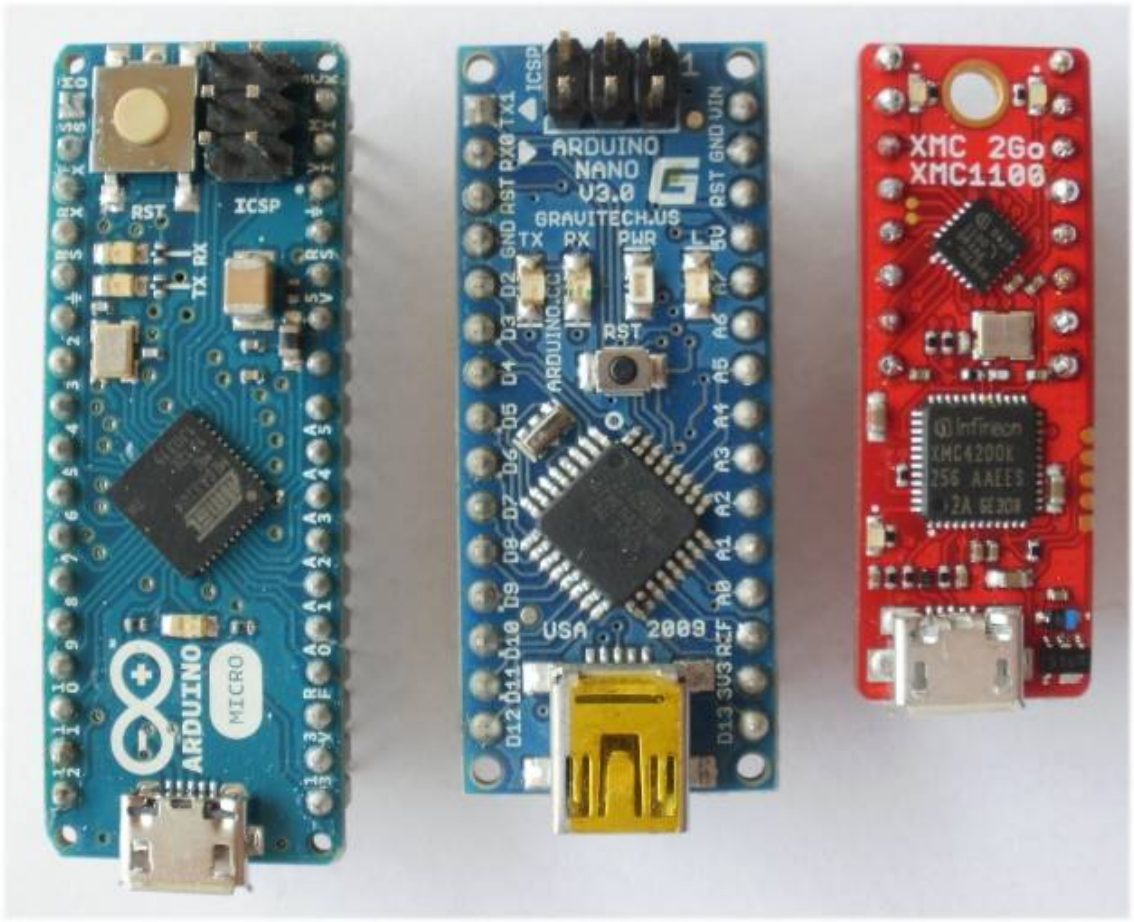
Arduino Micro e BMP180 Bosch per realizzare una Weather Station

Posted By *Adriano Gandolfo* On 4 giugno 2015 @ 5:50 In Arduino,Makers & Progetti Fai Da Te | 6 Comments

In un recente articolo pubblicato su Elettronica Open Source dal titolo "Quale scheda Arduino scegliere per il mio progetto? [1]" sono stati elencati vari modelli della serie Arduino. Analizziamo nel dettaglio uno dei modelli presentati: la scheda ARDUINO MICRO. Questa è stata sviluppata in collaborazione con Adafruit [2], e utilizza come processore l'ATmega32U4, la scheda su una superficie di neanche 9 centimetri quadrati mette a disposizione 20 pin I/O digitali, di cui 7 PWM e 12 ingressi analogici. Il modulo include tutto ciò che serve per il supporto del microcontrollore come l'interfaccia USB, due regolatori di tensione, vari led, con la possibilità di connetterlo con semplicità al computer ed essere subito operativo, per implementare i propri progetti. Perciò, se la vostra applicazione ha bisogno di una miniaturizzazione "spinta", sicuramente questa scheda potrà fare al caso vostro. Nell'articolo vedremo anche un'applicazione che permette di visualizzare su display TFT i dati provenienti dal sensore BMP180 e quindi avere i valori di pressione, temperatura e altitudine sul livello del mare.

Di schede di controllo di piccole dimensioni ve ne sono molte in commercio, nel caso di Arduino in un precedente articolo **Costruzione del robot LittleBot – Scheda di controllo** [3], avevamo visto per esempio **Arduino Nano** [4] , mentre in **un'altro articolo** [5] si vede un possibile utilizzo di una scheda **XMX 2Go** prodotta dalla Infineon.



[6]

Confronto di dimensioni tra Arduino Micro, Nano e la scheda XMC 2Go [5]

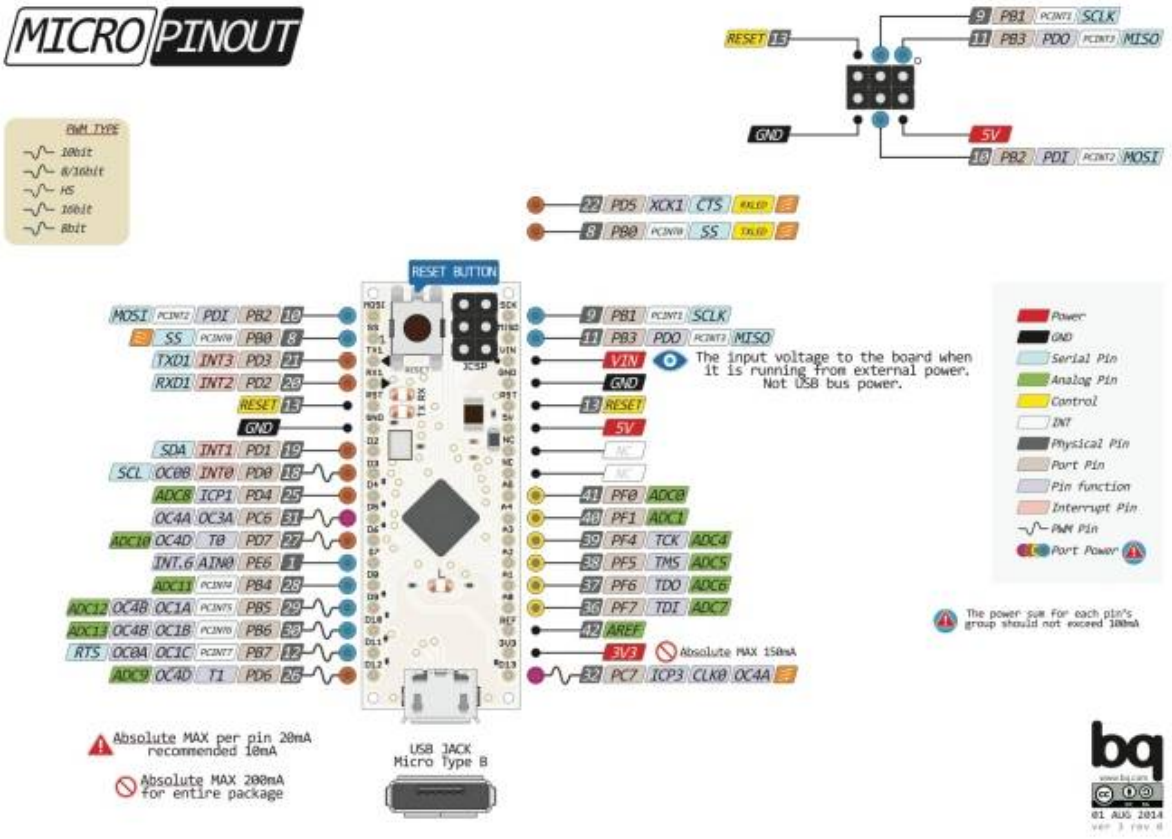
Vediamo ora di analizzare in dettaglio le caratteristiche offerte da **Arduino Micro** [7]

[8]

CARATTERISTICHE SALIENTI

Microcontroller	Atmel ATmega32u4

Velocità di clock	16 MHz
Tensione di funzionamento	5 V
Tensione in ingresso	7-12 V (consigliati)
Tensione in ingresso	6-20 V (limiti)
Pin I/O digitale	20 (7 forniscono in uscita segnali PWM e 10 possono essere usati come input analogici e 3 solo digitali)
Pin analogici	6
Corrente continua per I/O	40 mA
Corrente per Pin alimentati a 3.3V:	50mA
Flash Memory	32 KB (di cui 4 KB utilizzati dal bootloader)
SRAM	22.5 KB
EEPROM	1 KB
Dimensioni	48 x 18 mm



