prodotte sempre dalla Matrix, che prende il nome di E-block. Il sistema è costituito da una scheda madre alla quale possono essere connessi diversi blocchi per espanderne le funzionalità. Ogni blocco offre le risorse hardware necessarie per implementare una specifica interfaccia, come, ad esempio, porte USB o Ethernet, display grafici, interfacce CAN, driver per motori passo-passo, etc. La scheda scelta come scheda madre per lo sviluppo delle applicazioni che saranno presentate lungo la durata del corso è la EB006. La scheda EB006 è corredata di un CD di installazione che serve per installare tutti i driver necessari per poter utilizzare correttamente il sistema di sviluppo. Inseriamolo all'interno del nostro lettore CD e posizioniamoci all'interno della cartella "drivers\EB-006". In base all'architettura del nostro PC scegliamo `dpinst_amd64` (64 bit AMD o Intel), `dpins_ia64` (64 bit itanium) oppure `dpins_x86` (32bit) ed avviamo l'eseguibile. Comparirà la schermata di Fig. 8, che ci informa che premendo sul pulsante "Avanti" verrà avviata la procedura di installazione dei driver. Proseguiamo e, nel caso ci venga notificato da Windows che non è possibile verificare l'autenticità dei driver, scegliamo di proseguire con l'installazione. Al termine dell'operazione apparirà la

Fig. 8 Schermata iniziale installazione driver per E-block EB006.



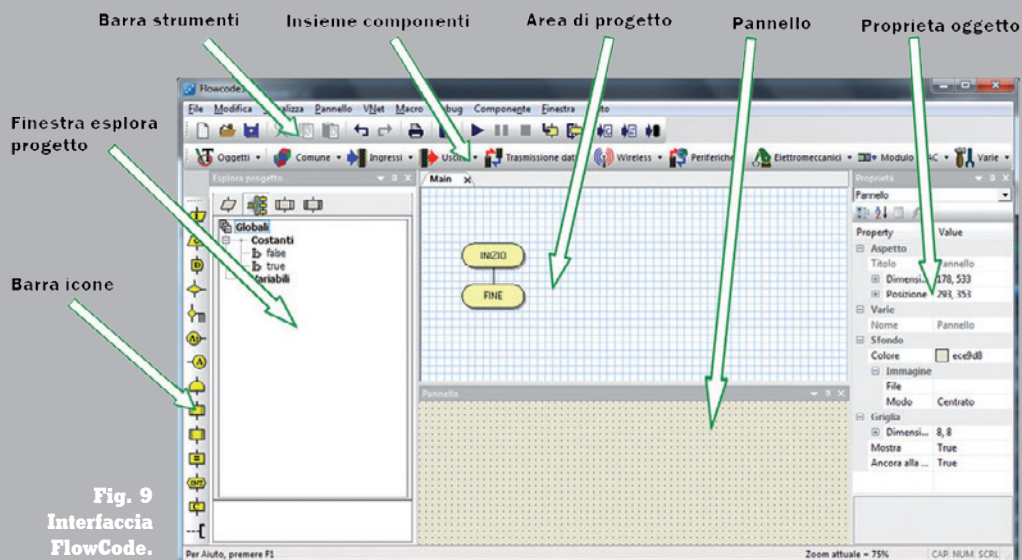


Fig. 9
Interfaccia
FlowCode.

schermata che ci indica che l'installazione è stata eseguita correttamente. Per concludere l'installazione premiamo il pulsante "Fine".

Collegiamo ora la scheda di sviluppo EB006 mediante il cavo USB (fornito in dotazione) al computer. Il sistema operativo installerà i driver aggiuntivi ed uno dei due led sulla scheda si accenderà, segnalandoci che il *debugger* è attivo.

Interfaccia di Flowcode

A questo punto anche il nostro hardware di sviluppo è configurato e pronto per essere utilizzato, passiamo quindi alla descrizione dell'interfaccia di programmazione dell'IDE. Come già detto nella parte introduttiva di questa puntata, Flowcode è un ambiente di sviluppo grafico: questo vuol dire che il programma non viene rappresentato, come avviene invece con i linguaggi di programmazione tradizionale, da un codice testuale, ma verrà composto facendo uso solo di oggetti grafici. Nel caso di Flowcode la rappresentazione scelta è quella dei diagrammi di flusso (o flowchart). Gli elementi base del diagramma di flusso implementano le strutture di controllo fondamentali del programma, come i blocchi decisionali, i cicli, i ritardi, le selezioni multiple, le operazioni logiche e matematiche e altro ancora. Per la gestione delle periferiche più complesse sono invece forniti una serie di componenti che non sono altro che dei contenitori che offrono delle funzioni predefinite, per semplificare

la gestione della periferica. Infine, tramite un opportuno *wizard* di configurazione, è possibile impostare la configurazione base del microcontrollore, come la frequenza del clock interno, eventuali protezioni, le impostazioni sulla memoria, etc.

L'interfaccia di FlowCode è suddivisa principalmente in barre e pannelli, ognuno dei quali ha uno specifico compito che ora andremo ad analizzare. Qualora non tutte le barre siano visualizzate, è possibile aggiungerle dal menu "*visualizza*" semplicemente aggiungendo una spunta vicino al nome che le identifica.

In Fig. 9 è rappresentata la schermata principale di Flowcode. Al centro dello schermo è presente un tab (che nel nostro caso si chiama "Main"), che chiameremo "*area di progetto*" nel quale verrà implementato il diagramma di flusso. Nella parte sinistra dello schermo è invece presente la barra delle icone, dalla quale è possibile scegliere quale blocco inserire all'interno del programma.

