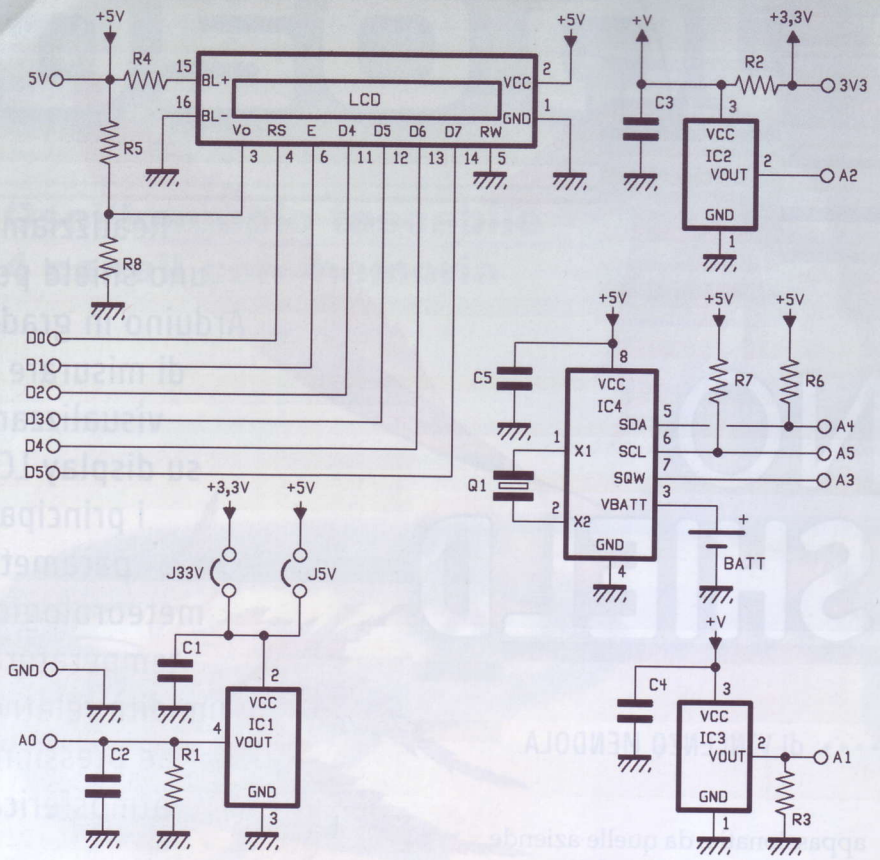
ARDUINO METEO SHIELD

Realizziamo uno shield per Arduino in grado di misurare e visualizzare su display LCD i principali parametri meteorologici: temperatura, umidità relativa e pressione atmosferica.

di VINCENZO MENDOLA

Una delle particolarità che ha reso Arduino tanto popolare è sicuramente la facilità con cui è possibile realizzare estensioni hardware personalizzate chiamati "SHIELD".
In commercio ce ne sono tantissime, praticamente per ogni esigenza, sia "ufficiali", cioè realizzati dal team di Arduino, che non ufficiali, ovvero dalla comunità di hobbisti e appassionati e da quelle aziende, come ad esempio Sparkfun o Seeed Studio, che hanno fatto dell'open hardware e del successo di questa piattaforma di studio e prototipazione il



loro business, tanto da proporre anche dei veri e propri cloni e delle versioni alternative; anche sul sito di Futura Elettronica potete trovare numerosi shield, molti dei quali presentati su questa rivista. Per chi volesse cimentarsi nella realizzazione di una propria versione, ricordiamo che l'unico accorgimento da seguire è quello di rispettare alcune specifiche, inerenti essenzialmente le dimensioni, il passo e le connessioni dei connettori SIL di Arduino, attenendosi alle quali chiunque può realizzare un proprio circuito di espansione compatibile.

IL METEO SHIELD

Sfruttando gli ingressi analogici di Arduino, abbiamo realizzato una di queste "estensioni", in grado di misurare e visualizzare i principali parametri meteorologici: temperatura, umidità relativa e pressione atmosferica; per questo motivo abbiamo deciso di chiamarlo METEO SHIELD. A questi tre sensori è stato aggiunto un RTC (Real Time Clock), realizzato con lo stesso integrato DS1307 della Maxim utilizzato nel RTC SHIELD presentato sulla rivista 163 del mese di Febbraio, questa volta in versione SMD. Aggiungendo questa scheda possiamo trasformare il nostro Arduino Uno in una vera e propria stazione meteo che non ha nulla da invidiare ai dispositivi commerciali oggi tanto di moda, questo perché, a differenza dei primi che non sono modificabili e con prestazioni non espandibili, la nostra piattaforma lo è, semplicemente cambiando il codice ed eventualmente, sfruttando la filosofia che ha reso Arduino tanto versatile, aggiungendo ulteriori schede.