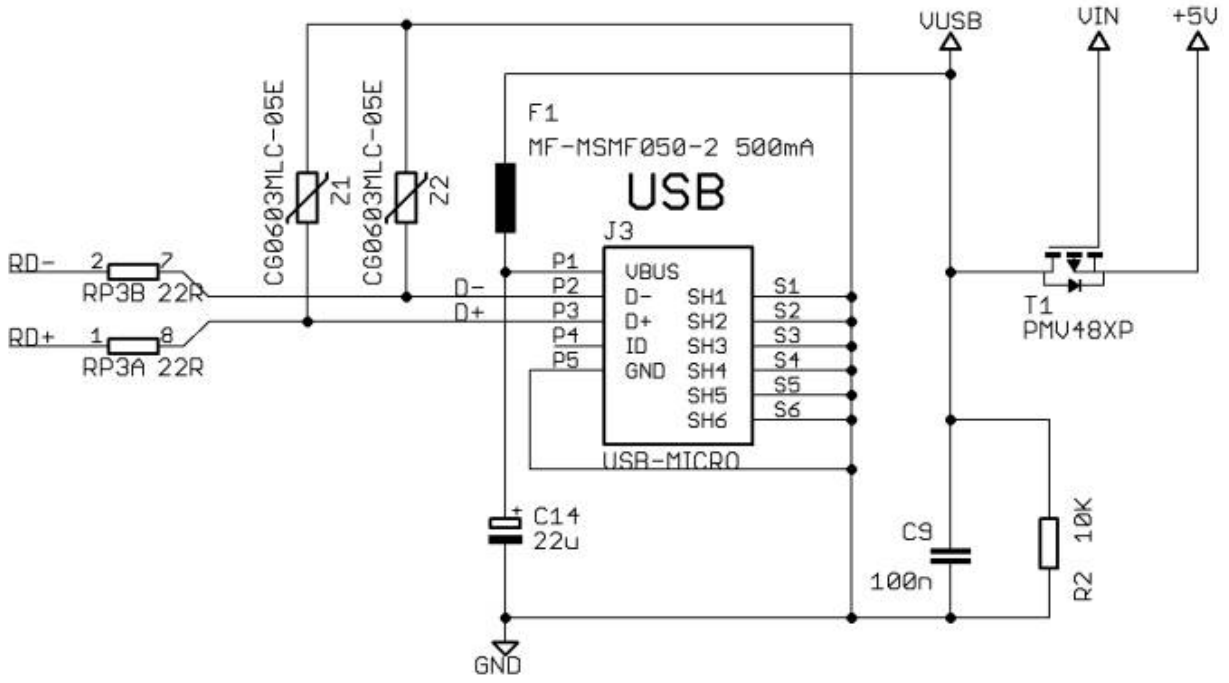
[9]

Passiamo quindi ad analizzare le varie sezioni in cui può essere suddiviso lo **schema elettrico** [10] della scheda: **alimentazione, processore, memoria, pin d'input e output, connettore ICSP, led di segnalazione.**

Sezione alimentazione

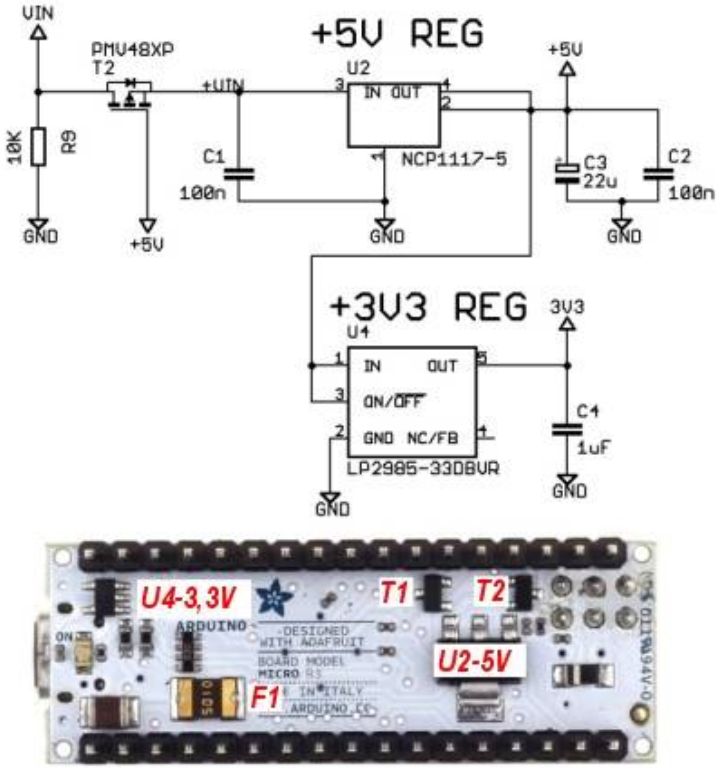
Arduino Micro può essere alimentato tramite **due diverse fonti di energia**: prelevata tramite il connettore dell'interfaccia **USB** micro o con un alimentatore esterno connesso al **Pin Vin.**

La fonte di alimentazione è selezionata automaticamente, tramite l'attivazione del mosfet, **T1** tipo **FND340P** [11] o **PMV48XP** [12], usato come switch, che si occupa di connettere o meno i 5V USB alla scheda.



[13]

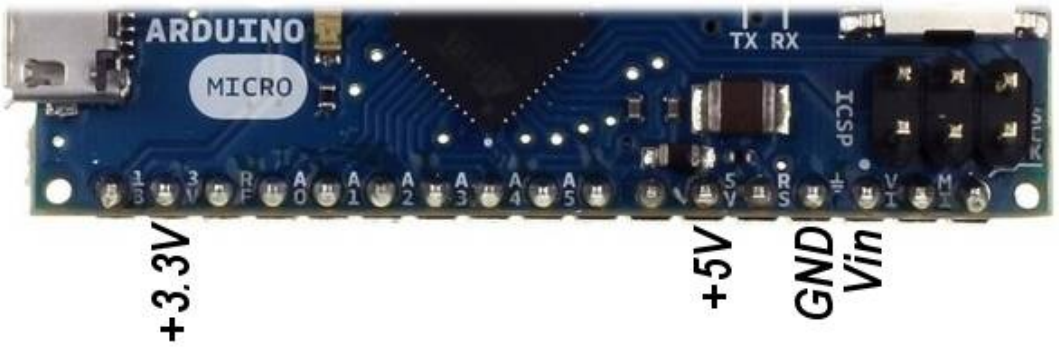
L'alimentazione da USB è protetta da un POLYFUSE azzerabile **F1** tipo **MF-MSF050-2** [14] da 500 mA, che protegge la porta USB del computer da cortocircuiti e sovracorrenti.
Sebbene la maggior parte dei computer abbia già una protezione interna, il fusibile fornisce un ulteriore livello di protezione.
In questo caso, se una corrente maggiore di 500 mA è prelevata dalla porta USB, il fusibile interromperà automaticamente il collegamento fino a quando il sovraccarico non sarà rimosso.
La tensione è livellata e filtrata tramite il condensatore C9 da 100 nF e l'elettrolitico C14 da 22uF.
La funzione della resistenza R2 da 10kΩ è quella di non far fluttuare il segnale di comando del mosfet T1.
L'alimentazione esterna (non USB) può essere fornita da un alimentatore DC o tramite una batteria, questi dovranno essere collegati ai pin **Gnd** e **Vin**.
La scheda può funzionare con una tensione che teoricamente può essere compresa tra i 6 e i 20 volt.
In pratica però, se si forniscono meno di 7V, la tensione continua generata potrebbe essere inferiore ai 5V e in questo caso il funzionamento della scheda può essere instabile.
D'altra parte, fornire una tensione maggiore di 12V, può portare al surriscaldamento con relativo danneggiamento del regolatore di tensione.
Per questo motivo l'intervallo raccomandato di tensione è compreso **tra i 7 e i 12 volt**.
La parte del circuito di regolazione è formata per quanto riguarda l'uscita a +5V, dall'integrato **U2** tipo **NCP1117-5** [15], che è un regolatore di tensione fisso a basso dropout con una corrente d'uscita massima di 1 A.
La tensione d'uscita è livellata tramite i condensatori C1 e C2 da 100nF e dall'elettrolitico C3 da 22µF.



[16]

Nel caso fosse disponibile la tensione di +5V fornita tramite la porta USB, oppure da fonte esterna tramite l'apposito PIN, l'attivazione del secondo mosfet, **T2** tipo **FND340P** o **PMV48XP**, usato anche in questo caso come switch, si occupa di bypassare l'integrato **U2**.
Nel caso invece, la tensione da USB non fosse presente, la tensione sarebbe prelevata tramite l'ingresso **Vin**.
La funzione della resistenza R9 da 10kΩ è quella di non far fluttuare il segnale di comando del mosfet.
Arduino Micro è anche in grado di fornire in uscita una tensione di +3,3V, questa non è direttamente utilizzata dalla scheda ma è presente per utilizzi esterni.
Per ottenere questa tensione che è ottenuta riducendo la tensione di +5V, è presente l'integrato **U4** tipo **LP2985-33DBVR** [17], si tratta di un regolatore di tensione fissa a basso rumore e basso dropout con possibilità di opzione shutdown, nel nostro caso non utilizzata dato che il pin risulta permanentemente connesso alla fonte di alimentazione.
La massima corrente d'uscita teorica è di 150 mA, anche se dalle caratteristiche della scheda è consigliato un assorbimento massimo di 50 mA.
La tensione è livellata dal condensatore elettrolitico C4 da 1uF.
Riassumendo i pin di alimentazione presenti sulla scheda Arduino Micro, sono i seguenti:

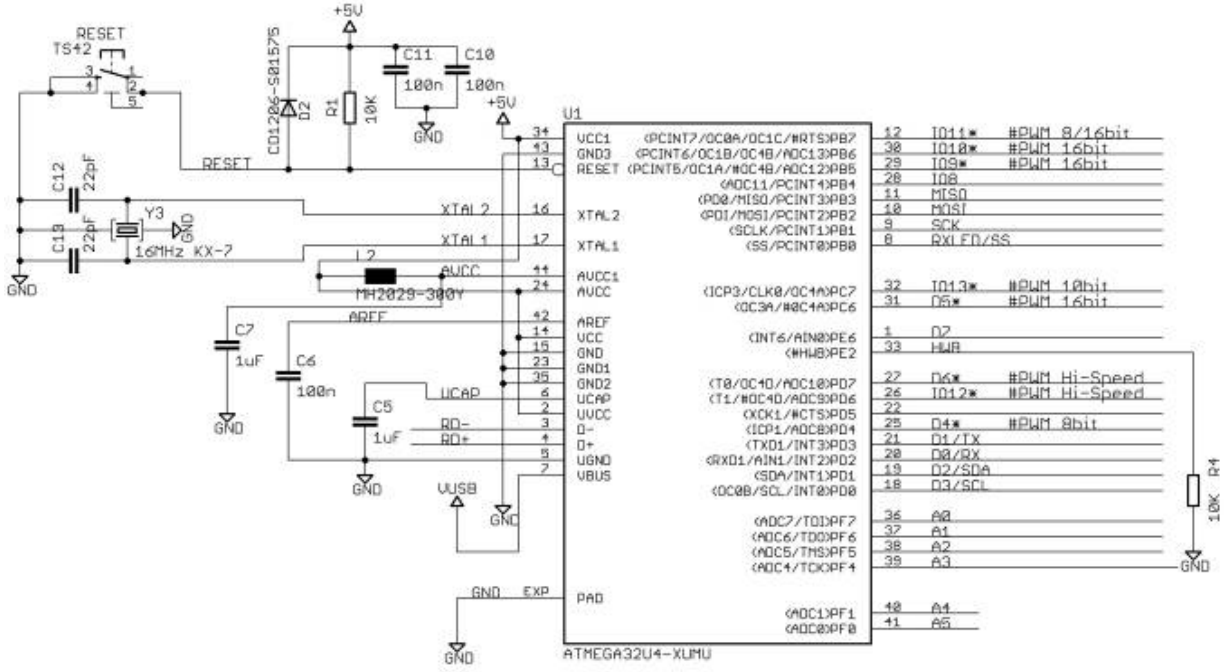
VI. La tensione d'ingresso alla scheda Arduino quando sta utilizzando una sorgente di alimentazione esterna (in contrapposizione a 5 volt dalla connessione USB o altra fonte di alimentazione regolata).
5V. tensione stabilizzata utilizzata per alimentare il microcontrollore e altri componenti sulla scheda. Questo può venire sia da VIN tramite un regolatore a bordo o essere fornita da USB o da un fonte esterna con un valore di 5V.
3V. Sono forniti 3.3 volt generati dal regolatore di bordo. Assorbimento massimo è di 50 mA.
GND pin di terra.



[18]

Sezione processore

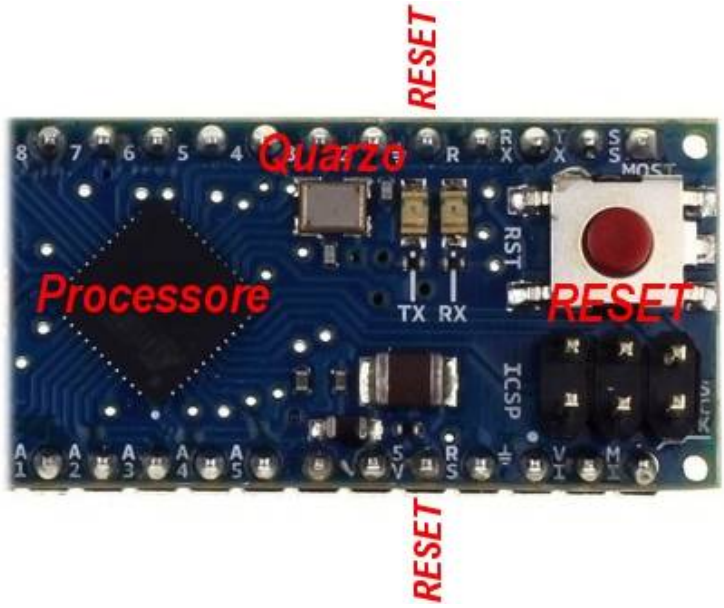
La scheda utilizza lo stesso tipo di processore della **Arduino Leonardo** [19] e **Arduino Esplora** [20], un microcontrollore prodotto dalla AVR tipo **ATMEGA32U4** [21] siglato sullo schema **U1**, che opera a una frequenza di **16 MHz**: è connesso a una porta USB ed è in grado di agire come un dispositivo client USB, come un mouse o una tastiera.



[22]

L'ATMEGA32U4 appare al computer come una porta COM virtuale, lo stesso processore funziona anche come dispositivo USB secondo lo standard 2.0.

L'integrato supporta, inoltre, la comunicazione **SPI** a cui si può accedere tramite la **libreria SPI** [23].
Nel circuito è presente un pulsante di reset (duplicato anche su due pin della scheda), con cui è possibile riavviare la scheda.



[24]

Memoria

Il processore **ATMEGA32U4** ha 32 KB di memoria flash, anche chiamata flash memory, di cui 4 KB sono utilizzati per il bootloader.
Sono poi presenti 2,5 KB di **SRAM** (acronimo di **S**tatic **R**andom **A**ccess **M**emory), che è un tipo di RAM che non richiede refresh.
E' inoltre presente 1 KB di memoria **EEPROM** (acronimo di **E**lectrically **E**rasable **P**rogrammable **R**ead-**O**nly **M**emory), in cui è possibile memorizzare piccole quantità di dati che devono essere mantenuti quando è tolta l'alimentazione elettrica (per esempio la configurazione di un dispositivo).

Per la lettura/scrittura di quest'ultimo tipo memoria è possibile utilizzare la **libreria EEPROM** [25].

Pin di Input e Output

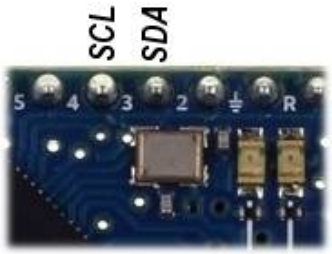
Ciascuno dei 20 I/O pin digitali di Arduino Micro può essere utilizzato sia come ingresso che come uscita, per fare questo si utilizzeranno le funzioni **pinMode ()** [26], **digitalWrite ()** [27], e **digitalRead ()** [27].
I pin operano a 5 volt e ognuno può fornire o ricevere un massimo di 40 mA di corrente.
Internamente è presente una resistenza di pull-up (sconnessa di default) del valore di 20-50 kΩ.
Alcuni pin hanno funzioni specializzate.

Serial: 0 (RX) e 1 (TX). La scheda è dotata di due pin per la comunicazione seriale di tipo TTL a 5V. I pin N.0 e N.1 sono collegati direttamente ai corrispondenti pin del processore 32U4 Piedino 21 TXD1 e 20 RXD1.

[28]

TWI: 2 (SDA) e 3 (SCL). La scheda supporta il protocollo hardware TWI detto anche I2C formato da due linee seriali di comunicazione:

SDA (**S**erial **DA**ta line) per i dati;
SCL (**S**erial **C**lock **L**ine) per il clock.



[29]

Per utilizzarla occorre far ricorso alla **Liberia Wire** [30].

Interrupt esterni: 0 (RX), 1 (TX), 2 e 3 Questi pin possono essere configurati per attivare un interrupt su un valore basso, un fronte di salita o di discesa, o una variazione di valore. Vedere la funzione **attachInterrupt ()** [31] per i dettagli.

Pin PWM, i pin 3, 5, 6, 9, 10, 11 e 13 sono utilizzabili come uscite **PWM** [32] (acronimo di **P**ulse **W**idth **M**odulation) a 8 bit utilizzando la funzione **analogWrite()** [33].

Ingressi analogici: A0-A5, A6 - A11 (su pin digitali 4, 6, 8, 9, 10, e 12) Arduino Micro può essere configurato per avere sino a 12 ingressi analogici, i pin da **A0 a A5** sono direttamente etichettati sui pin stessi, mentre per gli altri, da **A6 a A11**, sono disponibili accedendo tramite codice di programmazione utilizzando le costanti da A6 ad A11 poiché sono condivise rispettivamente sui pin digitali 4, 6, 8, 9, 10, e 12.
Tutto ciò può anche essere utilizzato come I /O digitale.
Ogni ingresso analogico fornisce 10 bit di risoluzione (cioè 1024 valori differenti).
Per default la misura ingressi analogici è per valori di tensione compresi tra 0 e +5 volt, anche se è possibile cambiare la tensione massima utilizzando il pin **AREF** e la funzione **analogReference ()** [34].

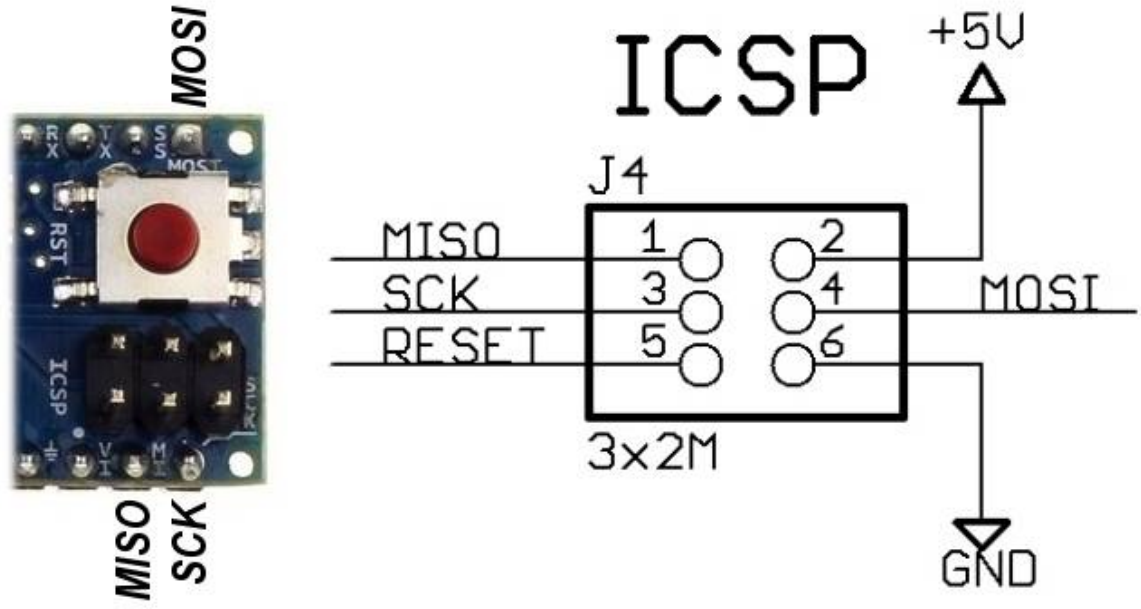
Oltre ai pin sin qui citati, sono presenti un paio di altri pin con funzioni speciali, questi sono:

AREF: Tensione di riferimento per gli ingressi analogici. Utilizzato con il comando **analogReference ()** [34].