Fig. 10
Barre degli
strumenti.

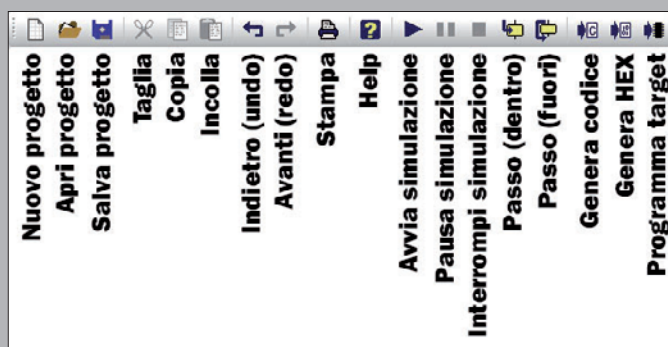
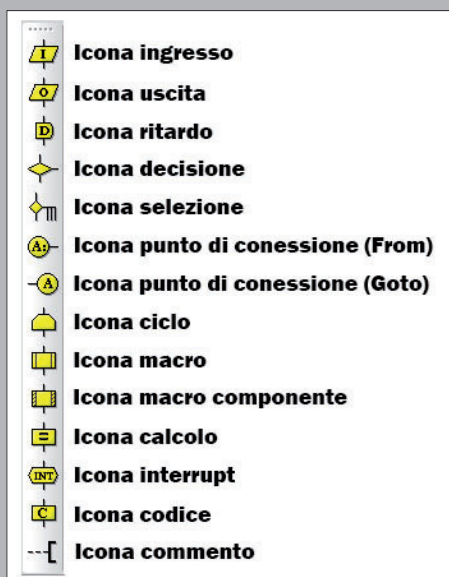


Fig. 11
Barre
delle
icone.



In alto, sopra l'area di progetto troviamo la barra degli strumenti (illustrata in Fig. 10), che ci fornisce i comandi base per gestire il progetto, generare codice, debuggare, compilare e simulare il nostro progetto. Tra la barra degli strumenti e l'area di progetto troviamo invece la barra dei componenti che ci consente di utilizzare, all'interno dei nostri progetti, componenti complessi quali USB, Ethernet, controllo di motori *brushless*, e molto altro ancora, semplicemente inserendoli all'interno del pannello che è visibile in basso al centro della schermata del programma.

La barra delle icone, rappresentata in Fig. 11, contiene tutti i possibili blocchi che possono essere aggiunti all'interno del diagramma di flusso. Per inserire un blocco è sufficiente trascinarlo all'interno del diagramma: una freccia gialla ci indicherà il punto esatto in cui lo stiamo inserendo e l'icona del mouse viene modificata prendendo la forma dell'oggetto che vogliamo inserire all'interno del nostro diagramma di flusso.

Descriveremo nelle prossime puntate le altre barre e pannelli che non abbiamo approfondito in questa sezione.

Creare un progetto

Per creare un progetto apriamo Flowcode e premiamo sul pulsante "Nuovo progetto" nella barra degli strumenti. Configureremo questo progetto in modo da poterlo utilizzare negli esempi che vedremo in seguito. Selezioniamo il microcontrollore scegliendo la famiglia 18 e successivamente

Fig. 12
Opzioni di progetto.

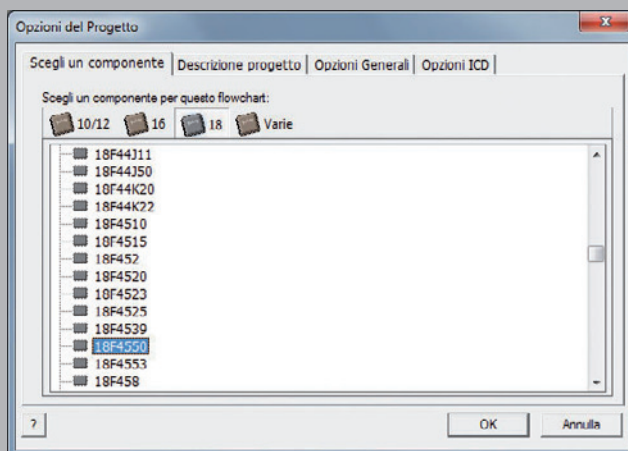
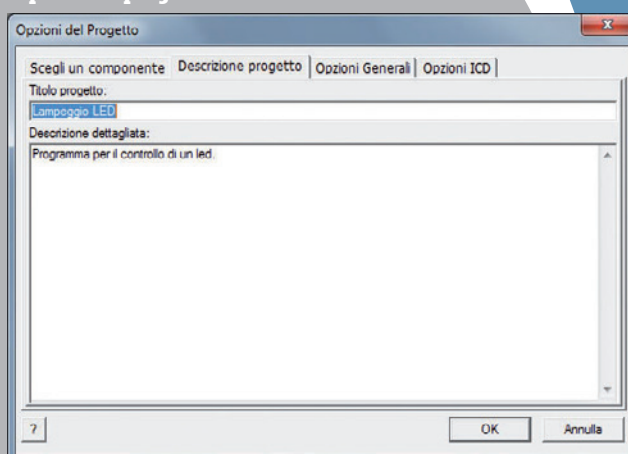
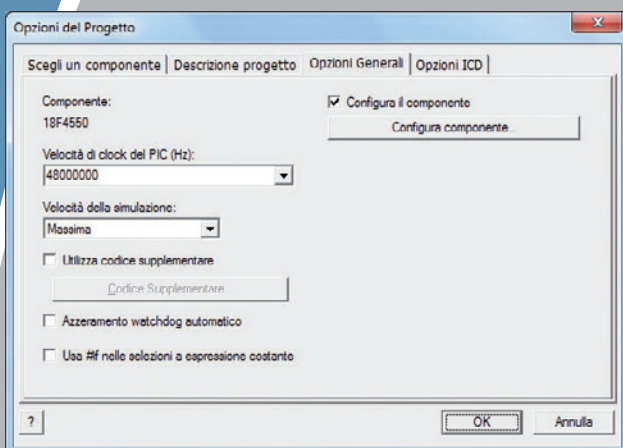