Il nostro meteo-shield è stato realizzato intorno a tre sensori analogici che sono il "cuore" del dispositivo:

- MCP9700A della Microchip per misurare la temperatura;
- HIH-5030-001 della Honeywell per determinare l'umidità relativa;
- MPXH6115A6U della Freescale per disporre dei dati inerenti la pressione atmosferica assoluta.

Questi tre trasduttori sono stati selezionati all'interno dell'ampia gamma di componenti in commercio, in quanto hanno un ottimo rapporto prezzo/accuratezza e sono sufficientemente diffusi per cui non dovrebbe essere un grosso problema recuperarne qualche unità per chiunque voglia cimentarsi nella realizzazione di questo sistema.

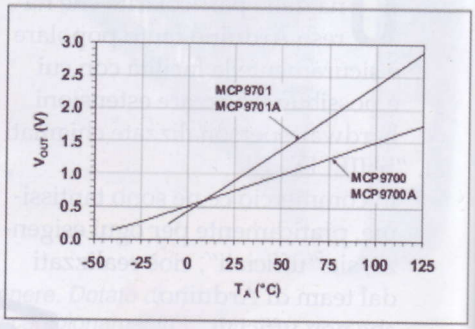


Fig. 1

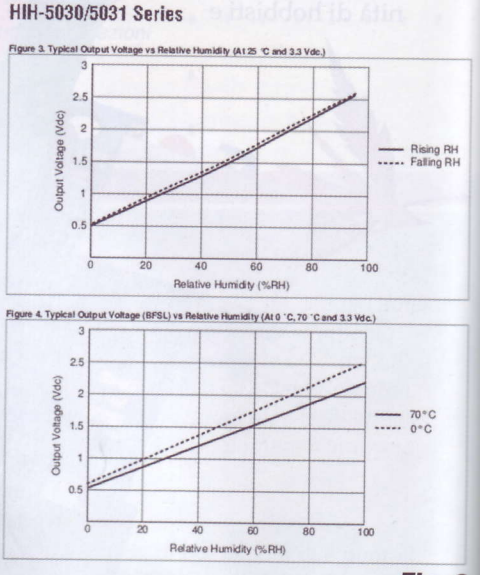


Fig. 2

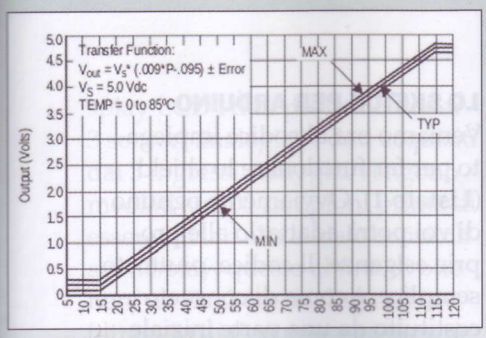
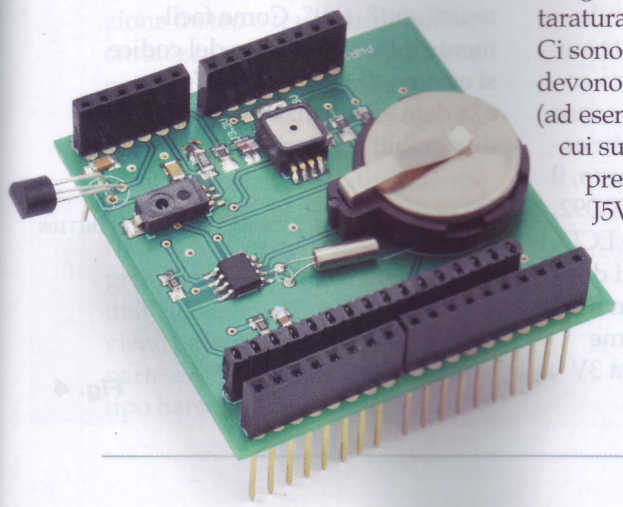