Reset: Portare questa linea LOW per resettare il microcontrollore. Tipicamente è utilizzata per aggiungere un pulsante di reset, quando quello presente sulla scheda non è raggiungibile.

Connettore ICSP

A lato del pulsante di reset, è presente il connettore **J4** denominato **ICSP** (acronimo di **I**n **C**ircuit **S**erial **P**rogramming) per l'eventuale programmazione senza rimozione dal circuito del processore.



[35]

I pin del connettore sono inoltre duplicati sulla scheda e sono etichettati **MISO**, **MOS**I e **SCK**.

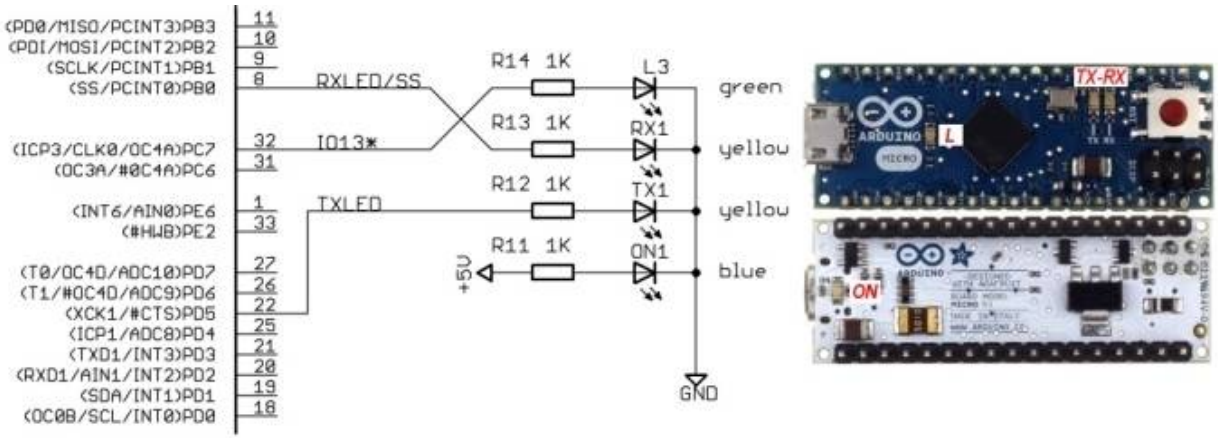
Pin J4	Pin del Arduino Micro	Funzione
1	MISO	MISO (Master Input Salve Output)
2	+5V	+5V
3	SCK	SCK (Serial Clock Signal)
4	MOSI	MOSI (Master Output Slave Input)
5	RESET	RESET
6	GND	GND

LED di segnalazione

Sulla scheda sono presenti quattro led:

- **due di colore giallo (RX1 eTX1)** che segnalano il traffico di dati sulla linea seriale, il led RX1 è connesso al pin RX_LED / SS, che indica l'attività di trasmissione durante la comunicazione USB, ma può essere anche utilizzato come slave.
- **uno di colore verde (L3)** collegato al pin digitale 13. Quando il pin è HIGH, il LED è acceso, quando il pin è LOW, esso è spento.
- **uno di colore blu (ON1)**, situato nella parte inferiore della scheda, segnala la presenza di alimentazione

Tutti i led sono di tipo SMD è hanno una resistenza di limitazione dal valore di 1kΩ.



[36]

Differenze tra Arduino Micro e Arduino Uno

Terminata la descrizione di come funziona la scheda, passiamo ora ad analizzare alcune differenze che esistono tra una normale scheda **Arduino UNO** e una scheda **Arduino MICRO**.
In generale, si programma e si utilizza la scheda Arduino Micro come si farebbe con altre schede Arduino. Vi sono, tuttavia, alcune differenze importanti.
Come abbiamo visto, la scheda Arduino Micro ha un solo processore sia per l'esecuzione del programma che per la comunicazione USB con il computer.
Arduino Uno e altre piattaforme utilizzano invece il classico FT232 o un Atmel dedicato, come il modello **ATMEGA8U2** su **Arduino UNO R3,** [37] per questa funzione, il che significa che la connessione USB al computer rimane connessa indipendentemente dallo stato del microcontrollore principale.
Combinando queste due funzioni in un unico processore, la porta seriale è finalmente libera per il programma utente in qualsiasi momento, lo stack USB prevede anche la parte HID che permette quindi ad Arduino Micro di emulare un mouse o una tastiera.
Occorre però segnalare che qualche volta, vi potrebbero essere dei problemi di restart quando si apre la connessione seriale, per cui si potrebbe perdere la connessione con la scheda quando si avvia il programma.

Rinumerazione della seriale al reset.

Dal momento che la scheda non ha un chip dedicato per gestire la comunicazione seriale, significa che la porta seriale è virtuale, è solamente una routine software, sia sul sistema operativo, che sulla scheda stessa.
Proprio come il computer crea un'istanza del driver della porta seriale quando si collega qualsiasi Arduino, la scheda Arduino Micro creerà un'istanza seriale ogni volta che esegue il suo bootloader.
Questo significa che ogni volta che si reimposta la scheda, la connessione seriale USB sarà interrotta e in seguito ristabilita.
La scheda scomparirà momentaneamente dall'elenco delle porte seriali e la lista si rinumererà.
Qualsiasi programma che ha una connessione seriale aperta con la scheda perderà la sua connessione.

Emulazione di tastiera e mouse.

Un vantaggio di utilizzare un singolo chip per l'esecuzione dei programmi e il collegamento USB è aumentata dalla flessibilità nella comunicazione con il computer.
Mentre la scheda appare come una porta seriale virtuale con il sistema operativo (chiamato anche **CDC**) per la programmazione e la comunicazione (come con Arduino Uno), può anche comportarsi come un (**HID** acronimo di **H**uman **I**nterface **D**evice) come una tastiera o come un mouse.

Separazione della comunicazione seriale dalla porta USB.

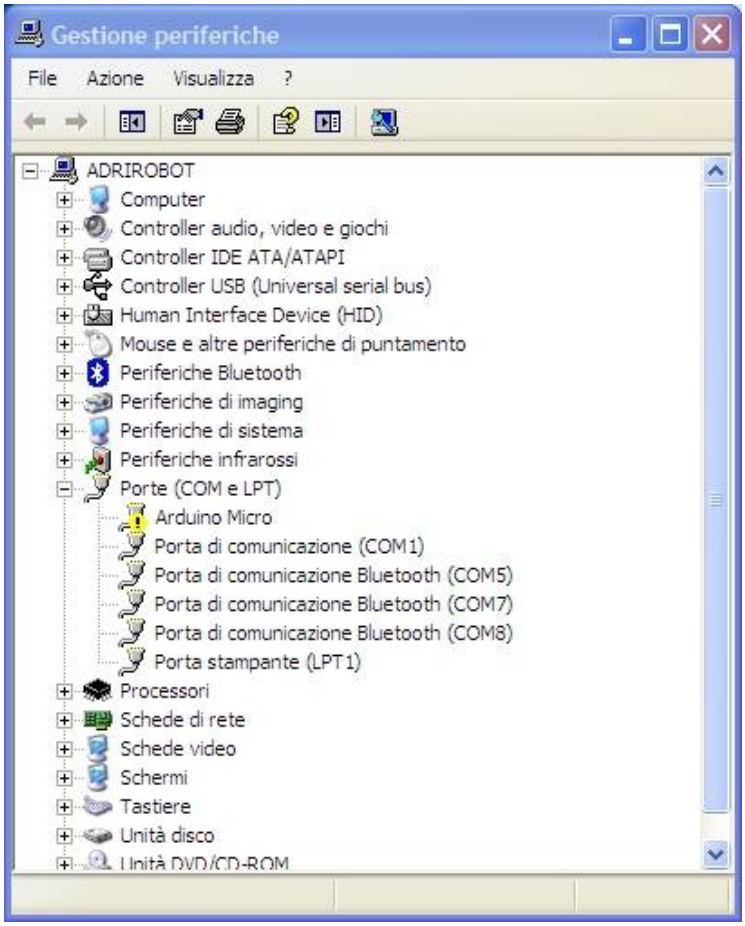
Sulla scheda Arduino Micro la comunicazione con il PC è realizzata con un driver seriale virtuale tramite la porta USB, non vi è collegamento ai pin fisici **0** e **1** come è sulla Arduino Uno.
Per utilizzare la porta seriale hardware UART TTL (5V) (**pin 0 e 1, RX e TX**), occorre utilizzare **Serial1**.
Il software Arduino include un monitor seriale che consente di inviare e ricevere dati dalla scheda Arduino.
I LED RX e TX sulla scheda lampeggeranno quando vi sarà traffico attraverso la connessione USB al computer (ma non per la comunicazione seriale sui pin 0 e 1).
La libreria **SoftwareSerial** [38]consente la comunicazione seriale su altri pin digitali di Arduino Micro.
Come abbiamo visto l'ATMEGA32U4 oltre alla comunicazione seriale, supporta sia la comunicazione **I2C** (TWI) sia la **SPI.**

Installazione del driver

Per installare il driver occorre collegare la scheda Arduino Micro al computer, per fare questo è necessario un cavo USB dotato di un connettore di tipo **Micro-B**.
Come abbiamo visto, il cavo USB permette sia la programmazione sia l'alimentazione della scheda.

