

Elettronica In

www.elettronica.in.it

€ 6,00 - Anno XXVII - n. 253 - APRILE 2021



ARLOK: il robot per tutti



■ Controllare
l'orto da Twitter

■ Orologio
da laboratorio

■ Filtri LC
per alimentatori

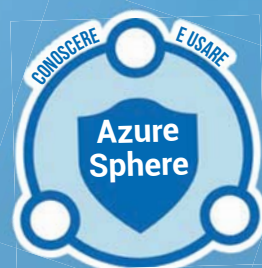
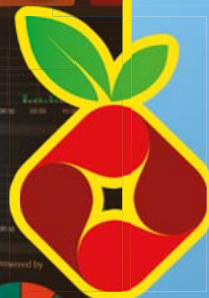
■ Pulsiossimetro
WiFi

■ Tobbie II

Trasmettitore DCF77



PI-HOLE, un server DNS indoor



Taglia e incidi con **Cricut MAKER**



Prezzo IVA inclusa

Macchina da taglio intelligente



€ 439,00

cod. CRICUTMAKER

Dai vita alla fantasia e alla creatività con Cricut Maker l'innovativa macchina da taglio. Grazie a Design Space, il software di progettazione gratuito, potrai realizzare praticamente qualsiasi progetto fai-da-te, dall'arte 3D all'arredamento per la casa, gioielli, progetti di carta e molto altro ancora e inviarti direttamente a Cricut Maker dal tuo tablet, dal PC o dallo smartphone. Cricut Maker taglia in modo rapido e accurato oltre 300 materiali: tessuti, pelle, carta, cartamodelli, cuoio, fogli di balsa e tanto altro. Espandi le potenzialità di Cricut Maker utilizzando utensili dedicati, alcuni inclusi nella confezione, altri acquistabili separatamente.





Come l'intelligenza artificiale ci ruberà il lavoro

Inghilterra, XIX secolo, nasce il movimento luddista. Nedd Lud (che darà il nome al movimento) è un giovane che combatte contro i poteri degli industriali: il suo scopo è quello di distruggere macchinari come il telaio meccanico che, automatizzando il processo produttivo, a suo giudizio "rubano" il lavoro agli operai.

Nel XXI secolo molte cose sono cambiate ma la paura che l'innovazione possa portar via il lavoro alle persone continua ad esserci, come secoli fa.

La pandemia ha accelerato l'adozione di tecnologie "intelligenti".

Prima erano i telai, poi i robot, e ora è l'Intelligenza Artificiale a minare il lavoro umano. È di qualche settimana fa, la notizia che un'azienda

israeliana (deepdub.ai) ha messo a punto un sistema per sostituire i doppiatori permettendo così di vedere l'ultimo film uscito in italiano o in qualsiasi lingua si preferisca. L'adozione dell'AI non è una cosa recente, risale infatti al 2017 la notizia che un'azienda di assicurazioni giapponese (fukoku-life.co.jp) sostituì 34 dipendenti con un software di intelligenza artificiale. Ma di certo la pandemia ha accelerato l'adozione di tecnologie "intelligenti" affidando ai computer problemi che prima solo gli uomini riuscivano a risolvere.

Si stima che entro il 2025 circa 85 milioni di persone dovranno cambiare il loro lavoro lasciando il posto a complessi algoritmi. Nedd Lud avrebbe sicuramente vita difficile in questo secolo, perché pensare di distruggere tutti i server del mondo per bloccare il progresso tecnologico sarebbe veramente complicato.

Ma una cosa dovrebbe tranquillizzarci: l'utilizzo dell'intelligenza artificiale creerà 97 milioni di nuovi impieghi. Con un unico imperativo: aggiornarsi. Verranno richieste sempre di più competenze legate al fattore umano, soprattutto personali (soft skill), come il pensiero critico, le capacità analitiche e le capacità di problem-solving; insomma, all'uomo verrà richiesta l'unica cosa che robot e computer non sanno fare (per il momento): pensare.

ON LINE

elettronica.in

SI RINGRAZIA

ADI
Futura Elettronica
Texas Instruments



In copertina:
realizziamo un
piccolo rover
stampato in 3D
gestito da Arduino.

Boris Landoni

boris.landoni@elettronica.in

Elettronica In

Rivista mensile, anno XXVII n. 253

APRILE 2021

EDITORIALE

Direttore Responsabile

Boris Landoni

boris.landoni@elettronica.in

Redazione

Stefano Garavaglia, Paolo Gaspari,
Davide Scullino, Gabriele Daghetta

redazione@elettronica.in

Grafica

Alessia Sfulcini

alessia.sfulcini@elettronica.in

Ufficio Pubblicità

Monica Premoli (0331-752668)

monica.premoli@elettronica.in

Ufficio Abbonamenti

Tel. 0331-752668

abbonati@elettronica.in

Direzione, Redazione, Pubblicità

FUTURA GROUP srl - Divisione Editoriale

via Adige 11 - 21013 Gallarate (VA)

Tel. 0331-752668

Abbonamenti

Annuo 10 numeri Euro 45,00

Annuo 10 numeri digitale Euro 39,00

Le richieste di abbonamento vanno inviate a:

FUTURA GROUP srl
via Adige 11, 21013 Gallarate (VA)

Tel. 0331-752668

oppure tramite il sito

<https://www.elettronica.in/abbonamenti/>

Distribuzione per l'Italia

SO.DI.P. Angelo Patuzzi S.p.A.

via Bettola 18 - 20092 Cinisello Balsamo (MI)

Tel. 02-660301 Fax 02-66030320

Stampa

ROTO3 Spa - Via Turbigo, 11/b
20022 CASTANO PRIMO (MI)

Tiratura
18.000 copie



Mensile
associato all'USPI,
Unione Stampa
Periodica Italiana

Rivista mensile registrata presso il Tribunale di Milano
con il n. 245 il 03/05/1995.

Prezzo di copertina Euro 6,00.

Gli arretrati nei formati cartaceo e digitale (pdf) sono acquistabili
sul sito della rivista al prezzo di Euro 6,00.

Poste Italiane Spa - Spedizione in abbonamento Postale - D.L.
353/2003 (conv. in L. 27/02/2004) art. 1 comma 1 - DCB Milano.

Futura Group srl è iscritta al Registro Operatori della
Comunicazione n. 23650 del 02/07/2013.

Impaginazione ed immagini sono realizzati in DeskTop Publishing
con programmi Adobe InDesign e Adobe Photoshop per Windows.
Tutti i contenuti della Rivista sono protetti da Copyright.

Ne è vietata la riproduzione, anche parziale, la traduzione e più in
generale la diffusione con qualsiasi mezzo senza l'autorizzazione
scritta da parte dell'Editore. I circuiti, il firmware ed il software
descritti sulla Rivista possono essere realizzati solo per uso
personale, ne è proibito lo sfruttamento a carattere commerciale
e industriale. Tutti possono collaborare con Elettronica In. L'invio
di articoli, materiale redazionale, programmi, traduzioni, ecc.
implica da parte del Collaboratore l'accettazione dei compensi e
delle condizioni stabilite dall'Editore (www.elettronica.in/ase.pdf).
Manoscritti, disegni e foto non richiesti non verranno in alcun caso
restituiti. L'utilizzo dei progetti e dei programmi pubblicati non
comporta alcuna responsabilità da parte della Società Editrice.

© 2021 Futura Group srl

Futura
Group
Editorial

Contenuti

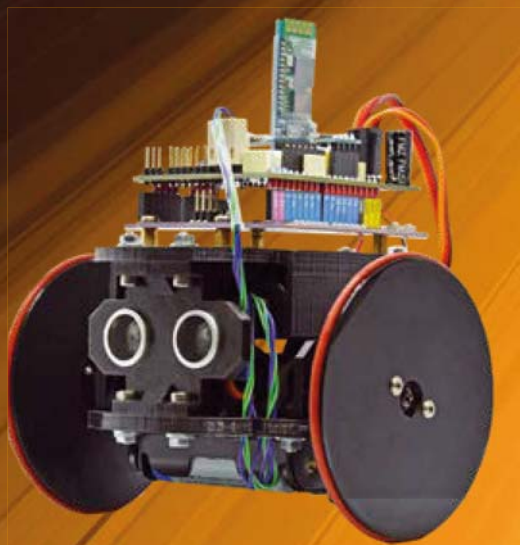
01

RUBRICHE

- 01 Editoriale
- 05 T&T Tips & Tricks
- 09 Embedded IoT
Digital Transformation of Things
- 13 Componenti & Sistemi
- 126 Scienza & Tecnologia

Progetto di Copertina

ARLOK: IL ROBOT PER TUTTI



47

Medicale

Pulsiossimetro WiFi

60



19

ARTICOLI

DIDATTICA

19 Filtri LC per alimentatori

Riduciamo il ripple e le spurie inevitabilmente presenti all'uscita dei convertitori DC/DC switching, in maniera semplice ed efficace.

NETWORKING

31 Anche l'orto twitta

Volete tenere aggiornati i vostri contatti Twitter sullo stato del vostro orto o giardino? Ecco il progetto che fa per voi.

Didattica

Filtri LC per alimentatori



19

Networking

Anche l'orto twitta



31

Networking**PI-HOLE, un server DNS indoor****73****Radio****Trasmettitore DCF77****83****Didattica****Corso Azure Sphere****115****Robotica****Tobbie II****107****Home Project****Orologio... da laboratorio****99****ROBOTICA****47 ARLOK: il robot per tutti**

Piccolo automa su ruote gestito da una scheda MakerUNO, versione elaborata di Arduino Uno, capace di muoversi evitando ostacoli.

MEDICALE**60 Pulsiossimetro Wifi**

Rileva battito cardiaco e percentuale di saturazione dell'emoglobina trasmettendo in wireless i valori, che verranno visualizzati sull'apposita app in smartphone e tablet.

NETWORKING**73 PI-HOLE, un server DNS indoor**

Rendiamo più sicura la connessione domestica bloccando alcuni siti web inappropriati per i bambini, gli elementi traccianti e le pubblicità.

RADIO**83 Trasmettitore DCF77**

Produce un segnale orario ricavato da un RTC molto preciso e lo irradia localmente sui 77,5 kHz con protocollo DCF77. Ideale per avere un riferimento orario o testare ricevitori.

HOME PROJECT**99 Orologio... da laboratorio**

Mostra giorno, data, ora, è completamente programmabile e dispone di una sveglia e di un sensore di luce ambiente per regolare automaticamente la luminosità del display.

ROBOTICA**107 Tobbie II**

Sperimentiamo la robotica con un piccolo automa governato dalla board micro:bit interfacciata ad uno smartphone.

DIDATTICA**115 Conoscere e usare Azure Sphere**

Prepariamo la connessione della nostra scheda al Cloud Azure IoT Central e configuriamo l'interfaccia di programmazione. Seconda puntata.

Ottimizza la tua certificazione di sicurezza funzionale con Texas Instruments

Ottieni in modo efficiente le certificazioni ISO 26262 e IEC 61508 e altre ancora, utilizzando i nostri prodotti, la documentazione disponibile e esperti di sicurezza ben informati.



Per ulteriori informazioni, tra cui analisi su richiesta e rapporti diagnostici, percentuale di FIT di sicurezza funzionale e il nostro processo di sviluppo certificato, visitare

ti.com/functionalsafety





La massima frequenza del 555

Anche in quest'era di microcontrollori e microprocessori usati anche per funzioni banali, uno dei più popolari, versatili e impiegati circuiti integrati è il classico timer 555, declinato in tutte le sue varianti e ormai costruito da tantissime Case; con esso è possibile realizzare svariati circuiti, soprattutto combinandone due o più. Uno da solo permette di realizzare monostabili, multivibratori astabili a frequenza fissa, variabile e controllata in tensione. Con due è possibile realizzare generatori di impulsi a durata costante e indipendente dalla frequenza (un 555, configurato come astabile, fa da trigger mentre l'altro è monostabile), VCO ed altro.

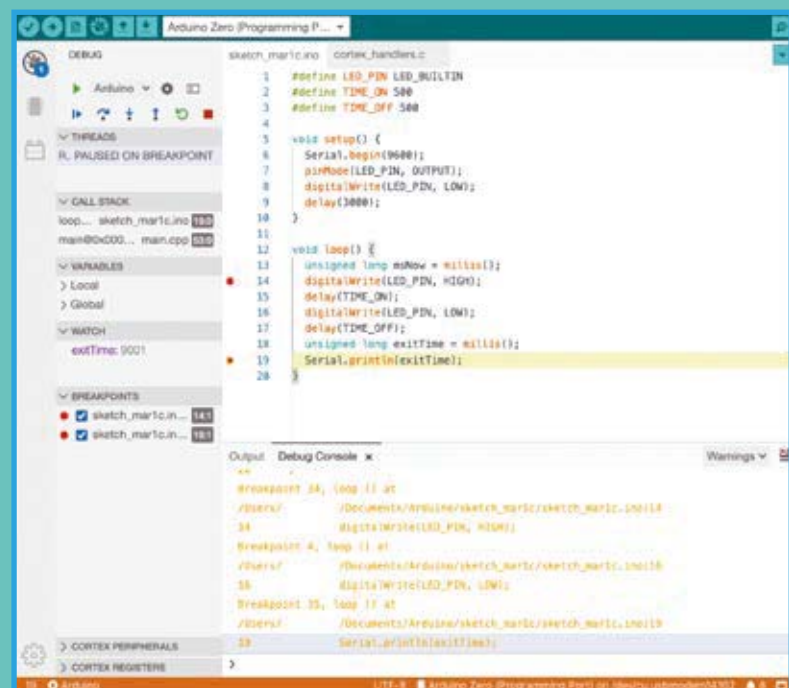
Ma soffermiamoci un istante sull'utilizzo come astabile e VCO, dove all'integrato è richiesto di generare una forma d'onda rettangolare: consultando i datasheet di vari produttori, risalenti ad epoche diverse, si notano differenze nella massima frequenza generabile: in alcuni casi è riportato 200 kHz mentre in altri è indicato 500 kHz. Questo può innescare non pochi dubbi. Ebbene, sono vere entrambe: il 555 così come è nato e com'è stato realizzato nei primi decenni arrivava al massimo a 200 kHz, mentre di recente alcuni costruttori ne hanno prodotte versioni "avanzate" capaci di operare fino a 500 kHz. Bisogna quindi fare sempre riferimento al datasheet del componente che si intende acquistare, ossia del relativo produttore, perché da uno all'altro determinate caratteristiche -per quanto non dovrebbe essere così- possono cambiare; un po' come per alcuni MOSFET della serie IRF, che in origine avevano il diodo di blocco tra Drain e Source di tipo comune e recentemente, invece, hanno uno Zener.

Tips & Tricks sono i nostri suggerimenti in pillole e i piccoli trucchi riguardanti sia l'hardware che il firmware, che sperimentatori, tecnici e progettisti elettronici vorrebbero conoscere per sviluppare meglio la propria passione: quella per l'Elettronica applicata. Ecco quelli che abbiamo preparato questo mese per voi.

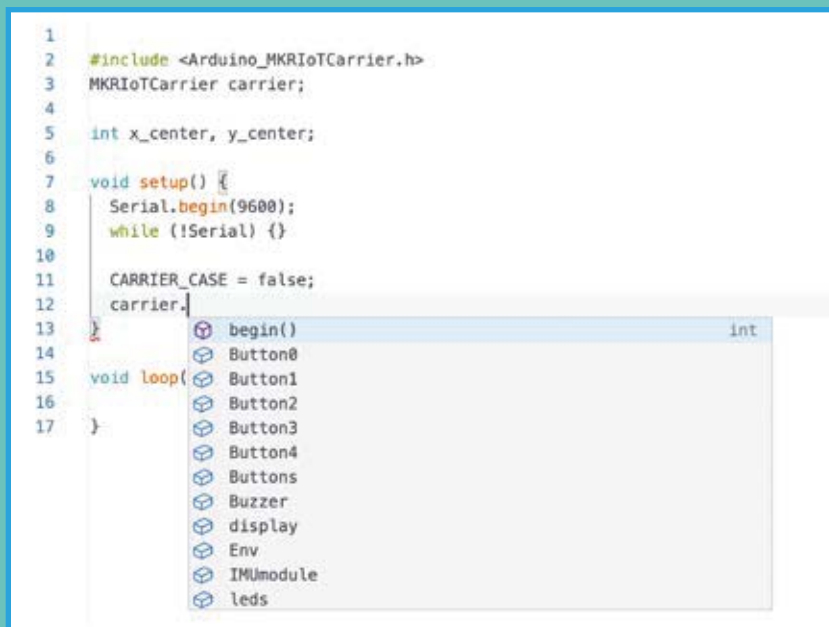
Arduino: l'IDE raggiunge quota 2.0

L'IDE è il software che chi realizza applicazioni con Arduino utilizza per programmare le proprie schede, ca-

ricandovi gli sketch. Il suo sviluppo è iniziato nel 2005 sulla base dell'interfaccia grafica del progetto Processing e da allora è stato costantemente sviluppato aggiungendo funzionalità, una release dopo l'altra, diventando lo standard de facto per la prototipazio-



↑ Il Live Debugger dell'IDE Arduino 2.0.



↑ La funzione di completamento automatico dell'IDE 2.0.

ne elettronica. Grazie a un framework estensibile basato su pacchetti di supporto per schede modulari, l'IDE supporta più di 1.000 schede ufficiali e non ufficiali (terze parti) ed è stato tradotto in 66 lingue; nell'ultimo anno è stato scaricato più di 39 milioni di volte. Mentre l'IDE di Arduino fornisce un'interfaccia semplice e chiara, ideale per gli utenti inesperti, gli utenti più avanzati spesso lamentano le capacità di modifica un po' troppo "basilari" rispetto agli editor moderni; tra queste l'indentazione del codice, il block folding, le parentesi a chiusura automatica, la ricerca e

la sostituzione di espressioni regolari, il comment toggling. Oltre a questo, molti utenti desideravano il debug live, ovvero la possibilità di eseguire il codice su una scheda allegata e fermarlo su una determinata riga per controllare il contenuto di variabili, memoria e registri. Purtroppo le release 1.x dell'IDE sono state sviluppate in Java e la base di codice monolitica rende difficile implementare tali funzionalità. Java sta anche diventando una tecnologia obsoleta per le applicazioni desktop e viene gradualmente eliminata dai sistemi operativi e dagli app store più recenti, il che implica

il dover risolvere problemi di compatibilità. Ecco perché il team di Arduino nel 2018 ha iniziato a effettuare il refactoring della toolchain arrivando al primo punto di svolta: arduino-cli, lo strumento da riga di comando di Arduino scritto in Golang che espone tutte le funzionalità principali dell'IDE, fornendo agli utenti avanzati uno strumento flessibile che possono integrare nel loro IDE.

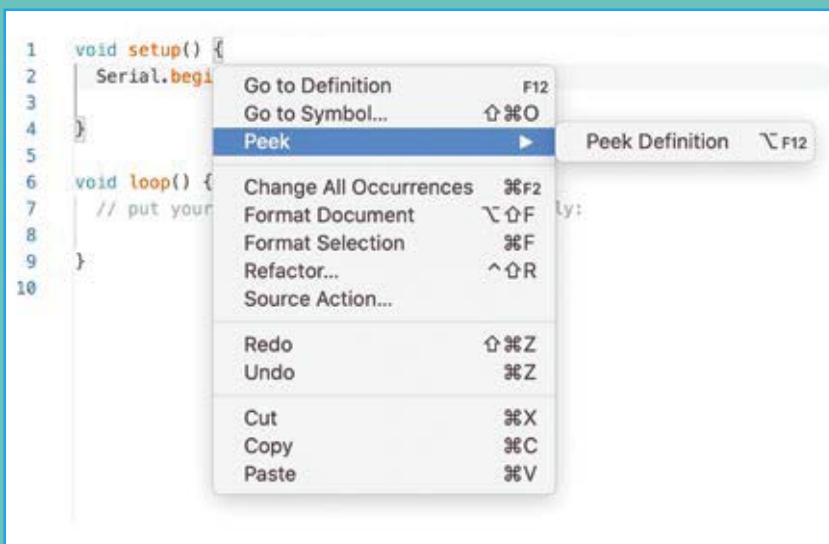
Nel 2019 è arrivata una nuova versione di IDE costruito su arduino-cli e basato su uno stack software moderno (Theia ed Electron) con il nome in codice di "Arduino Pro IDE" e nel 2020 è iniziato il lavoro per portare il nuovo IDE da un proof-of-concept a uno strumento completamente funzionale. Nasce così la versione 2.0, la cui principale innovazione è il debug live del codice. Porta con sé un editor moderno con interfaccia responsive e ridotti tempi di compilazione, pur rimanendo familiare e in continuità con le versioni precedenti.

Il debugger live consente di eseguire il codice in modo interattivo su una scheda e ispezionarne l'esecuzione senza scrivere decine di istruzioni "Serial.println ()" perché prima il debug si eseguiva in maniera grossolana così. Basta attivare il pannello di debug, impostare i punti di interruzione in cui si desidera sospendere l'esecuzione e ispezionare il contenuto delle variabili. È persino possibile cambiare il contenuto delle variabili al volo e riprendere l'esecuzione. Il debugger è uno strumento molto apprezzabile che ad oggi ha un unico limite: supporta tutte le schede Arduino basate sulle piattaforme SAMD e Mbed (famiglia MKR, Nano 33 IoT, Nano 33 BLE, Portenta, Zero).

I manutentori di core Arduino per schede di terze parti possono aggiungere il supporto per il debug aggiungendo i relativi parametri di configurazione; una guida tecnica per questo sta arrivando. Basta collegare una sonda di debug come il J-link Segger ai pin JTAG della board e si può debuggare.

Il nuovo IDE si basa sul framework Eclipse Theia, che è un progetto open source basato sulla stessa architettura di VS Code (protocollo del server del linguaggio, estensioni, debugger). Il front-end è scritto in TypeScript, mentre la maggior parte del backend è scritta in Golang.

Altra innovazione dell'IDE 2.0 è che durante la digitazione, l'editor suggerisce il completamento automatico di variabili e funzioni in base alle librerie incluse. Quando si clicca su una variabile o funzione, appare un menu contestuale



↑ Menu contestuale IDE 2.0 con scorciatoie e funzioni correlate.

che fornirà scorciatoie da tastiera per navigare più rapidamente e saltare alla riga o file dove sono dichiarate.

Misurare temperatura e umidità: quale sensore?

Arduino e i suoi derivati ci hanno abituati a risolvere semplicemente problemi come l'acquisizione ed elaborazione o trasmissione di parametri ambientali, primi fra tutti temperatura e umidità dell'aria; e ci hanno fatto conoscere specifici sensori nati nell'ecosistema di Arduino, come ad esempio l'usativissimo DHT-11. In commercio troviamo anche il meno noto DHT-12 e quindi è probabile che gli sperimentatori si trovino di fronte al dilemma di quale utilizzare tra i due, la cui sigla è, peraltro, simile.

Spieghiamo dunque che differenza c'è fra i due, a parte quella visiva. Sostanzialmente sono entrambi sensori combinati di temperatura e umidità ambientale che forniscono un'informazione digitale contenente i valori assunti da queste due grandezze fisiche.

Il sensore DHT-11 è un trasduttore di temperatura e umidità a quattro terminali di cui ne si utilizza 3, con uscita dei dati in formato digitale. Il doppio sensore utilizza un chip ed elementi traduttori che assicurano l'affidabilità e la stabilità. I suoi elementi sensibili, ossia termistore NTC e pellicola igrometrica polimerica (resistore sensibile all'umidità), sono connessi con un processore 8-bit single-chip il quale espone un'interfaccia single-wire o 1-wire.

Ogni sensore interno al DHT-11 è compensato in temperatura e calibrato in fabbrica in modo da ricavare il coefficiente di calibrazione per la correzione dell'errore e della non linearità, che nell'utilizzo verrà applicato dal firmware del microcontrollore che elaborerà i dati rilevati nel circuito dell'applicazione. I coefficienti di calibrazione vengono scritti in fabbrica nella memoria OTP e sono richiamati durante il processo di rilevamento e consentono di evitare di ricalibrare il sensore prima di ogni utilizzo.

Le ridotte dimensioni e il basso consumo, uniti alla possibilità di connetterlo al microprocessore host a una distanza fino a 20 m, fanno del DHT-11 il sensore ambientale ideale per Arduino.

La tensione generata dai due trasduttori viene convertita in formato digitale da un convertitore A/D a 8 bit e resa disponibile lungo una connessione seriale one-wire. La risoluzione della misura è di 1°C sulla temperatura (tolleranza $\pm 2^\circ\text{C}$) e dell'1% sull'umidità relativa (tolleranza $\pm 5\%$). Per quanto riguarda la temperatura, il campo misurabile è compreso fra -20 e $+60^\circ\text{C}$.

Quanto all'umidità, il DHT-11 misura fra 20 e 90% di umidità relativa, a una temperatura operativa compresa fra 0 e 50 gradi Celsius. La precisione dichiarata è per l'umidità $\pm 4\%$ RH (Max $\pm 5\%$ di umidità relativa) e per la temperatura $\pm 2^\circ\text{C}$. La risoluzione sull'umidità è dell'1% RH e sulla temperatura di 0,1 °C. Il tempo di misura è dell'ordine dei 2 secondi.

I dati sono forniti sotto forma di byte sull'interfaccia ad 1 filo più massa; l'alimentazione è a tensione continua compresa fra 3 e 5,5V.

E veniamo al DHT-12, che sostanzialmente è uguale al DHT-11 e differisce per l'interfaccia dati, che è del tipo a I²C-Bus, commutabile come 1-wire (in questo caso occorre collegare il piedino 4 – SCL, a GND – pin 3). Ci sono leggere differenze sul range della tensione di alimentazione, compresa fra 2,7 e 5,5V e sul campo di umidità relativa misurabile, che è tra 20 e 95% RH. Il resto è lo stesso del DHT-11.

In linea di massima un sensore vale l'altro, però il DHT-11 è l'ideale quando si dispone di poche linee libere sul microcontrollore o in Arduino, mentre il DHT-12 è più versatile perché supporta due interfacce di comunicazione.

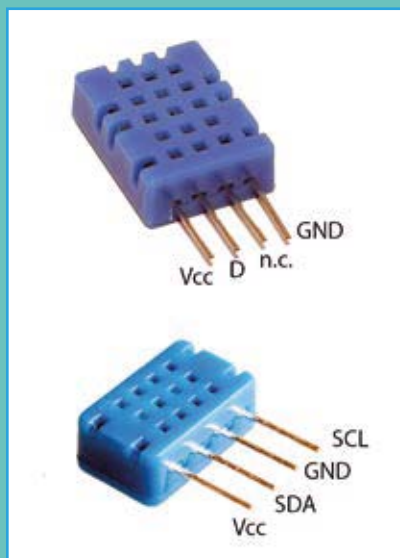
Per quanto riguarda i dati, una trasmissione completa consta di 40 bit, dove il dispositivo invia per primi i bit di valore maggiore: 8 bit sono il valore intero di RH, i successivi 8 bit sono i decimali

della RH, 8 bit successivi sono il valore intero della temperatura e gli ulteriori 8 bit i decimali della temperatura. A concludere ci sono 8 bit di checksum. Se i dati trasmessi sono corretti, il checksum dovrebbe essere formato dagli ultimi bit dei valori interi e decimali di temperatura e umidità.

Saldare gli SMD QFN

La componentistica a montaggio superficiale richiede alcune accortezze, soprattutto quando si tratta di dissaldare e saldare chip con i contatti sotto al corpo tipo QFN e BGA. Qui non ci si può affidare al saldatore ma serve la stazione ad aria calda e occorre, per evitare di surriscaldare la parte superiore del package e quindi il chip, riscaldare anche il circuito stampato a $120 \div 140^\circ\text{C}$ ponendolo a qualche centimetro da una piastra riscaldante (tipo bistecchiera). Per quanto riguarda la saldatura, una volta portata in temperatura la scheda e dopo aver applicato dello stagno e della pasta flussante alle piazzole, si appoggia l'integrato ben centrato su di esse e si scalda con la hot-air station impostata a circa 250°C , aspettando che lo stagno brilli, allorché si toglie il getto d'aria calda e si spegne la piastra riscaldante.

In dissaldatura, dopo aver scaldato la scheda inferiormente ed aver cosparso i bordi dell'integrato con pasta flussante a bassa densità, si applica il getto d'aria calda della hot-air station (sempre a una temperatura intorno ai 250°C) e si attende che le saldature brillino, ovvero che avvenga l'ebollizione del flussante, quindi con una pinzetta si verifica se il chip è ancora ancorato e, quando si mobilizza, lo si afferra con decisione (per non spostare componenti attigui) e lo si asporta.



↑ I sensori DHT-11 (in alto) e DHT-12 (in basso).

Web Forum e supporto tecnico

Hai un problema con uno dei circuiti pubblicati? Vorresti effettuare una modifica? Accedi al nostro Web Forum dove i nostri tecnici (ma anche gli altri lettori) ti aiuteranno a chiarire qualsiasi dubbio di natura tecnica. Collegati a:

www.elettronicaain.it/webforum

WHAT IF

E SE POTESSIMO BERE L'ACQUA DELL'OCEANO?

Il software e le competenze di ADI stanno contribuendo a portare la tecnologia dell'osmosi inversa a più persone, rendendo potabile l'acqua degli oceani.

Analog Devices. Where what if becomes what is.

E se.. Scopri di più: analog.com/WhatIf

 **ANALOG
DEVICES**
AHEAD OF WHAT'S POSSIBLE™

Silicon Labs propone BGM220 Explorer Kit, un kit di sviluppo progettato per dispositivi IoT che utilizzano il modulo BGM220P Wireless Gecko per disporre della connettività Bluetooth. Il modulo BGM220P si basa sul System-on-Chip EFR32BG22, ideale per lo sviluppo di applicazioni IoT connesse a basso consumo energetico, grazie al supporto BLE. Il kit BGM220 Explorer di Silicon Labs include una presa mikroBUS e un connettore Qwiic, consentendo agli sviluppatori di aggiungere funzionalità sfruttando l'ampia selezione di schede standard di MikroE, Sparkfun, AdaFruit e Seeed Studios. Inoltre, il kit

› IoT e Bluetooth: coppia vincente



presenta un'interfaccia USB, un debugger J-Link SEGGER integrato, un LED utente e un pulsante.

Il collegamento di hardware esterno al BGM220 Explorer Kit può essere ottenuto utilizzando i 20 breakout pad che sono una piedinatura comune alle periferiche del BGM220P Wireless Gecko come I²C, SPI, UART e GPIO. La connessione mikroBUS consente di inserire schede aggiuntive mikroBUS che si interfacciano con il BGM220P tramite SPI, UART o I²C-Bus, mentre il connettore Qwiic può essere utilizzato per collegare l'hardware dal sistema Qwiic Connect tramite I²C.

www.silabs.com

› Italtel e MADE animano l'Industria 4.0

Ha preso il via la collaborazione di Italtel con la Fabbrica Digitale di MADE dedicata alle PMI, finalizzata ad applicazioni di Industria 4.0 come monitoraggio e controllo smart dei processi industriali, manutenzione smart, logistica 4.0, cyber-security industriale e big data analytics. MADE è una fabbrica digitale e sostenibile che supporta le imprese manifatturiere nel percorso di trasformazione digitale verso l'Industria 4.0, rendendo disponibili conoscenze, metodi e strumenti sulle tecnologie digitali che spaziano dalla progettazione all'ingegnerizzazione, dalla gestione della produzione alla consegna, fino alla gestione del termine del ciclo di vita del prodotto. In un demo-center di 2.500 mq con aule per la formazione e spazi per co-working e riunioni, rappresenta una soluzione unica nel suo genere. Lo spazio è suddiviso in sei aree con competenze in diverse tecnologie 4.0: dalla robo-

tica collaborativa alla manutenzione da remoto, dai big data al gemello digitale, dall'efficienza energetica alla cyber-security industriale all'Industrial IoT. Nell'area Monitoraggio e Controllo Smart dei Processi Industriali, Monitoraggio e Controllo Energetico Smart, Manutenzione Smart" Italtel ha contri-

buito alla realizzazione di due casi d'uso: il primo legato all'analisi e al monitoraggio di KPI specifici per il Plant Manager; il secondo al monitoraggio ambientale remoto di un impianto presente presso uno stabilimento di un'azienda manifatturiera italiana (BLE)..

www.italtel.com/it



› IoT Industriale? Meglio geolocalizzato!

Digital Matter, fornitore di hardware per Industrial Internet of Things (IIoT) con supporto GPS e software di gestione di dispositivi IoT, ha utilizzato la piattaforma di asset management LoRa Edge di Semtech per sviluppare Yabby Edge, il suo innovativo dispositivo di localizzazione alimentato a batteria dedicato all'management indoor e outdoor.

Alla base del dispositivo c'è la piattaforma LoRa Edge LR1110 di Semtech (descritto sul numero 252 di Elettronica In) che semplifica la gestione delle risorse IoT in applicazioni chiave come il monitoraggio di pallet e attrezzature di magazzino, gestione degli stock di merce e delle spedizioni, il monitoraggio di carrelli e dei container, la gestione di

apparecchiature mediche e molto altro ancora, avvalendosi della geolocalizzazione implementata dal ricevitore GPS integrato nella piattaforma LoRa di Semtech.

Il dispositivo Yabby Edge offre calcoli avanzati sulla posizione basati su Cloud, riducendo in modo significativo il consumo energetico e potendo così contare su una durata della batteria fino a 12 anni, il che riduce i tempi di fermo e tutto quel che è connesso all'intervento di sostituzione riducendo nel contempo i costi.