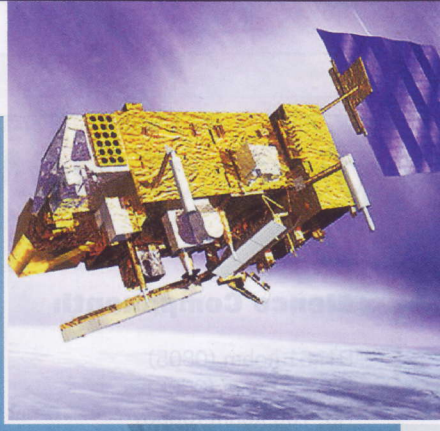
Fig. 3

SCHEMA ELETTRICO

Com'è possibile vedere dallo schema elettrico il circuito è piuttosto semplice: ognuno dei 3 sensori è connesso ad un ingresso analogico di Arduino, in particolare il sensore di pressione ad A0, quello di umidità ad A1 e quello di temperatura ad A2. Come riportato nel datasheet [1], il sensore MCP9700A può essere alimentato con una tensione di 3.3V o di 5V, o comunque compresa nell'intervallo 2.3÷5.5V ed ha una caratteristica di trasduzione lineare con una variazione di 10mV/°C con un'uscita di 0.5V a 0°C, come visibile chiaramente dalla **Fig. 1**, ricavata dalle specifiche fornite dal costruttore; tale trasduttore inoltre non necessita di calibrazione ed ha un'accuratezza di ±2°C nel range -40÷125°C (±1°C nel range 0÷70°C). L'MCP9700A è stato scelto in contenitore TO92 per facilitarne il posizionamento anche all'esterno del PCB qualora fosse necessario; per lo stesso motivo è stato collocato nella parte esterna del PCB, in modo da agevolare un eventuale contatto con la superficie di misura scelta. Oltre al condensatore di bypass C3 da 100nF, è stata introdotta anche la resistenza R2 da 220Ω per ridurre gli effetti del rumore sulla tensione d'uscita. Anche il sensore di umidità HIH-5030-001 può funzionare con una tensione di 3 o di 5V, o comunque un valore compreso tra 2.7V e 5.5V, e presenta una tensione d'uscita proporzionale all'umidità relativa secondo la funzione VOUT=(VSUPPLY) [0.00636 (sensor RH) + 0.1515 (valore tipico a 25°C)] con un' accuratezza di ±3% garantita entro l'intervallo 11-89% di RH [2]. Come visibile dalla **Fig. 2**, la relazione tra le grandezze è caratterizzata da una buona linearità; è possibile osservare un'isteresi trascurabile a 25°C ed uno scostamento contenuto delle caratteristiche di trasduzione a 0°C e a 70°C. Come richiesto da Honeywell è stata inserita una resistenza di carico da 100 kΩ in uscita.

La misura di pressione viene effettuata tramite il circuito integrato MPXH6115A6U che deve essere alimentato con una tensione di 5V, garantendo un'accuratezza di ±1.5% di VFSS (differenza algebrica tra la tensione d'uscita al valore massimo e minimo di pressione) [3]. Anche in questo caso la linearizzazione (**Fig. 3**) viene effettuata direttamente dal circuito integrato e non è necessaria alcuna taratura o regolazione dell'offset. Ci sono sensori di pressione che devono essere alimentati a 3.3V (ad esempio il MP3H6115A) per cui sul circuito stampato è stato previsto il ponticello J3.3V – J5V per selezionare la corretta tensione nel caso si abbia a disposizione uno di questi componenti. Gli altri due trasduttori, come abbiamo visto, possono essere alimentati





Meteop-B: tutto il meteo in 100 minuti

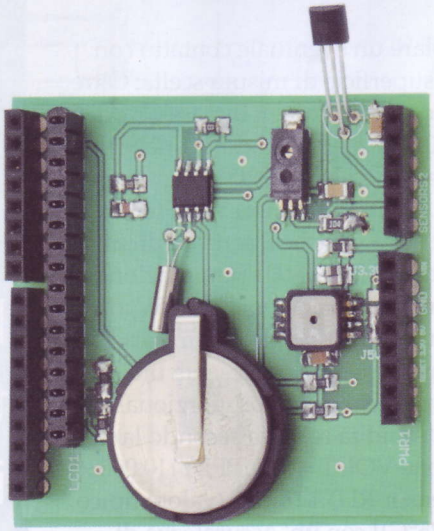
Si chiama Meteop-B ed è l'ultimo satellite meteorologico lanciato nello Spazio. Velocissimo, è capace di compiere un'orbita di rivoluzione intorno al nostro pianeta in appena 100 minuti, consentendoci di conoscere non solo la situazione meteo ma anche i rapidi cambiamenti quasi in tempo reale in tutto il mondo.