Fig. 13
Inserimento
descrizione
progetto.

18F4550, come indicato in Fig. 12. Ora passiamo al tab "Descrizione progetto" e inseriamo una descrizione significativa. Dal punto di vista della funzionalità del programma non cambia nulla se lasciamo vuoti questi campi ma, per poter capire meglio che cosa realizza il nostro progetto, sarebbe buona norma inserirne almeno una breve descrizione. Nel nostro caso abbiamo inserito il titolo e la descrizione del progetto che realizzeremo al termine di questa puntata: un semplice programma che fa lampeggiare un LED.

All'interno del tab "Operazioni Generali", visibile in Fig. 14, configuriamo il progetto in modo che rispecchi l'implementazione hardware del sistema. La casella "Velocità di clock del PIC" deve essere impostata a 48MHz e la casella "Configura componente" deve essere selezionata. Passiamo ora a configurare i settaggi specifici del microcontrollore cliccando sul pulsante

Fig. 14
Impostazioni generali
nuovo progetto.



“Configura componente”, che ci farà accedere a una schermata di configurazione come quella rappresentata in Fig. 15. Per motivi di tempo ci limiteremo a settare ogni campo come indicato in figura, senza soffermarci sul perché di ogni scelta, che potrà comunque essere approfondita leggendo il datasheet specifico del microcontrollore utilizzato. Dopo aver correttamente compilato ogni campo premiamo il pulsante OK, per tornare alla schermata di configurazione di progetto e nuovamente premiamo OK per iniziare a realizzare il diagramma di flusso.

Per poter utilizzare questa configurazione negli esempi successivi salviamo il progetto, premendo sul pulsante “Salva progetto” nella barra degli strumenti, con il nome “ConfigurazioneBase.fc”.

Aggiunta di blocchi nel diagramma di flusso

Una volta completata la configurazione dell’hardware, torniamo nell’area progetto per inserire blocchi del diagramma di flusso. Per inserire un blocco basta trascinarlo dalla barra delle icone all’interno dell’area di progetto. Una freccia gialla ci indicherà il punto esatto in cui lo stiamo inserendo. Per rimuovere un blocco basta selezionarlo e premere il tasto “Canc”. Cliccando sul blocco con il tasto destro è possibile eseguire altre operazioni quali tagliarlo, copiarlo, esaminarne le proprietà e altro ancora che vedremo nelle puntate successive.

Proviamo ora a esaminare le proprietà di

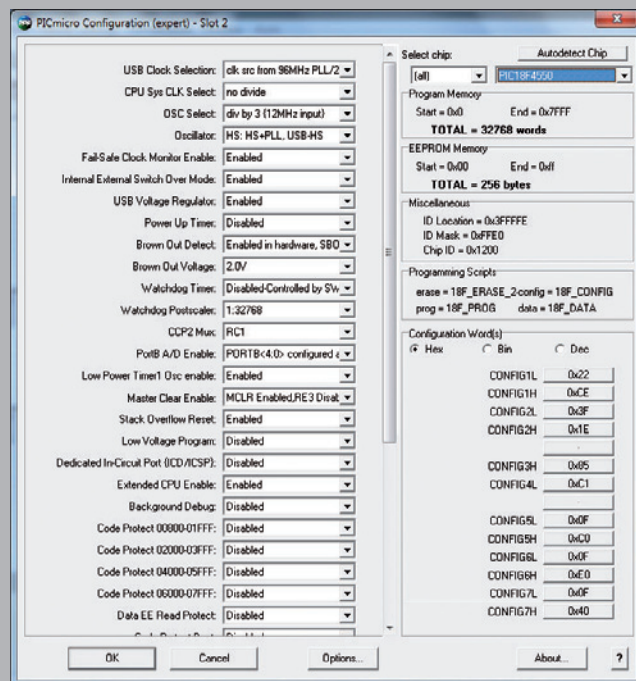
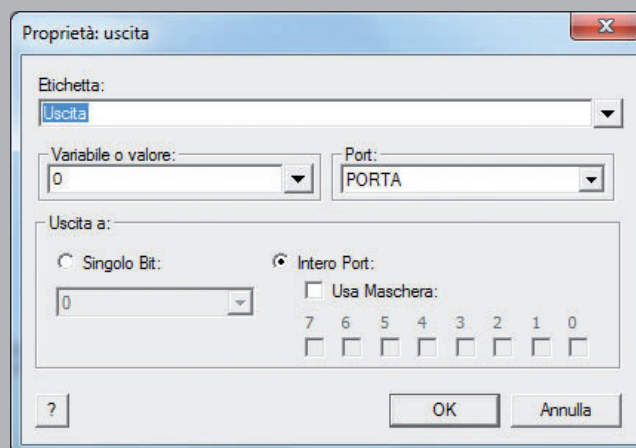