I dati sulla posizione possono essere facilmente inoltrati a qualsiasi piattaforma o sistema del cliente per una semplice integrazione con i software gestionali e di magazzino.

› Le foto animate dall'Intelligenza Artificiale

MyHeritage è un'azienda americana che si occupa di ricerche genealogiche, con la vocazione di ricongiungere famiglie trovando appartenenti, discendenti e parenti di cui spesso si ignora l'esistenza. Estendendo e forse estremizzando questo concetto, ha licenziato una straordinaria tecnologia per l'animazione delle foto da D-ID, un'azienda israeliana specializzata nella costruzione di video da foto utilizzando il deep learning. MyHeritage ha integrato nei propri servizi questa tecnologia per animare i volti nelle foto di famiglia dei propri clienti e creare sequenze video realistiche di alta qualità. La funzionalità è stata chiamata

Deep Nostalgia e utilizza modelli di Machine Learning realizzati da MyHeritage, ciascuno dei quali è un video che consiste in una sequenza di movimenti e gesti. Deep Nostalgia può applicare tali modelli con grande accuratezza ad uno o più volti presenti in una fotografia, creando un video che potrebbe animare la foto di una persona che non c'è più, per ricordarla meglio e per sentirla più vicina; in breve, per fare come se fosse viva, seppure nello schermo di un computer o di un qualsiasi dispositivo in grado di riprodurre filmati digitali. Questa tecnologia, stupefacente da un lato e per certi versi opinabile dall'altro, è l'ennesima dimostra-

zione di ciò che l'Intelligenza Artificiale permette di fare oggi, soprattutto della possibilità di creare filmati a partire da fotografie, cosa del resto già sperimentata in numerosi video fake ritraenti personaggi famosi intenti a compiere azioni che in realtà non sono mai state compiute. Per applicare l'animazione la funzionalità Deep Nostalgia richiede che l'immagine del volto sia ad alta risoluzione, ma i volti nelle foto storiche tendono ad essere piccoli e sfuocati; tuttavia l'azienda risolve brillantemente il problema mediante una funzionalità chiamata MyHeritage Photo Enhancer, che mette a fuoco le foto a bassa nitidezza e risoluzione, aumentandone la risoluzione e rendendo più nitidi i volti che vi compaiono. Questa combinazione garantisce risultati ottimali e produce animazioni video di alta qualità anche partendo da foto storiche, ritraenti, ad esempio, personaggi del passato: una dimostrazione di ciò è stata data in un video demo che ritrae i volti di Giuseppe Verdi, Abraham Lincoln, Charles Baudelaire, Giuseppe Garibaldi ed altri personaggi, animati. Per quanto potente, l'algoritmo di Intelligenza artificiale può creare il modello di un solo viso alla volta, ragion per cui pur potendo animare più persone che compaiono in una fotografia, l'animazione è fattibile indicando un solo volto.

www.myheritage.com



Le impostazioni del dispositivo possono essere configurate per adattarsi a qualsiasi applicazione di tracciamento. La piattaforma LoRa Edge di Semtech, con il suo hardware multitecnologico integrato e integrabile, il suo software modem nativo LoRa Basics Modem E operante sul protocollo LoRaWAN, e la sua geolocalizzazione LoRa Cloud con i relativi servizi di dispositivi e applicazioni, semplificano e accelerano lo sviluppo di soluzioni IoT. Il dispositivo Yabby Edge sarà disponibile nelle versioni con frequenze 868 MHz e 902+928 MHz del protocollo LoRaWAN.

www.semtech.com



› Data Analysis? Ci pensano IBM, RedHat e Siemens



Siemens, IBM e Red Hat hanno annunciato una collaborazione che utilizzerà un Cloud ibrido progettato per fornire una soluzione aperta, flessibile e più sicura per produttori e operatori di impianti industriali, finalizzata a ottenere valore in tempo reale dai dati operativi. In un mese, un singolo sito di produzione può generare più di 2.200 terabyte di dati, che però spesso non vengono sfruttati. Siemens Digital Industries Software applicherà l'approccio Cloud ibrido open di IBM, basato su Red Hat OpenShift, per estendere la flessibilità di implementazione di MindSphere, la soluzione IIoT as a service di Siemens, consentendo di eseguire MindSphere

on-premise, sbloccando velocità e agilità nelle operazioni di fabbrica e impianto, nonché attraverso il Cloud per supporto del prodotto, aggiornamenti e connettività aziendale senza interruzioni. MindSphere fornisce già ai clienti data-driven insight di lavorare meglio con l'Industrial IoT. Grazie al lavoro congiunto con IBM e Red Hat, Siemens può offrire ai clienti la flessibilità di scegliere di utilizzare MindSphere on-premise o nel Cloud per soddisfare al meglio le proprie esigenze operative e diventare più efficienti, agili e reattivi nei confronti del mercato. La maggior parte dei dati industriali viene generata al di fuori dell'IT, nelle

operazioni di produzione, nella supply-chain o nei prodotti connessi, ma per analizzarli occorre inviarli a un sistema informatico; oggi molte aziende scelgono di inviare i dati al loro Cloud aziendale o di ridurre la complessità dei sistemi IT.

MindSphere viene utilizzato dalle organizzazioni per raccogliere e analizzare i dati dei sensori in tempo reale da prodotti, impianti, sistemi e macchine, consentendo agli utenti di ottimizzare prodotti, risorse di produzione e processi di produzione lungo l'intera catena del valore per costruire un gemello digitale in tempo reale. Adottando Red Hat OpenShift, la piattaforma Kubernetes aziendale leader del settore, come architettura on premise preferita, i clienti avranno la flessibilità di eseguire le soluzioni MindSphere localmente in un Cloud privato o anche in applicazioni future attraverso un modello ibrido e multi-cloud come campo abilitante per le intuizioni aziendali. Attraverso questa offerta, Siemens e IBM consentono ai clienti di mantenere il pieno controllo fisico dei propri dati per far fronte meglio ai requisiti normativi e alla riservatezza dei dati.

www.ibm.com

INDISPENSABILE IN LABORATORIO!



L'alimentatore è lo strumento più importante di ogni laboratorio elettronico da cui si esige prima di tutto la possibilità di regolare con buona risoluzione la tensione e la corrente di uscita, requisiti soddisfatti da questi nuovi alimentatori. Mantengono i valori impostati in modo affidabile e indipendentemente dalle fluttuazioni della tensione di alimentazione o del carico collegato.



DISPONIBILE ANCHE NELLE VERSIONI:

DA 0-30 VDC / 0-10 A

COD. AL3010 € 106,00

DA 0-60 VDC / 0-5 A

COD. AL6005 € 106,00

€ 78,00

Cod. **AL3005N**

ALIMENTATORE LINEARE CON USCITA REGOLABILE 0-30 VDC / 0-5 A

Alimentazione: 110/220 VAC, 50/60 Hz

Uscita duale regolabile:

- Tensione: da 0 a 30 VDC

- Corrente: da 0 a 5 A

Risoluzione della tensione: 10/100 mV

Risoluzione della corrente: 1/10 mA

Precisione visualizzazione della tensione: $\pm 1\%$ +1 cifra

Precisione visualizzazione della corrente: $\pm 1\%$ +2 cifre

Dimensioni: 220x165x85 mm

Peso: 1,350 kg



Prezzi IVA inclusa.

Nuovo controller LED LIN RGB automotive da Melexis

Si chiama MLX81114 ed è il nuovo integrato controller LED RGB, che incrementa la famiglia di dispositivi a interfaccia LIN per la gestione dell'illuminazione interna automobilistica basata su LED RGB, nota anche come LIN RGB, che è una realtà ormai assodata di quasi tutti gli OEM del mondo.

L'integrato MLX81114 è una variante dell'MLX81113 con in più una protezione dell'accesso alla memoria interna che abilita le applicazioni antifurto ed è disponibile nel package SOIC-8.

Altre caratteristiche del controller includono memoria on-chip, corrente di uscita elevata e grande immunità ai disturbi elettromagnetici (EMC) accanto alla conformità alla sicurezza funzionale ISO 26262 per supportarne l'uso nei sistemi classificati ASIL-A.

L'illuminazione interna automobilistica è diventata una delle caratteristiche chiave per la personalizzazione e la dif-



ferenziazione all'interno dell'abitacolo; Melexis risponde a questa tendenza con la sua offerta di driver RGB(W) compatibili con lo standard LIN per consentire un'illuminazione ambientale sofisticata ma di facile implementazione per le auto, dal livello base ai

modelli di fascia media e di lusso.

Il controller è una periferica standard slave LIN basata su micro 16-bit pipelined RISC con 32 kB di Flash, System ROM con bootloader, 2kB di RAM e 576 Byte EEPROM.

www.melexis.com

Buffer hold-up TDK per alimentatori



TDK annuncia l'introduzione dei moduli buffer ZBM20 open frame in grado di offrire 380 ms di hold-up esteso per alimentatori AC/DC con tensioni di uscita 12V, 15V e 24V. Il tempo di hold-up esteso permette lo spegnimento delle apparecchiature collegate in modo sicuro senza perdita di dati. Le applicazioni tipiche sono nell'automazione industriale, robotica e attrezzature per la produzione di semiconduttori. Per prevenire scariche incontrollate dell'energia immagazzinata, può essere usata la funzione on/off remota per inibire l'uscita. Lo stato

del prodotto è accessibile da remoto tramite un relé DC OK o segnali foto-accoppiati. Più buffer possono essere collegati in parallelo per aumentare il tempo di hold-up. L'energia di mantenimento è immagazzinata in condensatori elettrolitici invece che in batterie, per eliminare la necessità di assistenza e manutenzione. Nel modello 24V si può usare un interruttore per selezionare livelli di buffer a tensione fissa o variabile. In modalità fissa, il modulo fornisce alimentazione quando la tensione di ingresso scende a 22,4V, in modalità variabile quando la tensione di ingresso diminuisce di 1V rispetto al valore iniziale. I moduli ZBM20 misurano 175 x 85 x 57 mm (LxWxH) e operano a temperature ambiente da -25°C a +70°C. Sono certificati IEC/UL/CSA/EN 62368-1.

www.tdk.com

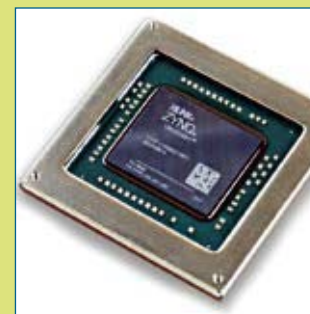
Xilinx supporta Fujitsu nello sviluppo 5G

Le unità radio Fujitsu 5G incorporano la tecnologia Xilinx UltraScale+ per il sito macro O-RAN e le reti greenfield. La tecnologia Xilinx RFSoc consente di ridurre i costi e il consumo energetico e le O-RU Fujitsu sono ideali per un'ampia gamma di applicazioni a spettro e multi-banda per reti O-RAN 5G. I dispositivi Xilinx UltraScale+ utilizzati nelle O-RU Fujitsu offrono il miglior equilibrio tra costo, adattabilità e scalabilità per soddisfare i requisiti di rete O-RAN 5G.

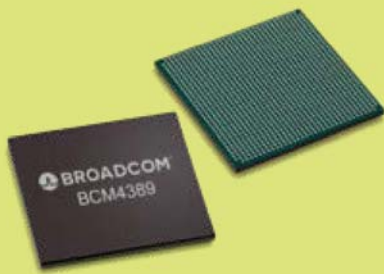
La famiglia di RFSoc Xilinx Zynq UltraScale+family comprende SoC per radiofrequenza che integrano i sottosistemi richiesti dall'implementazione di una Software-Defined Radio, compresi i Direct RF Sampling data converter, supportando CPRI ed Ethernet Gigabit Ethernet-to-RF in un SoC programmabile. Ogni RFSoc offre 8 ADC RF a

campionamento multiplo (RF-ADC) a 12-bit e 4.096 GSPS e 16 RF-DAC 14-bit, 6.554 GSPS, 1x (bypass) 2x, 4x, 8x decimation, Digital Complex Mixer e un Numerical Controlled Oscillator (NCO). Xilinx supporta, oltre a Fujitsu, altri attori dell'ecosistema O-RAN nella convalida continua dell'hardware e del software per le reti 5G mondiali. I primi sistemi Greenfield 5G O-RAN saranno implementati quest'anno.

www.xilinx.com



Il primo Smartphone con Wi-Fi 6E monterà il Broadcom BCM4389



Broadcom ha annunciato che Samsung Galaxy S21 Ultra, il primo smartphone Wi-Fi 6E, verrà equipaggiato con il transceiver BCM4389 a 2 Gbps dotato anche di multi-radio Bluetooth 5. Wi-Fi 6E è l'ultima generazione di Wi-Fi, capace di latenze dell'ordine di millisecondi operando nella banda dei

6 GHz. Il Broadcom BCM4389 porta le prestazioni del Wi-Fi 6 nella banda esistente dei 5 GHz, con supporto 160 MHz. Il BCM4389 è stato il primo chip Wi-Fi al mondo autorizzato dalla FCC a funzionare nella banda dei 6 GHz. Oltre al Wi-Fi 6E, l'architettura di connettività TBS del BCM4389 offre una

migliore QoS Wi-Fi, una precisione della posizione più precisa e un utilizzo della batteria cinque volte migliore delle attuali soluzioni di connettività. Il chip opera nelle bande senza licenza 2,4 e 5,1-7,125 GHz con la modulazione 1024-QAM, supportando MIMO e OFDMA.

www.broadcom.com

Da Microchip, soluzioni Vehicle Ethernet AVB

Microchip presenta la prima soluzione completamente integrata per il bridging Audio Video - Ethernet (AVB) in auto. La soluzione endpoint audio basata su hardware per AVB LAN9360 è un controller Ethernet single-chip con protocolli incorporati. Il controller endpoint audio LAN9360 di Microchip interconnette i dispositivi di infotainment dei veicoli tra cui altoparlanti, amplificatori, microfoni, sistemi di navigazione, sintonizzatori radio e poggiatesta intelligenti con l'Ethernet AVB. Il LAN9360 collega l'audio tra le interfacce audio locali Ethernet AVB e Inter-IC Sound (I²S), TDM (Time Division Multiplexing) e PDM (Pulse Density Modulation). Supporta completamente la trasmissione audio su Ethernet AVB, incluso gPTP Generalized Precision Time Protocol, timestamp, protocolli di trasporto e protezione dei contenuti con HDCP (High-bandwidth Digital Content Protection). Supporta inoltre l'avvio protetto e gli aggiornamenti remoti protetti tramite Ethernet. A differenza di altre soluzioni di rete bridging Ethernet che richiedono micro-



controllori (MCU) System-on-Chip (SoC) più stack di software di terze parti, il LAN9360 non richiede integrazione software. Il LAN9360 è stato convalidato secondo gli standard per l'interoperabilità Ethernet per i protocolli AVB, secondo IEEE 802.1BA-2011, IEEE 802.1AS, IEEE1722 e IEEE1733 per le reti Ethernet ed è certificato secondo gli standard di interoperabilità e affidabilità stabiliti dal consorzio Avnu Alliance. Altri dispositivi Microchip Ethernet per applica-

zioni automotive includono il LAN8770 100BASE-T1 PHY, un RTX a livello fisico a porta singola, che fornisce capacità di trasmissione e ricezione a 100 Mbps su singolo cavo UTP (Unshielded Twisted Pair). Microchip rende disponibile una scheda di sviluppo e l'MPLAB Network Creator (è un ambiente di configurazione grafica gratuito) per configurare il LAN9360 utilizzando un'interfaccia utente grafica molto intuitiva.

www.microchip.com

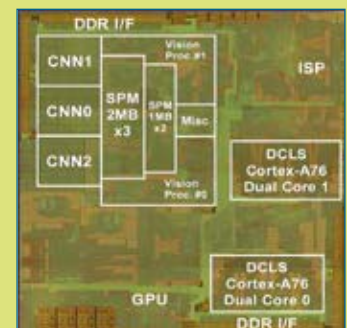
Nuovi SoC Automotive Renesas per ADAS

Renesas Electronics annuncia lo sviluppo di tecnologie di processore per SoC dedicati ad applicazioni come i sistemi avanzati di assistenza alla guida (ADAS) e la guida autonoma (AD) che mirano a ottimizzare sia le prestazioni che l'efficienza energetica, supportando un elevato livello di sicurezza funzionale.

La tecnologia include un core acceleratore hardware di rete neurale convoluzionale (CNN) che offre prestazioni di deep learning a 60,4 trilioni di operazioni al secondo (TOPS) e un'efficienza energetica di 13,8 TOPS/W; prevede inoltre sofisticati meccanismi di sicurezza per il rilevamento rapido e la risposta a guasti hardware

casuali. Integra, infine, un sistema che consente alle attività software con diversi livelli di sicurezza di operare in parallelo sul SoC senza interferire tra loro, rafforzando così la sicurezza funzionale per il controllo ASIL D. Queste tecnologie sono state applicate nel recente SoC R-Car.

www.renesas.com



Fotorelè ad alta corrente per uso industriale da Toshiba



TLP241B è il più recente fotorelè ad alta corrente di Toshiba Electronics Europe destinato all'uso nei componenti industriali, come i controllori logici programmabili (PLC) e le interfacce I/O, nonché nei sistemi di automazione degli edifici come le unità HVAC (riscaldamento, ventilazione e condizionamento dell'aria). Incorporando un MOSFET U-MOS ultra-avanzato di Toshiba quale componente di uscita, il TLP241B stabilisce nuovi parametri di riferimento per le prestazioni. È il primo dispositivo fotorelè sul mercato ad avere una tensione del termi-

nale di uscita in stato di off che va da 40V a 100V.

Di conseguenza, ha la capacità di sostituire i relè elettromeccanici di dimensioni più grandi, fornendo un'alternativa più affidabile, economica e salvaspazio. Basandosi su un sistema ottico allo stato solido piuttosto che su parti mobili, permette vita operativa più lunga, minore complessità del sistema e risposta più rapida. Inoltre, il TLP241B possiede altre caratteristiche chiave che ne aumenteranno la convenienza. Il dispositivo offre una corrente nominale in stato di ON di 2A in regime continuo e 6A pulsati. Offre inoltre capacità di isolamento da 5kV, con un intervallo di temperature di esercizio che va da -40 a 110. Questo nuovo fotorelè è alloggiato in package DIP4 ed opzionalmente ad M per il montaggio superficiale.

www.toshiba.semicon-storage.com

Umidità e temperatura di precisione

In linea con i sensori di umidità e temperatura che caratterizzano l'offerta di Sensirion, il nuovo SHT40 si distingue per il miglior rapporto qualità-prezzo. L'SHT40 si basa su un chip CMOSens completamente nuovo e ottimizzato che offre un consumo energetico ridotto e specifiche di precisione migliorate. Con la gamma estesa di tensione di alimentazione da 1,08 V a 3,6 V, è la soluzione per applicazioni mobili e alimentate a batteria, anche in virtù del ridotto assorbimento: 0,6 μ A (con frequenza di misura di 1 Hz) e 80 nA a riposo.

Le dimensioni ridotte in un robusto alloggiamento DFN consentono l'integrazione in circuiti dove l'ingombro è un fattore critico, soddisfacendo i requisiti di affidabilità



più elevati, come dimostrato dalla qualifica JEDEC JESD47. La precisione della misura di umidità relativa è l'1,8% e quella sulla temperatura, fino a 0,1 °C. Nel 2021 saranno disponibili anche le versioni ad alta precisione SHT41 e SHT45, che si distinguono per precisioni tipiche e massime perfezionate, fino a $\Delta RH \pm 1,5\%$ RH e $\Delta T = \pm 0,1$ °C. La membrana di filtro sul package garantisce la conformità IP67 per le applicazioni esposte ai liquidi. L'interfaccia dati è I²C high-speed.

www.sensirion.com

Il primo Data Acquisition System per BMS da Maxim

I progettisti di sistemi di gestione della batteria per autoveicoli (Battery Management System - BMS) possono ora contare sul MAX17852 di Maxim, un sistema di acquisizione dati ASIL-D a 14 canali ad alta tensione, progettato per scatole di giunzione intelligenti e altri sistemi di batterie per automobili fino a 400 V.

Il MAX17852 consente di progettare i sistemi per soddisfare i più elevati standard ASIL-D per tensione, corrente, temperatura e comunicazione, allo scopo di incrementare la sicurezza del pacco batteria.

L'integrato offre anche la massima precisione dei dati di tensione, corrente e temperatura, con una stretta sincronizzazione temporale.

Il MAX17852 consente una tipica misurazione della tensione delle celle di $\pm 0,45$ mV a temperatura ambiente e un errore massimo di ± 2 mV in un intervallo di temperatura compreso tra 5 e 40 °C, consentendo ai costruttori di automobili elettriche di massimizzare l'autonomia ottenibile dalle batterie delle auto. Con una tolleranza dell'amplificatore di rilevamento della corrente di ± 300 mV e un'im-

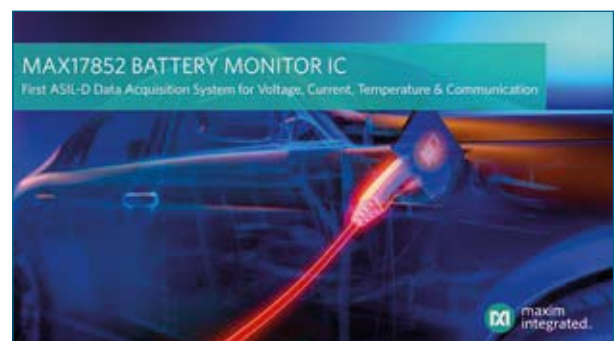
postazione del guadagno fino a 256 volte, nonché un errore nel guadagno del circuito di rilevamento della corrente di appena lo 0,3%, il MAX17852 fornisce i dati più velocemente e accuratamente possibile per i calcoli dei parametri inerenti alla gestione dell'alimentazione, allo stato di salute e di carica del pacco batteria.

Il sistema di acquisizione dati della batteria a 14 canali integra un amplificatore di rilevamento della corrente in modo che le informazioni sulla corrente vengano acquisite contemporaneamente alla tensione e alla temperatura della cella.

Il MAX17852 consente di utilizzare sia il sensore ad effetto Hall che i resistori di shunt come componenti di rilevamento della corrente.

L'elevata precisione di misurazione con il miglior errore di guadagno dell'amplificatore di rilevamento della corrente consente misurazioni di corrente accurate e allineate nel tempo per aumentare l'autonomia. L'alta integrazione consente dimensioni fino al 16% inferiori rispetto a una soluzione discreta e un risparmio sui costi del Battery Management System fino al 20%.

www.maxim-integrated.com





ABBONAMENTO
ANNUALE



Abbonati a Elettronica In!
Realizza i **Progetti** descritti sulla rivista,
rimani aggiornato con i nostri **Corsi**,
approfondisci la conoscenza delle
tecniche e dei componenti
più avanzati con i **Tutorial**
e le nostre **News**.



Visita: www.elettronica.in/abbonamenti



ABBONAMENTO ANNUALE PER RIVISTA DIGITALE

In **OMAGGIO** avrai
il **10% DI SCONTO** per i tuoi ordini
su **Futurashop.it**

COME ABBONARSI

→ **on-line**

compilando il modulo riportato
nella pagina "**Abbonamenti**"
del nostro sito www.elettronica.in.

→ **e-mail**

scrivendo a
abbonati@elettronica.in

→ **telefono**

telefonando a
+39-0331-752668

ABBONAMENTO ANNUALE PER RIVISTA **CARTACEA**

In **OMAGGIO** avrai
il **10% DI SCONTO** per i tuoi ordini
su **Futurashop.it**

GRATIS 1 VOLUME A SCELTA della collana
"L'elettronica per tutti" oppure

50% DI SCONTO per l'acquisto di un volume
della libreria "Elettronica In"

La possibilità di **LEGGERE GRATUITAMENTE**
la versione digitale di **ELETTRONICA IN**



Temperatura sempre sotto controllo!

Termometri a infrarossi senza contatto



€ 38,00
Cod. TP213N

Termometro IR con puntatore LASER (da -20°C a +400°C)

Termometro a infrarossi UNI-T con puntatore LASER per la misura a distanza della temperatura (da -20°C a +400°C). Dispone di display LCD con retroilluminazione, possibilità di visualizzazione in gradi centigradi o in gradi Fahrenheit, spegnimento automatico. La lettura della temperatura è istantanea. La confezione comprende: il termometro e due batterie da 1,5 V tipo AAA.

Termometro IR con puntatore LASER (da -32°C a +420°C)

Termometro a infrarossi UNI-T con puntatore LASER per la misura a distanza della temperatura (da -32°C a +420°C). Dispone di display LCD con retroilluminazione, possibilità di visualizzazione in gradi centigradi o in gradi Fahrenheit, spegnimento automatico. La lettura della temperatura è istantanea (250 ms) con la memorizzazione della soglia minima e massima in caso di scansione continua e prolungata. Memorizzazione fino a 5 set di valori. La confezione comprende: il termometro, la pila da 9 V e la borsa da cintura per il trasporto.

€ 78,00
Cod. TP224



Prezzi IVA inclusa.



€ 19,00
Cod. FR811

Termometro IR per la lettura della temperatura corporea o di superfici (da +34°C a +42,9°C)

Dispone di ampio display a LED retroilluminato che mostra la lettura in gradi centigradi o Fahrenheit (selezionabile). Gli infrarossi consentono al termometro di leggere la temperatura corporea a una distanza di 5-10 cm in meno di un secondo. L'apparecchio si spegne dopo 8 secondi di standby. Il termometro ha una precisione del display di 0,1°C con un errore massimo di $\pm 0,2^\circ\text{C}$ e un range $34,0^\circ\text{C} \sim 42,9^\circ\text{C}$. Dispone inoltre di segnalazione di allarme con soglia impostabile, cambio di colore del display per indicare le soglie di temperatura misurate. Offre la possibilità di memorizzare fino a 32 misurazioni.



FILTRI LC PER ALIMENTATORI

di GABRIELE PERETTO



Riduciamo il ripple e le spurie inevitabilmente presenti all'uscita dei convertitori DC/DC switching, in maniera semplice ed efficace.

La tecnologia degli alimentatori "switching" si è da tempo affermata, tanto che in molti ambiti dove si utilizzavano alimentatori lineari oggi si preferiscono quelli a commutazione e i DC/DC, utilissimi per passare da una tensione continua a un'altra: buck per ottenere valori minori di quello di partenza, boost per elevare la tensione. Gli switching hanno il pregio di presentare elevata efficienza (fino al 95%) e di essere molto compatti; per contro, spesso presentano un elevato "ripple" (ondulazione residua) in uscita. Normalmente negli alimentatori lineari AC/DC, composti da un trasformatore di rete, uno o più diodi raddrizzatori e alcuni condensatori di filtro, la frequenza è pari a quella di rete e cioè 50 Hz, oppure 100 Hz se il raddrizzatore è onda intera; qui il ripple è sempre funzione di queste due frequenze. Nel caso degli alimentatori a commutazione, la frequenza di lavoro va da qualche decina di kHz fin oltre 1 MHz, pertanto possiamo

trovare numerose armoniche talvolta di ampiezza anche di alcuni Vpp, che in alcuni casi possono creare dei problemi di funzionamento ai circuiti, soprattutto in HF, nonché creare dei disturbi a tutte le apparecchiature elettroniche situate nelle immediate vicinanze, trattandosi di veri e propri trasmettitori di rumore radio.

Per migliorare la situazione si può porre all'uscita dello switching un filtro in grado di attenuare sensibilmente questi disturbi. In questo articolo, andremo ad analizzare in dettaglio le caratteristiche e presenteremo un paio di progetti molto semplici, versatili e facilmente modificabili e adattabili in funzione delle varie necessità.

FILTRI IDEALI, FILTRI PASSIVI e FILTRI ATTIVI

Un filtro è un circuito elettrico o elettronico che consente il passaggio ad ampiezza inalterata di segnali a frequenze o intervalli di frequenze ben precisi, mentre ostacola il passaggio di segnali a frequenze diverse.

Un "filtro ideale" rappresenta un caso limite di questa definizione nella quale il passaggio dei segnali entro determinate frequenze, ovvero nella banda passante, avviene senza alcuna attenuazione mentre il blocco dei segnali su frequenze diverse, cioè nella banda di reiezione, avviene con attenuazione infinita.

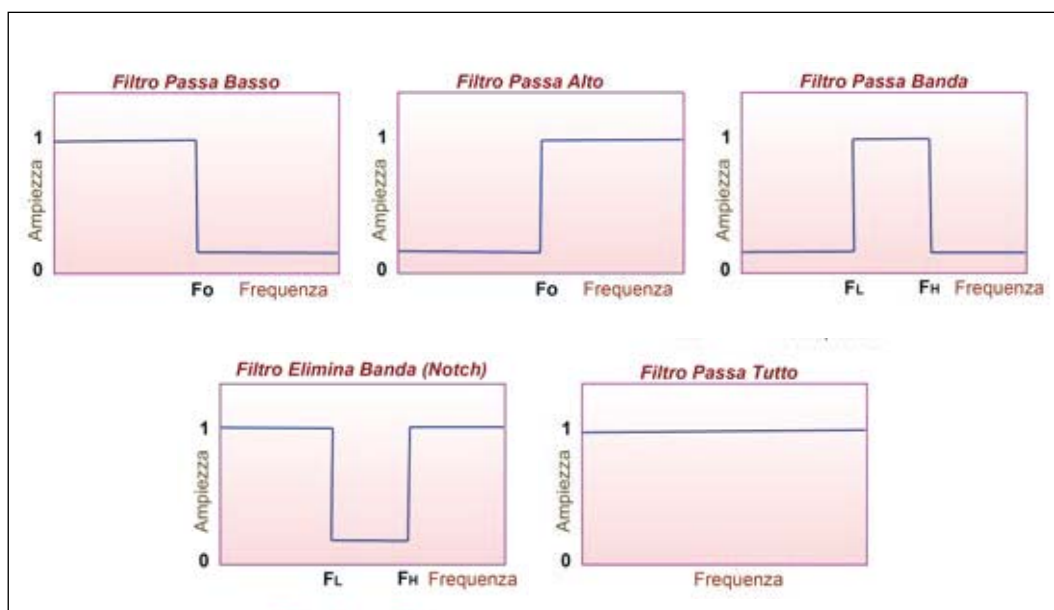
Tenendo presente questo comportamento si possono classificare cinque tipi fondamentali di filtri (**Fig. 1**), ognuno con caratteristiche peculiari ed in parte complementari.

- **Filtro passa-basso:** lascia passare i segnali con frequenza da zero fino alla sua frequenza di taglio F_o e blocca tutte le frequenze superiori a questa.
- **Filtro passa-alto:** blocca i segnali con frequenza inferiore alla sua frequenza di taglio F_o e lascia passare tutte le frequenze superiori a questa.
- **Filtro passa-banda:** lascia passare i segnali con frequenza compresa tra le sue due frequenze di taglio inferiore F_L e superiore F_H mentre blocca tutte le frequenze esterne a questo intervallo.
- **Filtro elimina-banda (notch):** blocca i segnali con frequenza compresa tra le sue due frequenze di taglio inferiore F_L e superiore F_H mentre lascia passare tutte le frequenze esterne a questo intervallo.
- **Filtro passa-tutto:** lascia passare i segnali di tutte le frequenze, quindi non è un vero filtro ma si usa per produrre uno sfasamento costante indipendentemente dalla frequenza.

I vari grafici proposti in **Fig. 1** illustrano "idealmente" il comportamento dei filtri, ovvero la sua risposta in frequenza. Questa rappresentazione assume per i filtri un ruolo primario per la quantità di informazioni che può fornire.

Filtri passivi e attivi

I filtri possono venire realizzati con tre diverse tecnologie. In ordine di complessità circuitale crescente troviamo i *filtri passivi* composti da sole



→ **Fig. 1**
Tipologie di filtri.

induttanze e capacità (**LC**), oppure resistenze e capacità (**RC**), o anche (**LR**) (induttanze e resistenze); i **filtri attivi** composti da resistenze e capacità nella rete di un amplificatore, solitamente di tipo operativo (**OP Amp**), oppure in una rete di capacità commutate (**SC**); i filtri DSP (Digital Signal Processing) composti da un microprocessore dedicato, che esegue elaborazioni matematiche sui segnali nel dominio digitale. In queste pagine tratteremo soltanto i filtri passivi di tipo LC, che sono quelli che ci interessano per le nostre applicazioni relative al filtraggio degli alimentatori. Il filtro passivo si caratterizza per i seguenti vantaggi: campo di frequenze applicative vastissimo, dagli Hz alle centinaia di GHz; assenza di distorsioni sul segnale (i componenti sono in prima approssimazione puramente lineari), assenza di alimentazione, operatività garantita indipendentemente dall'applicazione e svantaggi: possibile risonanza.

PARAMETRI E CARATTERISTICHE DEI FILTRI

Prima di sviluppare il progetto di un filtro è di primaria importanza definirne i parametri di funzionamento, ovvero significa impostare quale risposta deve presentare nel dominio della frequenza e del tempo.

Va sottolineato che i filtri, come tipologia di circuiti, richiederebbero complesse nozioni matematiche per una analisi dettagliata del loro comportamento. Non solo: a seconda dell'ambito di applicazione, talune concezioni assumono più o meno rilievo, richiedendo pertanto un approfondimento specifico. Tuttavia, accettando alcune semplificazioni, risulta possibile adottare un approccio che si basa su definizioni standard, valide nella pratica per buona parte delle situazioni reali.