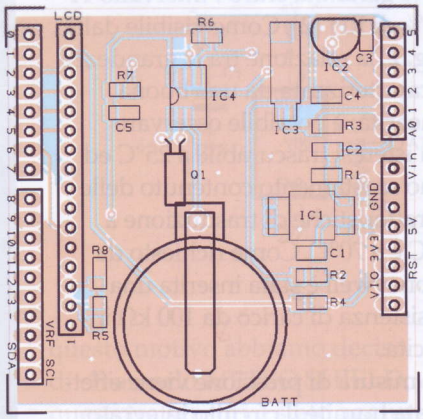
La velocità di esplorazione dallo Spazio è un fattore molto importante perché consente di tenere sotto controllo eventi in rapida variazione ed avere così previsioni del tempo affidabili anche in aree dove la situazione cambia rapidamente.

Metop-B è il secondo satellite meteorologico europeo, lanciato in orbita lo scorso 17 settembre dalla base russa di Baikonur, in Kazakhstan (Metop-A è stato posto in orbita nel 2006); ha una massa di oltre 4 tonnellate ed è in orbita non geostazionaria (compie 14 rivoluzioni al giorno) a circa 820 chilometri di distanza dalla Terra, con inclinazione di 90 gradi rispetto all'equatore. A bordo porta 11 strumenti di osservazione. Nel 2016 a Meteop-B si affiancherà Metop-C, destinato a rimpiazzare Meteop-A, che sebbene funzioni ancora è già alla fine della sua vita operativa.

Elenco Componenti:

R1: 51 kohm (0805)
R2: 220 ohm (0805)
R3: 100 kohm (0805)
R4: 100 ohm (0805)
R5: 1 kohm (0805)
R6: 1 kohm (0805)
R7: 1 kohm (0805)
R8: 22 ohm (0805)
C1: 100 nF multistrato
C2: 47 pF ceramico
C3: 100 nF multistrato
C4: 100 nF multistrato
C5: 100 nF multistrato
IC1: MPXH6115A6U
IC2: MCP9700A
IC3: HIH-5030-001
IC4: DS1307SO8
Q1: Quarzo 32,768 kHz
LCD: Display LCD 16x2 retroilluminato
Varie:
- Porta batteria per CR2032
- Batteria CR2032
- Strip Maschio 6 poli
- Strip Maschio 8 poli (2 pz.)
- Strip Maschio 10 poli
- Strip Maschio 16 poli
- Strip Femmina 16 poli
- Circuito stampato





anche a 5V; si è scelto di utilizzare 3.3V perché nei datasheet le specifiche e le curve di trasduzione sono riportate per questo valore. La sezione circuitale inerente l'RTC ricalca le semplici connessioni ai pochi componenti previsti dalla Maxim per far funzionare l'integrato [4] costituiti essenzialmente dal quarzo a 32.768 kHz che fornisce un riferimento temporale stabile all'oscillatore e dalle due resistenze di pull-up dell'I2C R6 e R7. L'uscita SQW del DS1307 che, come abbiamo visto nell'articolo inerente l'RTC Shield permette di avere un clock ausiliario, utile ad esempio, per far lampeggiare un led alla frequenza di 1Hz, è stato reso disponibile sul pin A3 di Arduino per usi futuri.

PIANO DI MONTAGGIO

Per il montaggio del circuito: è necessario utilizzare un saldatore di buona qualità, dotato di una punta conica di dimensioni adatte ai componenti SMD, i quali saranno saldati per primi, lasciando per ultimi i sensori, che andranno saldati evitando di indugiare eccessivamente sui terminali per evitare un loro eccessivo riscaldamento con conseguente possibile danneggiamento. Si proseguirà con il quarzo, il sensore di temperatura in TO92, il connettore per il display LCD, i connettori SIL caratteristici degli shield Arduino e infine con il portapile per le comunissime batterie al litio a bottone da 3V del tipo 2032.