Fig. 15
Configurazione
specifica del
microcontrollore.

un blocco. Come prima operazione inseriamo il blocco di gestione delle uscite del microcontrollore, trascinando all’interno dell’area di progetto l’icona uscita, (presente nella barra delle icone) tra i blocchi di INIZIO e FINE, già presenti di default all’interno del diagramma. Cliccando con il tasto destro sul blocco che abbiamo inserito appare la finestra a scomparsa di Windows. Scegliamo l’opzione “Proprietà” ed esaminiamo la finestra che si è aperta, riportata per comodità nella Fig. 16. Possiamo ora impostare tutte le proprietà disponibili per il blocco che abbiamo selezionato. Il blocco è utilizzato per controllare lo stato delle uscite del nostro microcontrollore. Nella lista a scorrimento “Port” possiamo selezionare la porta della quale vogliamo modificare il valore

Fig. 16
Proprietà
blocco uscita.



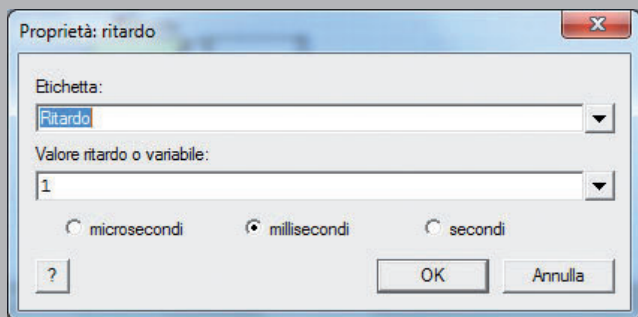


Fig. 17
Proprietà
blocco
ritardo.

di uscita, mentre nella casella “Variabile o Valore” indichiamo il valore alla quale vogliamo impostare la porta selezionata. Solitamente non vogliamo impostare il valore dell’intera porta di uscita, ma solo di uno o alcuni bit. Per far ciò ci viene in aiuto il pannello “Uscita a”. Se selezioniamo “Singolo bit” possiamo scegliere quale bit impostare lasciando invariato il valore degli altri pin della porta. Selezionando “Intero Port” e selezionando la casella “Usa Maschera” è possibile selezionare quali bit devono essere impostati. In questo modo è possibile lasciare invariati i valori dei pin della porta il cui bit non è contrassegnato nella maschera. La casella di testo “Etichetta” permette di modificare il testo che verrà indicato sopra il blocco all’interno del diagramma. È molto importante indicare per ogni blocco un’etichetta significativa perché, anche se per progetti molto semplici può sembrarci inutile, qualora volessimo realizzare diagrammi complessi, potrebbe esserci molto di aiuto avere a disposizione etichette che, con poche parole, ci indichino che cosa implementa quel particolare blocco. Sono in pratica l’equivalente dei commenti scritti

durante la stesura di un codice realizzato in un qualsiasi linguaggio di programmazione. Inseriamo adesso un ritardo software all’interno del nostro schema a blocchi. Per eseguire questa operazione selezioniamo l’icona del blocco di ritardo, presente sempre nella barra delle icone, e trasciniamola all’interno del diagramma, posizionandola proprio sotto il blocco precedentemente inserito. A questo punto accediamo alle proprietà del blocco, cliccando con il tasto destro del mouse, come abbiamo fatto precedentemente. Come è possibile vedere in Fig. 17, il blocco permette di inserire un ritardo nell’esecuzione del flusso del programma. Possiamo scegliere l’entità del ritardo da inserire indicandone il valore e l’unità di misura. Il valore che possiamo inserire deve essere di tipo intero, cioè senza virgola, ma possiamo comunque inserire ritardi minori del secondo passando all’unità di misura con risoluzione maggiore. Ad esempio per aggiungere un ritardo del valore di 0.5s basterà che inseriamo all’interno della casella “valore ritardo”, il numero 500, mentre come unità indichiamo “millisecondi”. Ricordiamoci che 1 millisecondo equivale a 1000 microsecondi e che 1 secondo equivale a 1000 millisecondi.

Le strutture di controllo flusso

Passiamo ora ad analizzare alcune delle

Fig. 18
Struttura
decisionale
binaria e
panello di
configurazione
del confronto.