Nel nostro caso tratteremo il tipo **passa-basso**

a **LC**, senz'altro la forma più diffusa e più utilizzata. Quanto illustreremo con pochi adattamenti, avrà poi validità generale indipendentemente dalla struttura del filtro. Una rete passa-basso può essere specificata in funzione di quattro parametri che la **Fig. 2** propone graficamente e che sono:

- **F₀** = frequenza di taglio; è la frequenza alla quale la risposta del filtro supera il segmento di banda passante e che tipicamente corrisponde a -3 dB per i filtri Butterworth;
- **F_R** = frequenza di reiezione; è la frequenza alla quale la risposta del filtro, raggiunge la minima attenuazione richiesta nella banda di reiezione;

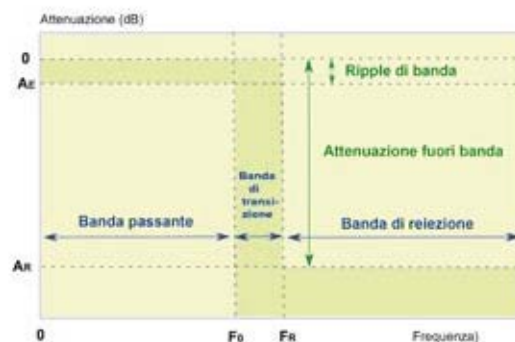


Fig. 2
Parametri di progetto per filtro passa basso.

- **A_E** = ripple di banda; è la massima attenuazione ammessa nel segmento di banda passante corrispondente tipicamente a -3dB;
- **A_R** = attenuazione fuori banda; è la minima attenuazione richiesta nella banda di reiezione.

In un progetto reale, non tutti questi parametri hanno uguale importanza; ad esempio, il "ripple di banda" potrebbe essere un fattore determinante nelle applicazioni audio di alta qualità dove si vuole alterare il meno possibile la risposta inferiore agli usuali 20 kHz. Nei filtri a banda medio/stretta, invece, spesso il fattore chiave è la "riduzione della banda di transizione". Il grafico poco sopra proposto evidenzia i limiti entro i quali la risposta in frequenza deve ricadere, un ipotetico filtro passa-basso che risponde ai requisiti che abbiamo imposto e visibile in **Fig. 3**. La curva di colore rosso rappresenta la risposta della rete, dato che si trova entro i limiti di progetto e si dimostra adatta alle nostre esigenze; tuttavia, come è intuibile vi sono diversi tipi di risposta in frequenza che possono soddisfare i requisiti per ogni particolare filtro. Esistono principalmente quattro famiglie che si distinguono per caratteristiche proprie. Nella **Tabella 1** sono elencate le caratteristiche base per ognuna di esse. La notazione (**n**) indica che la pendenza nella banda di transizione, qui presa a

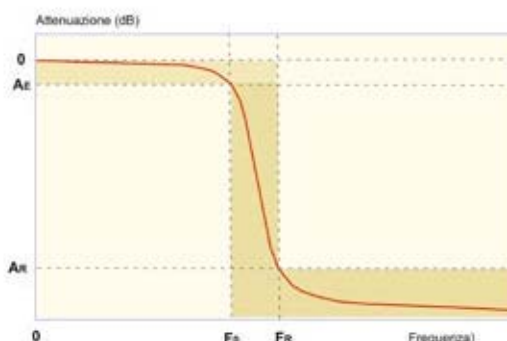


Fig. 3
Curva di risposta tipica di un filtro passa basso.

→ **Tabella 1**
Caratteristiche delle
quattro categorie
di filtro.

TIPO DI RISPOSTA	RIPPLE IN BANDA	PENDENZA (n)
Bessel Presenta la più lineare risposta in fase in banda passante, a scapito di una bassa pendenza di attenuazione oltre la frequenza di taglio.	3 dB	<6 dB/ottava
Butterworth Presenta il miglior compromesso tra attenuazione e risposta in fase. Non ha ondulazioni in banda passante a spese di una relativamente ampia banda di transizione. È il tipo di risposta più usato.	3 dB	6 dB/ottava
Chebyshev Presenta una elevata selettività, ovvero una maggiore pendenza di attenuazione oltre la frequenza di taglio. Ha ondulazioni in banda passante.	>0 dB	>6 dB/ottava
Ellittica Presenta la più elevata selettività, ovvero la maggiore pendenza di attenuazione oltre la frequenza di taglio, a scapito di una degradata risposta in fase. Ha ondulazioni sia nella banda passante che nella banda di transizione.	>0 dB	>6 dB/ottava

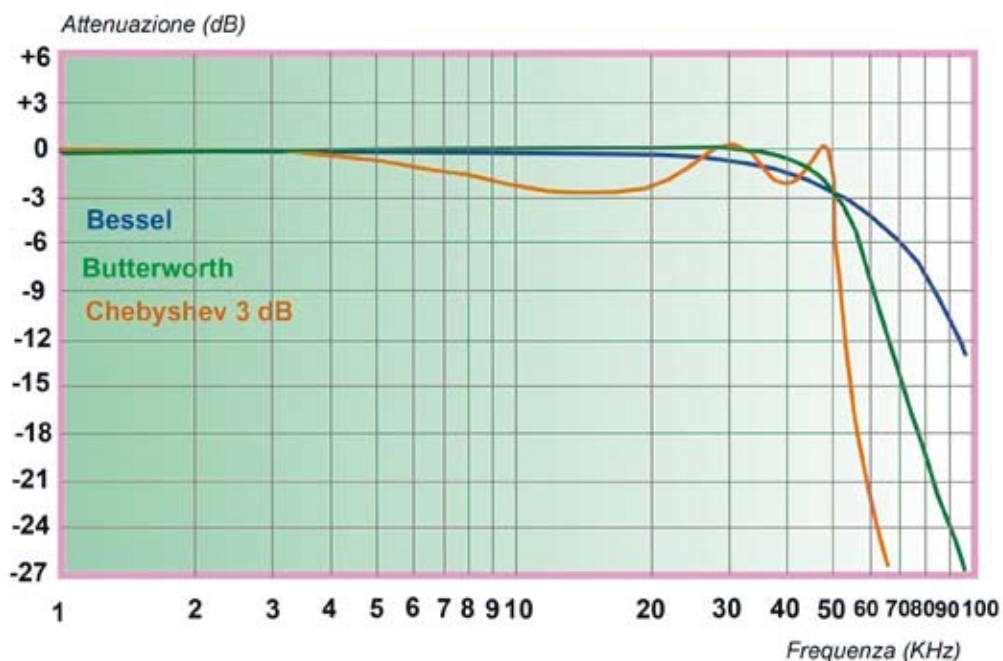
riferimento, concerne le reti di 1° ordine. Da notare che nei filtri Bessel e Butterworth la frequenza di taglio F_0 è per definizione coincidente con il punto ove l'attenuazione assume il valore di 3 dB. Invece per i filtri Chebyshev ed Ellittici invece si considera il punto ove si supera il valore di ripple, questo è imposto dal progetto essendo una variabile e non una caratteristica intrinseca al tipo di risposta. La pendenza di attenuazione lontano dalla frequenza

di taglio per tutti i filtri, ad esclusione degli Ellittici, è sempre data dalle seguenti relazioni:

$$A_L = N \text{ 6dB/ottava}$$

$$A_L = N \text{ 20dB/decade}$$

dove A_L è la pendenza di attenuazione in dB, N è l'ordine del filtro, dove per **Ottava** si intende un intervallo di frequenze con rapporto pari a 2 e per



→ **Fig. 4**
Curve di risposta
Bessel, Butterworth
e Chebyshev.

Decade si intende un intervallo di frequenze con rapporto di un ordine inferiore / superiore ($\times 10$). In **Fig. 4**, osserviamo che nella banda di transizione, cioè subito dopo la frequenza di taglio, ogni filtro ha un comportamento diverso come specificato nella tabella **Tabella 1**.

Per meglio quantificare la differenza tra le diverse risposte il grafico che segue mostra l'andamento della curva di attenuazione per filtri di 5° ordine con frequenza di taglio di 50 kHz rispettivamente per il tipo Butterworth, Chebyshev e Bessel con ripple di 3 dB. Il fattore che determina l'ammontare della pendenza di attenuazione, dipende dal numero di componenti utilizzati. La differenza tra le risposte viene ottenuta scegliendo opportunamente il fattore di smorzamento di ogni sezione. Questo parametro (indicato con "A") esprime la tendenza del circuito a creare oscillazioni in risposta a segnali impulsivi. Con diversa interpretazione matematica in molti casi invece del fattore "A" viene usato il fattore di qualità del filtro (indicato con "Q") che esprime il medesimo concetto ma con scale opposte: si ottiene una elevata tendenza alle oscillazioni quando Q è maggiore di 1. Nella **Tabella 2** sono riportati i valori di attenuazione/guadagno in tensione, rispetto ai dB. Come si può notare, poiché la scala in dB risulta logaritmica base 20 (Log_{20}), per ogni decade si ha un aumento di un ordine di grandezza ($\times 10$). Troveremo quindi rispettivamente: 1 a 0dB; 10 a 20dB; 100 a 40 dB; 1.000 a 60dB e così via. Abbiamo evidenziato anche l'attenuazione a 3dB che è 1,41, perché poi ci tornerà utile nelle prossime considerazioni.

FILTRI LC PASSA BASSO

Come anticipato, ci occuperemo ora dei soli filtri passa basso, che sono quelli che maggiormente ci interessano e sui quali abbiamo elaborato il nostro progetto. Innanzitutto esistono filtri in configurazione a L, a T e a π (pi greco) collegati in serie o parallelo e vanno dall'ordine 2 (attenuazione 12 dB/ottava) a ordine n . Nel nostro progetto potremo realizzare filtri di ordine 2, 3 e 4, sia del tipo a T che a π . Utilizzando più filtri in serie si potranno realizzare Filtri di ordine n volte superiore. Nella **Fig. 5** e nella **Fig. 6** sono rappresentati gli schemi base dei filtri di 2° ordine realizzabili sulle basette più piccole (LP1 e LP3) che in seguito descriveremo. Come potrete notare lo stesso filtro se realizzato con la tipologia a pi greco può risultare bidirezionale e che essenzialmente è quello che abbiamo voluto realizzare, ma nulla vieta di realizzare anche il tipo a L. In **Fig. 7** troviamo lo schema di un filtro

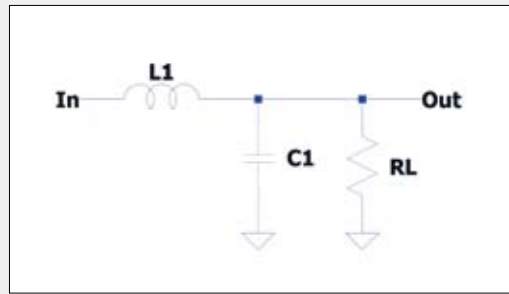


Fig. 5
Filtro PB LC di II° ordine.

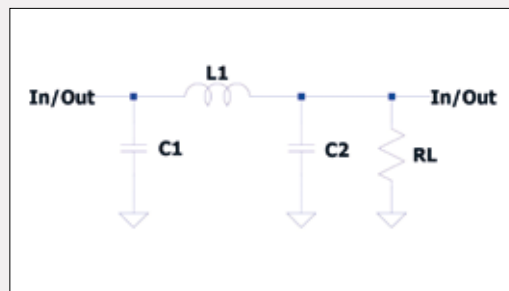


Fig. 6
Filtro PB LC di II° ordine a pi greco.

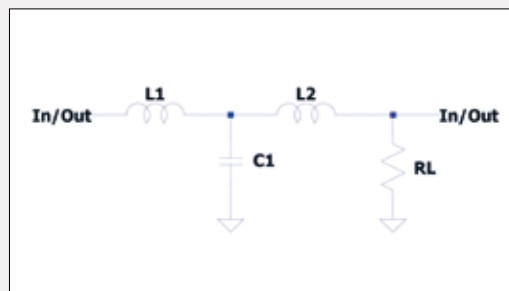


Fig. 7
Filtro PB LC a T di III° ordine.

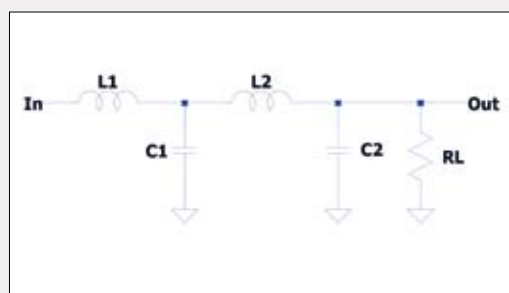


Fig. 8
Filtro PB LC di IV° ordine.

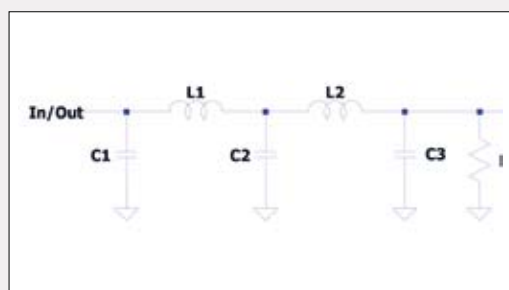


Fig. 9
Filtro PB LC di IV° ordine a pi greco.

Tabella 2
Valori di
attenuazione dei
filtri.

DECIBEL (DB)	GUADAGNO O ATTENUAZIONE IN TENSIONE	DECIBEL (DB)	GUADAGNO O ATTENUAZIONE IN TENSIONE	DECIBEL (DB)	GUADAGNO O ATTENUAZIONE IN TENSIONE
0	1	14	5,01	60	1.000
1	1,12	15	5,62	65	1.778,27
2	1,26	16	6,30	70	3.162,27
3	1,41	17	7,08	75	5.623,41
4	1,58	18	7,94	80	10.000
5	1,78	19	8,91	85	17.782,79
6	1,99	20	10	90	31.622,77
7	2,24	25	17,78	95	56.234,13
8	2,51	30	31,62	100	100.000,00
9	2,82	35	56,23	105	177.827,94
10	3,16	40	100	110	316.227,76
11	3,55	45	177,82	115	562.341,32
12	3,98	50	316,22	120	1.000.000
13	4,46	55	562,34		

a T di III° ordine, mentre nelle **Fig. 8** e **Fig. 9** sono rappresentate le due configurazioni di filtro (LL e π) relativamente ad un filtro di IV° ordine. Queste tre tipologie sono tutte realizzabile sulla basetta più grande (LP2), dove ovviamente c'è lo spazio per accogliere le due bobine L1 e L2.

PERCHÉ LC?

La stessa tipologia di filtro potrebbe essere realizzata utilizzando dei resistori al posto delle induttanze (*Filtro RC*), oppure delle resistenze al posto dei condensatori (*Filtro LR*) **Fig. 10** e il filtro risulterebbe molto più stabile e facile da calcolare, però come si può notare nel tipo LR è presente una resistenza in parallelo al carico (***RLoad***) che andrebbe a influenzare la corrente di uscita e anche la Fth. Nel tipo RC, la resistenza in serie (R1) introdurrebbe cadute di tensione importanti. Questa tipologia di filtro risulta tuttavia utilizzabile in talune applicazioni per piccole correnti di carico, essendo meno ingombrante e più economico.

Il filtro LC sopporta bene le correnti elevate e risulta stabile, benché sia necessario porre attenzione al fatto, come abbiamo visto prima, che potrebbe entrare in risonanza per via delle capacità parassite del circuito e in determinate condizioni dei picchi di frequenza. La **Fig. 11**, a titolo puramente esplicativo, rappresenta le due tipologie di circuiti

oscillanti a componenti passivi RCL tipo serie e parallelo. Fatte queste dovute premesse e appreso vantaggi e svantaggi delle varie tipologie, possiamo passare ai parametri di progetto.

PARAMETRI DI PROGETTO

Come abbiamo visto, i filtri, spesso richiedono complessi calcoli matematici, ma poiché per la nostra applicazione, non sono richieste elevate precisioni, ci accontenteremo di una buona approssimazione, stabilendo quali siano i parametri fondamentali da rispettare.

- **Frequenza di taglio.** Vorremmo che possibilmente fosse un ordine di grandezza (1/10) inferiore rispetto alla frequenza di lavoro dello switching, la quale normalmente è compresa tra 40 kHz fino a 1 MHz. Perciò sceglieremo valori possibilmente intorno a 4 kHz, ma se anche arrivassimo a 8÷10 kHz, andrà comunque bene.
- **Impedenza di uscita.** La vogliamo la più bassa possibile, possibilmente inferiore a 500 mΩ.
- **Dimensione dei componenti.** È una componente importante per non realizzare circuiti "mastodontici" che vanificherebbero la scelta per l'utilizzo di piccoli moduli di alimentazione.
- **Corrente di passaggio.** Gli induttori vanno scelti in funzione della corrente che le attraverserà e che può raggiungere facilmente i 2÷3 A o più.

- **Condensatori.** Abbiamo evitato condensatori elettrolitici sia per la polarizzazione, sia per le dimensioni e abbiamo optato per condensatori ceramici, che vanno però scelti nel range di classe tra X5R a X7R. La tensione dei condensatori dovrà anch'essa essere scelta in funzione della tensione di uscita dell'alimentatore, considerando almeno un 20% in più. Valori di 35Vf vanno normalmente bene un po' per tutto fino a 24 V di alimentazione. Oltre i 24V e fino a 36÷40 V, dovremo scegliere valori di 50Vf, anche se bisognerà forzatamente scendere sotto i 22 µF causa reperibilità e costo.

FORMULE DI CALCOLO

Anche se non andremo a toccare in modo dettagliato tutti gli aspetti intrinseci dei parametri di calcolo, vedremo qui di seguito alcune formule che risultano fondamentali per la scelta dei componenti.

$$f_{th} = \frac{1}{K \cdot \pi \cdot \sqrt{LC}} \quad \text{e} \quad R_L = \sqrt{\frac{L}{C}}$$

da cui si ricavano:

$$L = \frac{R_L}{K \cdot \pi \cdot f_{th}} \quad \text{e} \quad C = \frac{1}{K \cdot \pi \cdot f_{th} \cdot R_L}$$

Oppure:

$$L = \left(\frac{1}{K \cdot \pi \cdot f_{th}} \right)^2 / C \quad \text{e} \quad C = \left(\frac{1}{K \cdot \pi \cdot f_{th}} \right)^2 / L$$

Dove:

Fth = Frequenza di taglio in **MHz** (megahertz);

L = valore induttanza in **µH** (microhenry);

C = valore Condensatori in **µF** (microfarad);

RL = valore dell'induttanza (carico) in **Ω** (ohm);

K = costante.

Il valore della costante K, è funzione del tipo di ordine del filtro. Nella **Tabella 3** sono elencati i valori da moltiplicare sia per i condensatori che per le induttanze.

Per semplificarci la vita adotteremo un valore

medio (**Kpg**) che abbiamo valutato sperimentalmente e che riporta una buona approssimazione e semplificazione di calcolo.

Fig. 10
Filtro PB LR e filtro PB RC.

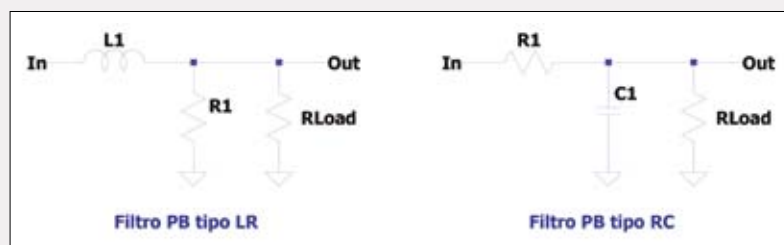


Fig. 11
Circuito risonante RLC serie e parallelo.

IL PROGETTO

Seguendo le linee guida che ci siamo dati, cioè avere come requisito fondamentale la bidirezionalità dei filtri, ci occuperemo solo dei seguenti circuiti:

- Filtro di II° ordine a π (schema di **Fig. 6**) **Fth** 3,86 kHz; RL 500 mΩ; attenuazione 12 dB/ottava; 40 dB/decade;
- Filtro di III° ordine a T (schema di **Fig. 7**) **Fth** 9,3 kHz; RL 830 mΩ; attenuazione >18 dB/ottava; >60 dB/decade;
- Filtro di IV° ordine $\pi \pi$ (schema di **Fig. 9**) **Fth** 10 kHz; RL 600 mΩ; attenuazione >24 dB/ottava; >80 dB/decade.

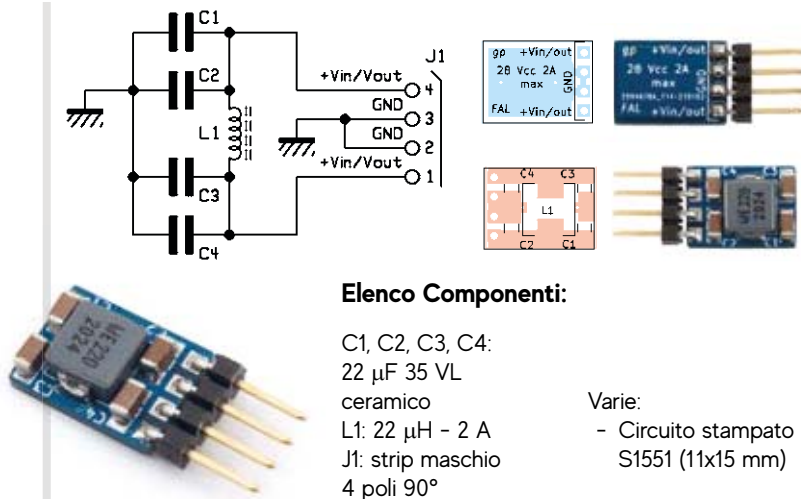
Per utilizzare le costanti (**Kpg**) indicate si prenda in considerazione quanto segue:

- per **L**, vale la somma di tutte le induttanze (µH);
- per **C**, vale la somma di tutte le capacità (µF);
- **Fth** indicato a -3dB;
- impedenza **Z** calcolata a 0,01dB.

Il filtro di II° ordine, è più che in grado di sopperire a tutte le esigenze per le quali è stato elaborato, ovvero la soppressione dei residui di alternata

ORDINE	K _{C1}	K _{L1}	K _{L2}	K _{C2}	Kpg -3dB
II	$\sqrt{2} = (1,4142)$	$\sqrt{2} = (1,4142)$			1,65
III	2	1	1		0,94
IV	1,8478	0,7654	1,8478	0,7654	0,64

Tabella 3
Valori della costante K.



(ripple) negli alimentatori CC, poiché riesce ad attenuare di ben 40 dB/decade, che come da **Tabella 2** vale 100 volte in tensione, il che significa che un ripple di 1V diventa da 10 mV. Ma già che c'eravamo, abbiamo voluto realizzare anche un progetto per i filtri di III° e di IV° ordine più che altro a scopo didattico, perché questa basetta risulta molto versatile e si possono realizzare sia filtri a T a tre elementi che a doppio π e sostituendo le induttanze con delle resistenze si possono realizzare filtri di tipo R/C. Per i **filtri RC**, la formula di calcolo per la Frequenza di taglio, è:

$$f_{th} = \frac{1}{2\pi RC}$$

Se ad esempio volessimo realizzare un filtro a 30 kHz per effettuare misurazioni sugli amplificatori audio, per il filtro a T (tre elementi) potremmo utilizzare due resistenze da 100 Ω e un condensatore da 47 nF, dove (sempre a -3dB):

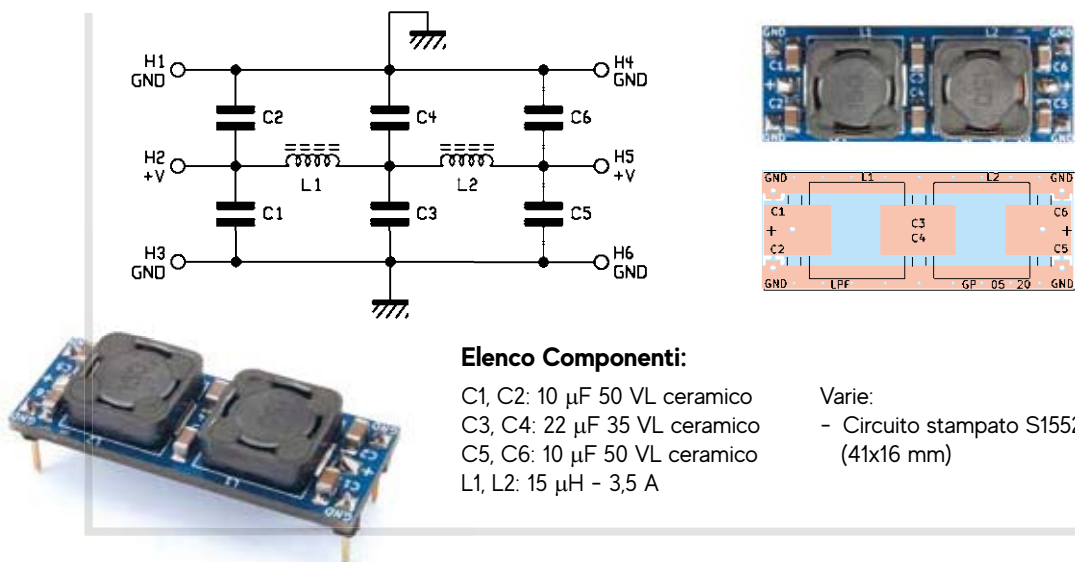
$$f_{th} = 1/(6.28 * 100 * 0.047) = 0.034 \text{ Mhz} = 34 \text{ kHz}$$

Qui apriamo una parentesi, evidenziando che tutti i circuiti sono stati verificati con il programma **LTSpice** che Analog Devices, mette a disposizione e scaricabile gratuitamente alla pagina www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html.

SCHEMA ELETTRICO

Andiamo ora a vedere alcuni esempi pratici di filtro. Gli schemi sono molto semplici e rispecchiano quanto ampiamente descritto sopra. Lo schema **LP1** rappresenta la struttura del filtro di II° ordine, realizzato in versione miniatura (stesse dimensioni di un transistor di potenza in case TO-220), mentre lo schema **LP3** rappresenta lo stesso filtro in versione più "robusta". In **LP2**, troviamo lo schema di quello di IV° ordine. Se da questo filtro eliminiamo i condensatori: C1, C2, C5 e C6 e montiamo soltanto C3 e C4, o anche solo uno dei due, otterremo il filtro di III° ordine con le caratteristiche anzi descritte nel paragrafo "PROGETTO". Sicuramente qualcuno si chiederà perché i componenti risultino posizionati su due linee opposte anziché tradizionalmente lineari; ebbene, è lo stesso, ma

modulo LP2



abbiamo voluto evidenziare la costruzione simmetrica e bidirezionale che rispecchia fedelmente il circuito della basetta e che sostanzialmente garantisce uniformità di impedenza al circuito, minimizzando la possibilità di introdurre frequenze di risonanza. Inoltre la possibilità di montare 2 condensatori in parallelo per ogni linea, ci garantisce la possibilità di gestire valori di capacità doppia e che risulta un parametro importante soprattutto perché i condensatori ceramici SMD, non sono disponibili con capacità e tensioni VL molto alte e oltremodo risultano costosi.

Nel caso di bassa impedenza, la simmetria non è poi così importante, ma qualcuno potrebbe avere necessità di accordare il filtro a 50 Ω , per applicazioni HF/RF e con questo circuito lo si può realizzare facilmente. Prendendo ad esempio il circuito del filtro di **LP1** (filtro di II° ordine), aumentando di 2 ordini di grandezza (100x) il valore di L e riducendo sempre di 2 ordini di grandezza (:100) il valore di C, otterremo:

$$R_L = \sqrt{\frac{2200(L)}{0.88(C)}} = 50\Omega$$

mentre rimane invariata la frequenza di taglio:

$$f_{th} = \frac{1}{1.65 \cdot \pi \cdot \sqrt{2200 \cdot 0.88(L \cdot C)}} = 3.86 \text{ kHz}$$

ossia la stessa frequenza di prima ma con impedenza a 50 Ω , avendo inserito L= 2.200 μH al posto di 22 μH e C=0,88 μF al posto di 88 μF . Le tipologie di filtro del progetto rientrano in classe Butterworth e la frequenza di taglio, come sopra citato, l'abbiamo perciò sempre calcolata a **-3 dB**. Riferendoci alla **Tabella 2** si può notare che 3 dB corrispondono ad una attenuazione o guadagno di 1,41 volte in tensione; questo valore corrisponde alla radice quadrata di 2. Peraltro:

$$\frac{1}{\sqrt{2}} = 0,707$$

corrisponde al seno di un angolo di 45° ovvero ad **1/4 π rad**, che equivale anche al valore di tensione efficace di una sinusoide (**V_{pk} x 0,707**).

REALIZZAZIONE PRATICA

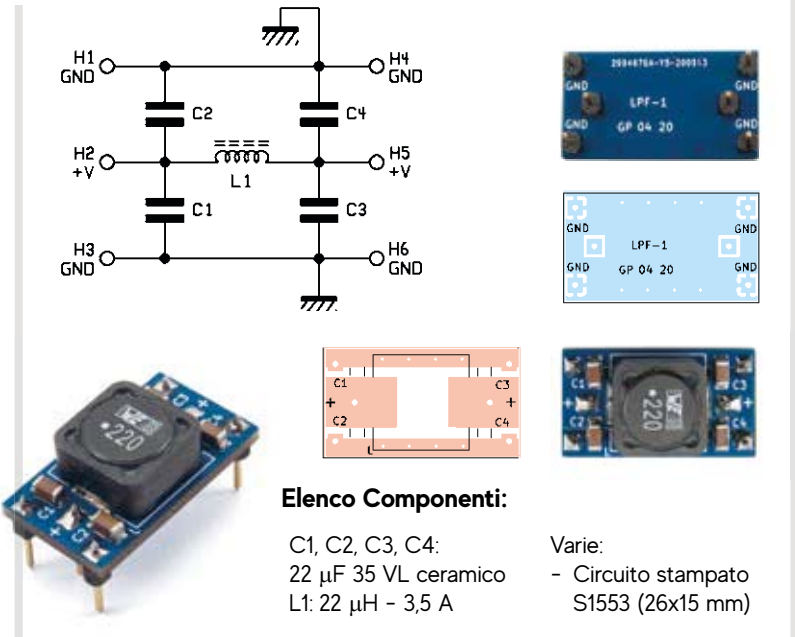
Per i nostri filtri abbiamo previsto tre formati di circuito, tutti con terminali a passo multiplo di 2,54 mm per permettere il montaggio su breadboard:

- basetta mini 11x14 mm montaggio verticale/orizzontale, per correnti fino a 2 A, filtro di II° ordine;

- basetta 14,5x26 mm montaggio orizzontale, per correnti fino a 4 A, filtro II° ordine;
- basetta 15,5x41 mm montaggio orizzontale, per correnti fino a 3 A, per filtri di III°, IV° e V° ordine.

La realizzazione di questi circuiti risulta molto semplice e i relativi PCB sono a doppia faccia, anche se il piano inferiore di massa funge solo da schermatura. Potrete scaricare come sempre i file dal nostro sito www.elettronica.in.it e richiedere i circuiti stampati alla Futura Elettronica utilizzando il servizio **PCBPRODUCTION**.

modulo LP3



Elenco Componenti:

C1, C2, C3, C4:
22 μF 35 VL ceramico
L1: 22 μH - 3,5 A

Varie:
- Circuito stampato
S1553 (26x15 mm)

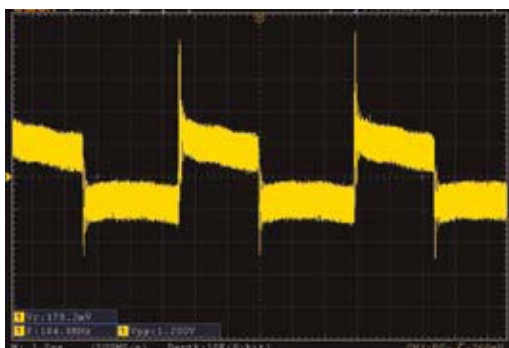
Per motivi di spazio, ma anche per evitare problemi di adattamento di impedenza, sono stati realizzati in tecnologia SMT, ma sono veramente facili da assemblare con un minimo di manualità.

Per il circuito "mini" suggeriamo di saldare per primo l'induttore e poi i condensatori; per gli altri due, consigliamo invece di montare per primi tutti i condensatori, poi con il saldatore applicare un leggero strato di stagno sulle piazzole per le induttanze, prima di deporle e saldarle, prima da un lato e poi dall'altro. I condensatori devono essere di classe X5R o meglio X7R e gli induttori vanno scelti in funzione delle correnti che li attraversano.

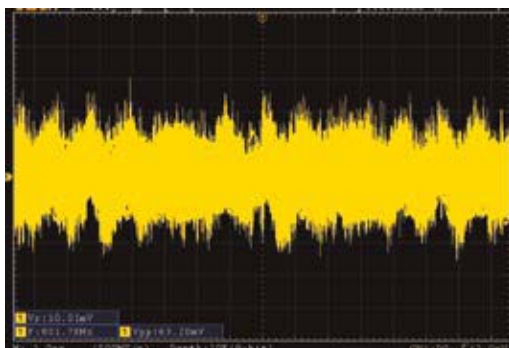
→ **Tabella 4**
Risultati delle
misure sui nostri
filtri.