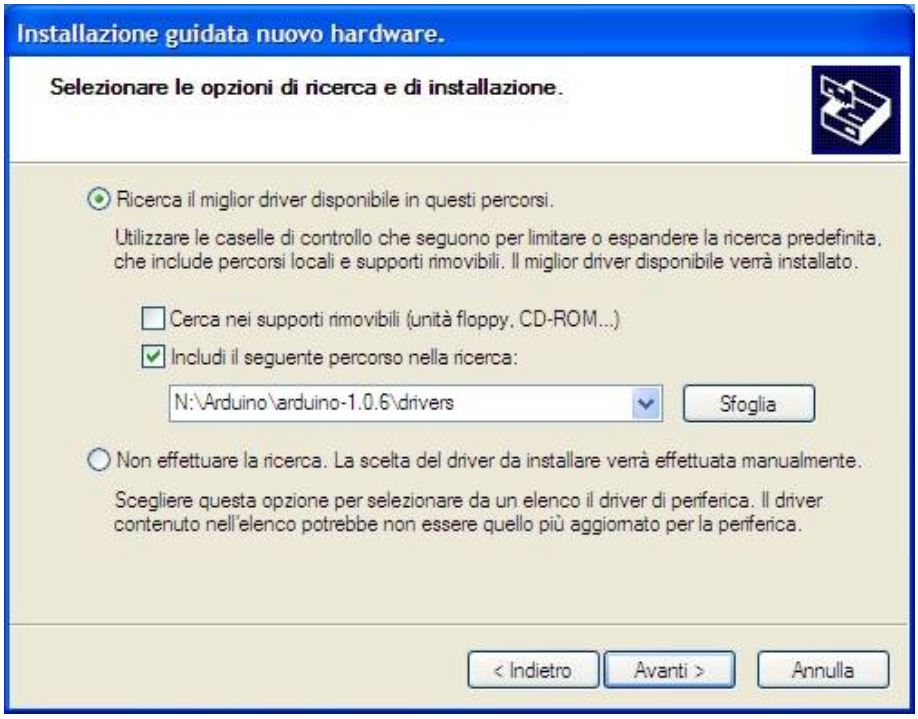
[39]

Le seguenti istruzioni si riferiscono alla versione per WINDOWS, sia per la versione XP che le successive, con piccole differenze nelle finestre di dialogo.
Una volta collegata la scheda, occorre attendere che Windows avvii il processo d'installazione del driver.
Se l'installazione non si avvia automaticamente, individuate tramite la Gestione periferiche di Windows (Start> Pannello di controllo> Hardware) nell'elenco **Arduino Micro,** che sarà evidenziato da un punto esclamativo giallo**.**
Fare clic con il tasto destro e scegliere **Aggiorna driver**.



[40]

Nella schermata successiva, scegliere "Cerca il software del driver" e fare clic su Avanti.
Fare clic sul pulsante Sfoglia. Un'altra finestra appare: passare alla cartella con il software Arduino che si è appena scaricata.
Selezionare la cartella in cui sono i driver e fare clic su OK, quindi fare clic su Avanti.



[41]

Si riceverà una notifica che la scheda non ha superato il test di Windows Logo. Fare clic sul pulsante Continua.

[42]

Dopo qualche istante, una finestra vi segnalerà che la procedura guidata per 'installazione del software per Arduino Micro è stata completata. Premere il pulsante Chiudi.

[43]

Al primo caricamento di un programma sarà richiesto d'installare il driver del Bootloader.
La procedura è simile a quella appena vista, il driver è posto nella stessa cartella impostata in precedenza.

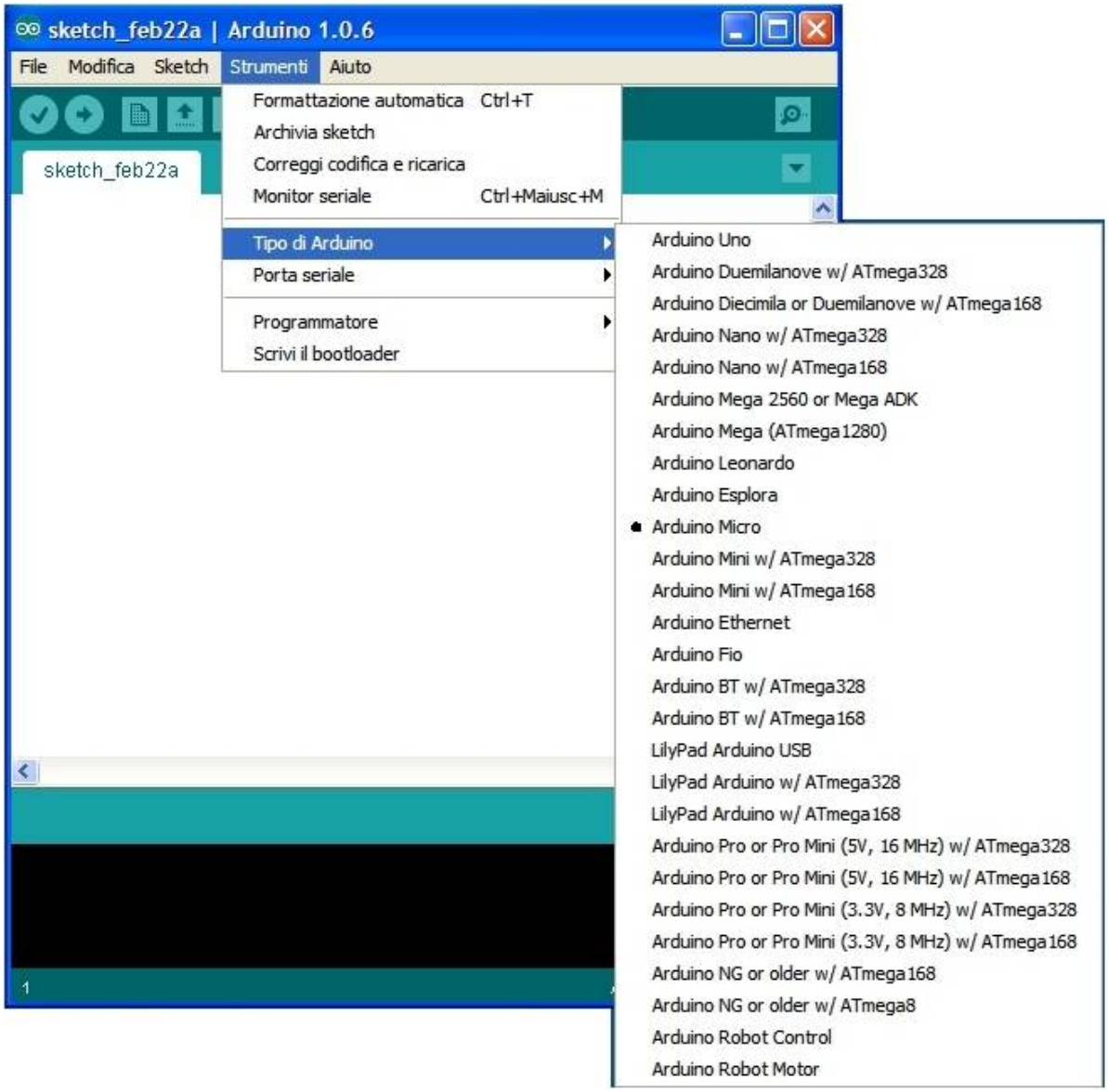
[44]

[45]

Programmazione

Arduino Micro può essere programmata con il **software IDE Arduino, ora alla versione 1.0.6** [46], che dovrebbe essere ben conosciuto dai lettori.

Per utilizzare la scheda occorrerà semplicemente selezionare "**Arduino Micro**" dal menu Strumenti> Tipo di Arduino.



[47]

Occorrerà inoltre indicare a quale porta seriale è collegato, selezionandola dal menu Strumenti> Porta seriale.
Come abbiamo visto l'ATMEGA32U4 su Arduino Micro è programmato con un bootloader che permette di caricare il nuovo codice senza l'utilizzo di un programmatore hardware esterno.
La comunicazione avviene utilizzando il protocollo AVR109.
È possibile comunque bypassare il bootloader e programmare il microcontrollore attraverso il connettore ICSP utilizzando eventualmente un altro Arduino come programmatore.

Test della scheda.

Per testare la scheda, non caricheremo il solito programma blink, ma qualcosa di più complesso: visualizzeremo su un **display TFT** i dati letti da un **sensore di BMP180** che fornisce i valori di: pressione, temperatura e altitudine sul livello del mare.
Perciò oltre alla scheda Arduino Micro sono necessari i seguenti componenti :

Una breadboard;
La scheda Arduino Micro;
Display LCD TFT Arduino;
Sensore di pressione BP180;
Alimentatore per breadboard;
Cavi colorati maschio-mascio per realizzare i collegamenti;
Una batteria 9V per l'alimentazione.
Vediamo le caratteristiche dei principali componenti utilizzati.

Display LCD TFT Arduino.

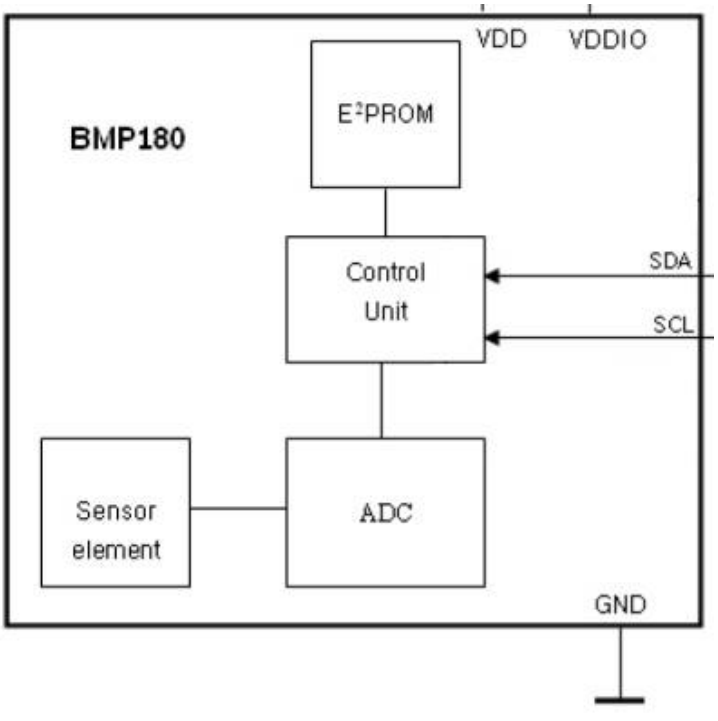
Il display qui utilizzato è lo stesso presentato nell'articolo **Scopriamo la nuova scheda Arduino Esplora** [20].