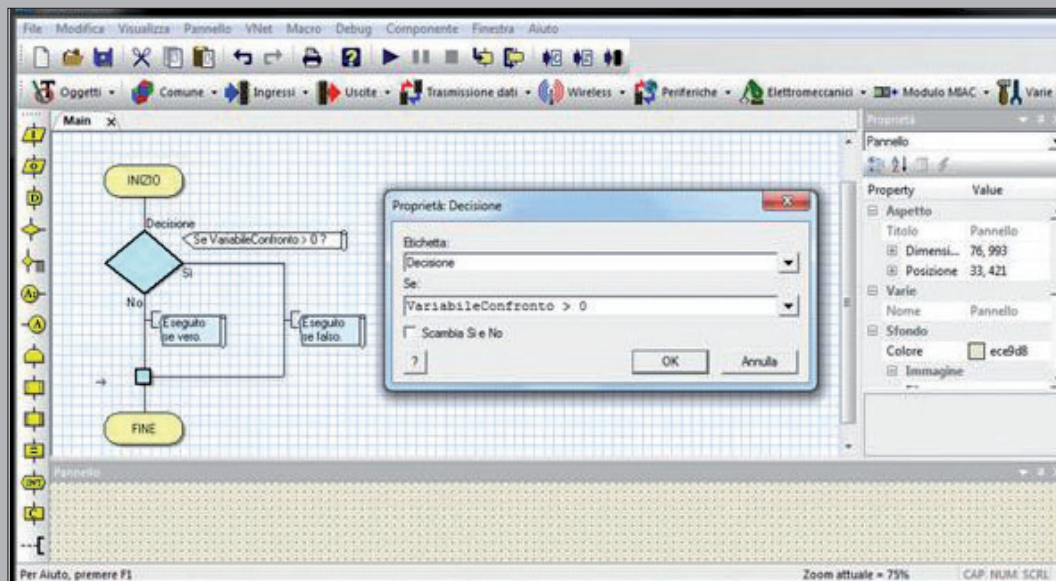


Fig. 19
Struttura
di selezione
multipla.

altre icone presenti nella “barra delle icone”. In Flowcode possono essere utilizzate tutte le normali strutture di controllo presenti normalmente nei linguaggi di programmazione come, a titolo di esempio, il C.

Decisione binaria

Mediante l'icona *decisione* è possibile implementare la funzionalità di decisione binaria. Il blocco fa sì che il flusso di programma passi per il primo o il secondo ramo, in base al valore del confronto effettuato all'interno del blocco. In pratica, se la condizione è verificata, il programma continuerà l'esecuzione dei blocchi presenti nel ramo identificato dall'etichetta “sì”, altrimenti eseguirà l'altro ramo. Per coloro che conoscono il linguaggio C il blocco è l'equivalente dell'istruzione “if”.

Selezione multipla

Utilizzando l'icona *selezione* è possibile implementare la funzionalità di selezione multipla. Il flusso di programma è deviato attraverso il ramo per il quale l'equazione di confronto soddisfa il valore indicato all'interno del caso specifico. Qualora nessuno dei casi indicati soddisfi l'equazione di selezione, il programma eseguirà le istruzioni presenti nel ramo di *default*. Possono essere realizzate selezioni con massimo 10 differenti casi, ma questo numero è più che sufficiente per realizzare la maggior parte dei possibili programmi. In pratica se uno dei rami indicati ha come etichetta il valore che soddisfa l'equazione viene eseguito, altrimenti viene scartato. Solo uno dei rami può essere eseguito ed ognuno deve avere un valore di selezione differente dall'altro.

È l'equivalente del costrutto “switch... case” di un programma implementato in linguaggio C. Un esempio d'uso della struttura di selezione multipla è riportato in Fig. 19.

Icona ciclo

L'icona *Ciclo* permette di realizzare tre differenti controlli di flusso:

- *Finché*
- *Esegui...finché*
- *Ripeti*

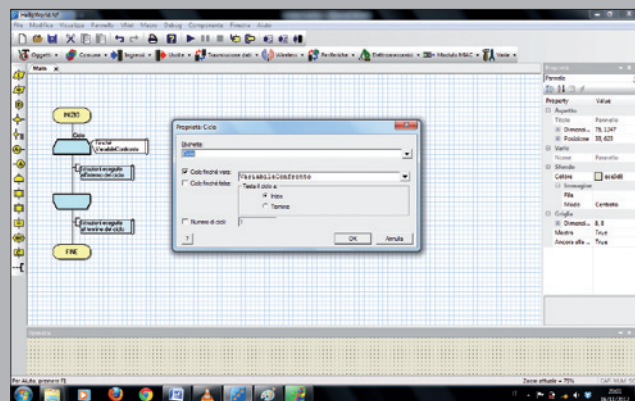
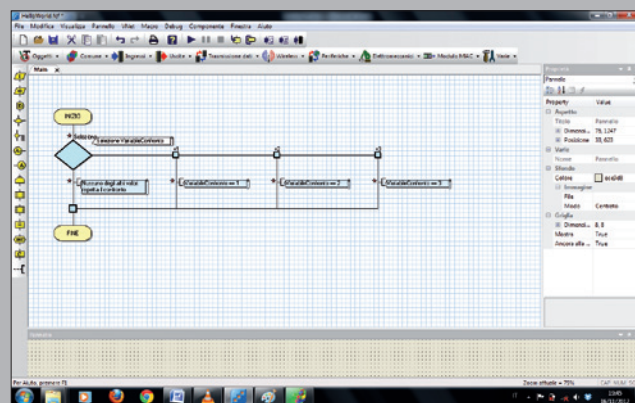


Fig. 20
Ciclo
“while”
(Finché
vero).

Il blocco implementa di default la struttura “Finché”, ma per realizzare gli altri due costrutti basta modificarne le proprietà come visto per gli altri blocchi nelle sezioni precedenti. Analizziamo le tre opzioni offerte:

Ciclo “Finché”

Fintanto che l'equazione di confronto risulta vera vengono eseguite le istruzioni all'interno del ciclo. Il confronto è effettuato al primo ingresso nel blocco di ciclo ed ogni volta che vengono terminate tutte le istruzioni interne al ciclo. Quando il confronto risulta falso le istruzioni interne non sono più eseguite e il programma continua attuando le istruzioni successive al ciclo. Nel caso in cui l'equazione di confronto risulti falsa al primo controllo, il codice interno al ciclo non viene eseguito neppure una volta e il programma continua la sua esecuzione ignorando le istruzioni contenute all'interno dei due blocchi di ciclo. Un esempio di ciclo di questo tipo è riportato in Fig. 20.