TIPO DI FILTRO	VIN [V]	IIN[A]	VOUT[V]	IOUT [A]	DROP OUT (MV)	FO PEAK (KHZ)	RIPPLE (MVPP) SENZA FILTRO	RIPPLE (MVPP) CON FILTRO
tutti	12	0	24	0	0	185 ≈	6 - 8	5 - 6
π	12	2.3	23.82	1	180	184.6	1200	33
π _min.	12	2.3	23.7	1	300	184.8	1200	30
$\pi \pi$	12	2.3	23.86	1	140	184.2	1200	20

→ **Fig. 12**
Ripple senza
filtro alimentatore
YXA347.



→ **Fig. 13**
Ripple filtrato
alimentatore
YXA347.



LA PROVA AL BANCO

Per la prova, abbiamo scelto uno switching step-up già testato e descritto sulla rivista n° 248: il 6300-YXA347 Futura Elettronica. Nella **Tabella 4** riportiamo i risultati dei test fatti con i filtri qui proposti. In **Fig. 12** vedete l'oscillogramma del ripple senza il filtro, che in tabella risulta uguale per tutti (1.200 Vpp) perché non c'è filtro in uscita. La **Fig. 13** propone, invece, il ripple presente a valle del filtro. Il valore riportato nell'ultima colonna è al netto del "rumore di fondo" introdotto dal sistema, che ha ampiezza di circa 30 mVpp. Emerge innanzitutto che la Fo di 185 kHz circa, sparisce per lasciar posto a un generico rumore di fondo ed altresì che vi è differente caduta di tensione sui vari moduli, correlata all'impedenza caratteristica (teorica) dei filtri. Maggior caduta di tensione (300

mV) su $Z=500 \text{ m}\Omega$ e minor caduta (140 mV) su $Z=600 \text{ m}\Omega$. Il filtro "mini" in teoria presenta anch'esso impedenza di $500 \text{ m}\Omega$, però troviamo una caduta di tensione leggermente superiore (180 mV) perché le caratteristiche degli induttori utilizzati sono un po' diverse. Uno di questi parametri è la resistenza R_{dc} , che per l'induttore da $22 \mu\text{H}$ del primo filtro risulta $35 \text{ m}\Omega$, mentre per il filtro "mini" è circa $165 \text{ m}\Omega$.

CONCLUSIONI

Realizzando e applicando i filtri qui proposti, potrete ridurre sensibilmente il ripple di ogni converter switching, sia esso step-up (boost), step-down (buck), buck-boost o Sepic. Facciamo notare che tutti i parametri qui esaminati sono teorici, perché gli induttori variano il loro valore in funzione della temperatura e della corrente che li attraversa e di conseguenza varia la Frequenza di taglio, anche per effetto della resistenza R_{dc} . Questo va tenuto da conto se si intendesse realizzare filtri per applicazioni HF/RF, ambiti in cui, a meno di non volersi studiare complesse nozioni matematiche, conviene applicare le nostre formule e poi testare i circuiti e "accordarli" sperimentalmente. □



Cosa occorre?

I filtri presentati in questo articolo possono essere costruiti facilmente. I PCB possono essere realizzati utilizzando il nuovo servizio di stampa circuiti stampati (cod. PCBPRODUCTION) disponibile dal sito www.futurashop.it. I prezzi si intendono IVA compresa.

Nuove telecamere



Telecamera WiFi motorizzata

- Sensore d'immagine CMOS 2Mpx
- Pixel effettivi 1920x1080
- Ottica 3.6mm
- 4 LED IR
- 4 LED bianchi
- Portata LED 30 metri
- Funzione PAN/TILT con rotazione orizzontale di 320° e verticale di 90°
- Compressione video H.264/H.264+H.265X
- Microfono incorporato
- Speaker
- Funzione Motion Detection
- Supporto ONVIF
- Registrazione su SD-Card (da 8GB a 128GB)
- Potenza assorbita 6W max
- Alimentazione 12Vdc/2A (con adattatore di rete)
- Dimensioni 160x107x107 mm
- Peso 500 grammi

€ 79,00
cod. **FR769**



Prezzi IVA inclusa

Telecamera Wi-Fi con LED dual light

- Sensore d'immagine CMOS 2Mpx
- Pixel effettivi 1920x1080
- Ottica 3.6 mm
- 2 LED IR
- 4 LED bianchi
- Portata LED 30 metri
- Compressione video H.264/H.264+H.265X
- Microfono incorporato
- Speaker
- Funzione Motion Detection
- Supporto ONVIF
- Protocolli supportati RTSP, FTP, PPPOE, DHCP, DDNS, NTP, UPnP
- Registrazione su SD-Card (da 8GB a 128GB) 2.4835GHz
- Potenza assorbita 6W max
- Alimentazione 12Vdc/2A (con adattatore di rete)
- Dimensioni 230x130x115 mm
- Peso 400 grammi

€ 49,00
cod. **FR768**



Scopri la gamma completa
su www.futurashop.it



Telecamera WiFi IP Dome Fisheye 360° con IR e sensore di movimento

- Sensore d'immagine CMOS 2Mpx
- Sistema PAL/NTSC
- Pixel effettivi 1920x1080
- Portata 10 m
- Registrazione su SD-Card (max 128GB)
- Alimentazione 12Vdc/1A
- Potenza assorbita 5W max
- Dimensioni: 15,2x16,5x4 cm
- Peso 250 grammi

€ 42,00
cod. **FR766**

Telecamera WiFi da 2MPX, IR, con pannello solare e batteria ricaricabile

- Sensore d'immagine FHD 1080P CMOS 1/2.7 PS5230
- Risoluzione Video 1920x1080/15 fps
- Portata LED IR 10 metri
- Alimentazione con 2 batterie al litio tipo 18650 ricaricate tramite pannello solare integrato
- Registrazione su SD-Card (da 4GB a 64GB)
- Grado di protezione IP67

€ 119,00
cod. **FR767N**



Telecamera WiFi, PIR motion con memorizzazione dati su SD Card e batteria ricaricabile



€ 98,00
cod. **FR789**



- Sensore d'immagine CMOS 2Mpx
- Risoluzione Video 1920x1080/25 fps
- Angolo di ripresa 100° wide-angle
- Portata LED IR 6 metri
- Ottica 3.6mm
- Sensibilità 0,1 LUX
- Registrazione su SD-Card (max 64GB)
- Alimentazione 2 batterie 18650 3,7V 3000Mah (incluse)
- Grado di protezione IP66
- Peso 550g
- Dimensioni 95x53x80 mm

Per lo smart working, per chattare con amici e parenti comodamente da casa!

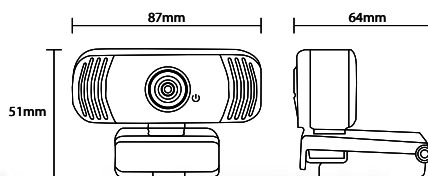
Prezzo IVA inclusa.



€24,00
Cod. FR795

WEBCAM FULL HD 1080P

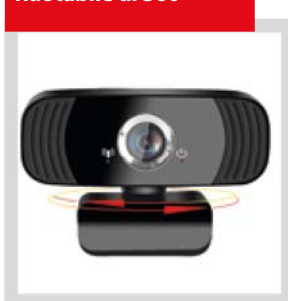
Webcam Full HD 1080P particolarmente indicata per seguire lezioni online, effettuare videoconferenze, videochat, videosorveglianza, registrazioni video, ecc. Dispone di lenti prive di distorsione, microfono omnidirezionale con riduzione del rumore, regolazione manuale della messa a fuoco, angolo di ripresa: 60° (verticale) – 120° (orizzontale), rotazione: 360°, LED di stato per alimentazione e attivazione webcam. Completamente Plug & Play (non richiede l'installazione di nessun driver), può funzionare con Windows 10/8/7/ Vista, Mac OS X, Linux, ChromeOS, Ubuntu e Android 4.0 o versioni successive. Basta collegarla alla porta USB del PC, attendere che il sistema operativo la rilevi e poi utilizzarla con Skype, Zoom, Facebook, ecc. Grazie alla sua base multifunzionale è possibile posizionarla e utilizzarla su PC fissi, PC portatili, smart TV, appoggiarla sulla scrivania oppure montarla su treppiede per macchine fotografiche.



- ✓ **Processore:** 3518EV300
- ✓ **Sensore immagine:** CMOS a colori
- ✓ **Risoluzione:** 1920 x 1080
- ✓ **Video stream:**
 - H.264: 1920 x 1080 / 1920 x 720 / 640 x 360
 - MJPEG: 1920 x 1080 / 1920 x 720 / 640 x 360
 - YUV: 640 x 360
- ✓ **Video frame rate (fps):** 30 fps (1280 x 720 / 1920 x 1080)
- ✓ **Uscita Video:** USB Video streaming
- ✓ **Controllo Immagine:**
 - messa a fuoco manuale
 - bilanciamento del bianco automatico
 - controllo automatico dell'esposizione (AEC, Automatic Exposure Control)
- ✓ **Otturatore elettronico:** automatico
- ✓ **Microfono:** omnidirezionale
- ✓ **Rapporto segnale/rumore:** > 50dB
- ✓ **Angolo di ripresa:** 60° (verticale) 120° (orizzontale)
- ✓ **Rotazione:** 360°
- ✓ **Sistemi operativi supportati:** Windows 10/8/7/Vista, Mac OS X, Linux, ChromeOS, Ubuntu e Android 4.0 o versioni successive
- ✓ **Alimentazione:** 5VDC / 160mA (tramite porta USB)
- ✓ **Temperatura di lavoro:** da -40°C a +70°C
- ✓ **Materiale:** ABS
- ✓ **Dimensioni (mm):**
 - Webcam: 87 x 64 x 51mm
 - lunghezza del cavo USB: 1,5m
- ✓ **Peso:** 180g



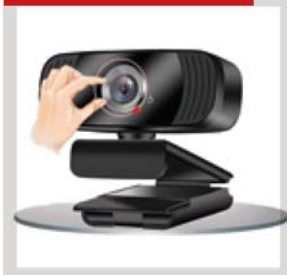
Ruotabile di 360°



Vari metodi di posizionamento



Supporta la messa a fuoco manuale



ANCHE L'ORTO TWITTA



Volete tenere aggiornati i vostri contatti Twitter sullo stato del vostro orto o giardino? Ecco il progetto che fa per voi.

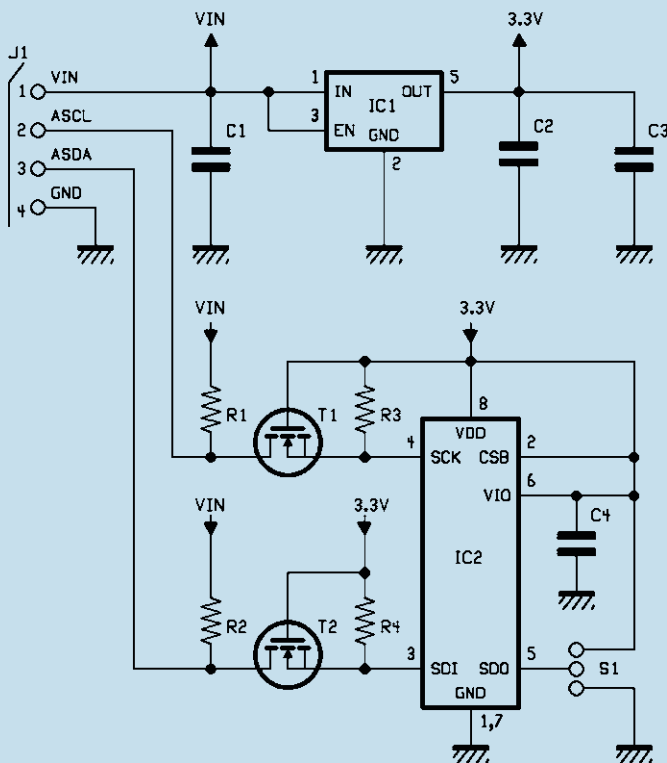
dell'Ing. **MIRKO UGOLINI**

C

omplice il periodo di clausura forzato, la riscoperta del suo valore terapeutico, e molto probabilmente anche la sua valenza economica nel salvaguardare le sempre più precarie economie familiari, ma un piccolo orticello anche sul balcone di casa sta diventando la valvola di sfogo di molti italiani che, dismesso

il lavoro ufficiale, si armano di zappa e rastrello e chini sulla terra si cimentano nel coltivare ortaggi vari. Al momento di strappare quel cespo di insalata si fa sempre un po' fatica ... perché ormai lo considera come uno di casa.

Visto il diffondersi di questa pratica ci è sembrato giusto, in linea coi tempi e le tendenze che vedono la condivisione quotidiana delle attività sui Social, dare al nostro sudato lembo di terra la possibilità di promuoversi su Twitter con dei cinguettii



che comunichino il progresso della crescita dei nostri ortaggi e le condizioni meteo dell'orto. Faremo questo con l'applicazione descritta in queste pagine, basata sul popolare modulo ESP32-CAM che utilizzerà la connettività WiFi e provvederà al monitoraggio di alcuni parametri interfacciandosi ad appositi sensori.

COS'È ESP32-CAM

L'ESP32-CAM Development Board della A.I. Thinker o equivalente è una piccola scheda con a bordo un SoC (System On Chip) ESP32 equipaggiata da una videocamera da 2M Pixel (OV2640) ad ottica fissa, uno slot per una memoria miniSD fino a 4Gbyte e una serie di linee GPIO per poter interfacciarsi al mondo esterno.

Abbiamo scelto questa piattaforma sia per l'ampia diffusione e il basso costo, sia per la possibilità di gestirla all'interno dell'IDE Arduino come hardware di terze parti, grazie al rilascio da parte della Espressif di specifiche librerie.

Per farvi comprendere come l'ESP32-CAM verrà impiegata proponiamo, nella **Fig. 1**, la pin-out. Come avrete notato, non è presente la porta USB (per ragioni di economia e semplicità) quindi per poter caricare il nostro sketch sul modulo dovremo dotarci di un convertitore USB TTL che permetta

di ottenere una tensione di alimentazione di 5V ed abbia la possibilità di selezionare il livello delle tensioni dei segnali RX e TX a 3,3 V.

Ricordiamo che il modulo SOC ESP32 non è 5V tolerant quindi i segnali dei sensori che andremo a collegare al pin GPIO devono essere tassativamente tutti nei limiti dei 3,3V.

Nella scheda sono presenti tre regolatori lineari di cui un **AMS1117-3.3** si occupa di convertire i 5V provenienti dalla alimentazione esterna ai 3,3V necessari per il modulo ESP32. A valle del regolatore è presente un MOS comandato dal segnale CAM_POWER il quale controlla (tramite la libreria Espressif) l'applicazione dei 3,3V ad altri due regolatori, uno da 2,8V ed uno da 1,2V usati entrambi per alimentare la videocamera.

Per il funzionamento della miniSD sono necessari diversi segnali, alcuni dei quali sono esposti sul connettore del modulo ESP32-CAM, ma dobbiamo essere consapevoli che se vogliamo utilizzare la miniSD dobbiamo rinunciare ad utilizzare questi PIN come GPIO standard.

Nella scheda sono presenti due LED: il primo (indicato come LED) è collegato al pin GPIO33, quindi impegna un I/O che non è neppure esposto esternamente. L'altro LED presente sulla scheda e posizionato frontalmente è un LED ad alta luminosità indicato come LED_FLASH ed è collegato al GPIO4 e controllato tramite un transistor NPN (Q1). Il GPIO33 tuttavia è condiviso (HS2_DATA1) con la scheda miniSD e quindi se viene impiegata la scheda miniSD il LED lo vedrete lampeggiare mentre la SD è in attività.

A bordo del modulo ESP32-CAM è presente un piccolo pulsante di reset del ESP32S collegato a pin E32_RST. Purtroppo i progettisti sono stati poco lungimiranti e non hanno previsto di portare esternamente questo PIN e lo hanno posizionato nel lato inferiore della schedina rendendolo difficilmente accessibile quando il modulo è in sede. I due pin U0TXD (TX) e U0RXD (RX) che ci permettono di caricare il nostro codice sul modulo ESP32-CAM sono esposti sul connettore esterno. Sul connettore del modulo ESP32-CAM è presente anche una seconda seriale U2RXD (RX) utilizzabile nel caso in cui serva ricevere dati tramite seriale. Il modulo comprende un'antenna WiFi su PCB. Nel caso in cui il modulo si trovi molto distante dal router o access-point, c'è la possibilità di collegare una antenna esterna tramite un micro connettore IPEX (U.FL) che si trova sul lato inferiore del PCB, in prossimità dell'antenna integrata. Però per attivare l'antenna esterna è necessario lo spostamento

di un ponticello realizzato con un resistore SMD a valore 0 ohm che deve essere spostato dal collegamento con l'antenna integrata al collegamento con il connettore IPEX.

Vi ricordiamo che anche se potrebbe apparire una buona idea bypassare il regolatore AMS1117 ed alimentare il tutto tramite una tensione esterna di 3,3V, è preferibile alimentare il modulo ESP32-CAM tramite l'ingresso 5V che fa, appunto, capo al Regolatore AMS1117.

Notate che il pin 4 (**P_OUT**) è marcato come 3,3V/5V perché è prevista la possibilità di farvi arrivare l'una o l'altra tensione montando una sola per volta delle due "resistenze" da 0 ohm; normalmente è montato il resistore che porta sul **P_OUT** la tensione a valle del regolatore AMS1117 (3,3V). Il segnale di reset della microcamera è derivato da una rete RC che permette alla telecamera di resettarsi nella fase di applicazione della tensione di alimentazione.

Potrebbe succedere che la porta del PC non sia in grado di alimentare in modo corretto il modulo stesso e in questo caso, una volta caricato lo sketch, è preferibile rimuovere il collegamento con il PC e alimentare il modulo tramite un alimentatore che possa erogare 5V con una corrente massima di almeno 1A (un caricatore USB per cellulare è più che sufficiente) oppure tramite un PowerBank. Quello che può accadere se utilizziamo l'alimentazione dalla porta USB è che il modulo risenta della poca capacità di alimentazione e soffra di ripetuti riavvii, blocchi o con dei malfunzionamenti.

SENSORE BME280

Bene, ciò detto andiamo a vedere il sensore con cui l'ESP32-CAM funzionerà: nello specifico si tratta del modulo multisensore Bosch BME280.

Sarà quindi possibile scattare ad intervalli predefiniti delle foto delle nostre piantine e contemporaneamente rilevare temperatura, pressione ed umidità. La foto scattata ed i dati meteo saranno quindi pubblicati su un account Twitter in tempo reale. Naturalmente si tratta di un esercizio puramente accademico che ha il solo scopo di investigare il funzionamento delle API di Twitter e permetterci di capire come farlo con un piccolo e modesto ESP32-CAM. Il BME280 è disponibile in una comoda breakout board che permette di ricavare i valori della temperatura, pressione ed umidità. Il protocollo di comunicazione è SPI oppure I²C. Noi adotteremo il modulo con il protocollo I²C anche perché i pin disponibili sul modulo ESP32-CAM sono veramente ridotti all'osso. Inoltre dovremo utilizzare una libreria particolare (da scaricare dal sito della rivista) e non quella che si usa normalmente sui moduli ESP32 standard perché la seconda va in conflitto con una variabile utilizzata nella libreria della ESP32-CAM. Il BME280 può misurare l'umidità relativa da 0 a 100% con una precisione del $\pm 3\%$, la pressione barometrica da 300 Pa a 1100 hPa con una precisione assoluta di ± 1 hPa e la temperatura da -40°C a 85°C con una precisione di $\pm 1,0^{\circ}\text{C}$. Le misurazioni della pressione sono così precise che il modulo potrebbe essere usato come altimetro con una precisione di ± 1 metro.

Il setup della libreria permette di settare i due pin per la comunicazione I²C, specificando le variabili `I2C_SDA` e `I2C_SCL`. Nel nostro caso abbiamo utilizzato il pin 16 per SDA e 14 per SCL. La lettura dei dati del sensore è veramente molto semplice. È importante utilizzare proprio questi due pin per la comunicazione I²C perché utilizzando altri pin liberi abbiamo notato dei malfunzionamenti che



Fig. 1
Pin-out
dell'ESP32-
CAM.

Fig. 2
Il modulo sensore
BME280.



interferiscono con il processo di Boot e quindi con la fase di caricamento del codice. L'indirizzo I²C predefinito del modulo BME280 è **0x76HEX** e può essere modificato facilmente in **0x77HEX** rimuovendo un sottile ponticello di pista e facendo una piccola saldatura (**Fig. 2**). Da notare anche che la Breakboard BME280 ospita un regolatore di tensione lineare LDO da 3,3 V quindi deve essere alimentato sul Pin VIN applicando 5V. Sono presenti anche dei traslatori di livello che permettono di interfacciare il modulo con dei microcontrollori che lavorino a 5V. Significa quindi che il cuore del modulo, il sensore Bosch, potrebbe lavorare direttamente con modulo ESP32 alimentandolo a 3,3V ed interfacciandosi con i pin del ESP32. Purtroppo la breakout board non espone i pin di alimentazione a 3,3V né i pin diretti del sensore, quindi dobbiamo sacrificare un po' di energia ed alimentare la Breakboard a 5V tramite

il modulo ESP32. In **Fig. 3** vediamo il collegamento del modulo **BME280** con il modulo **ESP32Cam**.

CREARE UNA APP TWITTER

Per poter pubblicare un post su Twitter in modo programmatico, ovvero da un dispositivo hardware quale il modulo ESP32-CAM non è sufficiente possedere le credenziali di un account Twitter, ma è necessario essere in possesso anche di un account sviluppatore Twitter. La sottoscrizione quale sviluppatore non comporta costi, ma deve essere richiesta compilando alcuni form sul relativo portale <https://developer.twitter.com> attendere poi l'esito della richiesta via e-mail che normalmente richiede un paio di giorni di valutazione. Se non siamo stati molto convincenti nel rispondere ai pochi ma perentori quesiti proposti (**Fig. 4**) qualche volta veniamo ricontattati via e-mail e sollecitati a rispondere ad altre domande che volgono a sincerarsi del fatto che lo sviluppatore non voglia fare un uso fraudolento o comunque non lecito delle API (Application Programming Interface) messe a disposizione dell'ambiente di sviluppo di Twitter. Una volta entrati in possesso delle credenziali, sarà necessario terminare la compilazione (**Fig. 5**) del profilo di sviluppatore inserendo un numero di cellulare valido e convalidarne la correttezza tramite la ricezione di un codice. A questo punto si rende necessario accedere alla Dashboard, creare un progetto assegnandogli

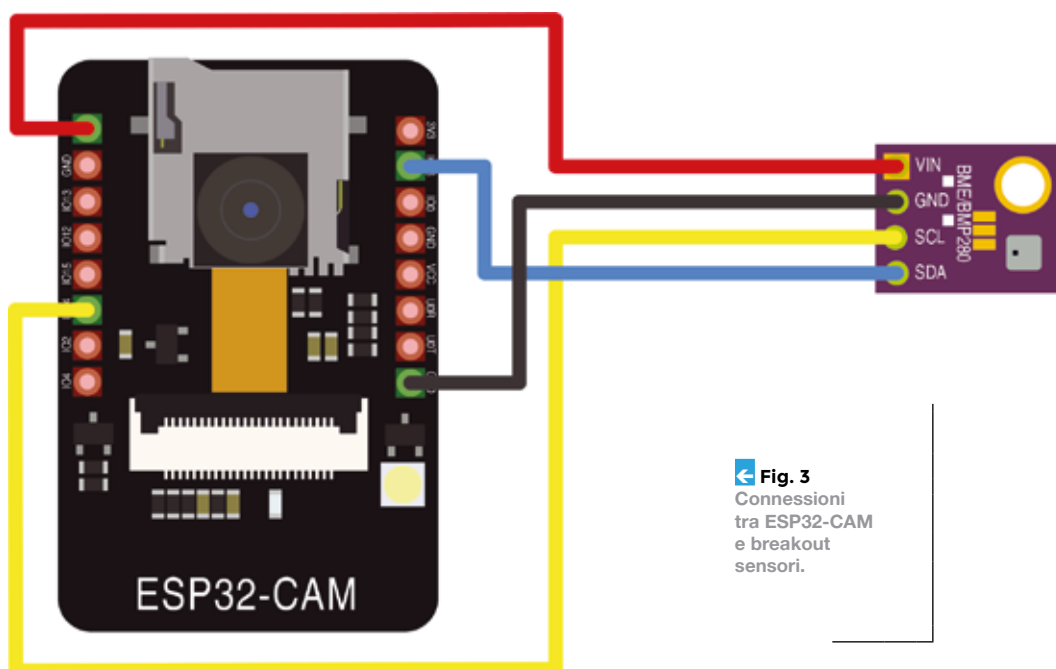


Fig. 3
Connessioni
tra ESP32-CAM
e breakout
sensori.

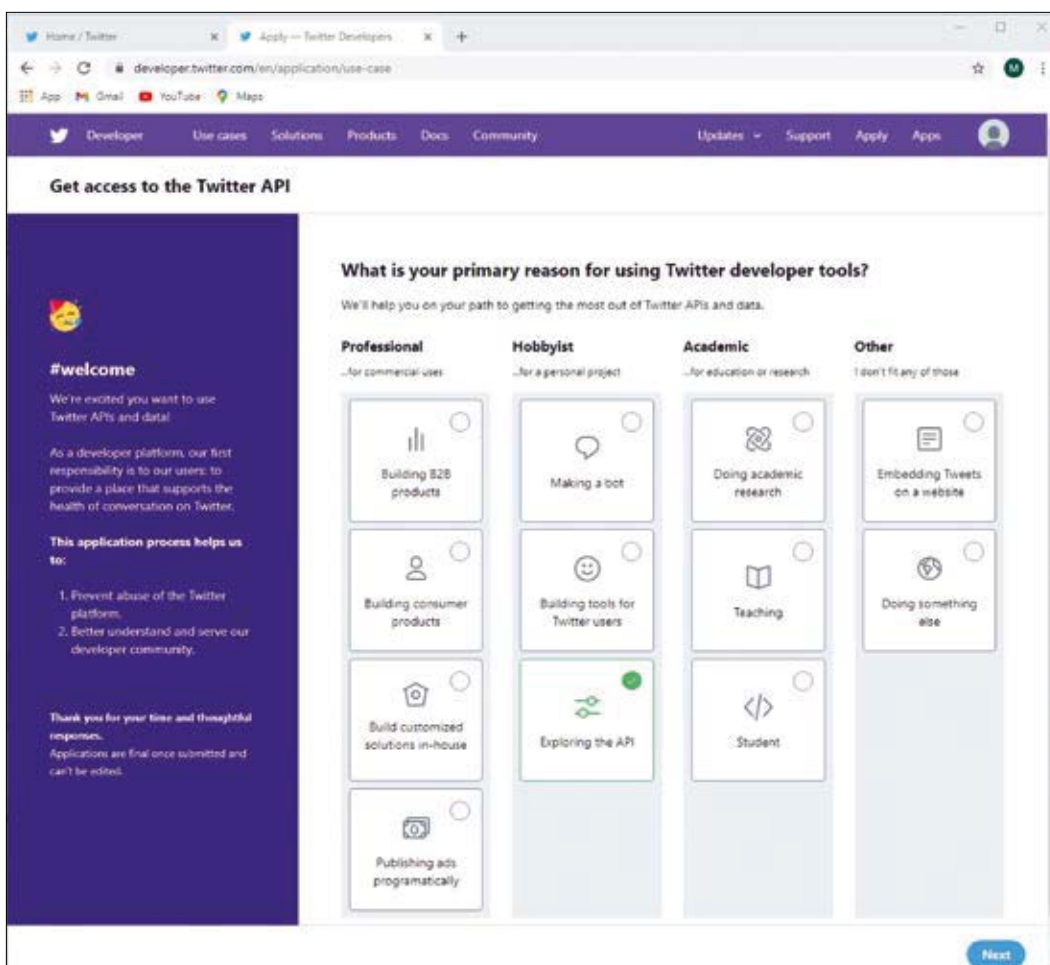


Fig. 4
Pagina di
supporto
di Twitter.

un nome, nel nostro caso abbiamo usato come nome del progetto **"Social Gardening"** (Fig. 6) ed all'interno del progetto creare una **App**. Alla **App** abbiamo assegnato il nome **"ESP32 Chirping"** (Fig. 7). È poi necessario assegnare dei permessi alla **App**, per la nostra abbiamo selezionato "Read + Write + Read and post direct messages". Nella schermata che elenca la nostra **App** è possibile selezionare il pulsante che specifica le **Keys** ed i **Token** da utilizzare per l'autenticazione (Fig. 8). Aperta questa schermata è possibile individuare due porzioni di nostro interesse: **"API Key & Secret"** che è in diretta relazione con il nostro account Developer di Twitter e la voce **"Access Token & Secret"** che è connesso alla specifica **App**. In pratica per poter postare sul nostro account Twitter tramite codice presente su ESP32 abbiamo necessità di due coppie di stringhe: API Key e API Key secret - Access token e Access token secret. È assolutamente raccomandato che queste coppie di chiavi non siano divulgate, alla pari delle vostre

credenziali Twitter standard, perché permettono di impersonarvi in modo completo. Nel caso in cui per qualche motivo la sicurezza fosse compromessa, è possibile rigenerare le chiavi.

Gli endpoint, ovvero gli indirizzi che vengono utilizzati per effettuare le chiamate verso le API di Twitter, sono ovviamente in https:

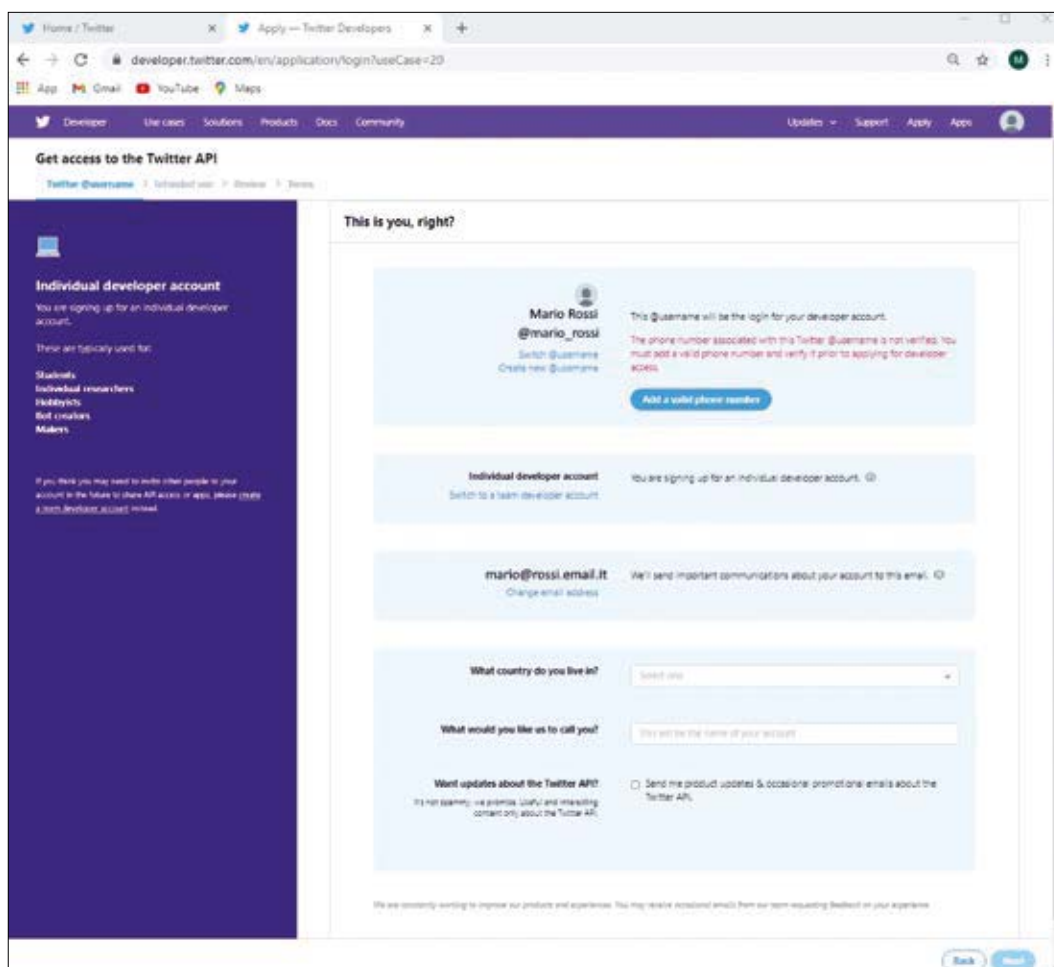
<https://api.twitter.com/1.1/statuses/update.json>
<https://upload.twitter.com/1.1/media/upload.json>

Dall'URL possiamo evincere anche che vengono usate le API nella Versione 1.1, quella attualmente consolidata. Dalla Dashboard di Twitter è possibile notare che sono in fase di definizione anche delle nuove API 2.0, ma il nostro progetto fa attualmente uso della versione 1.1.