LO SKETCH PER ARDUINO

Veniamo ora al codice impiegato per far funzionare lo shield (**Listato 1**). Ovviamente ognuno di voi potrà adattarlo alle proprie esigenze. Il codice, piuttosto semplice, è come di consueto costituito da una parte iniziale in cui vengono incluse le librerie necessarie al funzionamento dello shield: possiamo notare la presenza della libreria "LiquidCrystal", utilizzata per rendere il meteo-shield indipendente, dotandolo di un proprio display, un comune LCD 16X2 affiancato dal codice per la visualizzazione dei dati sulla seriale. Per far funzionare l'RTC, facente capo come già visto ai piedini A4 e A5 utilizzati da Arduino per l'I2C, si utilizza la consueta libreria "Wire" e la stessa libreria "RTClib" e la relativa porzione di codice presentata sul numero 163 relativa al funzionamento dello RTC shield; seguono le righe di definizione ed inizializzazione delle variabili con l'assegnazione dei piedini dotati di ADC A0, A1 e A2 di Arduino che hanno il compito di acquisire i dati da visualizzare; l'istruzione visibile in **Fig. 4** è stata scritta al fine di realizzare il carattere "°C" personalizzato, come spiegato in dettaglio nella sezione "Reference" del sito ufficiale di Arduino [5], la quale è associata alla riga *"lcd.createChar(0, degree);"* in *"void setup()"*, eseguita mediante *"lcd.write((uint8_t)0)"*. Come facilmente intuibile, il resto del codice si occupa di visualizzare l'ora e la data ed acquisire e rendere disponibili alla seriale ed al di-

```
byte degree[8] = { // CHARACTER "°C" DEFINITION
  B10111,
  B01000,
  B10000,
  B10000,
  B10000,
  B01000,
  B00111,
};
```

Fig. 4

splay LCD i dati provenienti dai 3 sensori analogici. I dati acquisiti dai 3 ingressi analogici vengono mediati su 100 campioni prima di essere inviati alla seriale e visibili sul display, questo al fine di ottenere un risultato analogo ad un filtraggio di tipo passa basso e ridurre l'effetto del rumore e le fluttuazioni; le funzioni di trasferimento utilizzate per determinare i valori delle tre grandezze meteorologiche sono quelle riportate sui datasheet dei rispettivi produttori.
Il codice è stato testato sull'ultima versione dell'IDE disponibile al momento della realizzazione dell'articolo, la 1.0.1 per cui non dovreste incontrare alcun problema nella sua compilazione; si ricorda che è possibile servirsi anche della versione 0.23 dell'IDE, semplicemente modificando l'istruzione *"lcd.write((uint8_t)0);"* con *"lcd.write(0);"*.
Come abbiamo detto i sensori scelti non necessitano di calibrazione; ad ogni modo, qualora si volesse migliorare l'accuratezza dei dati e compensare gli scostamenti dovuti alle tolleranze come precedentemente discusso, è possibile correggere i valori forniti dai tre integrati assegnando opportunamente un valore alle grandezze "TCORR", "RHCORR" e "PCORR", in base a quanto sperimentalmente riscontrato caso per caso; per fare ciò è necessario, ovviamente, disporre di una stazione meteo o trasduttori dall'accuratezza nota e ben definita con cui effettuare la comparazione.
Il valore di pressione indicato dai barometri rappresenta la pressione atmosferica relativa mentre il trasduttore da noi impiegato riporta la pressione assoluta. La pressione atmosferica dipende dall'altitudine e questo legame viene utilizzato dalla maggior parte degli altimetri che sono di tipo barometrico; la pressione