Si tratta di un **display LCD tipo TFT** [48] (Thin Film Transistor), retroilluminato che misura 1,77" pari a circa 45 mm di diagonale, con risoluzione di 160 x 128 pixel.
Il modulo misura 40x44mm circa e nella parte posteriore è presente uno slot per schede micro-SD, che tra le altre cose, permette di memorizzare le immagini bitmap da visualizzare sullo schermo.
Per la comunicazione, il display utilizza un **protocollo SPI**, per facilitare la gestione del display. E' disponibile **un'apposita libreria** [49] che è inclusa in quelle disponibili per l'IDE 1.0.5 e successive.

[50]

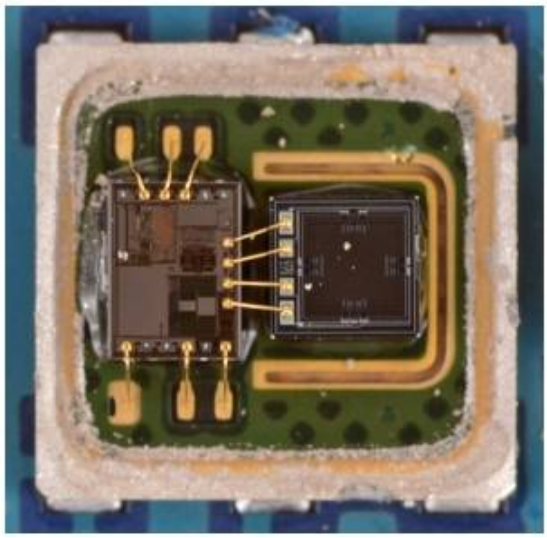
Sensore di pressione BMP180.

Il sensore utilizzato è il **BMP180 prodotto dalla BOSCH** [51], permette di misurare pressioni da 300 a 1100hPa (da -500m a +9000m relativi al livello del mare), la tensione di alimentazione è compresa tra 1.8-3,6 (VDD) e 1.62-3.6V (VDDIO), il consumo è di soli 5uA con una lettura al secondo.



[52]

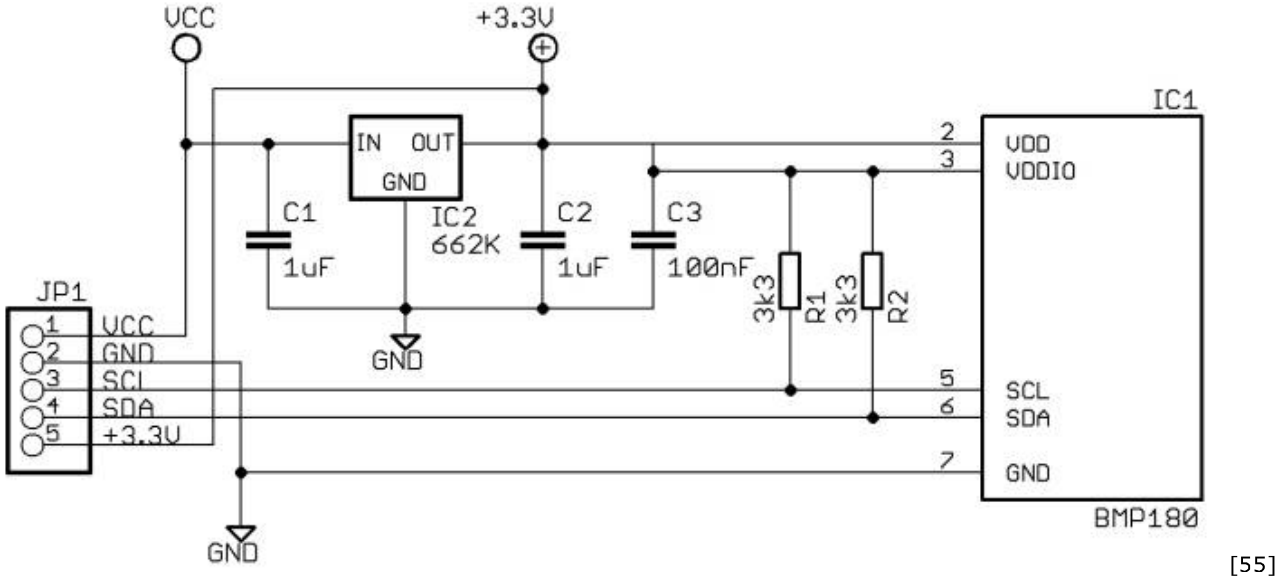
Il sensore è contenuto in un package tipo LGA con una dimensione di 3,6x3,8 mm.



[53]

Oltre al valore di pressione, il sensore fornisce anche la misurazione della temperatura; i dati sono trasmessi tramite un'interfaccia I2C, con un indirizzo fisso 0x77.
La calibrazione è fatta durante la produzione e i valori sono salvati nella memoria del sensore.
Per facilitare l'uso del sensore, questo si trova di solito già montato su una piccola basetta, dove oltre al sensore, trovano posto i pochi altri componenti necessari al suo funzionamento.

Il modulo utilizzato dispone anche di un piccolo regolatore di tensione **tipo 662K** [54] che fornisce in uscita una tensione di 3,3 V, che permette di alimentare il modulo anche con una tensione di 5V.



[55]

Schema del modulo utilizzato nel test

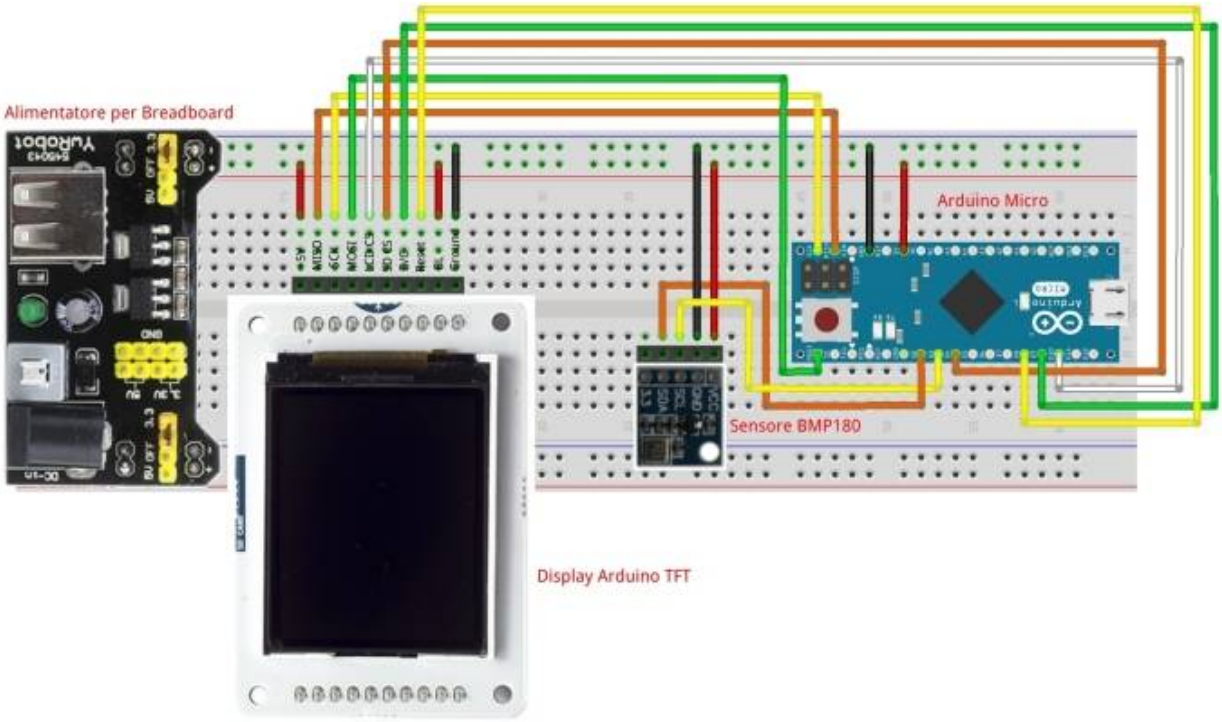
Alimentatore per breadboard.

Per alimentare i vari componenti utilizzati, si è utilizzato un'alimentatore compatto (dimensioni: 55 x 35mm) studiato appositamente per essere impiegato con le breadboard.
Al modulo è possibile applicare una tensione compresa tra 6,5 e 12 Vdc, per ottenere in uscita due tensioni distinte a 3,3 V e 5 V con una corrente massima di circa 700 mA.
Ha un interruttore on/off, uscita USB 5V (non utilizzata nel nostro caso), LED di alimentazione e jumper di selezione tensione. Potrete trovarlo in rete cercando "**YwRobot MB-V2**"

[56]

Schema di collegamento.