COMUNICAZIONE HTTPS

Normalmente nell'instaurare un canale di comunicazione in HTTPS sono richieste molte risorse

Fig. 5
Creazione del
profilo sviluppatore.



se computazionali crittografiche; nel caso del nostro semplice ESP32, per poter instaurare una comunicazione sicura in HTTPS ci si accontenta di verificare se l'url al quale intendiamo collegarci sia da ritenersi attendibile, e per poter fare questo si ricorre alla verifica di validità del certificato dell'endpoint basandosi sulla catena di certificazione e quindi andando a ritroso verificando che il certificato dell'ente Emittente (CA) sia valido. Un po' come funziona il sistema notarile: un notaio emette un atto, chi viene in possesso dell'atto, per verificarne la validità controlla che il Notaio che l'ha redatto sia in effetti iscritto all'albo Notarile. Per poter fare questo abbiamo bisogno di salvare sul nostro sketch, la FingerPrint del certificato della CA che ha emesso il certificato per i siti degli endpoint di Twitter. Per estrarre il certificato utilizzeremo il browser Chrome. Apriamo quindi Chrome e digitiamo nella barra degli indirizzi l'url di uno degli endpoint di Twitter ("<https://api.twitter.com>"), il lucchetto chiuso alla sinistra dell'url ci indica che il certificato

e valido e si è quindi in comunicazione protetta con l'endpoint. Se clicchiamo sul lucchetto si apre una finestra, clicchiamo su "certificato (Valido)" poi sul tab "Percorso di certificazione" e poi sul primo elemento della catena di certificazione "**DigiCert**", poi clicchiamo sul pulsante "Visualizza Certificato", clicchiamo sul tab "Dettagli" e poi su "Copia su file..."; pulsante Avanti, selezioniamo "Codificato Base 64 X.509" ed infine pulsante Avanti, selezionando il nome del file (il nome non è importante) in cui vogliamo salvare il certificato. Dobbiamo copiare il contenuto del file in una variabile **root_ca** nel nostro sketch avendo cura di terminare in modo corretto le varie righe di cui si compone il certificato con il carattere "\n" \ come si potrà vedere dal codice. Abbiamo riportato la procedura per ottenere il certificato di Root a scopo di completezza, ma non preoccupatevi: potete saltare la procedura perché l'abbiamo fatto per voi e lo abbiamo incluso nel file **twitter_settings.h**. Come avrete notato analizzando il certificato in Chrome, la scadenza è 10/11/2031, quindi fino a



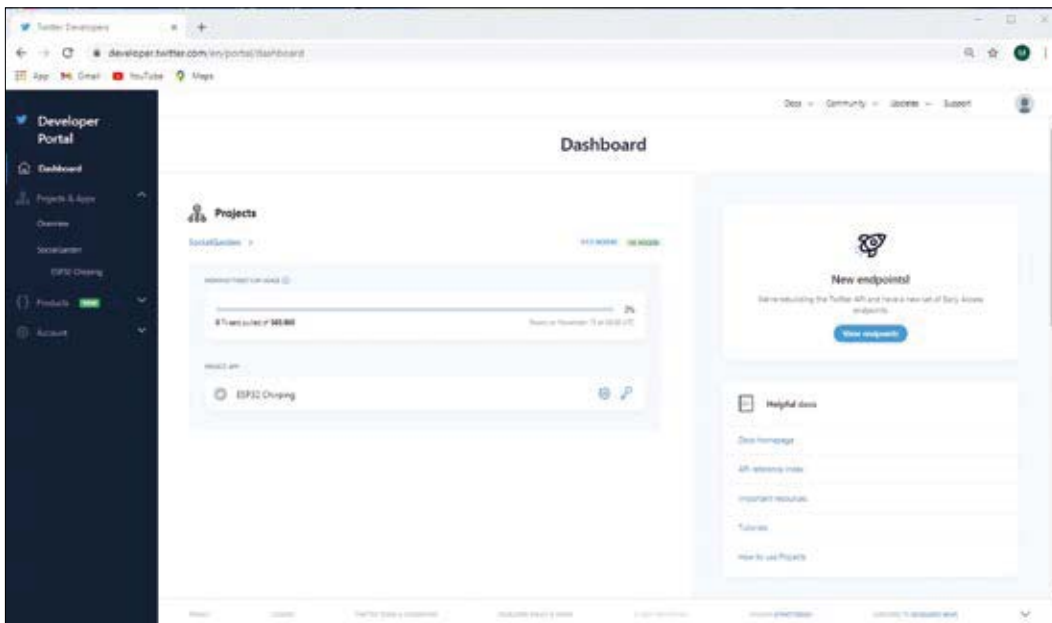


Fig. 6
La Dashboard
Twitter con il
progetto.

quella data potremo stare sicuri che la comunicazione con gli endpoint avverrà in modo corretto.

SERVER NTP (NETWORK TIME PROTOCOL)

Le API di Twitter richiedono che il dispositivo ESP32 fornisca durante la comunicazione dei

Timestamp corretti. Per ottemperare a questo requisito è necessario che il Modulo ESP32 utilizzi il servizio fornito da un server NTP. Inoltre abbiamo necessità di attivare l'invio del Twitt ad una particolare ora e quindi dobbiamo sincronizzare l'RTC (Real Time Clock) interno al modulo ESP32 con il

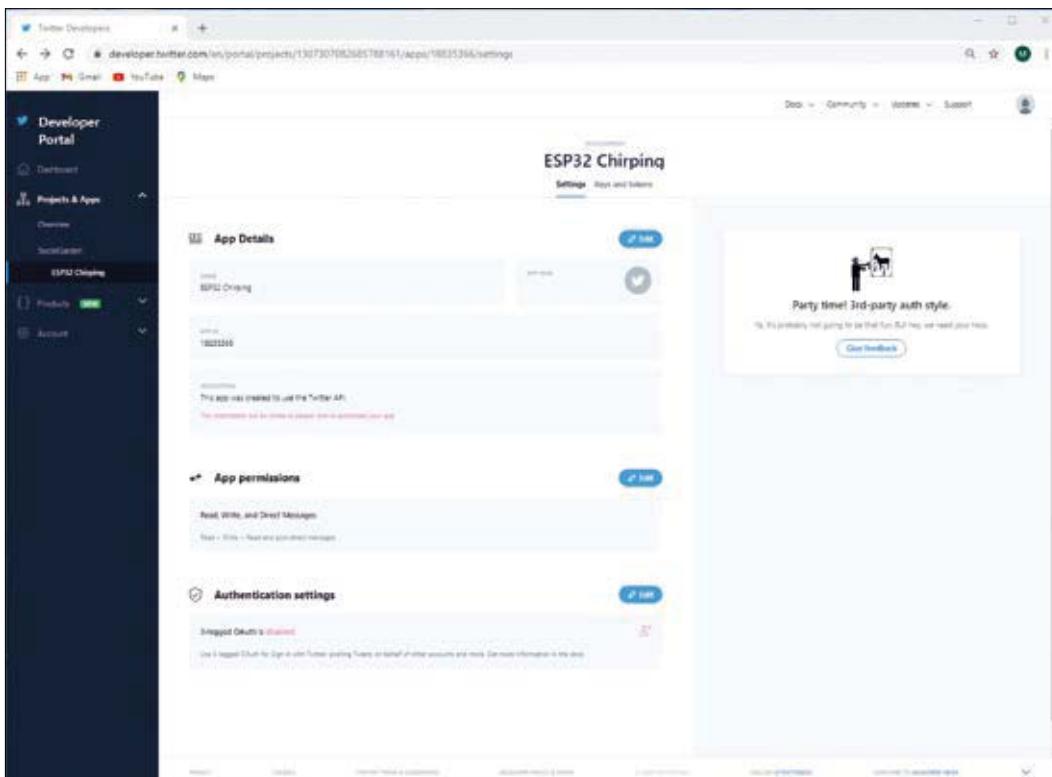
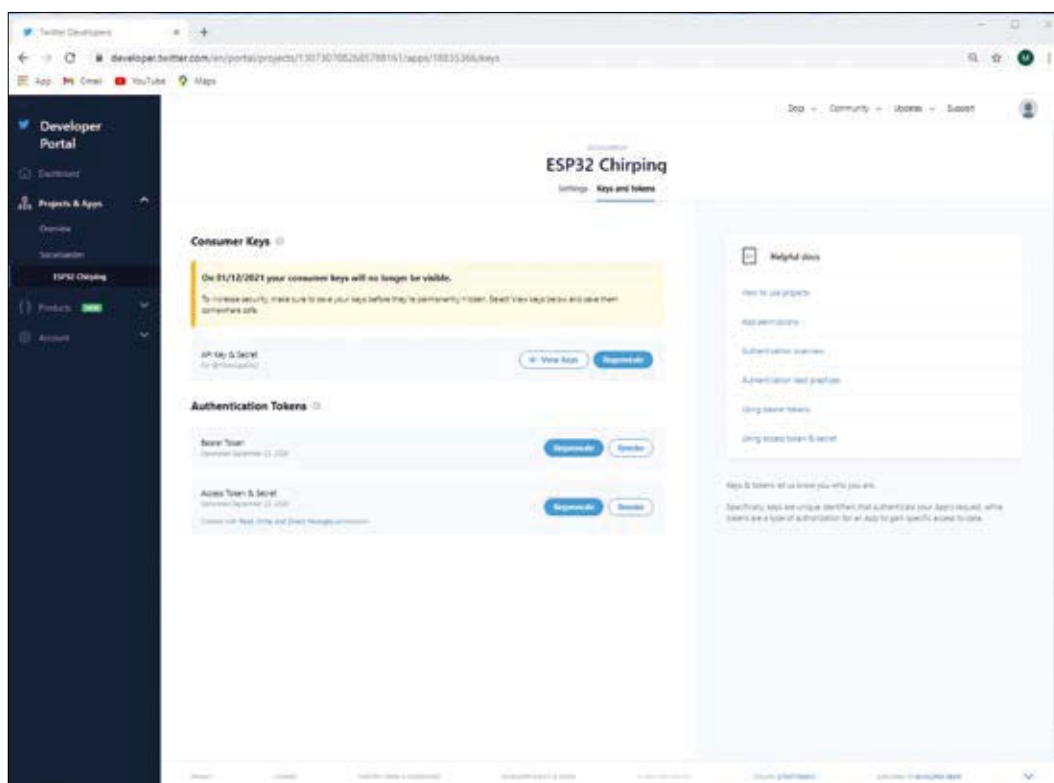


Fig. 7
Assegnazione
del nome
all'App.

Fig. 8
Schermata di
definizione dei
parametri di
autenticazione.



servizio NTP.
NTP sta per Network Time Protocol. È un protocollo Internet (IP) standard per sincronizzare gli orologi del computer tramite rete internet. Il protocollo può essere utilizzato per sincronizzare un dispositivo con l'ora **UTC** (Coordinated Universal Time). L'ora UTC è uno standard mondiale, strettamente correlato al **GMT** (Greenwich Mean Time). L'UTC non varia, è lo stesso in tutto il mondo. NTP imposta l'ora sulla base di UTC, qualsiasi offset del fuso orario locale o dell'ora legale deve essere applicato dal client.
In questo modo i client possono sincronizzarsi ai server indipendentemente dalla posizione e dalle differenze di fuso orario.
Il dispositivo client come ESP32 si connette al server utilizzando il protocollo UDP (User Datagram Protocol) sulla porta 123.

- Il client trasmette quindi un pacchetto di richiesta a un server NTP.
- In risposta a questa richiesta, il server NTP invia un pacchetto timestamp.
- Un pacchetto timestamp contiene più informazioni come timestamp, accuratezza, ritardo o fuso orario UNIX.
- Un client analizza i valori di data e ora correnti e setta l'orologio interno (RTC)

Per fare questo la funzione che dovremo usare per ottenere la sincronizzazione del RTC con il servizio NTP è:

```
configTime(gmtOffset_sec, daylightOffset_sec,  
ntpServerName)
```

dove:

gmtOffset_sec

lo possiamo ricavare dalla seguente url: https://en.wikipedia.org/wiki/List_of_UTC_time_offsets
Nel nostro caso viene impostato a **3600** secondi
= 1h

daylightOffset_sec

è in pratica il parametro che definisce lo scostamento dell'ora solare. Lo possiamo settare a 0 e probabilmente lo sarà anche per gli anni a venire visto che il 2020 sembra che sia stato l'ultimo in cui La UE utilizzerà l'ora solare.

NtpServerName

Si tratta dell'URL del server che andremo ad



interrogare. Lo settiamo a “**pool.ntp.org**”, come suggerisce il prefisso, si tratta di un pool (batteria di server) che assicurano la risposta in tempi brevi e garantisce una notevole ridondanza.

Il modulo RTC integrato sul modulo ESP32 non eccelle come precisione e quindi è buona norma risincronizzarlo abbastanza regolarmente. Abbiamo fatto in modo che la sincronizzazione avvenga al primo boot del dispositivo ed ogni 5 riavvi successivi.

Una volta eseguita la sincronizzazione, per interrogare l'RTC e ricavare l'ora e la data corrente, possiamo usare una funzione **getLocalTime(&timeinfo)** dove **&timeinfo** è il puntatore alla struttura dati che “riceverà” la data e l'ora corrente.

Bene, se chiamassimo questa funzione contestualmente alla sincronizzazione con il server NTP, l'ora ritornata sarebbe corretta, ma se come abbiamo intenzione di fare non facessimo una sincronizzazione con il server NTP ad ogni boot del dispositivo, andando a recuperare l'ora e la data come visto sopra, avremmo l'amara sorpresa di trovare uno scostamento rispetto a quella effettiva.

Per scongiurare questo dobbiamo “informare” il codice della nostra TZ (Time Zone) per cui prima di chiamare la funzione **getLocalTime()** dobbiamo settare il TZ con la funzione seguente **setenv(“TZ”, “CET-1”, 1)** anche qui è possibile gestire l'ora legale, ma la sintassi della stringa è un poco più articolata. Per esempio **setenv(“TZ”, “CET-1CEST-2,M3.5.0/02:00:00,M10.5.0/03:00:00”, 1);**

Vi rimandiamo al seguente sito per una lista <https://remotemonitoringsystems.ca/time-zone-abbreviations.php>

Per la stessa motivazione addotta in riferimento al parametro **daylightOffset_sec**, anche qui non setteremo l'ora legale e quindi adotteremo solo **setenv(“TZ”, “CET-1”, 1).**

GESTIONE FOTOCAMERA

Il modulo ESP32-CAM espone esternamente diversi pin sui due Header, ma non tutti sono utilizzabili. Alcuni sono dedicati alla comunicazione con il chip PRAM, altri sono impegnati per la scheda SD, ed alcuni per gestire la CAMERA. Inoltre la gestione della Camera impone l'utilizzo di risorse anche software che potrebbero compromettere il funzionamento di altri sensori aggiuntivi. Per questo motivo abbiamo adottato la tecnica di

inizializzare la camera solo nel momento in cui si renda necessario scattare la foto ed al termine effettuare un **deinit()** per liberare le risorse software.

Tra le cose che devono essere fatte nella fase di inizializzazione della camera c'è l'esigenza di portare a livello basso il pin CAM_POWER nella fase di inizializzazione e a livello alto in quella di de inizializzazione tramite i comandi **digitalWrite(CAM_POWER, LOW); digitalWrite(CAM_POWER, HIGH);** Lo scopo del pin è quello di controllare l'applicazione dei 3.3V derivanti dal regolatore AMS1117 ai due regolatori a 2,8V e 1,2V necessari per l'alimentazione della Camera.

La foto catturata dalla camera non sarà salvata né sulla SD card (che non adotteremo) né sul Filesystem SPIFF il quale in seguito a ripetute scritture potrebbe deteriorare il funzionamento del modulo. Ricordiamo che il filesystem SPIFF è derivato dalla Flash memory e quindi ha un limite di circa 10.000 cicli di scrittura. La foto viene invece mantenuta in memoria e viene quindi inviata al server attingendo alla variabile (Buffer) in cui il modulo ESP32CAM salva i dati. Non possiamo neppure utilizzare la MicroSD perché poi non avremmo disponibili i pin GPIO che ci servono per comunicare con il sensore Bosch.

MODALITÀ LOW POWER

Il nostro scopo è quello di mantenere il più possibile il modulo ESP32CAM in modalità basso consumo (LP), risvegliarlo dopo un certo intervallo di tempo, scattare una foto, leggere i parametri dal sensore BME280, comporre un messaggio di testo, postare il tutto su Twitter e poi tornare in modalità LowPower.

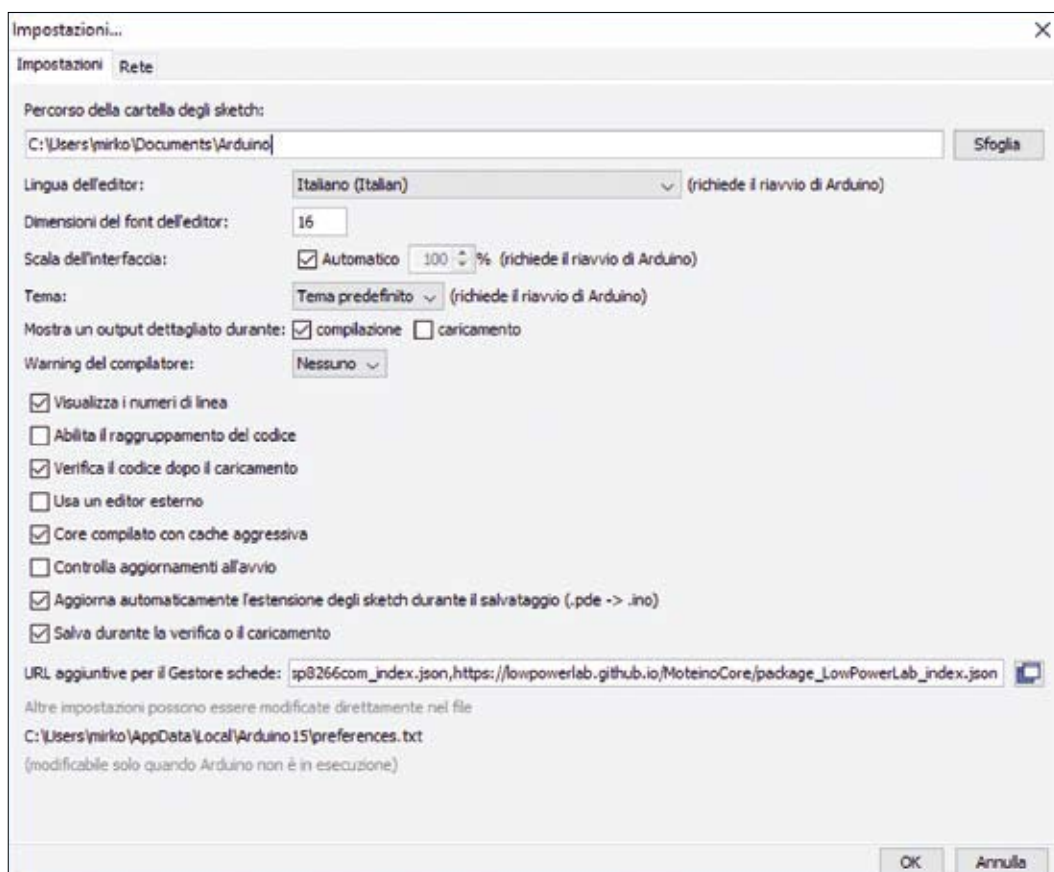
Naturalmente non ha senso inviare più di un messaggio al giorno e quindi abbiamo deciso di specificare che il risveglio debba avvenire definendo un orario in una variabile e quindi l'ideale sarebbe un risveglio ogni 24 ore in un orario specificato per esempio alle 10 del mattino.

Purtroppo non è possibile mantenere in modalità LP il modulo per un tempo così lungo e quindi quello che abbiamo fatto e definire il max intervallo di sleep, poi ad ogni risveglio andremo a controllare se l'orario corrente corrisponde a quello impostato per lo scatto della foto.

Se non coincide torneremo velocemente in modalità LP senza nemmeno inizializzare la connessione WiFi, la camera o altro. Se invece l'orario corrisponde con quello impostato, attiveremo il WiFi, leggeremo i dati del sensore, scatteremo la foto e la posteremo su Twitter per poi tornare velocemente



Fig. 9
Definizione del
percorso di ricerca
dei driver Espressif.



in modalità LP, in attesa di un altro ciclo. Inoltre, ogni 5 riavvii andremo ad interrogare il server NTP per sincronizzare l'orario del modulo interno RTC.

IL MODULO ESP32-CAM NELL'IDE ARDUINO

Come anticipato nella parte introduttiva, grazie alle librerie rilasciate da Espressif è possibile utilizzare il modulo tramite l'ambiente di sviluppo di Arduino. Come di consueto, se non avete già utilizzato dei moduli ESP32 nei vostri progetti precedenti seguite passo-passo le indicazioni seguenti per installare il Core dei moduli SoC ESP32. Consigliamo vivamente di assicurarsi di aver la versione della IDE di Arduino più recente possibile. Al momento della stesura del seguente articolo la versione più aggiornata è la 1.8.13. Apriamo quindi dal menu

file la voce File/Impostazioni. Poi individuiamo la casella di testo "URL aggiuntive per il gestore di schede" (Fig. 9) e clicchiamo sul pulsante a destra per aprire una finestra (Figura 10) nella quale poter inserire l'URL di definizione dei core SoC fornito da Espressif:

https://dl.espressif.com/dl/package_esp32_index.json

ed inseriamolo (oppure aggiungendolo ad altri eventualmente presenti) nella casella di testo e salviamo cliccando su OK.

Riavviamo quindi L'IDE di Arduino. Poi andiamo a selezionare Strumenti/Gestione schede, digitiamo "ESP32" nella casella di testo. Dopo una breve attesa dovrebbe apparire la voce visibile in Fig. 11, selezioniamo la versione più recente del pacchetto. Nel nostro caso 1.0.3 e clicchiamo su installa. Una barra di progressione apparirà in calce alla scheda informandoci sul progresso della installazione. Al termine della quale potremo riavviare L'IDE di Arduino e alla prossima apertura avremo un sottomenu con una nuova serie di schede basate sul chip SoC ESP32 installate sul nostro ambiente di sviluppo. Il modulo che dobbiamo selezionare

Fig. 10
Inserimento
dell'URL.



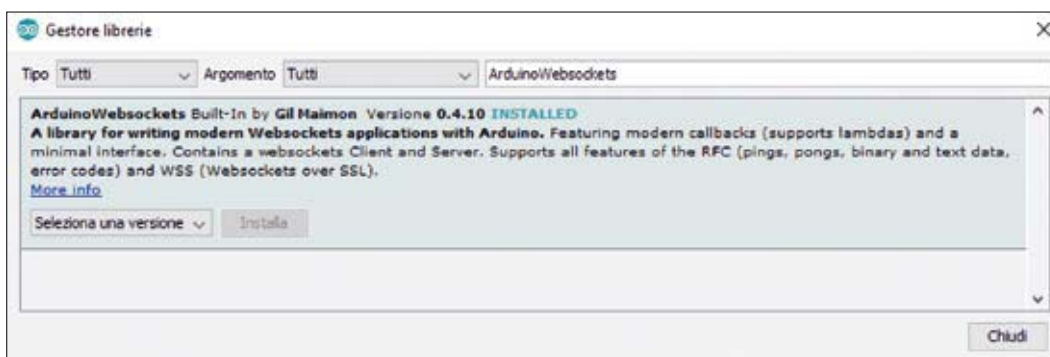


Fig. 11
Installazione
dei driver per
la ESP32-
CAM.

per il progetto corrente dovrà essere “ESP32 Dev Module”.

IL CODICE

Il codice è organizzato in quattro file: il file principale **ESP32CamMEM.ino**, il file di **ap_settings.h** che definisce i dati di accesso al nostro Router WiFi e i parametri del server NTP, il file che specifica i pin del modulo **camera_pins.h** ed il file **twitter_settings.h** che specifica i parametri di connessione con gli endpoint di Twitter ed il certificato CA degli stessi. Per il progetto di questo articolo avremo necessità di aggiungere le seguenti librerie ad Arduino:

ArduinoJson <https://github.com/bblanchon/ArduinoJson>

BME280 https://github.com/BoschSensortec/BME280_driver

TimeLib <https://github.com/PaulStoffregen/Time>

Nel caso le abbiate già presenti nel vostro ambiente, assicuratevi di aggiornarle all'ultima versione. Per inserire le librerie, utilizziamo lo strumento **Gestione Librerie** dell'IDE per **ArduinoJson** (Fig. 13) mentre per **BME280** e **TimeLib** (non essendo tuttora disponibili nel repository di Arduino) dovremo inserirle a mano nella cartella **Libraries** di Arduino. Dovremo quindi scaricare il file zippato **BME280_driver-master.zip** dal rispettivo repository GitHub, scompattarle la cartella, rimuovere il suffisso **-master** dal nome della cartella e spostare la stessa dentro la cartella che solitamente si trova in: `C:\Program Files(x86)\Arduino\libraries` ed analogamente per **TimeLib**. Inserite le librerie e riavviato L'IDE, saremo pronti per il nostro codice. Non riporteremo lo sketch completo nell'articolo, ma solo le parti salienti.

Potrete comunque trovare tutto il codice sul nostro sito; qui di seguito vediamo la parte di importazione delle librerie:

```
#include <TimeLib.h>
#include <time.h>
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include <ArduinoJson.h>
#include <base64.h>
#include <esp_camera.h>
#include <Arduino.h>
#include <BME280.h>
#include <soc/rtc_cntl_reg.h>
#include <esp_task_wdt.h>
#include "ap_settings.h"
#include "twitter_settings.h"
#include "camera_pins.h"
```

Il cuore della nostra applicazione è chiaramente la libreria **esp_camera.h** resa disponibile da Espressif che espone il funzionamento della scheda ESP32-CAM. Le altre librerie degne di nota sono **WiFiClientSecure** che viene impiegata per effettuare le chiamate alle API di Twitter in https, poi a seguire la Libreria **WiFi.h** per la gestione del WiFi, **ArduinoJson.h** per parsare i response delle chiamate API. La libreria **BME280.h** per la lettura dei dati dal sensore.

Abbiamo anche un paio di librerie del Core del modulo ESP32 Core, come **soc/rtc_cntl_reg.h** usata per disattivare brownout detector (il rilevamento delle sotto tensioni di alimentazione) e **esp_task_wdt.h** usata per realizzare una funzione per riavviare il modulo che impiega il Watch Dog, poi una libreria crittografica **base64.h** impiegata per codificare in base64 alcuni parametri durante le chiamate alle API di Twitter.

Vediamo poi nella parte terminale della stessa sezione l'inclusione dei file locali. Notare la mancanza delle parentesi angolari (< >) cosa che sta ad indicare che i file si devono trovare nella stessa cartella del file principale dello sketch.

Abbiamo poi due righe di codice molto interessanti



che definiscono due variabili particolari:

```
RTC_DATA_ATTR int bootCount = 0;  
RTC_DATA_ATTR int NTP_set = 0;
```

L'attributo **RTC_DATA_ATTR** anteposto al tipo (int) specifica che queste due variabili sono destinate a "sopravvivere" durante la fase di passaggio in modalità LowPower e successivo risveglio. In pratica **RTC_DATA_ATTR** specifica che la collocazione del valore delle due variabili deve essere posizionata nella memoria (registri) RTC che mette a disposizione fino ad 8K di spazio. Il mantenimento del valore delle variabili avviene in virtù del fatto che durante la fase di basso consumo uno dei pochi moduli che rimane alimentato è proprio l'RTC ed i suoi registri di memoria.

Naturalmente non bisogna abusarne (vista la ridotta dimensione), ma in questo caso è essenziale mantenere il numero di cicli di LowPower eseguiti ed il fatto che la chiamata al server NTP sia andata a buon fine. All'interno del codice potremo utilizzare queste variabili nel modo tradizionale. Naturalmente se resetteremo il modulo tramite il relativo pulsante oppure togliendo l'alimentazione al modulo queste variabili verranno resettate.

Dopotutto vivono in una memoria RAM.

Nel blocco successivo abbiamo la definizione del fattore di conversione da secondi a microsecondi, utilizzato nella definizione dell'intervallo di durata della modalità LP, poi abbiamo la durata massima in secondi che possiamo assegnare al modulo e l'orario al quale deve essere scattata la foto e postata su Twitter.

```
#define uS_TO_S_FACTOR 1000000ULL long SleepDuration = 7200;  
int ShootAt = 10;
```

Di seguito abbiamo la definizione dei pin dedicati alla comunicazione I²C con il sensore BME280 e la definizione del pin utilizzato per esempio per fare triggerare l'uscita del modulo dalla modalità LP:

```
#define P_BUTTON GPIO_NUM_13  
#define I2C_SDA GPIO_NUM_16  
#define I2C_SCL GPIO_NUM_14
```

Abbiamo inoltre il blocco che definisce SHA1 di porzioni di testo nella fase di invio agli endpoint delle API di Twitter utilizzando delle librerie esterne scritte in C (**Listato 1**). A seguire troviamo l'asse-

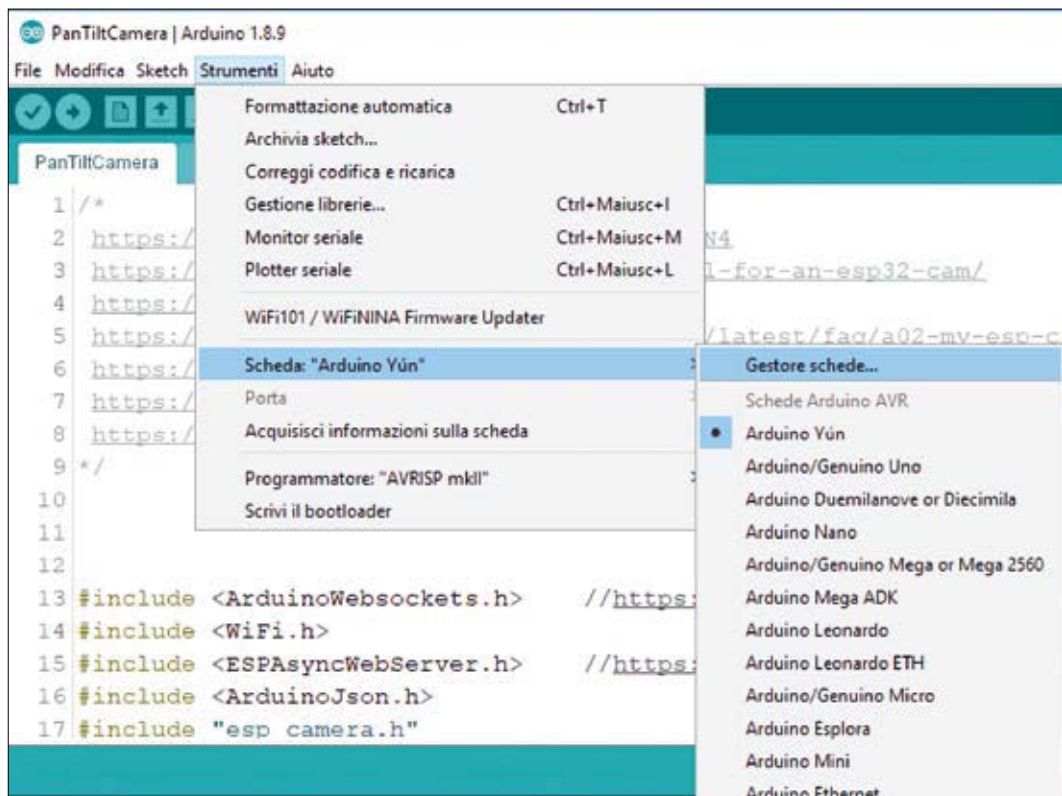


Fig. 12
Librerie Arduino
Json installate.



gnazione dello SSID e della password (che dovrete personalizzare) della nostra rete WiFi, che sono definiti nel file esterno **ap_settings.h** :

```
const char* ssid = WIFI_SSID;
const char* password = WIFI_PASSWORD;
```

Proseguendo nel codice abbiamo l'attribuzione delle variabili utilizzate per le chiamate alle API di Twitter.

```
char consumer_key[] = API_KEY;
char consumer_secret[] = API_KEY_SECRET;
char access_token[] = ACCESS_TOKEN;
char access_secret[] = ACCESS_TOKEN_SECRET;
char base_host[] = API_BASE_HOST;
char base_url[] = API_BASE_URL;
char base_uri[] = API_BASE_URI;
char message_array[141];
```

Da notare la variabile che dovrà contenere il messaggio testuale del Twitt **char message_array[141]**; che quindi come ci si poteva aspettare può contenere al massimo un messaggio di 140 caratteri. La dimensione è 141 per il carattere terminatore.

Nel blocco successivo abbiamo poi la definizione della variabile destinata a contenere il FrameBuffer **fb** della immagine ripresa della camera, la classe del sensore BME280 e la classe del client Http che useremo per postare i dati verso gli endpoint delle API di Twitter in Https.

```
camera_fb_t * fb = NULL;
Bme280 bme280;
WiFiClientSecure client;
```

Nel blocco seguente (**Listato 2**) abbiamo la sezione di Setup del codice.