Listato 1

```
//CODE MADE BY VINCENZO MENDOLA USING THE SAME RTC LIBRARY AND RTC CODE MADE
BY FUTURA ELETTRONICA//

// LIBRARIES

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

// LCD PINS

LiquidCrystal lcd(7, 6, 5, 4, 3, 2);

RTC_DS1307 RTC;

// VARIABLES DEFINITION AND INITIALIZATION

#define TEMP 2      //TEMPERATURE ACQUISITION ON ANALOG PIN 2
#define UMID 1      //HUMIDITY ACQUISITION ON ANALOG PIN 1
#define PRESS 0     //PRESSURE ACQUISITION ON ANALOG PIN 0
float val = 0.0;
float T= 0.0;
double umidita = 0.0;
double RH = 0.0;
double RHout = 0.0;
double UM = 0.0;
double Pascal=0.0;
double PS=0.0;
double P=0.0;
float VADC= 5;
int DPR = 0;
int RHCORR = 0;
int PCORR = 0;
int TCORR= 0;
double STAMPA_T = 0;
double STAMPA_U = 0;
double STAMPA_P = 0 ;
byte degree[8] = { //  CHARACTER "°C" DEFINITION
  B10111,
  B01000,
  B10000,
  B10000,
  B10000,
  B01000,
  B00111,
};

void setup() {
  Serial.begin(9600);
  lcd.begin(16, 2);
  lcd.createChar(0, degree); // "°C" SYMBOL
  Wire.begin();
  RTC.begin();
  RTC.sqw(1);       //0 Led off - 1 Freq 1Hz - 2 Freq 4096kHz - 3 Freq 8192kHz - 4 Freq 32768kHz

  if (! RTC.isrunning()) {
    Serial.println("RTC is NOT running!");
   // following line sets the RTC to the date & time this sketch was compiled
   // RTC.adjust(DateTime(__DATE__, __TIME__));
  }
}

void loop() {
  // RTC LCD OUTPUT

  DateTime now = RTC.now();
  lcd.setCursor(0, 1);
  if (now.hour() < 10) {
    lcd.print("0");
  }
  lcd.print(now.hour(),DEC);     // HOUR
  lcd.print(":");
  if (now.minute() < 10) {
    lcd.print("0");
  }
  lcd.print(now.minute(), DEC);// MINUTES

  lcd.setCursor(6, 1);
  if (now.day() < 10) {
    lcd.print("0");
  }
  lcd.print(now.day(), DEC);  // DAY
  lcd.print('-');
  if (now.month ()<10) {
    lcd.print("0");
  }
  lcd.print(now.month(), DEC);//MONTH
  lcd.print('-');
  lcd.print(now.year(), DEC);  //YEAR


  // SERIAL RTC OUPUT

  if (now.hour() < 10) {
    Serial.print("0");
  }
```

(Continua)

Listato 1 - SEGUE

```
Serial.print(now.hour(),DEC);
Serial.print(":");
if (now.minute() < 10) {
  Serial.print("0");
}
Serial.print(now.minute(), DEC);
Serial.print(":");
if (now.second() < 10) {
  Serial.print("0");
}
Serial.print(now.second(), DEC);
Serial.print("; ");
if (now.day() < 10) {
  Serial.print("0");
}
Serial.print(now.day(), DEC);
Serial.print('-');
if (now.month ()<10) {
  Serial.print("0");
}
Serial.print(now.month(), DEC);
Serial.print('-');
Serial.println(now.year(), DEC);
Serial.println();

// SERIAL METEO OUTPUT

STAMPA_T= (temp());
STAMPA_U= (readUMID());
STAMPA_P = (pressure());
Serial.print("TEMPERATURA ");
Serial.print(STAMPA_T);
Serial.write(176);
Serial.print("C; ");
Serial.print("UMIDITA' ");
Serial.print(STAMPA_U);
Serial.print("%; ");
Serial.print("PRESSIONE ");
Serial.print(STAMPA_P);
Serial.println("mbar");

// LCD METEO OUTPUT

lcd.setCursor(0, 0);
lcd.print(STAMPA_T,1); //SHOW ONLY THE FIRST DECIMAL
lcd.write((uint8_t)0); //PRINT "°C" CHARACTER (IDE 1.0.1)
delay(200);

lcd.setCursor(6, 0);
lcd.print(STAMPA_U,1);//SHOW ONLY THE FIRST DECIMAL
lcd.setCursor(10,0);
lcd.print("%");
delay(200);

lcd.setCursor(12, 0);
lcd.print(STAMPA_P,0);//SHOW ONLY THE INTEGER PART
delay(200);
}

float temp() {
  double nread = 100.0;           // NUMBER OF READINGS
  double somma = 0.0;
  for (int i=0; i<nread; i++)
  {
    val = analogRead(TEMP);
    T = (((VADC/1024.0*val)-0.5)* 100)+TCORR;   //TEMPERATURE
    somma += T;
  }
  delay(100);
  return (somma/nread);
}

double readUMID(){
  double nread = 100.0;           // NUMBER OF READINGS
  double somma = 0.0;
  for (int i=0; i<nread; i++)
  {
    UM = analogRead( UMID );
    RHout=(((UM*VADC/1024.0/3.3)-0.1515)/0.00636)+RHCORR;    //HUMIDITY
    somma += RHout;
  }
  delay(100);
  return  ( somma / nread );
}

float pressure(){
  double nread = 100.0;           // NUMBER OF READINGS
  double somma = 0.0;
  for (int i=0; i<nread; i++)
  {
    Pascal=analogRead(PRESS);
    P=(((Pascal*VADC/1024)/VADC+0.095)/0.009)*10+DPR+PCORR;  //PRESSURE
                                                  // TRANSFERT FUNCTION
    somma += P;
  }
  delay(100);
  return  ( somma / nread );
}
```