Ciclo “Esegui...Finché”

Impostando la proprietà del blocco “Testa il ciclo a:” al valore “Termine”, come in

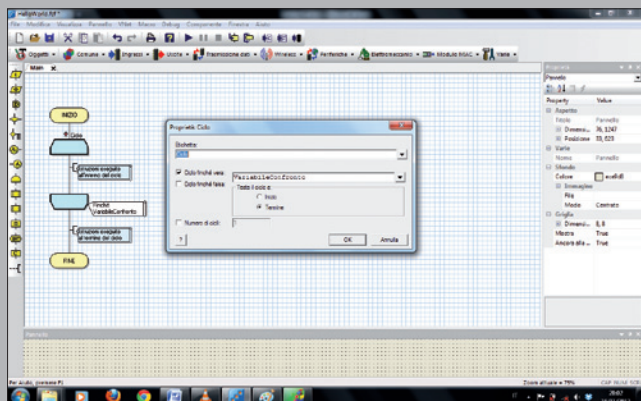


Fig. 21 - Ciclo "Esegui ...Finché". Il confronto è effettuato al termine del blocco.

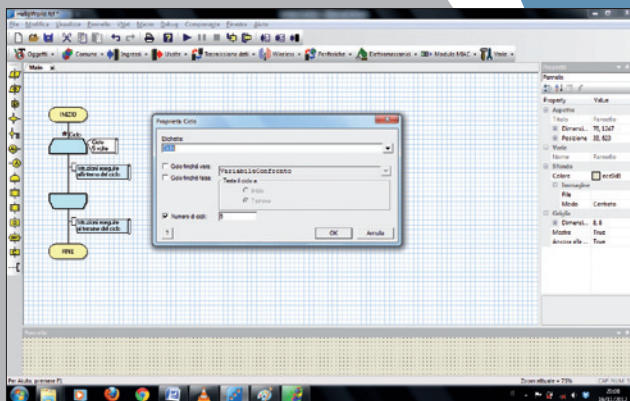


Fig. 22 - Ciclo "Ripeti".

Fig. 21, lo rendiamo equivalente al blocco realizzato in C con le istruzioni "Do While". In questo caso l'istruzione di confronto è eseguita al termine di ogni attuazione delle istruzioni contenute all'interno del ciclo. Il codice interno è quindi eseguito almeno una volta.

Ciclo "Ripeti"

Selezionando la proprietà del blocco a "Numero cicli:", come è stato fatto in Fig. 22, il ciclo viene impostato per funzionare come un "for". Configurato in questo modo le istruzioni interne vengono eseguite un numero di volte pari al numero indicato

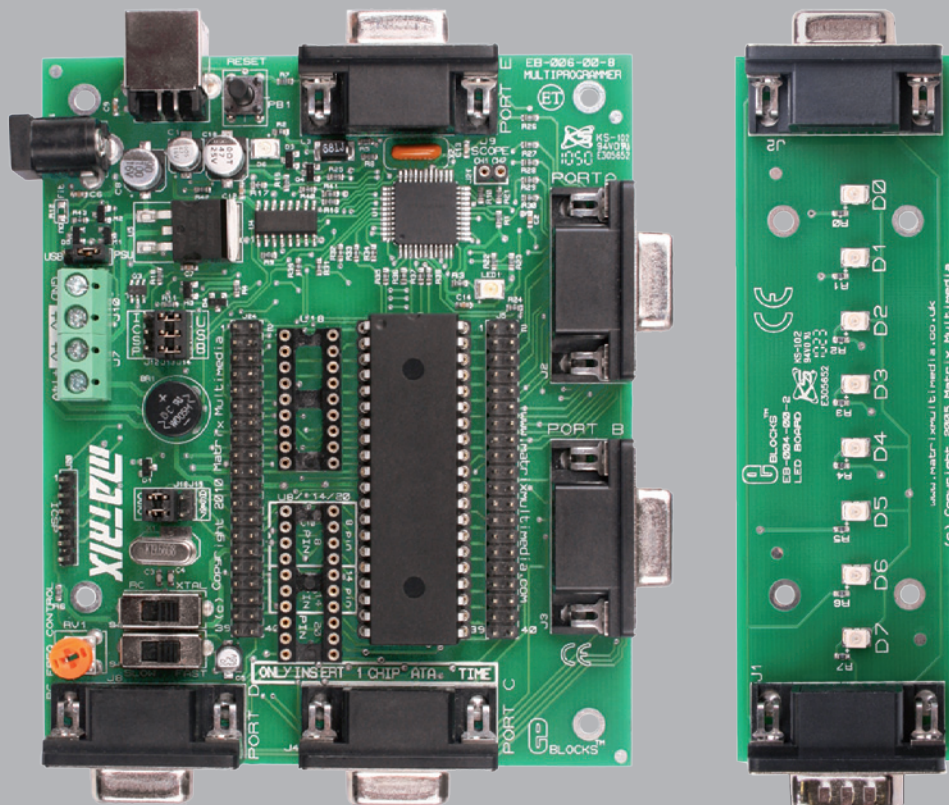
nella cella a fianco dell'impostazione "Numero cicli:". Il blocco è fondamentale quando abbiamo bisogno che una porzione del nostro diagramma di flusso venga eseguita per un numero predefinito di volte.

Esempio pratico: facciamo lampeggiare un LED...