Le parti degne di nota della sezione di setup sono: La definizione dei due I/O il pulsante P_BUTTON come INPUT_PULLUP e LED_BUILTIN come OUTPUT, subito sotto abbiamo la definizione della assegnazione del pulsante P_BUTTON alla routine di Wakeup dalla modalità LP sul fronte di discesa dello stato del PIN. Abbiamo poi l'incremento del contatore di risvegli bootCount che viene incrementato ad ogni risveglio del modulo. Viene poi inserita a scopo di debug, (e quindi in produzione la riga può essere commentata) la visualizzazione del motivo che ha comportato il risveglio del modulo, raggiungimento del tempo massimo di SleepMode, oppure la pressione del pulsante

Listato 1

```
extern "C" {
    typedef struct{
        uint32_t Intermediate_Hash[SHA1_SIZE / 4];
        uint32_t Length_Low;
        uint32_t Length_High;
        uint16_t Message_Block_Index;
        uint8_t Message_Block[64];
    } SHA1_CTX;
    void SHA1Init(SHA1_CTX *);
    void SHA1Update(SHA1_CTX *, const uint8_t *
msg, int len);
    void SHA1Final(uint8_t *digest, SHA1_CTX *);
}
```

Listato 2

```
Void setup () {
    bool result = false;
    Serial.begin(115200);
    pinMode(P_BUTTON, INPUT_PULLUP);
    pinMode(LED_BUILTIN, OUTPUT);
    esp_sleep_enable_ext0_wakeup(P_BUTTON, 0);
    ++bootCount;
    Serial.println("Boot number: " + String(bootCount));
    Serial.println("NTP_set: " + String(NTP_set));
    print_wakeup_reason();
    if (NTP_set == 0 || bootCount%5==0) {
        wifi_connect();
        Sync_With_NTP();
    }
}
```

Listato 3

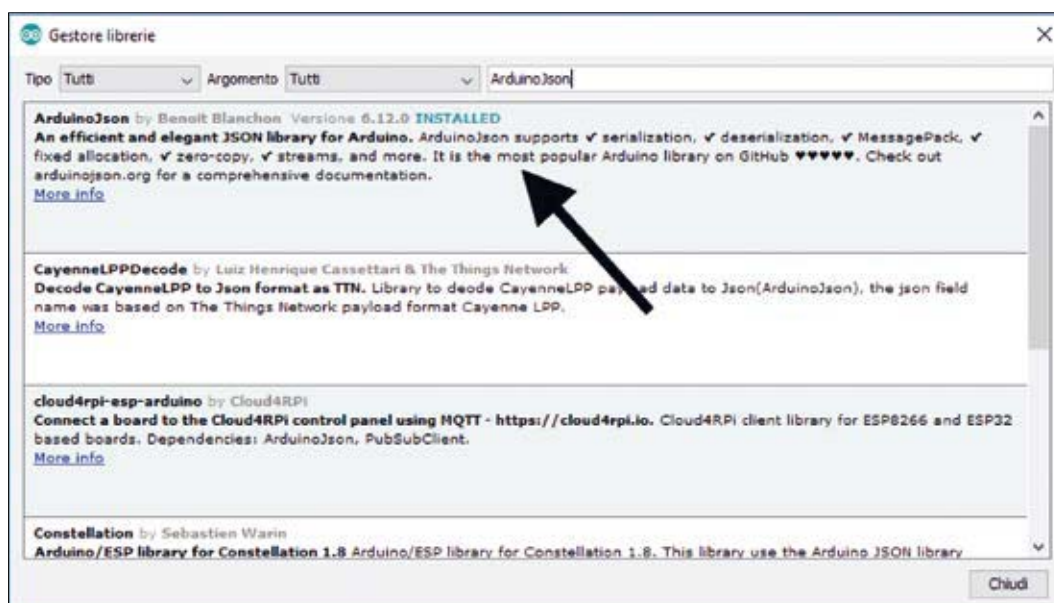
```
void loop() {
    // Acquisisce l'ora corrente
    CurTime ct = getTime();
    Serial.print("CurrentHour:");
    Serial.println(ct.CurrentHour);
    Serial.print("CurrentMinutes:");
    Serial.println(ct.CurrentMin);
    Serial.print("CurrentSeconds:");
    Serial.println(ct.CurrentSec);
    // Comparo l'ora corrente con l'ora di invio
    Twitt
    if (ct.CurrentHour == ShootAt) {
        if (WiFi.status() == WL_CONNECTED) {
            if (! twitt_done ) {
                initialize_CAM();
                twitt_done = send_Twitt()
            }
            ;
            BeginSleep();
        }
        } else {
            wifi_connect();
        }
        } else {
            Serial.println("NOT IN TIME WINDOW");
            BeginSleep();
        }
    }
}
```

P_BUTTON.

Abbiamo poi la verifica di una condizione: se non è



Fig. 13
Impostazioni
dell'IDE.



stata eseguita la sincronizzazione oppure se il numero dei cicli LP è multiplo di 5 vengono eseguite la connessione WiFi e la sincronizzazione con il server NTP.

Nella sezione di Loop() (**Listato 3**), viene letta l'ora attuale dal RTC tramite la funzione `getTime()` che ritorna una `Struct CurTime`, viene quindi confrontata l'ora corrente con l'ora per la quale desideriamo che venga effettuato il Post. Se l'ora non coincide il dispositivo viene messo nuovamente in modalità LowPower. Se invece l'ora corrisponde, viene verificato se la connessione WiFi è attiva, se attiva viene inizializzato il sensore BME280, viene inizializzata la camera, viene scattata la foto ed inviato il Tweet e poi il dispositivo viene messo in modalità LowPower. Se non attiva viene eseguita la connessione WiFi.

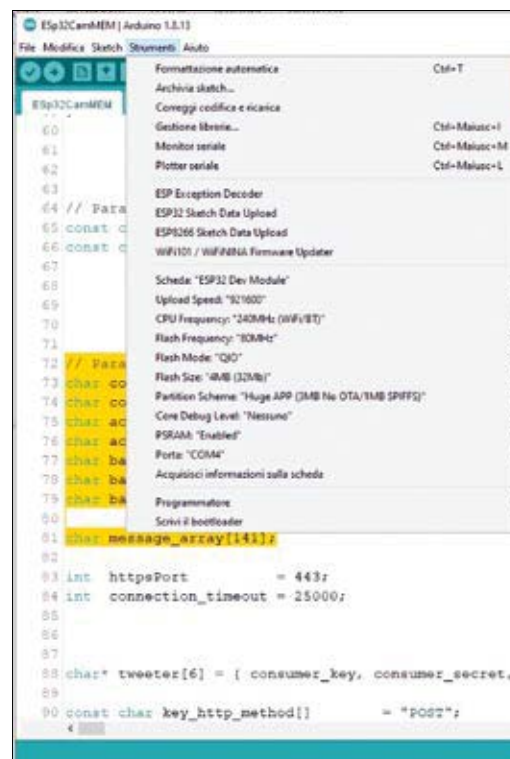
CARICHIAMO LO SKETCH SUL MODULO

Come primo passo compiliamo il codice. Apriamo la voce "Strumenti" dal menu della IDE di Arduino e settiamo le varie voci evidenziate come in **Fig. 13**. La porta seriale potrebbe essere diversa da quella che abbiamo indicato, in tal caso selezionate dal menu a discesa quella corretta nel vostro PC. Chiaramente dovremo poi mantenere le stesse impostazioni quando avvieremo il processo di caricamento. Terminata la configurazione cliccate sul pulsante con il segno di spunta nell'IDE di Arduino. Se il processo di compilazione si è concluso con successo, possiamo passare al caricamento dello sketch; se invece si verifica un errore di compilazione, dobbiamo prima

risolverlo.

Per poter caricare lo sketch sul modulo, dobbiamo per prima cosa collegare (prestando attenzione alla pin-out ed al settaggio dei livelli di tensione) il convertitore USB/seriale alla porta USB del nostro PC e al connettore che abbiamo predisposto sulla motherboard, settare in modalità programmazione il modulo chiudendo con

Fig. 14
Preparazione della
scheda.



un ponticello il connettore SV5 che fa capo al GPIO0. Il modulo del convertitore USB proposto in **Fig. 15** ha una piedinatura che si adatta al connettore della schedina, ma non è scontato. Se avete un convertitore diverso prestate attenzione alla corrispondenza dei pin. Poi dobbiamo cliccare sul pulsante di upload dell'IDE di Arduino e dopo qualche secondo pigiare brevemente e rilasciare il pulsante di Reset che si trova sotto il modulo. Se tutto procede in modo corretto vedremo apparire nella parte inferiore della IDE dei messaggi di log che ci informano dello stato del caricamento. Terminato il quale saremo invitati a cliccare nuovamente sul pulsante di reset, dobbiamo però prima rimuovere il ponticello sul GPIO0 che setta la modalità di programmazione (**Fig. 16**). Rimosso il ponticello, apriamo il monitor seriale e pigiamo il pulsante di reset. Dopo qualche secondo dovrebbero apparire dei messaggi di log che ci informano del funzionamento del nostro sketch.

REALIZZAZIONE PRATICA

Per la programmazione monteremo l'ESP32-CAM sulla scheda madre FT1502 Futura Elettronica proposta nel progetto "Microcamera WiFi con ESP32-CAM" presentato sul numero 243, che consente il caricamento da PC tramite un convertitore USB-TTL; diamo per scontato che abbiate già disponibile tale scheda adattatrice, altrimenti per la sua realizzazione rimandiamo al fascicolo n° 243. Chi non se la sente di realizzare da sé la scheda FT1502 può far realizzare il PCB utilizzando il servizio PCBPRODUCTION su www.futurashop.it.

Il progetto si realizza facilmente connettendo tra loro la breakout board BME280 e la ESP32-CAM come proposto nella **Fig. 3**; il tutto andrà innestato nella scheda adattatrice giusto il tempo per programmare il modulo. Notate che la camera OV2640 presente sul modulo ESP32-CAM, normalmente è già collegata al connettore e assicurata con un pezzetto di nastro biadesivo alla parte metallica dello slot miniSD. Per evitare di danneggiarla, non flettete in modo ripetuto il sottile connettore flessibile di cui è dotata e sinceratevi che sia fissata con il biadesivo.

CONCLUSIONI

Bene, per questo articolo abbiamo terminato; con questa nuova applicazione vi risulterà chiaro che il modulo ESP32-CAM è molto versatile. Per il

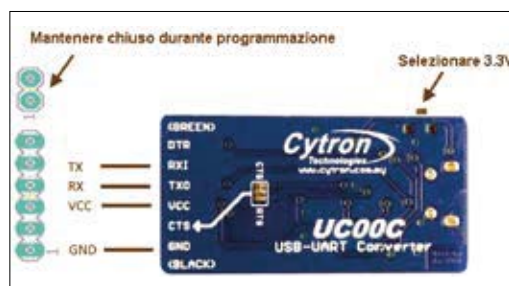


Fig. 15
Il nostro convertitore seriale/USB.

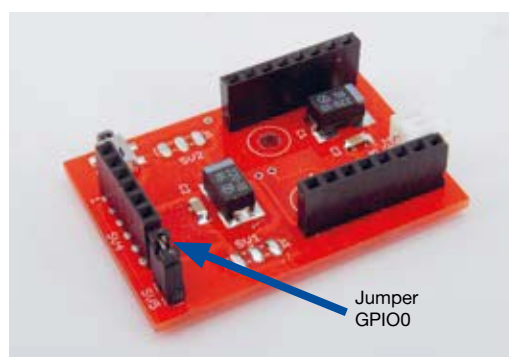


Fig. 16
La scheda madre su cui innestare l'ESP32-CAM per la programmazione.

momento abbiamo appena iniziato a sfruttarne le enormi potenzialità. Sperimentate con il progetto che vi abbiamo introdotto in questo articolo cercando di prendere confidenza con il modulo ESP32-CAM e con la relativa motherboard. Il codice e le librerie utilizzate nell'articolo sono disponibili al seguente url di Github: <https://github.com/open-electronics/ESP32-CAM-TWITTER>. ■



Cosa occorre?

La camera con ESP32 (cod. ESP32-CAM) è disponibile presso Futura Elettronica a Euro 16,50, il convertitore USB/TTL (cod. UC00A) è disponibile a Euro 15,50, il PCB della scheda adattatrice può essere realizzato utilizzando il nuovo servizio di stampa circuiti stampati (cod. PCBPRODUCTION) disponibile dal sito www.futurashop.it. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)
Tel: 0331-799775 - <http://www.futurashop.it>



Fishino

Design With Simplicity

I TUOI PROGETTI DIVENTANO WIRELESS

Fishino UNO, Fishino MEGA, Fishino32, Fishino GUPPY e Fishino SHARK sono schede di sviluppo Arduino-compatibile dotate di WiFi e lettore di schede microSD. Grazie ad un completo pacchetto di librerie (scaricabile dal sito www.fishino.it), si ha la possibilità di gestire tutti i propri progetti dal web in maniera semplice e veloce!

Fishino UNO



cod. FISHINO UNO
€ 36,00



- Controller: ATmega328
- Alimentazione:
 - da 7 a 12 Vdc (tramite plug)
 - 5 Vdc (tramite porta USB)
- Compatibile al 100% con Arduino UNO
- Modulo WiFi
- Interfaccia per scheda MicroSD
- Modulo RTC con batteria di mantenimento
- Sezione di alimentazione a 3,3 V potenziata
- Compatibile con shield e schede millefori
- Peso 25 g
- Dimensioni 76,5 x 53,5 x 14 mm

- Controller: ATmega2560
- Alimentazione:
 - 3,6 Vdc (tramite batteria esterna al litio)
 - da 3,5 a 20 Vdc (tramite plug)
 - 5 Vdc (tramite porta USB)
- Circuito di ricarica per batteria al litio
- Compatibile al 100% con Arduino MEGA
- Modulo WiFi
- Interfaccia per scheda MicroSD
- Modulo RTC con batteria di mantenimento
- Stadio di alimentazione switching (5 Vdc e 3,3 Vdc)
- Compatibile con shield e schede millefori
- Peso 37 g
- Dimensioni 101,5 x 53,5 x 15 mm



Fishino MEGA



cod. FISHINOMEGA
€ 55,00

Fishino32



cod. FISHINO32
€ 65,00



- Controller: PIC Microchip a 32 bit
- Alimentazione:
 - 3,6 Vdc (tramite batteria esterna al litio)
 - da 3,5 a 20 Vdc (tramite plug o sull'ingresso Vin)
 - 5 Vdc (tramite porta USB)
- Circuito di ricarica per batteria al litio
- Compatibile con Arduino UNO
- Modulo WiFi
- Modulo RTC con batteria di mantenimento
- Clock 120 MHz, riducibile via software
- Interfaccia per scheda microSD
- Codec audio stereo
- Peso 29 g
- Dimensioni 76,5 x 53,5 x 14 mm

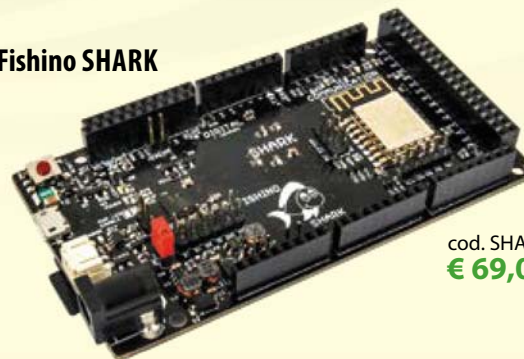
Fishino GUPPY



cod. GUPPY
€ 39,00

- Controller: ATmega328
- Alimentazione:
 - 3,6 Vdc (tramite batteria esterna al litio)
 - da 6,5 a 20 Vdc (tramite PIN VIN)
 - 5 Vdc (tramite porta USB)
- Circuito di ricarica per batteria al litio
- Compatibile al 100% con Arduino NANO
- Modulo WiFi
- Interfaccia per scheda MicroSD
- Peso 10 g
- Dimensioni 75 x 20 x 18,5 mm

Fishino SHARK



cod. SHARK
€ 69,00

- Formato standard Arduino MEGA
- Processore PIC32MX470F512
- 512 kB di ROM
- 128 kB di RAM
- Frequenza di clock di 105 MHz
- Interfaccia USB nativa, sia device che host
- RTC incorporato
- Lettore per schede microSD
- Modulo WiFi integrato
- Codec audio
- Tensione di alimentazione: 3-20Vdc
- Alimentazione a batteria, plug e/o via connettore USB
- Funzionamento interno a 3,3 V
- Pin DIGITALI 5V-tolerant
- Ricarica automatica di una batteria LiPo
- Spegnimento da software con processore in standby
- Wake-up tramite pin esterno o in tempi prefissati

Fishino PIRANHA



cod. PIRANHA
€ 49,00

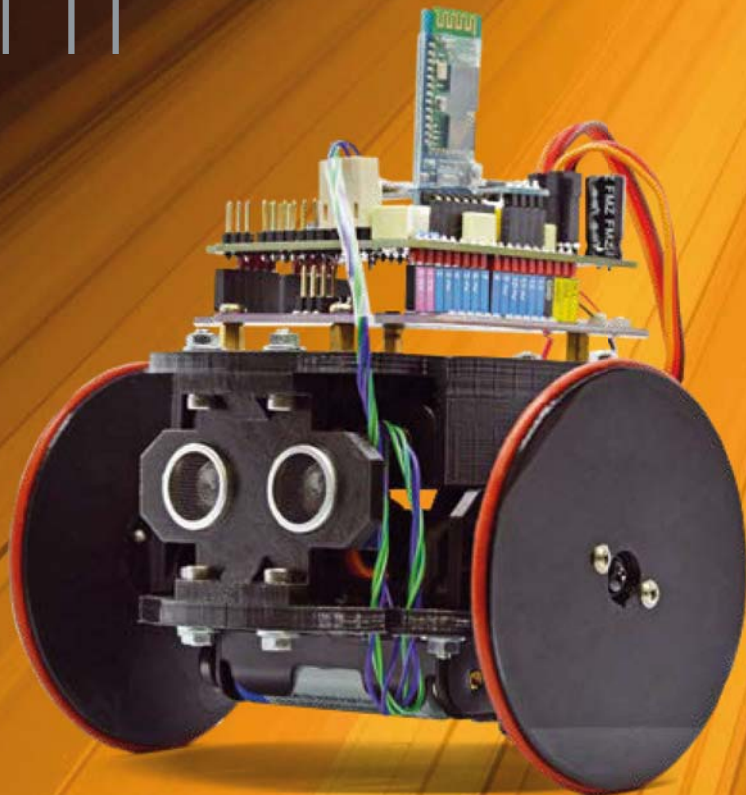


- Alimentazione:
 - 3,7 Vdc (tramite batteria esterna al litio)
 - da 3,5 a 20 Vdc (tramite plug)
 - da 3,5 a 20 Volt sull'ingresso Vin
 - 5 Vdc (tramite porta USB)
- Circuito di ricarica per batteria al litio
- Compatibile al 100% con Arduino MKR1000
- Modulo WiFi
- Interfaccia per scheda MicroSD
- Peso 10 g
- Dimensioni 62 x 25 x 16 mm

Prezzi IVA inclusa.

ARLOK: IL ROBOT PER TUTTI

di GIOVANNI BERNARDO



Piccolo automa su ruote gestito da una scheda MakerUNO, versione elaborata di Arduino Uno, capace di muoversi evitando ostacoli.

La robotica educativa è una tematica sempre più presente nella didattica e raccoglie sempre più adesioni da parte di giovani e meno giovani; in questo contesto si colloca ARLOK, un semplice robot educativo su ruote ideato per gli studenti e gli sperimentatori alle prime armi.

Il telaio è molto semplice da realizzare con la stampa in 3D: non sono necessari costosi macchinari di precisione. Il telaio di ARLOK è fatto da pezzi abbastanza grandi per poter essere maneggiati e assemblati con successo anche da chi ha poca manualità.

Il cuore di ARLOK è un'elettronica di controllo basata sulla scheda Arduino-like Maker UNO, per la quale rendiamo disponibili i codici sorgente di esempio e relative istruzioni. In questo articolo introdurremo il robot, spiegandovi quali sono le sue potenzialità e cosa ci si può fare, guidandovi nella

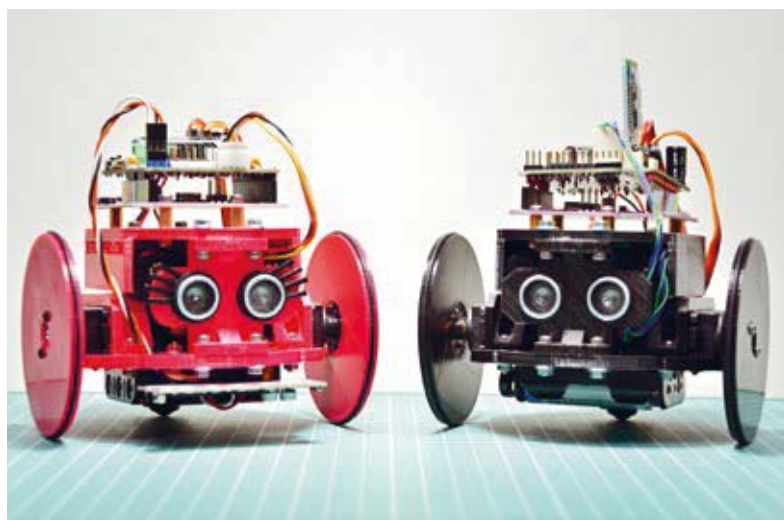


Fig. 1
Il robot su
ruote ARLOK.

costruzione e nei primi esperimenti.

ARLOK ha una storia molto particolare: nasce dalla continua sperimentazione che ha portato l'autore a mettere insieme vari componenti, fino a far nascere un robot; il primo nome assegnato ad esso doveva essere *Arlecchino*, perché il suo "vestito" cioè la sua scocca, era stata realizzata con avanzi di stampa 3D e risultava di vari colori.

Arlecchino è stato trasformato in **AR.L.O.**, che è poi diventato l'*acronimo inverso* di *ARduino Loaded with O-rings* (non ha quindi nulla a che vedere col dinosauro della Disney/Pixar) perché le ruote hanno per "gomme" degli O-ring. AR.L.O. è stato il capostipite della generazione da cui poi è nato il robot descritto in questo articolo: ARLOK, che sostanzialmente è AR.L.O. in versione basata sulle schede CYTRON MakerUNO e Maker Line. ARLOK è un robottino a 2 ruote a guida differenziale, ovvero munito di due ruote entrambe collegate ad un proprio motore, nel quale la sterzata si ottiene facendo girare le ruote in senso opposto alla stessa velocità (il baricentro rimane fermo) oppure facendo girare le ruote nella stessa direzione ma a velocità differenti (virata). L'equilibrio è mantenuto tramite un terzo punto di appoggio fornito da un ball-caster (una sferetta montata in un supporto, libera di rotolare). Il firmware base di ARLOK è uno sketch Arduino che fa muovere il robot sempre dritto fino a che non trova un ostacolo, rilevato da un SONAR, allorché provvede a cambiare direzione. A completamento dell'hardware, ma come accessori, sono previsti un piccolo display OLED e un modulo Bluetooth. Il robottino può essere espanso a piacere man mano che si acquisisce confidenza con la programmazione e con le dinamiche di movimento. Di seguito analizziamo la struttura del robot.

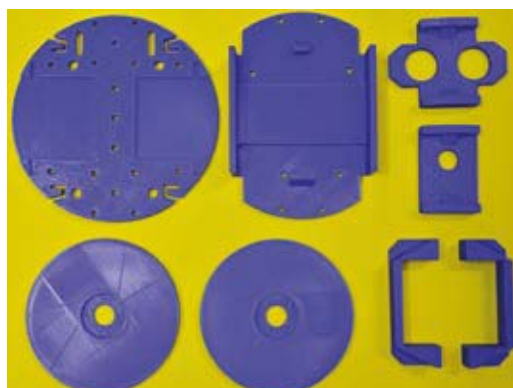
Fig. 2
Pezzi di cui si
compone il telaio di
AR.L.O.

TELAIO

Il telaio è realizzato mediante la stampa 3D e le parti da stampare sono state create con Tinkercad; i relativi file STL, insieme ai programmi di esempi ed altro, sono disponibili sul repository Github di ARLOK (<https://github.com/settezero/arlok>). I file vanno stampati in modo che i pezzi risultino abbastanza solidi, quindi impostando un riempimento (fill-rate) del 25-30% e scegliendo magari come filamento il PLA+, che rispetto al PLA presenta delle caratteristiche migliori. Il telaio si compone di 8 parti (6 file singoli) che sono quelle mostrate nella **Fig. 2**.

Abbiamo un pianale, rotondo, dello spessore di 5mm, che è il pezzo in alto a sinistra della **Fig. 2**; il pianale (file *arlok_plate_bottom.stl*) ha i fori per il fissaggio delle staffe dei servo, le scanalature per accogliere i servo, le forature sia avanti che indietro per il modello di ball-caster scelto e due fori centrali per accogliere un portabatterie standard per 3AA posto verso il basso, nonché delle gole e dei fori più grandi utili per raccogliere i cavi. Le scanalature per accogliere i servo sono profonde 2.5mm. Considerando che un servo standard (coricato sul lato più lungo col perno di uscita parallelo al piano) ha un'altezza di circa 20mm e che il perno si trova al centro (a 10mm), la distanza tra punto centrale del perno e terra (considero il servo incastrato nella base, con la base appoggiata a terra) è pari a 12,5mm: useremo dopo questo valore per alcune considerazioni.

Ci sono quindi le due staffe per bloccare i servo (*arlok_servo_bracket.stl*, da stampare 2 volte) e un pianale superiore (*arlok_plate_top.stl*) con le forature per accogliere una scheda Arduino e con delle alette per trattenere il portabatterie da 4 stilo che andrà appoggiato sulle staffe dei servo, usato per alimentare i servocomandi. Sono presenti due montanti per il pianale superiore: uno anteriore con i fori per incastrare il sensore ultrasuoni



(*arlok_pillar_sonar.stl*) presente anche nella versione rounded con le parti laterali arrotondate anziché squadrate) e uno posteriore presente in tre versioni: con foro da 6 mm (*arlok_pillar_hole_06.stl*) con foro da 10 mm (*arlok_pillar_hole_10.stl*) e con foro da 12 mm (*arlok_pillar_hole_12.stl*). In questo foro va l'interruttore per scollegare il pacco da 4 batterie stilo. Ci sono tre versioni con foro di diverso diametro perché il foro da 6 mm può essere utilizzato per i classici interruttori a levetta, mentre quelli da 10 e 12 mm sono destinati a ospitare i pulsanti autobloccanti.

Per l'assemblaggio del telaio andranno usate 12 viti M3 con la parte filettata lunga 12mm, consiglio il tipo TCEI (*Testa Cilindrica con Esagono Incassato*) dato che in alcuni punti meno accessibili è facile stringerle con una chiave esagonale a 90°. Il ball-caster andrà montato con le viti in dotazione nel relativo kit: basta mettere solo le due viti posteriori. La scheda MakerUNO, essendo piatta nella parte inferiore (non ci sono saldature o pin che sporgono), può essere fissata al pianale usando del biadesivo o, meglio, del velcro per poterla rimuovere agevolmente, oppure ancora, per ottenere un ancoraggio più stabile, dei distanziali F/F da almeno 10mm con 8 viti da 5 mm.

LE RUOTE

Le ruote di ARLOK si ottengono per stampa 3D da un progetto presente su Thingiverse, chiamato Servo Wheel 4.0 (by Obijuan). Questo progetto è costituito da un file OpenScad che permette di realizzare delle ruote personalizzate adatte ad essere montate sulle squadrette dei servo. OpenScad è un tool molto interessante perché permette di parametrizzare i vari aspetti dei pezzi da stampare. Le ruote generate con questo script sono piene, hanno una scanalatura sul bordo che serve ad accogliere un O-Ring di gomma per dare grip e un incavo circolare (o a stella a 4 o 6 braccia) per accogliere la squadretta da montare sul servo, con quattro minuscoli fori intorno per il fissaggio della ruota alla squadretta. Tutti questi aspetti della ruota sono parametrizzati. Il file STL della ruota si trova anch'esso nel repository Github con il nome *arlok.wheel.stl*; la ruota ha uno spessore di 4mm, un diametro adatto ad accogliere un O-Ring di 7 cm di diametro e un incavo circolare del diametro di 21mm per le squadrette dei servo.

Il diametro finale delle ruote è influenzato dal diametro interno dell'O-Ring e va scelto insieme all'altezza del ball-caster per fare in modo che il



Fig. 3
Squadrette per servo. In inglese vengono chiamate "horn".

robot si mantenga dritto, o meglio, leggermente inclinato verso la parte posteriore per fare più peso sul ball-caster e mantenere meglio l'equilibrio. Il ball-caster scelto, della Tamiya, può essere montato in diversi modi, così da ottenere diverse altezze. Riprenderemo il discorso tra breve.

Il diametro della ruota finita sarà quindi uguale al diametro interno dell'O-Ring + due volte lo spessore (la sezione) dello stesso: avendo utilizzato un o-ring che ha una sezione di 2,5 mm e un diametro interno di 70 mm, la mia ruota ha un diametro esterno finito di 75 mm. Tenete conto che poi il diametro finale aumenta leggermente quando andrete ad installare l'O-Ring.

Vediamo come si può personalizzare la ruota; per prima cosa apriamo il file in OpenScad:

```
//-- Wheel parameters
wheel_or_idiam = 50; //-- O-ring inner diameter
wheel_or_diam = 3; //-- O-ring section diameter
wheel_height = 2*wheel_or_diam+0; //-- Wheel
height: change the 0 for
```

Le dimensioni sono espresse tutte in mm. Il primo parametro (*wheel_or_idiam*) imposta il diametro interno dell'o-ring. Il secondo imposta lo spessore (il diametro della sezione circolare) dell'o-ring e l'ultimo indica lo spessore della ruota: normalmente viene spessa il doppio dell'o-ring con l'o-ring al centro. Se vogliamo la ruota più spessa cambiamo lo zero con un numero maggiore, ma senza esagerare. Seguono quindi i parametri usati per tutti i tipi di squadretta (horn):

```
//-- Parameters common to all horns
horn_drill_diam = 1.5;
horn_height = 6; //-- Total height: shaft + plate
horn_plate_height = 2; //-- plate height
```

Il primo parametro indica il diametro dei fori di fissaggio squadretta (default 1,5 mm). Il secondo parametro indica lo spessore totale della squa-

dretta (parte piana+imbocco perno servo) e l'ultimo il solo spessore della parte piana (senza tener conto, quindi, dell'imbocco del perno): questi due parametri serviranno a "scavare" la parte centrale della ruota per fare in modo che la squadretta entri dentro la ruota ad incastro. Subito dopo ci sono parametri relativi ad una squadretta di tipo circolare. I valori predefiniti sono quelli relativi alle squadrette forniti con i servo della Futaba:

```
//-- Futaba 3003 servo rounded horn parameters
rh_diam1 = 8.5; //-- Rounded horn small diameter
rh_diam2 = 21.5; //-- Rounded horn big diameter
rounded_horn_drill_distance = 7.3;
```

Il primo parametro indica il diametro esterno dell'imbocco del perno del servo (*small diameter*) e il secondo il diametro esterno (*big diameter*) della squadretta. Il terzo infine indica la distanza dei fori di fissaggio della squadretta dal centro. Seguono quindi parametri relativi ad altri tipi di squadrette, che non prendo in considerazione perchè è già difficile mettere d'accordo le tolleranze della stampante 3D con le misure prese a mano della squadretta circolare, figuriamoci con le squadrette a forma di stella. La generazione delle ruote si trova alla fine dello script:

```
//-- Test!
Servo_wheel_rounded_horn();
translate([wheel_or_idiam+10,0,0]) Servo_wheel_4_
arm_horn();
translate([-wheel_or_idiam-10,0,0]) Servo_wheel_6_
arm_horn();
```

La ruota per la squadretta tonda è generata dalla funzione *Servo_wheel_rounded_horn()*; per cui le altre due funzioni, che generano le ruote per le squadrette a 4 e 6 braccia, le potete commentare con un doppio slash. Per avere le ruote piene al 50%, ci vogliono 37 grammi di filamento (circa 12,4 metri). In totale, quindi, telaio+ruote necessitano di circa 123 grammi di filamento (41,4 metri). Questi dati sono approssimativi e dipendono da tanti fattori quali percentuale di riempimento e spessore delle pareti.

BALL CASTER

Dato che ci sono due ruote mosse ciascuna da un proprio motore, poste più o meno al centro, e che non si tratta di un balancing-robot, ARLOK

per non ribaltarsi ha bisogno almeno di un terzo punto di appoggio sul quale deve cadere gran parte del peso. Il ball-caster impiegato in ARLOK è abbastanza comune e ingegnerizzato meglio rispetto alla media; si tratta del **Tamiya 70144**, reperibile presso la Futura Elettronica (www.futurashop.it). In questo modello di Ball Caster la sfera d'acciaio non poggia sulla plastica della struttura ma rotola su 3 piccole sbarrette metalliche e questo consente di avere un attrito e un'usura minori rispetto ad altri modelli. Questo ball-caster si può montare in 4 differenti modi che portano a 4 diverse altezze: 11mm, 16 mm, 27mm e 37mm. Le ultime 2 altezze, 27 e 37mm, in realtà vengono raggiunte montando tra le due parti principali della struttura delle rondelle di plastica (indicate come H1 nello schema di montaggio) che sono alte circa 2 mm: togliendole si arriva quindi a 25 e 35mm di altezza. Dunque, il gruppo di locomozione di ARLOK si basa su:

- due ruote diametro 75mm (o-ring sezione 2,5 mm e diametro interno 70 mm);
- Ball-Caster Tamiya 70144 nella versione da 27 mm ma senza le rondelle H1 (altezza ball-caster: 25 mm).

Esistono comunque tanti modelli di ball-caster stampabili in 3D. Cambiando Ball-Caster bisogna poi valutare l'altezza da terra e cambiare il diametro delle ruote (e di conseguenza trovare altri due O-Ring) e capire se possono essere montate le batterie al piano inferiore eventualmente vogliate utilizzarle (vedremo tra poco se sono necessarie). Oltre ai ball-caster esistono anche i *ruotini pivotanti*, tipo di quelli usati per i mobili, ma sono sconsigliati perchè a volte, ruotando di 90° rimangono incastrati; per contro vanno meglio su pavimenti con piastrelle "in fuga".