relativa è la pressione calcolata al livello del mare.
Un modo semplice per avere l'indicazione della pressione relativa è quello di aggiungere al nostro codice un valore, che abbiamo per comodità definito DPR, che può essere facilmente ottenuto dalla differenza tra il valore di pressione visualizzato da una stazione meteo che indica appunto la pressione assoluta e il valore fornito dal nostro trasduttore di pressione.
Un ottimo complemento del METEO SHIELD (**Fig. 5**) è il SOLAR SHIELD (**Fig. 6**), venduto da Futura Elettronica con il codice 7300-SOLARSHIELD con cui è possibile trasformare facilmente l'Arduino dotato di METEO SHIELD (**Fig. 7**) in una completa ed autonoma stazione meteorologica in grado di funzionare utilizzando solamente una

per il MATERIALE

Tutti i componenti utilizzati in questo progetto sono di facile reperibilità. Il master del circuito stampato può essere scaricato dal sito della rivista così come il firmware utilizzato. Meteo-Shield è anche disponibile in kit o già montato al prezzo di 42,00 Euro; sono compresi tutti i sensori, la basetta forata e serigrafata, la batteria al litio, e le minuterie. Non è compreso il display LCD retroilluminato con LED bianchi disponibile separatamente (cod.1446-LCD16X2WB, Euro15,00) né la board Arduino che costa 24,50 Euro. Tutti i prezzi si intendono IVA compresa.