Ora che conosciamo i blocchi principali e le possibili variazioni per il controllo di flusso, possiamo implementare un semplice diagramma per far lampeggiare un LED. Per realizzare questo esempio abbiamo bisogno dei seguenti componenti:

- EB006 "USB PICmicro multiprogram-

Fig. 23
Scheda EB006
a sinistra e
scheda
E-Blocks
EB004 a
destra.



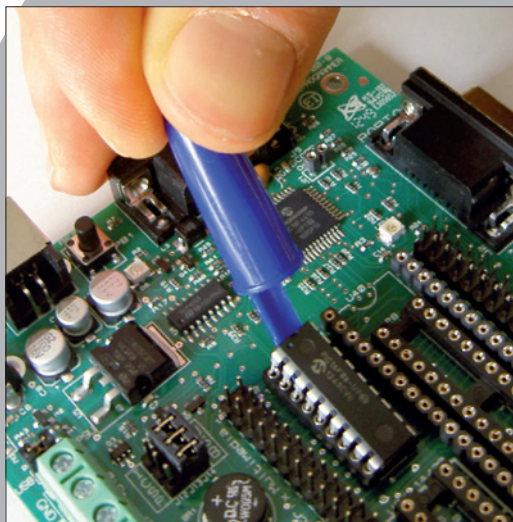


Fig. 24
Metodo per la rimozione del microcontrollore PIC.

con uno a 12MHz e spostiamo il *jumper* di selezione della sorgente di clock su "OSC" e l'interruttore di selezione oscillazione su "XTAL". Un riassunto grafico dell'impostazione dei vari jumper è riportato in Fig. 25. Collegiamo ora la scheda EB006 con la E-Blocks EB004 utilizzando la PORT A, come illustrato in Fig. 26.

mer board"

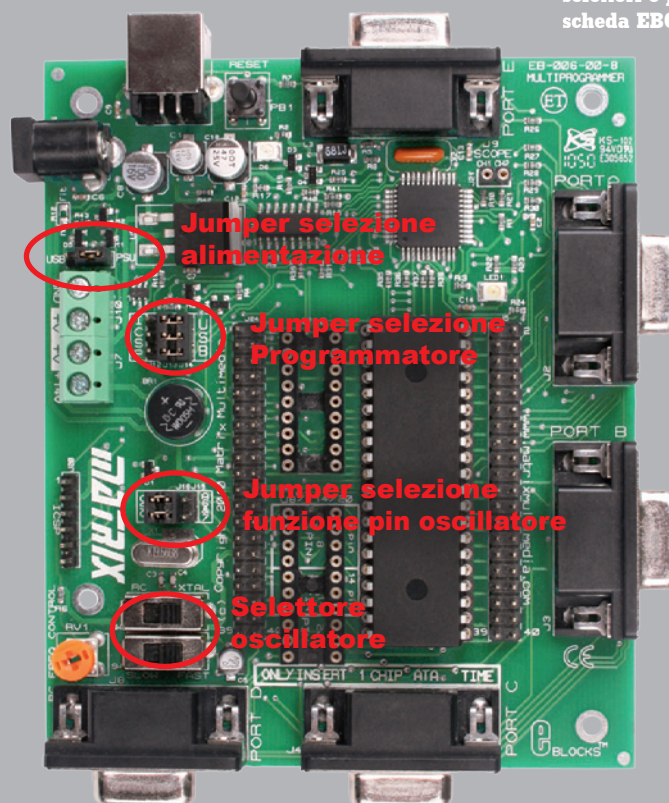
- EB004 "LED Board"
- PIC18F4550
- Quarzo 12MHz

La scheda EB006 contiene il microcontrollore e l'hardware necessari per interfacciarsi con le altre schede E-Blocks e il PC dal quale lo programiamo; deve essere configurata per poter essere utilizzata nel nostro esempio. I jumper di selezione dell'alimentazione devono essere posizionati sul lato di sinistra, in modo da far sì che l'alimentazione provenga dall'USB. Il jumper di programmazione deve anch'esso essere settato su USB, in modo da poterlo programmare direttamente da PC, senza l'ausilio del programmatore ICSP della Microchip. Sostituiamo ora il microcontrollore che è attualmente montato sulla scheda con il PIC18F4550. Questa operazione è molto delicata, quindi dobbiamo prestare molta attenzione nel rimuovere il microcontrollore senza danneggiare la scheda di sviluppo. Per evitare danni aiutiamoci con il tappo di una biro ed eseguiamo l'operazione come indicato nella sequenza di Fig. 24.

Facendo attenzione alla direzione dell'allargiamento a 40 pin, inseriamo il PIC18F4550, assicurandoci che tutti i piedini entrino senza piegarsi, all'interno dello zoccolo. Per piegare correttamente i piedini del PDIP40, possiamo aiutarci con il piano della nostra scrivania, appoggiando il PDIP di traverso e facendo leva sul case, finché i piedini risultano paralleli al bordo. A questo punto sostituiamo il quarzo

L'hardware è ora pronto e possiamo dedicarci all'implementazione del programma. Apriamo Flowcode e carichiamo la configurazione di base che avevamo salvato durante la configurazione del nostro primo progetto. Premiamo sul pulsante "Apri progetto" nella barra degli strumenti e selezioniamo il file "ConfigurazioneBase.fcf". Per evitare di "sporcare" il file, dato che Flowcode durante alcune operazioni salva automaticamente il progetto, salviamo immediatamente il progetto con un nuovo nome. Dal menu scegliamo "File/Salva con nome..." e indichiamo come nuovo nome "LampeggioLed.fcf". Proviamo ora a pensare

Fig. 25
Configurazione selettori e jumper scheda EB006.