ALIMENTAZIONE

Per l'alimentazione dell'insieme è stato scelto un pacco di 4 pile stilo (AA) alloggiato in un porta-batterie, montato tra le staffe dei servocomandi e la base del piano superiore che ha delle ali per accoglierlo. Utilizzando un'unica fonte di alimentazione a 6V non è possibile fornire tale tensione direttamente ad Arduino, quindi senza scomodare regolatori di tensione è stato applicato un diodo al silicio in serie alla linea che dai 6V porta alla MakerUNO e alla logica.

Il diodo determina una caduta di tensione pari a circa 0,7V tale che alle logiche arrivano tra 5,3V e 5,4V.



Il microcontrollore ATmega328 a bordo della MakerUNO può funzionare fino a 5,5V, così come gli altri integrati a bordo della scheda. A bordo del robot, però, oltre ad Arduino ci sono altre logiche che devono funzionare a 5V: abbiamo il sensore ad ultrasuoni HC-SR04 ed eventualmente il display OLED e il modulo Bluetooth.

Il sensore HC-SR04 può funzionare fino a 5,5V. Il display OLED è basato sul classico SSD1306, un controller che funziona a 3,3V grazie a un regolatore che consente di alimentare il modulo con 5 ed anche 6V. Tensione di alimentazione a parte, c'è da considerare la parte relativa al level-shifting dei segnali logici, dato che i livelli sulle linee SDA/SCL devono giungere al controller necessariamente a 3,3V. Ebbene, nei display OLED 128x32 Adafruit la conversione dei segnali su SDA/SCL da livello logico alto 5V a 3,3V è affidata a MOSFET collegati alla linea dei 3,3V, per cui non ci sono problemi ad alimentare la logica a 5 o 5,5V. Su quelli cinesi, dato il basso costo non vengono usati MOSFET, che comunque non sono necessari nel caso in cui sul bus I²C il livello di tensione venga stabilito dal display stesso: sappiamo bene che le linee dell'I²C sono open-drain (i dispositivi che comunicano tra loro tirano solo la linea verso il livello basso, mentre quando bisogna andare a livello alto, vanno in alta impedenza e il livello è fornito da resistenze di pull-up).

Tipicamente SDA e SCL sono portati a livello alto (verso i 3,3V forniti dal regolatore on-board) da due resistenze da 4,7 kΩ e in più Arduino UNO e Leonardo riportano i due pin SDA e SCL sull'header senza resistenze di pull-up, il che significa che il livello logico alto non può arrivare ai 5,3V che stiamo fornendo alla Maker UNO, ma viene determinato dall'altro dispositivo posto sul bus (che ha il pull-up verso 3,3V), quindi anche per il display non c'è problema.

Il modulo Bluetooth viene alimentato a 5V ma a bordo ha un regolatore di tensione a 3,3V che "regge" come minimo a 6V in ingresso o 5,5V nel peggiore dei casi. Nessun problema, invece, per la logica, perché questi moduli Bluetooth richiedono sempre che esternamente venga messo un partitore di tensione sulla loro linea di ricezione nel caso in cui il dispositivo con cui comunicano non lavori a 3,3V. Sul PCB della breakout board del modulo si legge chiaramente: *Level: 3.3V*.

Sullo shield di ARLOK è stato predisposto il partitore formato da R1 e R2 dalla linea TX di Arduino verso la linea Rx del modulo Bluetooth. Con i valori specificati nello schema (R1=15kΩ e

R2=22kΩ), al modulo Bluetooth arriverà un livello logico massimo di 3,3V se l'alimentazione è 5,5V.

$$V_{OUT}=V_{IN}*(R2/R1+R2)=V_{IN}*0.5945$$

Il range di tensione sul pin RX del modulo è quindi da 3,3V (massimo, alimentando con Vservo a 6V) a 2,67V (minimo, alimentando con 3 pile AA): non superiamo il massimo ammesso e rientriamo nei 2V per far considerare valido il livello. Per la linea di comunicazione opposta, da TX del modulo a RX di Arduino, non c'è bisogno di fare nessun adattamento perché i 3,3V forniti dal modulo vengono già interpretati correttamente dalle logiche a 5V. Il PCB ha comunque due ingressi: uno per la tensione dei servo (6V) e uno per la tensione logica (VDD). È quindi possibile utilizzare un portatile aggiuntivo per 3 stilo (4,5V) da mettere sotto al pianale per alimentare la sola logica; in questo modo le tensioni di alimentazione (servo e logica) sono separate e si minimizzano anche eventuali disturbi: questa è la configurazione raccomandata. Volendo è possibile utilizzare il portabatterie per 4 stilo con batterie ricaricabili NiMH: dato che queste hanno una tensione di 1,2V, avremo un totale di 4,8V: in questo caso, però, non tutti i servo potrebbero funzionare correttamente e al posto di D1 dovremmo mettere un ponticello, perché se scaliamo la tensione arriviamo a 4,1V e le logiche potrebbero non funzionare correttamente.

SERVOCOMANDI

Perché usare un servocomando al posto di un motoriduttore? Innanzitutto perché i servo hanno dimensioni standard, contengono un motoriduttore, costano molto poco e sono molto facili da trovare; ma la cosa forse più importante di tutte è che hanno l'elettronica di pilotaggio del motore a bordo, il che semplifica l'elettronica richiesta da ARLOK. Chiaramente non è possibile usare i servocomandi così come sono dato che hanno una rotazione limitata. Anche se in commercio esistono già servocomandi a rotazione continua, vediamo di seguito come modificare due servocomandi standard RC per farli diventare a rotazione continua. Sul prototipo del robot sono stati usati dei servocomandi: sono molto comuni e hanno gli ingranaggi in metallo, una costruzione molto solida con cavi più spessi e invece dei cuscinetti hanno delle boccole in metallo sui perni di uscita. Sullo shield sono previsti connettori per servo con una pin-out standard: ovvero con il terminale di alimentazione (6V) al centro.



Generalmente i servo hanno una pin-out del tipo segnale-tensione-GND (S + -) anche se possono cambiare i colori dei fili. Se utilizzate servocomandi con diversa piedinatura bisogna separare i contatti del connettore. La modifica per la rotazione continua è molto semplice e consiste in due operazioni: rimozione del notch dall'ingranaggio del perno di uscita e sostituzione del potenziometro (che funge nei servo da sensore di posizione) con due resistenze di uguale valore (consigliato 22 Kohm). In pratica il servo, leggendo sempre un valore "centrale" sull'IO del potenziometro, crederà che l'albero si trovi sempre al centro e quindi l'elettronica a bordo continua far ruotare il motore anziché fermarlo. Il notch viene messo solo per sicurezza.

Alcuni servo più sofisticati, però, rilevano la condizione di rotazione continua perché quando danno corrente al motore e questo gira (tra l'altro ne rilevano anche l'assorbimento) rilevano che la posizione restituita dal potenziometro di feedback rimane sempre al punto centrale; questo determina anomalia e l'arresto del motore. Pertanto bisogna utilizzare dei servo analogici classici.

DISPLAY OLED

Lo shield e il codice sono predisposti per l'utilizzo di un piccolo display OLED da 0,91" con una risoluzione di 128x32 pixel (**Fig. 4**). Sebbene sia utile per la taratura del punto zero dei servocomandi e per mostrare altre informazioni, non è necessario perché la taratura si può fare anche senza.

MODULO BLUETOOTH

Per la connessione Bluetooth è stato previsto il modulo HC-05, ma va bene anche l'HC-06: l'importante è che sia quello montato su breakout

board con pin-strip che porta all'esterno i pin RX/TX. A bordo è presente un regolatore di tensione a 3,3V per il chipset, i cui livelli logici sono quindi 0/3,3V; ecco perché è necessario mettere un partitore tra la linea TX di Arduino, che lavora a 5V, e la linea RX del modulo.

Quando innestate il modulo bluetooth controllate la piedinatura: il pin RX del modulo va allineato con la piazzola TX e viceversa.

Il modulo Bluetooth è un accessorio e può essere utile per sperimentare ulteriormente: avere un debug, ad esempio, sul cellulare utilizzando un software qualsiasi che fa da terminale o possiamo anche realizzare un'app, magari tramite MIT App Inventor, per poter pilotare il robottino dal cellulare o semplicemente avere una dashboard con indicazioni e letture di sensori fornite dal robot. Ancora potremmo utilizzare un'app android, Bluetooth Electronics, che è un semplice tool di comunicazione seriale over Bluetooth che ci dà la possibilità di costruire interfacce con pulsanti ai quali possiamo associare comandi da inviare via seriale: sul repository GitHub è presente proprio questo esempio.

Il modulo HC-05 può anche fungere da master oltre che da slave, a differenza dell'HC-06 che funziona solo come slave; l'uno o l'altro, per il nostro ARLOK è indifferente perché sarà sempre il cellulare a fare da master. Un'altra differenza tra i due moduli sta nella modalità di funzionamento attiva all'avvio: il modulo HC-06 si avvia in modalità AT (può accettare da subito comandi per la configurazione), mentre l'HC-05 non accetta comandi AT all'avvio e per mandarlo in questa modalità bisogna premere il pulsantino on board. Questi moduli, oltre all'alimentazione e alle linee UART hanno altri due pin: STATE, che è il pin a cui è collegato il led on-board che lampeggia in modo diverso a seconda della modalità attiva, e un altro pin che a volte è riportato come KEY, altre volte come EN. In genere quando c'è EN si tratta di un pin per spegnere il modulo: mandando un livello basso si spegne, lasciandolo scollegato o a 3,3V il modulo si accende.

Quando è riportato come KEY potrebbe non servire a nulla (HC-06) o per attivare la modalità AT sul modulo HC-05: in questo caso dato che il pin per la modalità AT è tenuto normalmente a livello basso con una resistenza di pull-down, per entrare in questa modalità dal pin (cioè senza premere il pulsante) bisogna mandarlo a livello alto (3,3V). Il problema di questi moduli è che ogni produttore è d'accordo sul pin STATE ma ognuno

Fig. 4
Il display OLED
I²C 0,91" 128x32
utilizzato.



fa un po' quello che gli pare con l'altro pin: KEY o EN; per tale motivo sul PCB è stato previsto un connettore a 6 pin per il modulo ma i due pin laterali non li ho collegati a nulla. Esistono poi alcune breakout board che non hanno i pin disposti allo stesso modo: a quel punto dovete utilizzare l'area millefori e fate voi un adattatore.

Il connettore per il modulo Bluetooth lo potete utilizzare anche per collegare altri dispositivi di comunicazione, basta che stiate attenti alla tensione di alimentazione del modulo che volete usare (i moduli ESP-01 vanno alimentati a 3,3V) e al pin-out.

Non abbiamo una stretta necessità della modalità AT: questa viene utilizzata per cambiare la configurazione del modulo ovvero cambiarne il baud-rate (default 9600bps sull'HC-05), la password di accesso (default 1234 sull'HC-05, a volte 0000), il nome (default HC-05) ecc. Sul modulo HC-05, quando si entra in modalità AT il baud-rate non è più 9600 ma 38400 e durante questa modalità non è possibile collegare un dispositivo Bluetooth. Se volete cambiare i parametri di configurazione del modulo vi è necessario un adattatore UART/USB e un programma di terminale.

SHIELD ARLOK

Per affacciare MakerUNO a sensori e attuatori del robot è stato realizzato uno shield specifico che accoglie i componenti elettronici a bordo del robot. Trovate lo schema elettrico dello shield in queste pagine, insieme al relativo piano di montaggio. Su un lato dello shield ci sono le due morsettiere



Fig. 5
Modulo Bluetooth HC-05 montato su Breakout Board con collegamento a 6 pin.

a vite per l'alimentazione con l'indicazione di positivo e negativo: su VServo andrà la tensione fornita dalle 4 batterie AA. Nel caso in cui volessimo utilizzare un solo pacco batterie (quello da 4 stilo), allora il jumper posto dietro i connettori a vite (JP1) va messo sulla posizione VServo (verso destra, in direzione del diodo): in questo modo, alle logiche arriverà la 6V scalata dal diodo D1 (quindi circa 5,3-5,4V). Un altro jumper (Power) può essere utilizzato per disconnettere la sola tensione logica: qui può anche essere messo un piccolo interruttore con passo 2,54.

L'altro connettore a vite (VDD) va usato nel caso in cui alle logiche volessimo fornire un'alimentazione separata: in questo caso J1 va chiuso verso la dicitura VDD (verso sinistra, in direzione dei connettori dei servo).

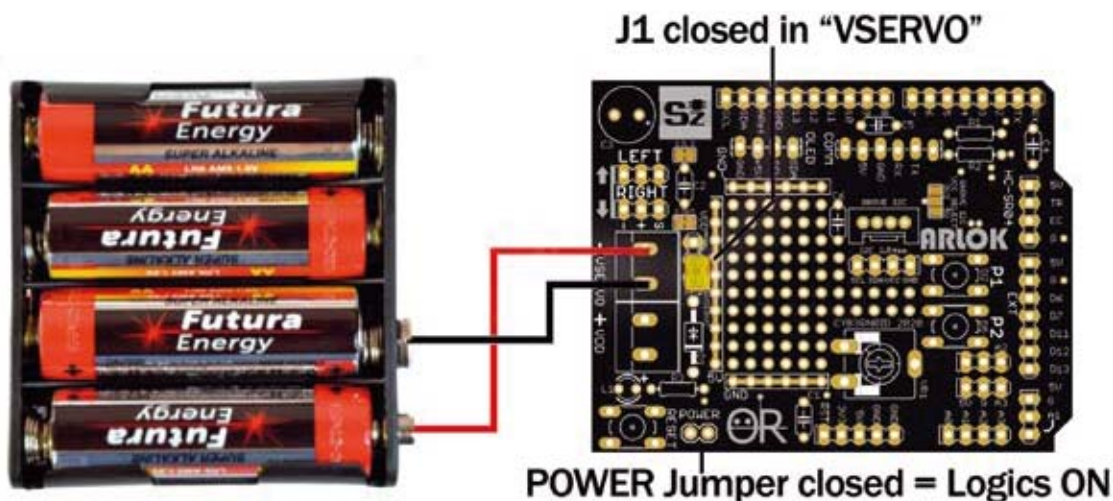
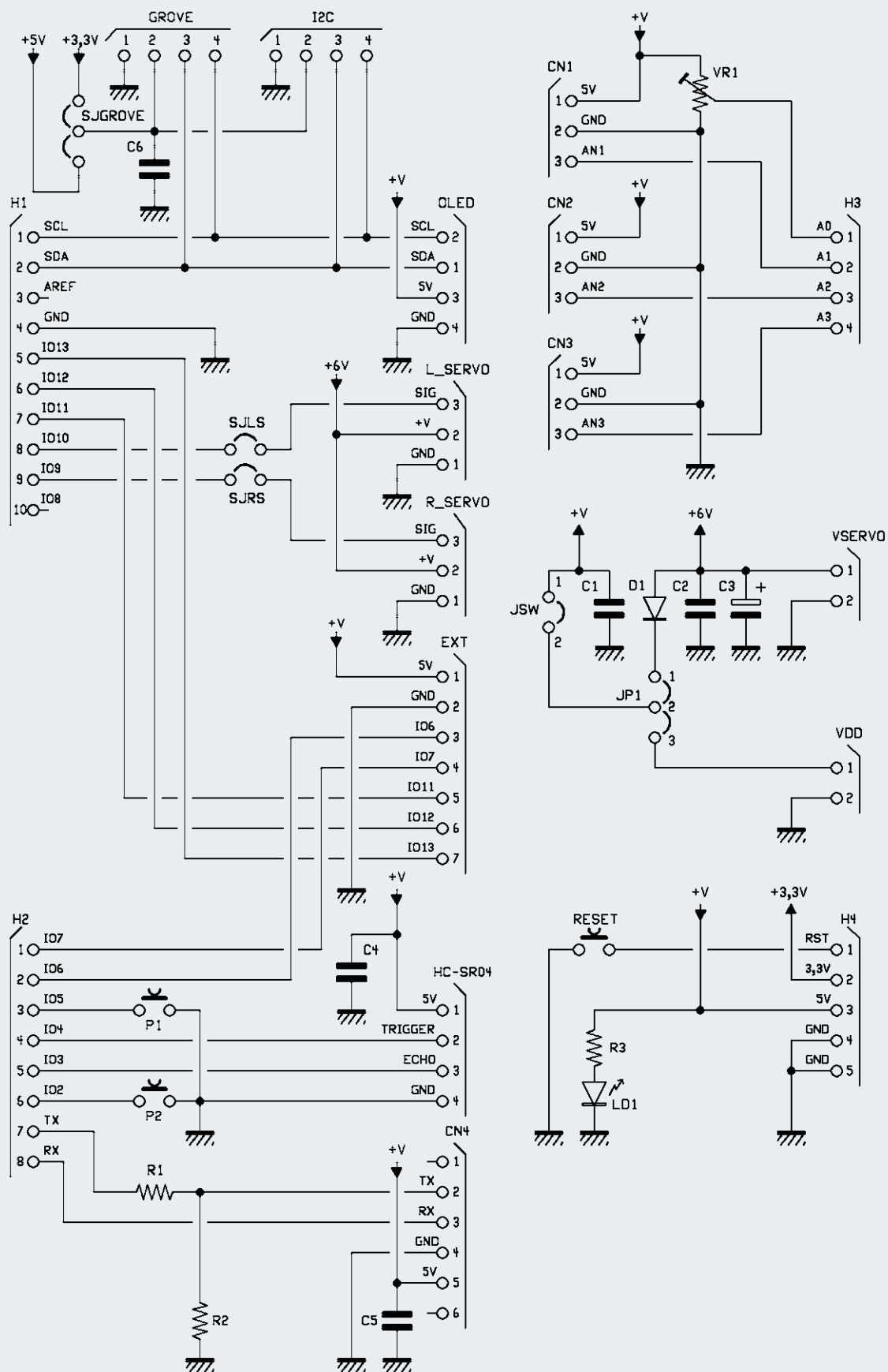


Fig. 6
Configurazione ad unica fonte di alimentazione del robot.



Nella **Fig. 6** viene mostrato come fare per alimentare il robot (sia servocomandi che logiche) con una sola sorgente costituita dal pacco batterie da 4 stilo.

Nella **Fig. 7**, invece, viene illustrato come configurare la scheda per poter utilizzare due alimentazioni separate (il pacco da 3 stilo andrebbe installato sotto al pianale inferiore, tra ruote e ball-caster, sfruttando i due/tre fori centrali). Se i servo assorbono poca corrente e quindi non creano sbalzi di tensione, si può utilizzare la prima soluzione (solo le 4 batterie AA); se invece notate che il modulo Bluetooth si resetta o addirittura si resetta la Maker UNO, provate prima ad aumentare il valore dei condensatori sulla scheda e anche ad aumentare l'ampiezza della rampa di decelerazione dei servo durante uno stop o un cambio direzione (i cambi improvvisi portano a picchi di assorbimento). Se anche questo non risolve i reset, allora è necessario utilizzare la seconda soluzione con il pacco batterie aggiuntivo per alimentare la sola logica.

All'altra estremità del PCB va collegato il sensore ultrasuoni rispettando la piedinatura riportata sulla serigrafia, che combacia con quella del sensore orientato con il quarzo verso il basso. Potete collegare il sensore utilizzando 4 cavetti jumper femmina/femmina oppure rimuovere l'header dal sensore ultrasuoni e saldare 4 fili che terminano

con un connettore Molex femmina passo 2,54 mm, da innestare sul maschio saldato sul PCB. Diversamente dovrete piegare verso l'alto l'header del sensore durante il montaggio perché tocchi sul vano batterie e non potete innestare i cavetti jumper. Potete utilizzare o un'accoppiata connettore femmina munito di cavi + maschio da montare sul circuito stampato oppure un cavo maschio/femmina da innestare direttamente sull'header del sensore (dopo averlo piegato). Nello schema di cablaggio in queste pagine vedete come collegare i moduli allo shield.

REALIZZAZIONE PRATICA

Passiamo ora alle note costruttive, iniziando con lo shield, il cui circuito stampato può essere ottenuto per fotoincisione partendo dalle tracce lato rame scaricabili dal nostro sito www.elettronica.in.it oppure potete realizzarlo utilizzando il servizio PCBPRODUCTION di Futura Elettronica.

Quanto al montaggio dei componenti, andate in ordine di altezza nel caso in cui non venga utilizzata la Maker Uno ma un Arduino Uno standard, è necessario usare headers più lunghi del normale altrimenti lo shield tocca sul connettore USB. Particolare attenzione va prestata al modulo Bluetooth: normalmente il verso di innesto è quello indicato nella foto: il pin RX del modulo deve entrare nel pad contrassegnato con TX e il pin TX in quello RX. I due pin esterni (STATE e KEY o EN) non sono collegati, quindi attenzione a non

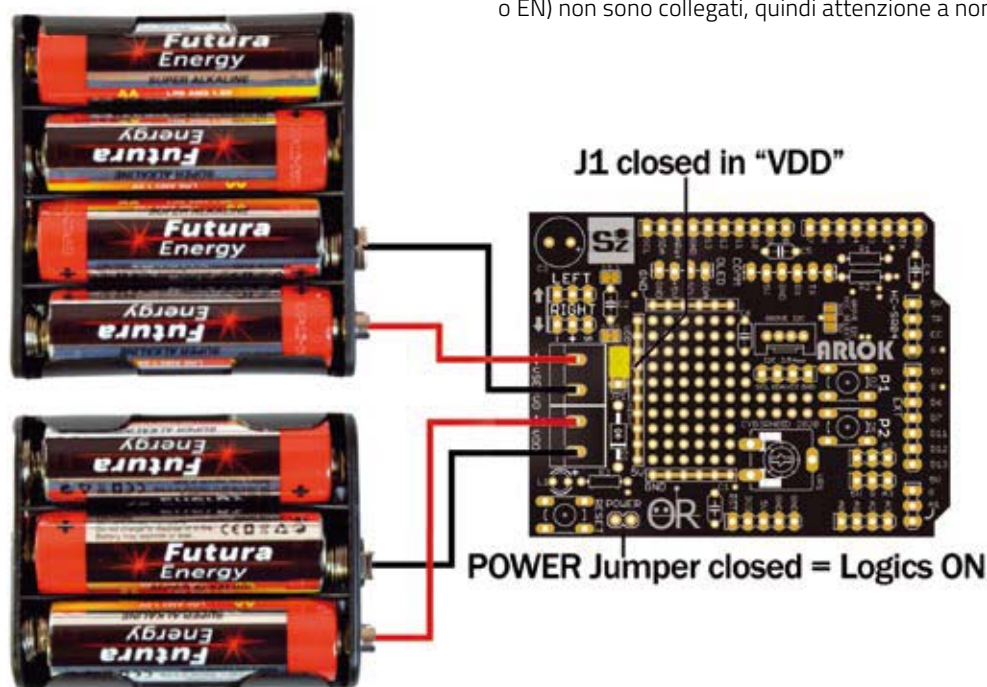
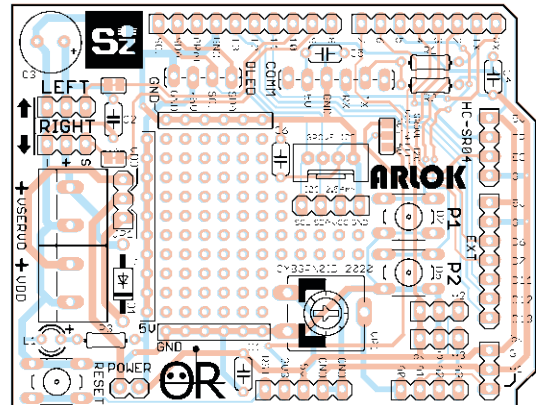


Fig. 7
Alimentazione
distinta per
la logica e gli
attuatori.



Elenco Componenti:

R1: 22 Kohm
R2: 15 Kohm
R3: 330 ohm
C1, C2, C4, C5, C6: 100nF
C3: 1000 uF 16V
LD1: led rosso 3mm
VR1: trimmer 10Kohm MO
D1: 1N4004

P1-P2: microswitch NO
RESET: microswitch NO

Varie:

- pinheader 10pin
- pinheader 8pin
- pinheader pin
- pinheader 4pin
- strip maschio 2 poli
- strip maschio 3 poli

- strip maschio 4 poli
- strip maschio 7 poli
- strip maschio 8 poli
- strip maschio 10 poli
- strip femmina 6 poli
- strip femmina 4 poli
- morsetto 2 poli p5
- jumper
- connettore grove
- Circuito stampato S1543 (69x54mm)

inserirlo al contrario. Dovete fare molta attenzione anche alla pin-out dei servocomandi. Nota importante: il sensore ultrasuoni deve stare tutto fuori: se le capsule vanno a filo con la staffa, le letture sono alterate e il robot comincia a rilevare ostacoli anche se non ci sono. Sul PCB sono riportati anche due header (EXT1 e EXT2) con i pin rimasti liberi. In particolare l'header EXT1, sotto al connettore per ultrasuoni, è per un eventuale modulo aggiuntivo **line-follower** per cui ci sono 3 I/O da sfruttare per il sensore di linea centrale e i due laterali: utilizzando il sensore makerline della Cytron, oltre ai PIN digitali per il rilevamento della linea è possibile, in alternativa, utilizzare un unico ingresso analogico. Gli ingressi analogici, tranne lo zero che è occupato dal trimmer, sono riportati su 3 headers separati, ognuno dotato anche della linea 5V e GND per facilitare il montaggio di eventuali sensori analogici. Il trimmer collegato ad A0 servirà nel firmware di esempio per la regolazione del punto zero dei servocomandi. Nella parte centrale del PCB c'è una utile area millefori che si presta a modifiche e aggiunte. Se intendete utilizzare gli ingressi analogici A4 e A5, sappiate che sono condivisi con SDA e SCL. Per la MakerUNO bastano degli header maschio

normali (quelli da 11mm): in questo caso utilizzate lo stesso header standard maschio da 40 pin indicato sopra (per i servo e JP1) e avrete cura, nella zona dell'header alimentazioni, di montare un pezzo da 6 pin anziché da 8 lasciando liberi i pin *NC* e *Vin*, che si trovano alle estremità e non servono.

IL FIRMWARE

Passiamo adesso all'analisi dello sketch installato nella scheda MakerUNO per gestire ARLOK. Prima di tutto è necessario installare su Arduino IDE alcune librerie. Basta andare in *Strumenti > Gestione librerie* scrivere il nome della libreria (in grassetto nella lista che segue), premere Invio per cercare e cliccare sul pulsante *Installa* nella casella dove compare la libreria precisa. Dato che possono esserci varie librerie con nomi simili, leggete bene quali sono gli autori, riportati in corsivo:

- **Adafruit SSD1306** di *Adafruit* (questa ha bisogno di una libreria aggiuntiva che verrà installata automaticamente dopo un pop-up che vi richiede il permesso di farlo);
- **Adafruit GFX** di *Adafruit*
- **TimerOne** di *Jesse Tane, Jérôme Despatis, Michael Polli, Dan Clemens, Paul Stoffregen;*

Bisogna installare ancora un'altra libreria che non compare nella lista di Arduino IDE e dovremo farlo a mano: andiamo all'URL <https://github.com/nabontra/ServoTimer2>, clicchiamo sul pulsante *Clone or download* e quindi *Download ZIP*. Estratto il file ZIP si genera la cartella *ServoTimer2-master*: rinominiamola in *ServoTimer2*. Copiamo questa cartella nella cartella delle librerie Arduino in documenti (per Windows: Documenti/Arduino/libraries). Dopo averla copiata, giusto per sicurezza, controllate che all'interno di questa cartella ci sia una cartella *examples*. Se l'IDE era aperto, dobbiamo chiuderlo e riavviarlo per fargli vedere questa nuova libreria. Ora è possibile scaricare il programma di esempio per far muovere ARLOK, dal sito <https://github.com/settorezero/arlok>. Copiate solo il codice e incollatelo nella finestra di Arduino IDE. Selezionate la scheda che state utilizzando e provate solo a compilarlo senza caricare (cliccate il tasto di spunta). Se la compilazione va a buon fine, siete già a buon punto, altrimenti vuol dire che non avete installato qualche libreria o avete selezionato la scheda sbagliata.

Funzionamento del codice

Partiamo dalla gestione del SONAR: questo viene controllato tramite interrupt per non bloccare il programma aspettando di fornirgli il trigger e controllare la risposta: in tutto questo tempo il robot potrebbe sbattere senza accorgersi dell'ostacolo. L'idea della lettura del SONAR tramite interrupt è di Steve Garrat. Per questo motivo abbiamo installato la libreria *TimerOne* che consente di impostare con facilità un interrupt per overflow sul *Timer1* con il quale gestiremo il triggering del sensore. Il *Timer* viene impostato per generare un interrupt ogni 50µS. Questo interrupt viene agganciato alla funzione **timer1_ISR**. Qui utilizziamo una macchina a stati ed è bene che vi ricordiate come funziona il sonar. La prima volta che scatta l'interrupt ci troviamo nello stato 1: viene mandato a livello alto il pin di trigger, si imposta un contatore **trigger_time_count** e ci si posiziona per lo stato 2. La seconda volta che scatta (dopo altri 50µS) ci troviamo quindi nello stato 2 e il pin di trigger viene portato a livello basso: l'impulso sul trigger è durato quindi 50µS, da datasheet deve durare almeno 10µS. Si torna allo stato zero, ovvero si fa nulla: quando **trigger_time_count** raggiunge lo zero si ricomincia daccapo inviando un nuovo impulso sul trigger.

Il contatore arriva a zero dopo 225mS, impostati da *TICK_COUNTS* ($4500 \times 50\mu S = 225000\mu S$). Nel frattempo l'impulso di ritorno viene monitorato da un pin al quale abbiamo agganciato un interrupt sul cambio di stato che abbiamo chiamato **sonarEcho_ISR**. In quest'altra funzione controlliamo se il cambio di stato del pin di echo è stato da livello basso a livello alto o viceversa; nel primo caso vuol dire che l'impulso è appena arrivato e la sua durata (cioè fino a che non ridiventa basso) sarà proporzionale alla distanza rilevata. Viene quindi memorizzato il valore del contatore **micros()** di Arduino.

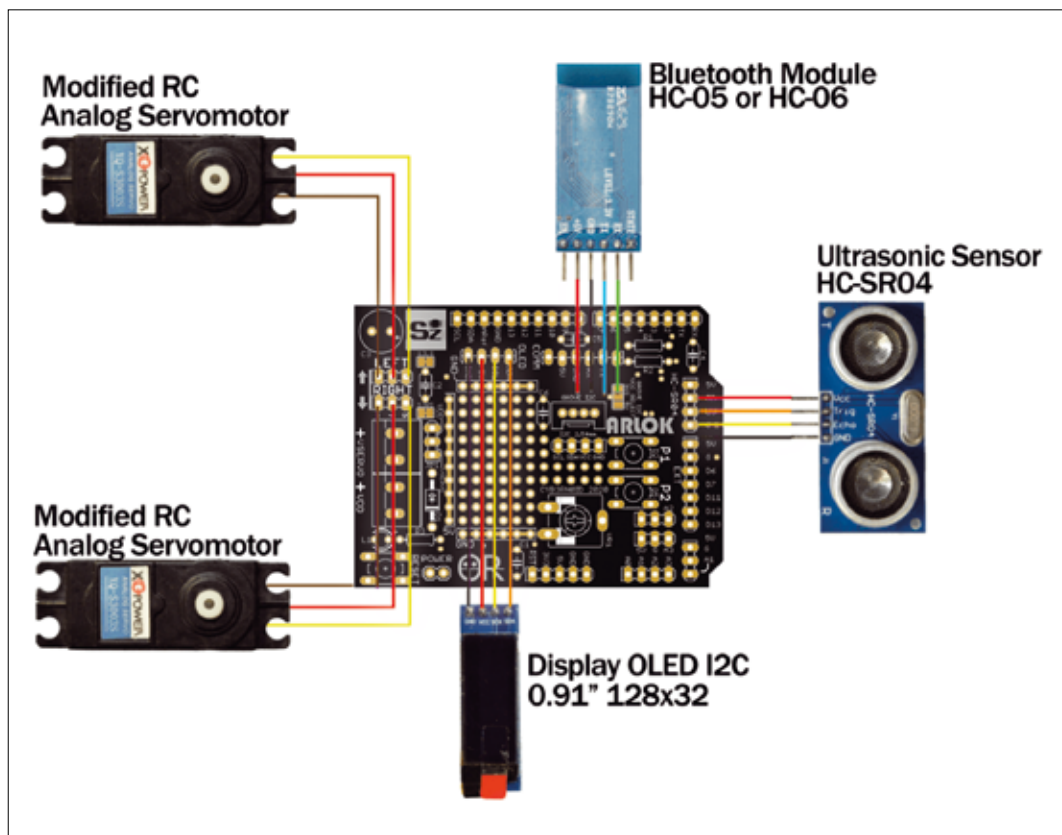
Dopo un po' scatterà un nuovo interrupt causato dal nuovo cambio di stato di echo da alto a basso: a questo punto dal contatore **micros()** attuale si sottrae il valore memorizzato prima e abbiamo quindi la durata dell'impulso di echo da cui ricaviamo la distanza. Il valore di distanza rilevato viene salvato in una variabile globale in modo che sia disponibile anche al Loop. Tutto questo deve essere eseguito nel tempo tra un trigger e il successivo. Per la gestione dei servo non abbiamo utilizzato la funzione standard di Arduino ma una modificata, *ServoTimer2*, che per la generazione del PWM per pilotare i servo si avvale del *Timer2* piuttosto che dell'1, che, come avete visto, stiamo sfruttando per gestire il SONAR e non possiamo quindi utilizzare per altro. Per generare l'interrupt potremmo usare una libreria che sfrutta il *Timer2* per generare l'interrupt e lasciar perdere la libreria alternativa per i servo. Questa libreria, diversamente da quella standard di Arduino, non accetta come parametro l'angolazione in gradi, bensì la durata in microsecondi dell'impulso di pilotaggio.