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

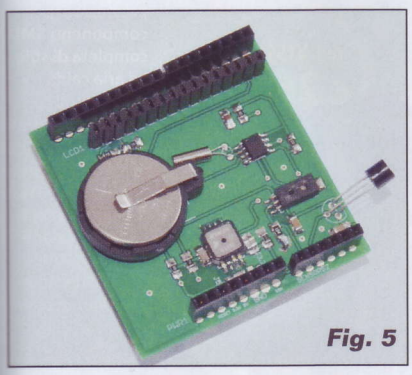


Fig. 5

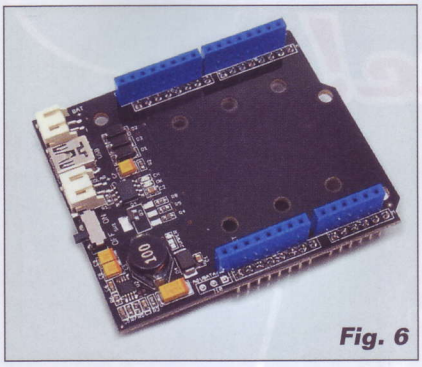


Fig. 6

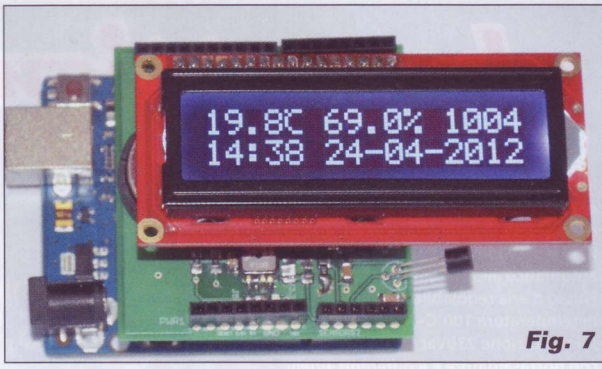
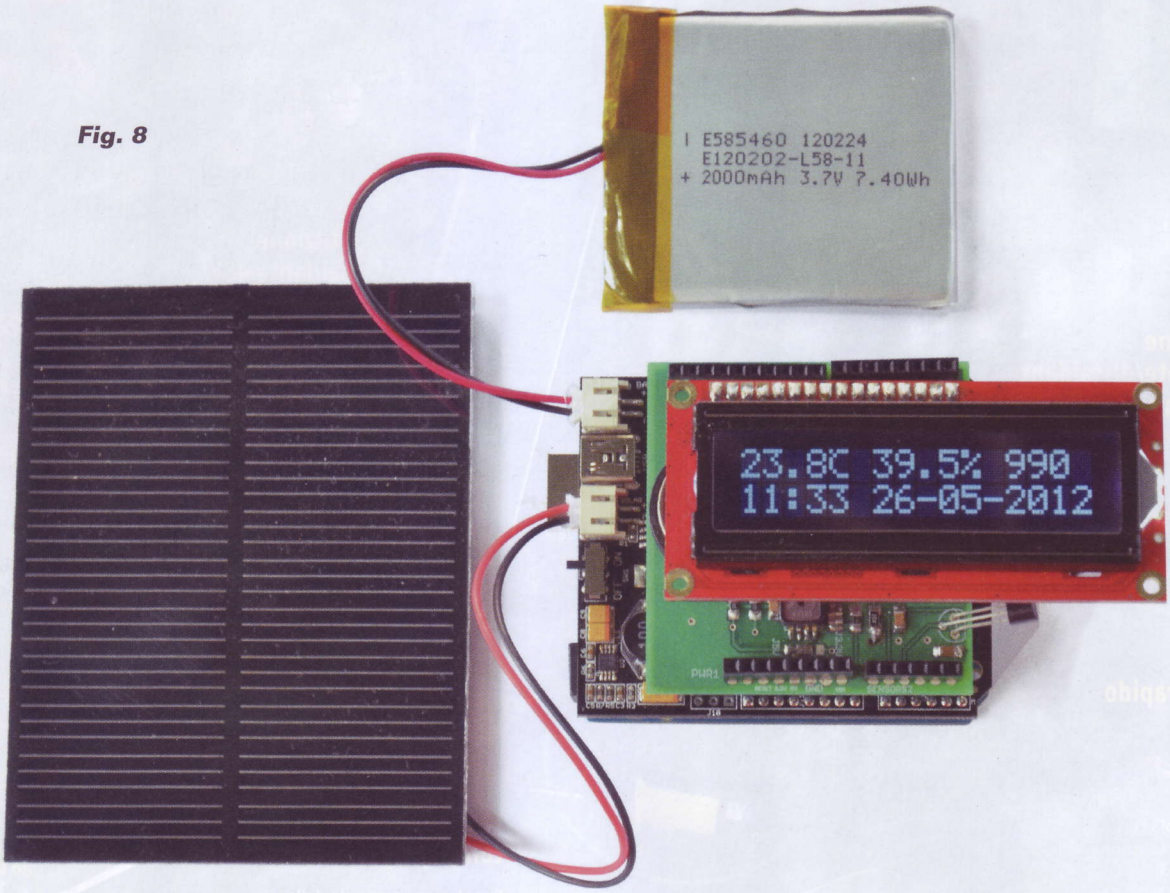


Fig. 7

Fig. 8



batteria al litio, ricaricata da un pannello solare mediante appunto il SOLAR SHIELD (**Fig. 8**). Questo è munito oltre che di due connettori a cui collegare rispettivamente la batteria e il pannello solare (ad esempio il 7500-SOL-PAN1W della Futura Elettronica), anche di una presa mini USB che può essere utilizzata per ricaricare la batteria tramite il PC: un LED rosso segnala la corretta applicazione della tensione di carica e l'inizio di questa mentre un LED verde ne indica il termine. Sullo stesso lato in cui sono disposti questi connettori è presente un piccolo switch: in posizione "ON" il circuito applica la tensione di 5V al relativo terminale del connettore SIL, alimentando di conseguenza l'Arduino ed eventuali altri shield a questo connessi.

È possibile trovare ulteriori dettagli inerenti la scheda, tra cui lo schema elettrico completo ed i file Eaglecad, sul wiki ufficiale [6]. ■

RIFERIMENTI:

[1] http://ww1.microchip.com/downloads/en/DeviceDoc/21942e.pdf

[2] http://sensing.honeywell.com/index.cfm/ci_id/155943/la_id/1/document/1/re_id/0

[3] http://www.freescale.com/files/sensors/doc/data_sheet/MPXA6115A.pdf

[4] http://datasheets.maxim-ic.com/en/ds/DS1307.pdf

[5] http://arduino.cc/en/Reference/LiquidCrystalCreateChar

[6] http://www.seeedstudio.com/wiki/Solar_Charger_Shield_v2.0b