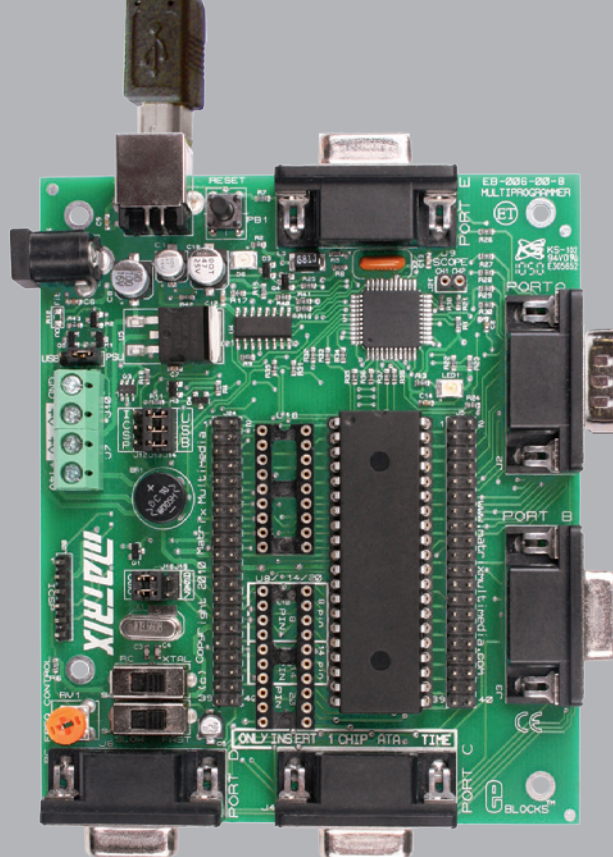
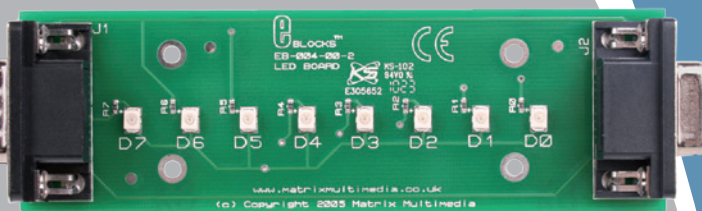
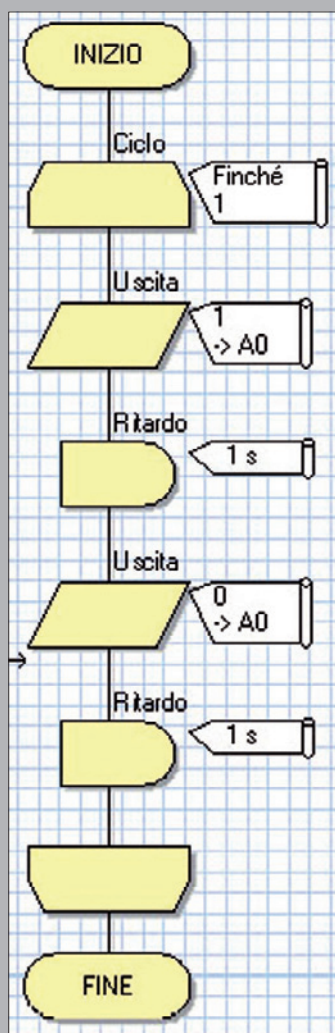


Fig. 26
Configurazione
completa per la
realizzazione
dell'esempio.



alle operazioni che devono essere eseguite per fare in modo che si veda lampeggiare un LED sulla scheda. Innanzitutto dobbiamo realizzare un ciclo infinito che il programma dovrà eseguire per tutto il tempo di attività e per questo scegliamo il blocco "Finché". Impostiamo in modo che il ciclo venga sempre eseguito come descritto nella sezione precedente. Abbiamo poi bisogno di qualcosa che ci permetta di modificare il valore di un pin del microcontrollore ed il blocco che fa al caso nostro è l'icona uscita. Inseriamo due blocchi all'interno del ciclo e impostiamoli in modo che il primo setti il bit 0 al valore 1 e il secondo il bit 0 al valore 0. Se eseguiamo il flusso così com'è attualmente implementato,

Fig. 27
Diagramma di flusso per il
lampeggio del led collegato
alla porta A pin 0.



vedremmo a causa dei fenomeni fisici interni al nostro occhio il LED sempre acceso, ma con un'intensità minore rispetto a quella massima. Per evitare questo problema inseriamo un blocco di ritardo dopo il set a uno della porta e un altro – identico – dopo il blocco di set a zero della porta. Impostiamo il valore del ritardo a 1 secondo per entrambi i blocchi inseriti. Se avete eseguito tutto correttamente dovreste ottenere un diagramma di flusso simile a quello riportato in Fig. 27.

Il nostro flusso è ora pronto per essere caricato sulla scheda. Salviamo il progetto premendo il pulsante "salva progetto" nella "barra degli strumenti" perché lo riutilizzeremo in seguito come base di partenza per altri esempi. Premiamo il pulsante "compila e trasferisci" per caricare il programma all'interno del microcontrollore presente sulla nostra scheda di sviluppo. Attendiamo che il programma esegua le operazioni e al termine del caricamento potremo notare che il nostro primo programma viene eseguito sulla scheda con il LED che si accende e si spegne ogni secondo.