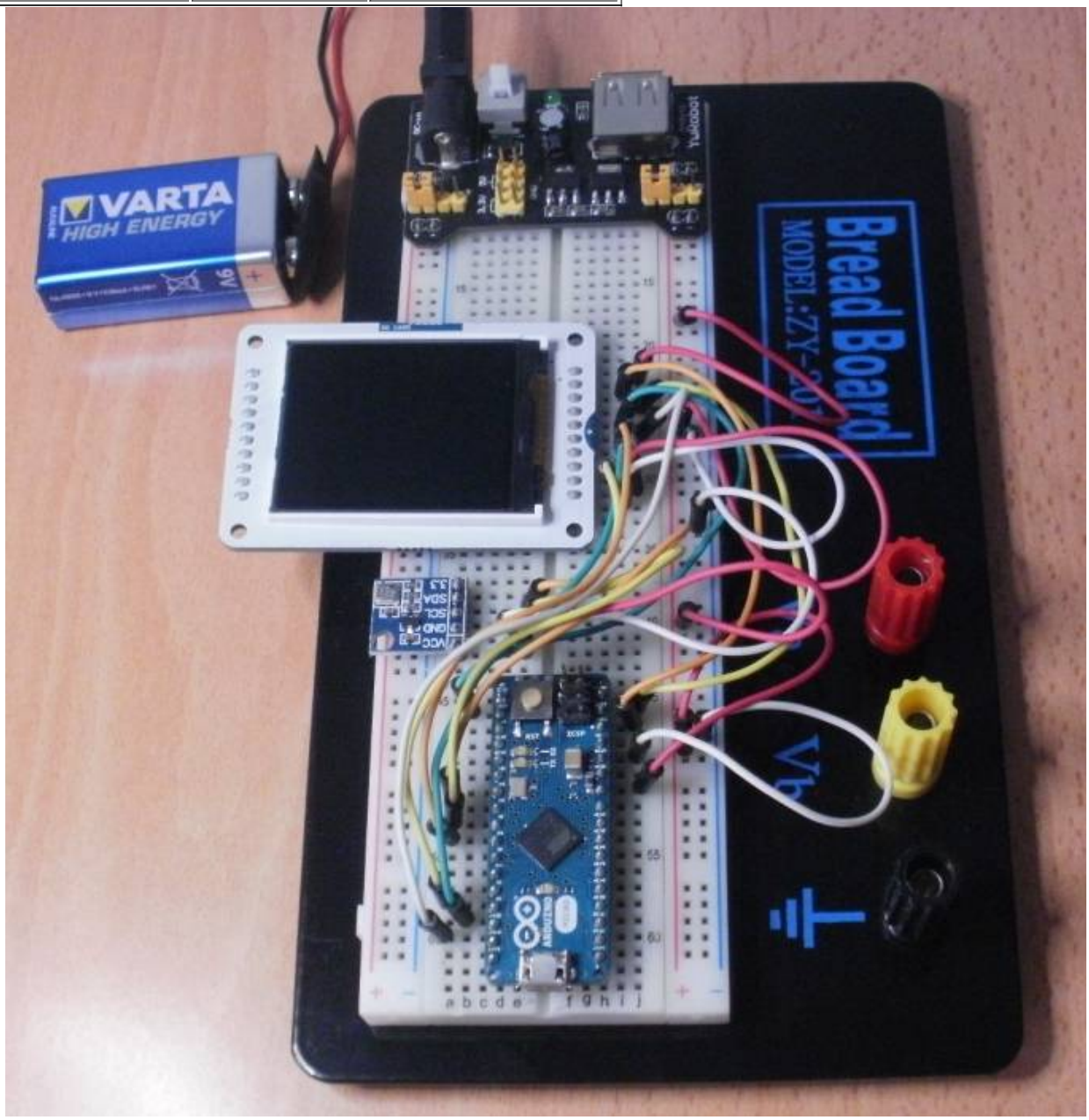
Nell'immagine sotto riportata sono visibili i collegamenti da realizzare per l'effettuare il test.



[57]

Pin Arduino Micro	Pin Display TFT	PIN Sensore BMP180
+5V	+5V, BL	+5V
GND	GND	GND
MISO	MISO	
SCK	SCK	
MOSI	MOSI	
SDA		SDA
SCL		SCL
D4	SD CS	
D10	LCD CS	
D9	D/C	

D8	Reset	



[58]

Libreria per utilizzo sensore BMP180

Per facilitare l'uso del sensore è disponibile un'apposita libreria (**bmp180** **[59]**), questa funziona in unione alla **libreria Wire.h** [30] già presente come libreria standard dell'IDE di Arduino che permette l'utilizzo del protocollo I2C.

In rete ve ne sono molte versioni, quella utilizzata per questo test possiede vari comandi:

Initialize();
GetUncompensatedTemperature();
GetUncompensatedPressure();
Compensate Temperature();
CompensatePressure();
SoftReset();
GetTemperature();
GetPressure();
GetAltitude();
SetResolution();
IsConnected()
EnsureConnected()

GetErrorText().
Nel programma di test saranno utilizzati solamente i seguenti:

Initialize() - Inizializza il sensore
IsConnected() – verifica che il sensore sia effettivamente connesso.
EnsureConnected() - verifica che sia possibile connettersi con il sensore.
SoftReset() - garantisce che quando si è collegato il sensore vi sia un avvio pulito.
GetTemperature() - acquisisce dal sensore il dato della temperatura.
GetPressure() - acquisisce dal sensore il dato della pressione in Pascal.
GetAltitude() - acquisisce dal sensore il dato dell'altitudine in metri, il risultato ottenuto deve essere corretto per la pressione relativa al punto in cui avviene la misurazione, nel programma deve essere inserito il valore locale nella variabile seaLevelPressure, che di default è pari alla pressione a livello del mare pari a 101325 Pa;

Nota: il dato della pressione è fornito in Pascal (Pa) nel Sistema internazionale pari a 1 newton su metro quadrato (1 N/m²). In ambito meteorologico, la pressione atmosferica si misura in centinaia di Pascal (o ettopascal, abbreviato con hPa). Si ha: 1 013,25 millibar = 101 325 Pa = 1 013,25 hPa

Per utilizzare la libreria occorrerà scaricarla e salvarla all'interno della cartella Library dell'IDE di Arduino. Successivamente per utilizzarla all'interno del programma si dovranno aggiungere le seguenti righe:

```
#include <Wire.h>
#include <BMP180.h>
```

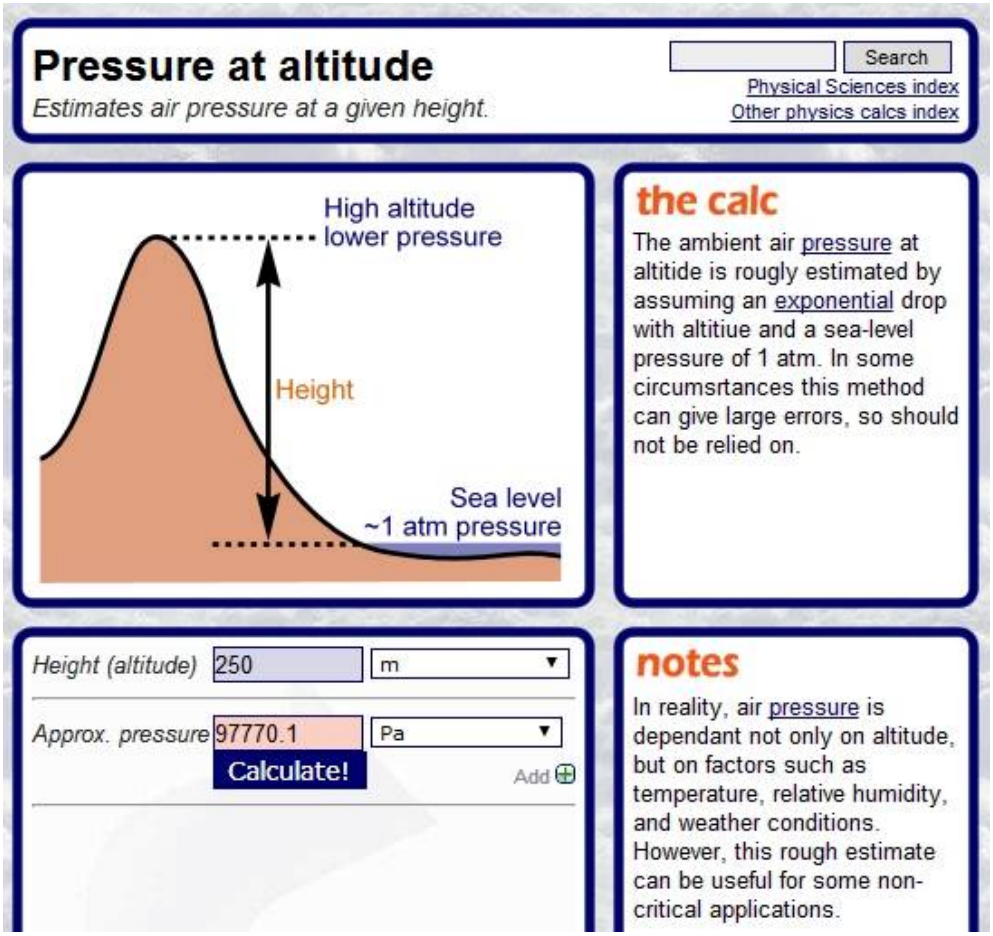
Programma di test.

Il programma di test (**Sensore_BMP180** [60]) mostrerà sul display TFT, dopo una pagina iniziale di presentazione, i valori letti dal sensore BMP180.

[61]

Per il funzionamento del programma oltre alle librerie **Wire.h** e **BMP180.h** citate prima per la gestione del sensore occorre utilizzare anche altre due librerie che sono già incluse in quelle standard dell'IDE versione 1.0.6, queste sono **TFT.h** e **SPI.h** che occorrono per la gestione del display LCD.
Il listato è sufficientemente commentato per comprenderne il funzionamento.
Per ottenere i valori corretti dell'altitudine occorre inserire il valore relativo alla propria posizione nella

variabile **seaLevelPressure**, che di default contiene il valore 101325 (espresso i Pa), per determinare il valore si può utilizzare l'utility trovata in rete.



[62]

Il risultato del calcolo è solamente una stima in cui si assume un calo esponenziale della pressione considerando una pressione a livello del mare di 1 atm (101325 Pa).

Questo valore potrebbe non essere corretto in quanto la pressione dell'aria dipende non solo dell'altitudine, ma anche da altri fattori quali temperatura, umidità relativa, e le condizioni atmosferiche. Tuttavia, questa stima può essere utile per applicazioni non critiche. **Nota:** Per i calcoli impostare il dato in uscita in Pa.

```
#include <TFT.h>  // Arduino LCD library
#include <SPI.h>
// gestione sensore BMP180
#include <Wire.h>
#include <BMP180.h>

// Definizione dei pin utilizzati
// per il display TFT
#define cs   10
#define dc   9
#define rst  8

#define indicatorLed  13
// array per contenere i dati di pressione
char PressurePrintout[5];
String Pressure;
// array per contenere i dati di temperatura
char TemperaturePrintout[5];
```

```
String Temperature;
// array per contenere i dati di altitudine
char AltitudePrintout[5];
String Altitude;

// Crea un'istanza per il senore BMP180.
BMP180 barometer;
// Inserire la pressione a livello del mare della
// vostra posizione a Pascal.
float seaLevelPressure = 101325;

// create an instance of the library
TFT TFTscreen = TFT(cs, dc, rst);

void setup() {
  // Scrittura pagina di presentazione
  TFTscreen.begin();
  TFTscreen.background(0, 0, 0);
  TFTscreen.setTextSize(2);
  TFTscreen.stroke(255,255,0);
  TFTscreen.text(" Test sensore",0,0);
  TFTscreen.text("   BMP180",0,20);
  TFTscreen.text("Arduino Micro",0,40);
  TFTscreen.stroke(0,255,255);
  TFTscreen.text("  adrirobot",0,60);

  // inizalizzazione della libreria I2C per la
  // comunicazione con il sensore BMP180.
  Wire.begin();
  // Impostazione LED indicatore.
  pinMode(indicatorLed, OUTPUT);
  // Creiamo un'istanza del nostro sensore BMP180.
  barometer = BMP180();
  // Si verifica che ci si può collegare al sensore.
  if(barometer.EnsureConnected())
  {
    TFTscreen.setTextSize(1);
    TFTscreen.stroke(0,255,0);
    TFTscreen.text("BMP180 collegato",30,100);
    // Accensione led se sensore collegato
    digitalWrite(indicatorLed, HIGH);

    // Dopo il, collegamento, il sensore viene
    //resettato per garantire un avvio pulito.
    barometer.SoftReset();
    // Ora inizializziamo il sensore.
    barometer.Initialize();
  }
  else
  {
```

```
      TFTscreen.setTextSize(1);
      TFTscreen.stroke(255,0,0);
      TFTscreen.text("BMP180 non collegato",20,100);
      // Spegnimento led se sensore non collegato
      digitalWrite(indicatorLed, LOW);
    }
    // Tempo di visualizzazione schermata iniziale
    delay(2000);
    TFTscreen.background(0, 0, 0);
  }

  void loop() {
    if(barometer.IsConnected)
    {
      //Lettura e visualizzazione pressione in Pascal.
      TFTscreen.stroke(255,0,0);
      TFTscreen.setTextSize(2);
      TFTscreen.text("Pressione:",0,0);
      TFTscreen.setTextSize(3);
      Pressure = String (barometer.GetPressure());
      Pressure.toCharArray(PressurePrintout, 6);
      TFTscreen.text(PressurePrintout, 0, 18);
      TFTscreen.text("Pa",115,18);

      // Lettura e visualizzazione temperatura in gradi Celsius.
      TFTscreen.stroke(0,255,0);
      TFTscreen.setTextSize(2);
      TFTscreen.text("Temperatura:",0,45);
      TFTscreen.setTextSize(3);
      TFTscreen.stroke(0,255,0);
      Temperature = String (barometer.GetTemperature());
      Temperature.toCharArray(TemperaturePrintout, 6);
      TFTscreen.text(TemperaturePrintout, 0, 63);
      TFTscreen.circle(110, 65, 2);
      TFTscreen.text("C",115,63);

      //Lettura e visualizzazione quota attuale in metri.
      TFTscreen.stroke(255,255,0);
      TFTscreen.setTextSize(2);
      TFTscreen.text("Altitudine:",0,90);
      TFTscreen.setTextSize(3);
      Altitude = String (barometer.GetAltitude(seaLevelPressure));
      Altitude.toCharArray(AltitudePrintout, 6);
      TFTscreen.text(AltitudePrintout, 0, 108);
      TFTscreen.text("m",115,108);

      // Attesa tra le misure
      delay(1000);

      // Cancellazione dati per nuova misura
```

```
    TFTscreen.stroke(0,0,0);
    TFTscreen.text(PressurePrintout, 0, 18);
    TFTscreen.text(TemperaturePrintout, 0, 63);
    TFTscreen.text(AltitudePrintout, 0, 108);
  }
}
```

Filmato illustrativo

E' possibile vedere l'esecuzione del programma visionando il filmato:

Arduino Micro - Lettura sensore di pressione BMP180

Conclusioni

Siamo al termine di quest'articolo di presentazione di questo piccolo/grande Arduino, abbiamo anche mostrato l'utilizzo di un sensore con cui il lettore potrebbe creare, con l'utilizzo di altri sensori quali sensore di umidità e un modulo clock, **una piccola stazione meteo** [63], oppure realizzare magari un altimetro come visibile in **questo filmato** [64].
Non resta che liberare la propria immaginazione.

Article printed from Elettronica Open Source: **https://it.emcelettronica.com**

URL to article: **https://it.emcelettronica.com/arduino-micro-e-bmp180-per-realizzare-una-weather-station**

URLs in this post:

[1] Quale scheda Arduino scegliere per il mio progetto?: **https://it.emcelettronica.com/quale-scheda-arduino-scegliere-per-il-mio-progetto**
[2] Adafruit: **http://www.adafruit.com/**
[3] **Costruzione del robot LittleBot – Scheda di controllo**: **https://it.emcelettronica.com/costruzione-del-robot-littlebot-scheda-di-controllo**

[4] **Arduino Nano**: **http://arduino.cc/en/Main/arduinoBoardNano**
[5] un'altro articolo: **https://it.emcelettronica.com/levaluation-kit-xmc-2go-di-infineon-come-scheda-sensori-i-robot-robi**
[6] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_confronto.jpg**
[7] **Arduino Micro**: **http://arduino.cc/en/Main/arduinoBoardMicro**
[8] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_foto.jpg**
[9] Image: **http://www.pighixxx.com/test/pinouts/boards/micro.pdf**
[10] **schema elettrico**: **http://arduino.cc/en/uploads/Main/arduino-micro-schematic.pdf**
[11] **FND340P**: **http://www.adrirobot.it/datasheet/transistor/pdf/FDN340P.pdf**
[12] **PMV48XP**: **http://www.adrirobot.it/datasheet/transistor/pdf/PMV48XP.pdf**
[13] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_usb.jpg**
[14] **MF-MSF050-2**: **http://www.adrirobot.it/datasheet/vari/pdf/MF-MSMF.pdf**
[15] **NCP1117-5**: **http://www.adrirobot.it/datasheet/integrati/pdf/NCP1117ST33T3G.pdf**
[16] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_schema_regolatori.jpg**
[17] **LP2985-33DBVR**: **http://www.adrirobot.it/datasheet/integrati/pdf/lp2985-33.pdf**
[18] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_alimentazione.jpg**
[19] **Arduino Leonardo**: **http://arduino.cc/en/Main/arduinoBoardLeonardo**
[20] **Arduino Esplora**: **https://it.emcelettronica.com/scopriamo-nuova-scheda-arduino-esplora**
[21] **ATMEGA32U4**: **http://www.adrirobot.it/datasheet/processori/pdf/ATMega32U4.pdf**
[22] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_processore.jpg**
[23] **libreria SPI**: **http://arduino.cc/en/Reference/SPI**
[24] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_processore_foto.jpg**
[25] **libreria EEPROM**: **http://www.arduino.cc/en/Reference/EEPROM**
[26] **pinMode ()**: **http://arduino.cc/en/Reference/PinMode**
[27] **digitalWrite ()**: **http://arduino.cc/en/Reference/DigitalWrite**
[28] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_seriale.jpg**
[29] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_scl-sda.jpg**
[30] **Liberia Wire**: **http://arduino.cc/en/Reference/Wire**
[31] **attachInterrupt ()**: **http://arduino.cc/en/Reference/AttachInterrupt**
[32] **PWM**: **http://it.wikipedia.org/wiki/Pulse-width_modulation**
[33] **analogWrite()**: **http://arduino.cc/en/Reference/AnalogWrite**
[34] **analogReference ()**: **http://arduino.cc/en/Reference/AnalogReference**
[35] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_schema-connettore_ICSP.jpg**
[36] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_schema-led.jpg**
[37] **Arduino UNO R3,**: **http://www.arduino.cc/en/Main/arduinoBoardUno**
[38] **SoftwareSerial** : **http://www.arduino.cc/en/Reference/SoftwareSerial**
[39] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_Cavo-USB.jpg**
[40] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_driver.jpg**
[41] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_posizione_drive.jpg**
[42] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_driver-micro1.jpg**
[43] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_driver_micro.jpg**
[44] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_bootloader_1.jpg**
[45] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_bootloader.jpg**
[46] **software IDE Arduino, ora alla versione 1.0.6**: **http://arduino.cc/en/main/software**
[47] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_IDE.jpg**

[48] **display LCD tipo TFT**: **http://arduino.cc/en/Main/GTFT**
[49] un'apposita libreria : **http://arduino.cc/en/Reference/TFTLibrary**
[50] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/GLCD_pins.jpg**
[51] **BMP180 prodotto dalla BOSCH**: **https://www.bosch-sensortec.com/en/homepage/products_3/environmental_sensors_1/bmp180_1/bmp180**
[52] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_bmp180-schema-a-blocchi.jpg**
[53] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_sensore-bmp180.jpg**
[54] **tipo 662K**: **http://www.adrirobot.it/datasheet/integrati/pdf/XC6206.pdf**
[55] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_bmp180_schema-modulo.jpg**
[56] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_alimentatore_breadboard.jpg**
[57] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_schema_circuito.jpg**
[58] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_test.jpg**
[59] bmp180: **https://it.emcelettronica.com/wp-content/uploads/2015/03/bmp180.zip**
[60] Sensore_BMP180: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Sensore_BMP180.zip**
[61] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_display_programma.jpg**
[62] Image: **https://it.emcelettronica.com/wp-content/uploads/2015/03/Arduino_micro_Pressione-locale.jpg**
[63] una piccola stazione meteo: **https://it.emcelettronica.com/stazione-meteo-online-con-arduino**
[64] **questo filmato**: **https://www.youtube.com/watch?v=mMiMNTv25Bw**