Il codice non fa altro che far andare ARLOK dritto (i due servo, dato che si trovano montati in maniera speculare, dovranno girare uno in un senso e l'altro nel senso opposto). Nel momento in cui la distanza rilevata dal SONAR è al di sotto di una certa soglia (parametro *OBSTACLE*), il robotino si ferma e con una funzione random sceglie di andare a destra o sinistra.

Il valore di *OBSTACLE* non deve essere troppo basso perchè c'è una certa lentezza di aggiornamento dovuta alla gestione del display, il valore attuale nel codice è un buon compromesso.

Lo stop del robot, però, non è immediato perchè ne causerebbe il ribaltamento in avanti per inerzia ma è stata prevista una rampa di decelerazione ottenuta giocando sugli impulsi di pilotaggio dei servo. Infatti tenendo 1.500 µS come valore di





servo in posizione centrale (fermo, nel nostro caso), al di sopra di tale valore il servo ruoterà in un verso e al di sotto nel verso opposto, ma più siamo vicini al valore di 1500 tanto più il servo girerà lentamente, altrimenti in condizioni normali (servo non modificato) non riuscirebbe a fermarsi in tempo per riportarsi precisamente al centro, per cui il comportamento normale è che la velocità di rotazione del servo diminuisce man mano che si avvicina al punto centrale. Il servo che abbiamo modificato crede di trovarsi sempre al punto centrale; c'è però da gestire il punto zero del servo, dal momento che non tutti i servo si fermano precisamente al valore di 1.500 μ s, specie questi che sono stati modificati. A ciò provvede la funzione di setup, per la quale avete bisogno di qualcosa su cui appoggiare il robotino in modo che le ruote siano libere di girare in aria senza toccare e di un piccolo cacciavite a taglio che entri nel cursore del trimmer VR1. Il setup è facilitato dall'utilizzo del display.

CONCLUSIONI

Il progetto ARLOK nella sua interezza è rilasciato sotto licenza Creative Commons BY-SA-NC 4.0,

ovvero siete liberi di condividerlo e farne modifiche a patto che sia indicato l'autore originale e le eventuali modifiche fatte, il progetto o qualsiasi sua parte non siano utilizzati a scopi commerciali e che la condivisione avvenga allo stesso modo di come l'ha fatta l'autore. □

Cosa occorre?

I componenti utilizzati in questo progetto sono disponibili presso Futura Elettronica. La board MakerUNO (cod. MAKERUNO) è in vendita a Euro 12,50, il modulo bluetooth (cod. HC05STRIP) è disponibile a Euro 9,00, il sonar (MISDIST04) costa Euro 5,00, il Ball-caster (cod. BCASTER) è in vendita a Euro 9,90, i servo a rotazione continua (cod. SERVO360LC) sono disponibili a Euro 9,90 cad. mentre i servo da modificare con ingranaggi in metallo (cod. MG996R) costano Euro 9,00 cad., il display OLED (cod. I2COLED091) è in vendita a Euro 5,90. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)
Tel: 0331-799775 - <http://www.futurashop.it>



IMPARA L'ELETTRONICA FACILMENTE CON TOTEM MINI LAB



€ 112,00
cod. TOTEMMINILAB

Piccolo ma potente laboratorio di elettronica ideale per sperimentare e apprendere le basi dell'elettronica. Pienamente compatibile con l'ambiente di programmazione Arduino, Totem Mini Lab comprende una varietà di strumenti di misura e blocchi elettronici pronti all'uso per realizzare esperimenti. Pensato per essere utilizzato insieme al sistema di costruzione Totem consente agli utenti di costruire solidi banchi da lavoro con breadboard integrate. Inoltre, il laboratorio è compatibile con schede di espansione I/O, sensori e audio che possono espandere ulteriormente le capacità del Mini Lab. L'utilizzo di Totem Mini Lab è semplice e divertente, ideale sia per ragazzi sia per adulti!

COMPLETALO CON LE SCHEDE DI ESPANSIONE!



ESPANSIONE AUDIO

Con questo pannello sarà possibile avere disponibili e facilmente accessibili i seguenti componenti: amplificatore audio in classe AB con 1 W di potenza, altoparlante integrato e alimentazione a 12 volt, VU-meter con barra a 10 LED, generatore di funzioni in grado di generare segnali fino a 1 MHz.

€ 39,50
Cod. TOTEMAUDIOSIDE



ESPANSIONE I/O

Contiene componenti elettronici che solitamente sono difficili da inserire sulle breadboard. Con questo pannello sarà possibile avere disponibili e facilmente accessibili i seguenti componenti: 3 interruttori a pulsante con indicatori LED, 3 potenziometri da 10 kOhm, 1 encoder rotativo, 3 pulsanti, 1 LED RGB, 1 presa Jack audio da 3,5 mm, 1 Relè da 10 A / 250 V, con LED di stato.

€ 37,00
Cod. TOTEMIOSIDE



ESPANSIONE SENSORI

Con questo pannello sarà possibile avere disponibili e facilmente accessibili i seguenti componenti: un microfono preamplificato, un display OLED 128x64 con interfaccia I²C, un sensore umidità DHT11, un termistore NTC da 10 kohm, un buzzer e un piccolo driver per motori DC.

€ 39,50
Cod. TOTEMSENSORSIDE

PULSIOSSIMETRO

WiFi

di BORIS LANDONI



Rileva battito cardiaco e percentuale di saturazione dell'emoglobina trasmettendo in wireless i valori, che verranno visualizzati sull'apposita app in smartphone e tablet.



P

er quanto esistano da parecchio tempo e, soprattutto nella pratica clinica siano molto utilizzati in varie discipline e in diversi reparti, i dispositivi per quantificare la saturazione dell'emoglobina contenuta

nel sangue e misurare il battito cardiaco sono praticamente finiti sulla bocca di tutti da quando è arrivata l'epidemia di COVID-19 o pandemia che dir si voglia. Tali dispositivi sono chiamati, quando integrati in uno solo (come nel caso di quelli optoelettronici applicati all'estremità delle dita), pulsiossimetri e sono l'insieme di un saturimetro (che intuitivamente rileva la cosiddetta "saturazione") di un misuratore della frequenza cardiaca. Hanno conquistato la ribalta sia mediatica, sia clinica proprio con il COVID-19 giacché una delle sue principali complicanze è l'insufficienza respiratoria e in questo quadro il saturimetro, che è un dispositivo medico in grado di rilevare il livello di ossigenazione del sangue gioca un ruolo determinante in quanto permette di comprendere la qualità degli scambi gassosi e l'efficienza dell'apparato respiratorio nel complesso. Ugualmente importante è la misura della frequenza del battito cardiaco, giacché un eccessivo aumento è spesso conseguenza del carente apporto di ossigeno alle cellule.

Dopo avervi introdotto la tematica con il progetto proposto nel fascicolo di maggio/giugno 2020, torniamo a parlare di pulsiossimetria in questo articolo, proponendovi uno strumento dalle analoghe funzioni, ma più avanzato perché non si limita alla visualizzazione locale dei parametri e della curva rilevati, ma invia tramite WiFi i dati, che quindi possono essere visualizzati su un'apposita app per smartphone e tablet, chiamata Blynk e appoggiata all'omonimo servizio web. Per rilevare battito e saturazione impiegheremo l'integrato Maxim IC MAX30102 (Fig. 1), che fun-



Si tratta di una piccola scheda a interfaccia I²C basata sul sensore per pulsossimetria MAX30102 in grado di misurare, in maniera non invasiva, la quantità di emoglobina legata nel sangue e pertanto la percentuale di saturazione dell'emoglobina.

La scheda analizza il sangue sfruttando due LED (uno a luce rossa alla lunghezza d'onda di 660 nm e l'altro a infrarossi emittente sugli 880 nm) con cui dirige la luce sul dito, quindi rilevando la luce riflessa mediante un fotodiode.

L'analisi della luce riflessa consente di determinare la percentuale di emoglobina saturata dall'ossigeno (per saturazione si intende quando l'emoglobina ha assimilato tutto l'ossigeno

possibile). La breakout board dispone inoltre di una funzione di prossimità per risparmiare energia e ridurre l'emissione di luce visibile quando il dito della persona da monitorare non è sul sensore.

Il modulo può essere spento tramite software.

In standby il consumo di corrente è quasi pari a zero (0,7 μ A).



zione analizzando la trasparenza dei vasi periferici del polpastrello del dito, investito da un raggio di luce di cui si rileva il riflesso o l'attraversamento del dito. La qualità della radiazione giunta al diodo rivelatore di luce fornisce indicazioni sul tenore di ossigeno, giacché più il sangue è scuro e meno è ossigenato e, viceversa, più è chiaro (rosso vivo) e maggiore è la percentuale di ossigeno. La saturazione si esprime in percentuale e quella massima ovviamente è il 100%, mentre sono buoni valori sopra 92-93%. Naturalmente la percentuale rilevabile dipende, oltre che dallo stato dei polmoni, anche dalla qualità dell'aria inspirata e dalla presenza di fattori che possono ridurre la disponibilità dell'ossigeno presente nell'aria.

IL PROGETTO

Per realizzare il pulsossimetro elettronico qui proposto, ci siamo avvalsi di un misuratore di ossigenazione del sangue e rilevatore del battito cardiaco a riflessione di luce su breakout board, basato sull'integrato MAX30102, ossia di questo integrato montato su una piccola scheda reperibile in commercio (da Futura Elettronica, www.futurashop.it) che ne facilita l'utilizzo; per l'esattezza parliamo di un sensore per "pulsossimetria" ossia in grado di misurare, in maniera non invasiva e combinata, la quantità di emoglobina legata all'ossigeno nel sangue ed anche la frequenza delle pulsazioni (battito) cardiache attraverso la variazione del calibro dei vasi sanguigni. Il sensore comunica via I²C-Bus i dati sulla saturazione ed anche quelli sulla pulsazione del sangue (che permetteranno di tracciare un grafico tipo elettrocardiogramma, nonché di desumere la frequenza del battito cardiaco) quindi abbiamo utilizzato una scheda compatibile con l'IDE Arduino (la vedete in **Fig. 2**) ovvero una board equipaggiata con la connettività WiFi e un display ideale per acquisire tali informazioni, elaborarle e trasformarle in un formato adatto a comandare sia il piccolo display OLED sul quale vedremo la frequenza del battito (BPM, ossia battiti per minuto o, se preferite i termini anglosassoni, Beats Per Minute...) e la percentuale di saturazione dell'emoglobina. Ma non solo, perché i dati elaborati verranno poi trasmessi attraverso l'interfaccia WiFi, come spiegheremo tra breve.

IL RILEVAMENTO DI OSSIGENO E BATTITO

Il sensore della Maxim IC già montato su breakout board è quello che vedete nel riquadro ad esso dedicato nella pagina accanto.



Il nostro pulsossimetro WiFi è composto da una breakout board basata sul MAX30102 interfacciata a un modulo ESP32OLED.



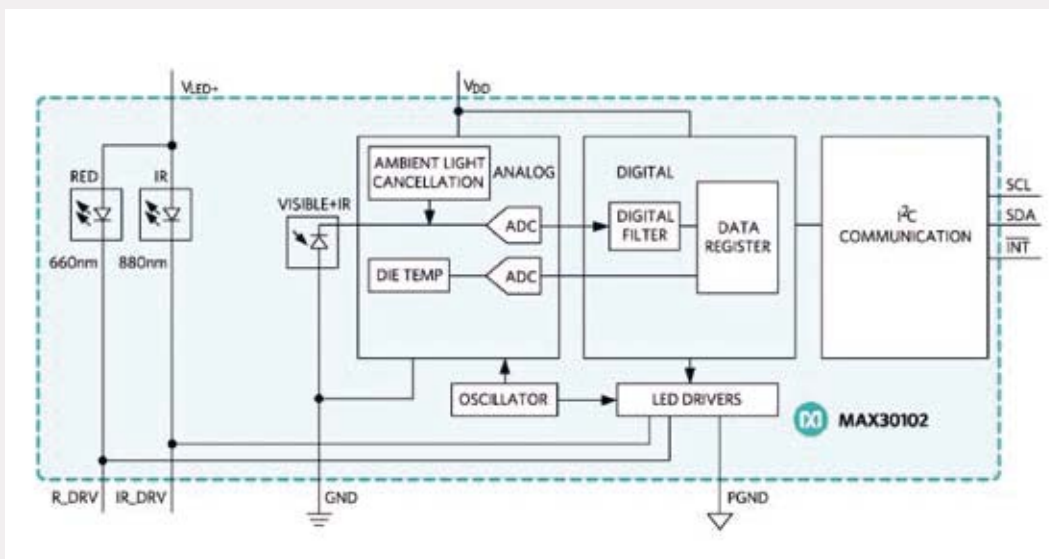


Fig. 1
Schema
interno
del
MAX30102.

A differenza della gran parte dei sensori per saturimetria e pulsiossimetria da applicare al polpastrello (che proiettano un raggio di luce sulla cute e rilevano l'illuminamento dalla superficie dell'unghia) il trasduttore del MAX30102 funziona emettendo verso l'alto del chip la luce prodotta da due LED: uno rosso (luce rossa alla lunghezza d'onda di 660 nm) e l'altro operante nell'infrarosso (emittente a 880 nm). Appoggiandovi sopra il dito, una parte della luce penetrata in esso per la trasparenza dei tessuti viene riflessa e l'ammontare della luce riflessa captato da un fotodiodo sulla stessa superficie da cui è emessa la luce combinata permette di determinare, sulla base della trasparenza e delle variazioni di trasparenza nel tempo sia l'andamento del flusso sanguigno, sia la percentuale di ossigeno in esso presente (la percentuale di emoglobina saturata dall'ossigeno). Nello specifico, il segnale elettrico prodotto dal fotodiodo viene inviato ad uno stadio amplificatore low-noise e condizionatore del segnale, quindi ciò che ne risulta entra in un blocco di analog signal processing che permette di estrarre i dati sull'ossimetria e sul battito cardiaco (riferitevi allo schema a blocchi interno dell'integrato). Quest'ultimo si ottiene analizzando le variazioni nell'andamento del flusso sanguigno attraverso la dilatazione e il restringimento dei vasi periferici determinati dalla pulsazione e dal rilascio dei ventricoli del cuore, che causano aumento e conseguente calo della pressione arteriosa. Il componente Maxim dispone inoltre di una funzione di rilevamento della prossimità, per risparmiare energia e ridurre l'emissione di luce visibile

quando il dito dell'utente non è sul sensore. I segnali che vengono digitalizzati dal blocco signal processing diventano quindi numerici e sono inviati sul bus I²C, dal quale vengono letti dalla nostra scheda ESP32OLED (questo il codice della board), il cui firmware, sfruttando una libreria specifica del sensore, può calcolare la frequenza cardiaca e la saturazione dell'ossigeno in valore percentuale.

I dati elaborati permettono di preparare i pacchetti di dati che seguiranno due strade: tramite opportuni comandi verranno visualizzati dal piccolo display OLED di bordo (che sarà la nostra interfaccia utente locale) che riporterà saturazione e battito; attraverso la connessione ed il protocollo WiFi saranno trasmessi verso l'apparato mobile nel quale sarà installata l'apposita app, la quale diverrà il remoto pannello sul quale vedremo il grafico dell'andamento della pressio-



Fig. 2
La scheda
ESP32OLED.

ne del sangue (quindi l'effetto della pulsazione del cuore) la frequenza cardiaca e la saturazione rilevati dal sensore.

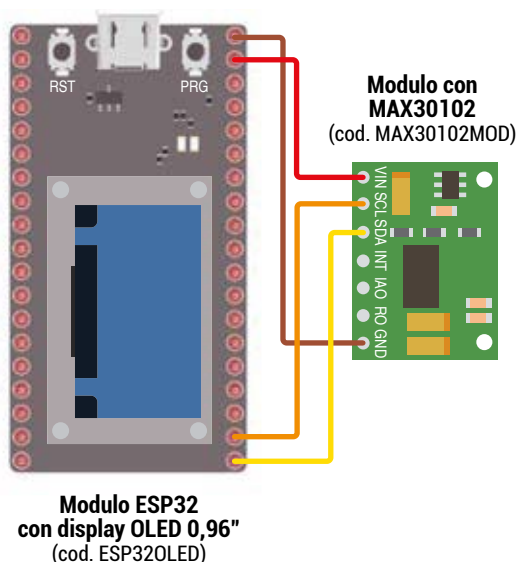
Il MAX30102 consente di impostare, attraverso il bus I²C e in fase di inizializzazione, il sample-rate e quindi il numero di campionamenti nell'unità di tempo che il converter analogico/digitale posto a valle dell'analog signal processor deve eseguire: una maggior frequenza determina un'analisi e una visualizzazione più accurata ma accresce il consumo di elettricità, mentre la riduzione della frequenza di campionamento (e perciò della precisione della misura) si traduce in una riduzione dei consumi; quest'ultima impostazione rappresenta un buon compromesso nelle applicazioni in cui si desidera realizzare un saturimetro portatile e ci si accontenta di una precisione accettabile per la gran parte delle situazioni pratiche.

IL NOSTRO CIRCUITO

Bene, ciò detto andiamo a vedere nel dettaglio da cosa è composto il sistema, ossia a descrivere le unità da cui è formato. L'hardware del progetto è dunque composto da:

- modulo ESP32 con display OLED 0,96" (3085-ESP32OLED);
- modulo breakout con MAX30102 (2846-MAX30102MOD);
- batteria al litio 1100 mAh 1s (singola cella).

schema di CABLAGGIO



La batteria ci permetterà di ottenere un pulsossimetro portatile che non richiede fili per il collegamento, peraltro più sicuro di uno alimentato con un alimentatore da rete.

Guardando lo schema di cablaggio vediamo che la breakout board su cui è montato il sensore Maxim si connette alla scheda ESP32OLED, la quale ha architettura Arduino ma dotata di modulo WiFi e di un piccolo display OLED blu da 0,96".

Il sensore viene connesso mediante due fili più la massa e l'alimentazione a 5V (prelevata rispettivamente da GND e il pin 5V della scheda ESP32OLED); le due linee sono SCL ed SDA del bus I²C.

Le linee SDA ed SCL sono open-drain, ragion per cui è opportuno prevedere le resistenze di pull-up, che nella breakout board sono da 4,7 kohm ma terminanti sulla linea di alimentazione interna. Per questa ragione, essendo la ESP32OLED alimentata internamente a 3,3 volt ed avendo gli I/O funzionanti a tale tensione (ma comunque capaci di leggere livelli logici 1 di valore più basso) non è necessario rimuovere i resistori di pull-up interni né aggiungere resistori esterni, dato che la nostra scheda legge correttamente i livelli logici 0/1,8V caratteristici della breakout board.

Questa scheda è una piattaforma di sviluppo basata sul chip ESP32 della Espressif, un microcontrollore Dual-Core Tensilica LX6 a 32 bit con una frequenza di clock di 240 MHz, una memoria SRAM interna da 520 kB, Wi-Fi 802.11 b/g/n, Bluetooth dual-mode (classico e BLE) e 28 pin I/O. Sulla scheda è presente anche una memoria flash da 32 MB, un'antenna WiFi su circuito stampato, un display OLED da 0,96", un caricabatteria per batterie ai polimeri di litio, un convertitore USB-seriale CP2102, due pulsanti (uno per il reset e l'altro per avviare la programmazione) e due LED di stato. Ideale per applicazioni IoT (Internet of Things), può essere programmata tramite l'IDE di Arduino.

Le caratteristiche della scheda sono le seguenti:

- display OLED monocromatico da 0,96";
- microcontrollore Dual-Core Tensilica LX6 32bit;
- frequenza di clock impostabile fino a 240 MHz;
- SRAM interna da 520 kB;
- memoria Flash da 32 MB;
- transceiver WiFi 802.11 b/g/n integrato;
- Bluetooth dual-mode integrato (classico e Low Energy);
- tensione di alimentazione da 3,3 a 7 Vcc;
- 28 GPIO (alcuni riservati per gestire il display OLED);

- caricabatteria per batteria ai polimeri di litio integrato, per 1 cella (3,7V);
- temperatura di esercizio da -40°C a +90°C.

Le connessioni da realizzare tra scheda la ESP320LED e la breakout board MAX30102 sono le seguenti:

- 3,3V > VIN
- GND > GND
- 22 > SCL
- 21 > SDA

Per le interconnessioni troverete molto comodo l'utilizzo di jumper (fili con pin o femmina) di quelli per Arduino; allo scopo di facilitare le connessioni, raccomandiamo di dotare sia la breakout board MAX30102MOD, sia la scheda ESP320LED di pin-strip maschi.

L'APP MOBILE

Per visualizzare i dati del pulsiossimetro WiFi utilizziamo una specifica app chiamata Blynk, la quale sfruttando la connessione wireless, estrae le informazioni e le presenta sulla sua interfaccia utente. Più esattamente, il nostro sistema si connette tramite WiFi a un access-point o router wireless dal quale ottiene l'accesso a Internet, allo scopo di pubblicare i dati sul servizio Blynk; poi l'app punta a quest'ultimo per ottenere i dati da visualizzare sull'interfaccia utente. Blynk (www.blynk.io) è una piattaforma IoT indipendente dall'hardware, che supporta app mobile white label e che si appoggia a un cloud privato; gestisce circa 400 piattaforme hardware che si possono rivolgere ad essa e supporta l'analisi dei dati ed esempi di machine learning. Blynk può controllare l'hardware di acquisizione da remoto, visualizzare i dati dei sensori collegati e memorizzarli. Le componenti principali sono:

- le **Blynk App**, ossia le app che consentono di creare le interfacce utente utilizzando i widget forniti nell'ambiente di sviluppo;
- il **Blynk Server** che è il responsabile di tutte le comunicazioni tra smartphone e hardware; può utilizzare il Blynk Cloud pubblico, ma è anche possibile eseguire un server Blynk privato, in locale; è open-source e potrebbe gestire migliaia di dispositivi (può anche essere eseguito su una scheda Raspberry Pi).
- le **Blynk Libraries**, che sono le librerie per tutte le piattaforme hardware più diffuse e abilitano la comunicazione con il server ed elaborano tutti i comandi in entrata e in uscita.

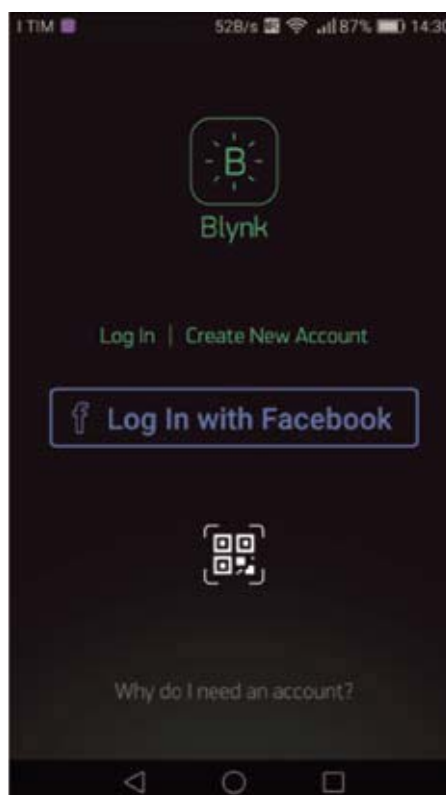
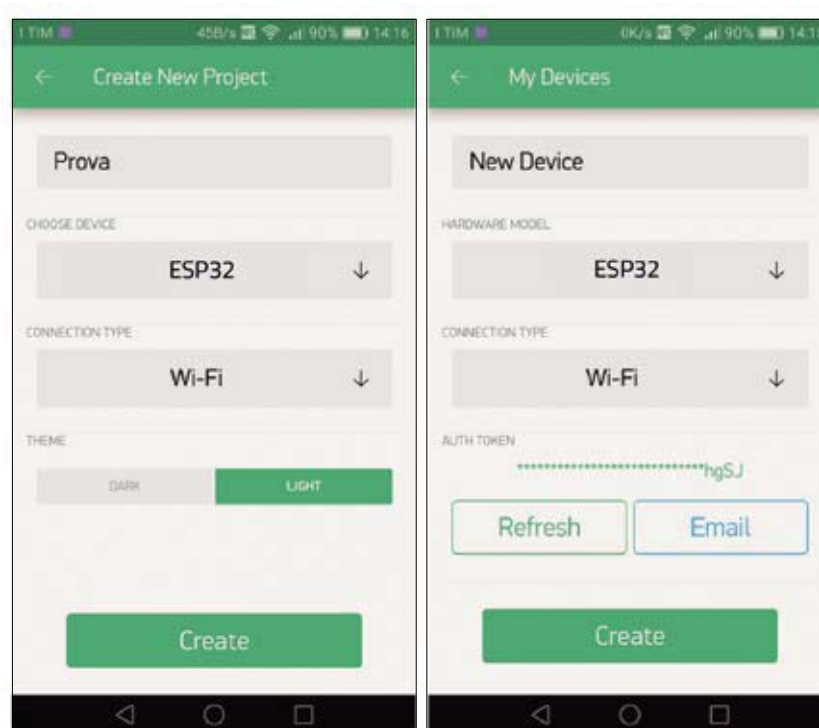


Fig. 3
Creare un account all'avvio di Blynk e scegliere l'hardware associato.

Ogni volta che si preme un pulsante nell'app Blynk viene generata una richiesta diretta al Blynk Cloud, dove la stessa viene elaborata e diretta tramite Internet all'hardware associato, che si trova in remoto. Lo stesso avviene quando un hardware pubblica dei dati, i quali vengono inviati all'app del dispositivo mobile associato. Per iniziare a lavorare con Blynk occorrono dunque un hardware completo di interfaccia wireless e uno smartphone o tablet Android. L'hardware è, nel nostro caso, la board ESP320LED con collegata la breakout board MAX30102 e per l'interfacciamento da utente sfrutteremo uno smartphone su cui installeremo l'app Blynk, che permette di generare interfacce utente sia su iOS che su Android.

Queste vanno quindi configurate per associarle, nella piattaforma, al nostro hardware; l'app non è altro che un "costruttore di applicazioni" per lo smartphone e permette di creare l'interfaccia utente come vedremo tra breve. Occorre poi scaricare la libreria Blynk per Arduino (ora alla versione 0.6.1) dal sito <https://github.com/blynk/blynk-library/releases/tag/v0.6.1>, e installarla nell'IDE di Arduino, quindi creare un nuovo

Fig. 4
Creare un nuovo progetto scegliendo il titolo, il device hardware e il tipo di connessione alla rete (WiFi); subito dopo si riceverà il Token per e-mail oppure lo si vede nella schermata.



account Blynk associandolo a un indirizzo e-mail reale (**Fig. 3**). L'account è necessario per salvare i progetti e avere accesso ad essi da più dispositivi da qualsiasi luogo ed è anche qualcosa che va a vantaggio della sicurezza, perché consente di avere il controllo sul proprio private Blynk Server eventualmente impostato.

Dopo aver effettuato l'accesso al proprio account, si può iniziare a creare un nuovo progetto.

L'APP PER ANDROID

Ogni progetto di Blynk ha quattro elementi base che si scelgono quando si inizia la costruzione di una nuova app (Create New Project), che possiamo riassumere con Project Name, Device, Theme. Project corrisponde al nome assegnato all'app, Device permette di scegliere la piattaforma da connettere (nel nostro caso la ESP32 Espressif) e con quale tipo di connessione (WiFi). Il Theme è il colore dello sfondo.

Stabiliti questi, si va a creare il progetto e viene reso disponibile il cosiddetto Token di autenticazione, il quale è un identificatore univoco necessario per collegare l'hardware allo smartphone; verrà inviato per e-mail una volta completata la creazione del progetto. Ogni nuovo progetto creato avrà quindi il proprio token di autenticazione. Oltre che automaticamente, il codice token

può essere copiato manualmente e inviato per posta elettronica cliccando sul pulsante "e-mail"; l'e-mail verrà inviata all'indirizzo utilizzato per la registrazione.

Nell'ambiente di editing dell'app, la riga in alto, oltre al nome che abbiamo dato all'app, ci sono quattro simboli: il primo da sinistra ci permette di spostarci tra le eventuali app già create o di entrare sulla pagina per creare una nuova app, oppure di uscire dal software. Il secondo simbolo (un esagono con un cerchio inscritto) è il pulsante che permette di variare i settings dell'applicazione che stiamo scrivendo (nome, condivisione di accesso all'app); da qui possiamo usare Shortcut, impostare l'hardware cui è connessa l'app, le informazioni sul token, il tema dell'app e l'eventuale sospensione dello schermo, notificare se l'app è on/off (con questa opzione abilitata il server invierà i comandi "App Connessa" e "App Disconnessa" all'hardware quando l'app Blynk è online/offline). Possiamo infine copiare il progetto oppure cancellarlo.

Il terzo simbolo (un + cerchiato) permette di aprire il menu dei widget disponibili (si può anche ottenere lo stesso effetto toccando con il dito in un punto qualunque dello schermo in fase di editing). Infine il triangolo è il comando di "esecuzione" dell'applicazione.

Dunque, una volta premuto il pulsante “Crea” per iniziare a costruire l’interfaccia utente ci troviamo sullo schermo lo sfondo del progetto vuoto, in cui aggiungere i widget desiderati; tra questi sceglieremo **gauge**, **SuperChart** e **LCD** che ci serviranno rispettivamente per visualizzare un indicatore dello stato di ossigenazione, un grafico del battito cardiaco e il rispettivo rate.

Ogni widget è un modulo che svolge una specifica funzione di input/output durante la comunicazione con l’hardware. Ci sono quattro categorie di widget, ma noi ne utilizziamo tre della categoria **Display**, nella quale rientrano quelli utilizzati per la visualizzazione dei dati da sensori e altre fonti; tra essi troviamo Led, caselle di testo ove appare la variabile da visualizzare, gauge, LCD (il classico LCD a una riga o a due righe), Terminal (una specie di visualizzatore di messaggi da/per l’hardware remoto), chart (SuperChart, cioè la possibilità di graficare le variabili in modi diversi), streaming di video e visualizzazione di immagini.

SuperChart ci permetterà di osservare l’andamento dei parametri del pulsiossimetro; notare che siccome i dati sono inviati su un cloud anche quando la nostra app è spenta, le variabili vengono comunque memorizzate (nel cloud) e quindi una volta accesa la app possiamo rivedere tutto quello che è successo nel periodo in cui l’app è stata off.

Una volta trascinato il widget nell’area di editing possiamo ridimensionarlo e toccando con il dito il widget, possiamo visualizzare i suoi settings: per esempio possiamo dare un titolo al grafico, ma anche collegare le variabili passate dall’ESP32-LED (si tratta di **Blynk.virtualWrite(V4,ESp02)** e **Blynk.virtualWrite(V5,beatAvg)** che trovate nel **Listato 1**). Questo si fa semplicemente premendo “Add datastream” che permette di scegliere quale variabile visualizzare usando Input, e settando il tipo di grafico, il suffisso (in questo caso %) e così via, in maniera molto intuitiva e semplice. Terminati i settings del widget SuperChart passiamo ad inserire un LCD a due righe 16 caratteri che ci permette di visualizzare le due variabili relative a saturazione e battiti per minuto. Anche qui impostiamo i parametri per farlo funzionare, indicando la frequenza di aggiornamento (è meglio usare il sistema Push, cioè aggiornare quando è disponibile) ed eventuali messaggi esplicativi sul display. Infine aggiungiamo il widget gauge, che conformemente alla **Fig. 5** posizioneremo in alto nell’interfaccia utente; fatti i settings anche di questo, abbiamo completato la nostra interfaccia.

A questo punto possiamo avviare l’app con il pulsante a triangolo in alto a destra.

IL FIRMWARE

Passiamo adesso allo sketch Arduino che permette alla board ESP32OLEAD di gestire il pulsiossimetro e dialogare con la piattaforma Blynk e attraverso essa con l’app. Lo sketch gestisce quindi l’interfacciamento con il sensore MAX30102 per acquisire i dati, la comunicazione WiFi per accedere alla rete wireless cui il nostro apparato si collegherà, nonché il protocollo di comunicazione (e autenticazione) con Blynk. Dunque, la prima parte dello sketch riguarda l’inclusione delle librerie:

```
#include "heltec.h"
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <Wire.h>
#include "MAX30105.h" //MAX3010X library
#include "heartRate.h"
```

La prima riguarda la gestione della ESP32OLEAD, la seconda e la terza sono per il supporto WiFi, mentre la *Wire.h* supporta la connessione dati alla breakout board MAX30102; il modulo pulsiossimetro viene gestito attraverso la libreria *MAX3015.h*, corredata dalla *heartRate.h* che estrae i relativi dati. Oltre a includere le suddette librerie, viene inclusa la libreria di Blynk relativa alla nostra ESP32OLEAD, ossia *BlynkSimpleEsp32.h*. Tale libreria permette di dialogare mediante WiFi e cloud Blynk con la nostra app, instradando i messaggi mediante l’utilizzo del token di cui siamo in possesso e che bisogna introdurre nello sketch attraverso la variabile `auth[]`:

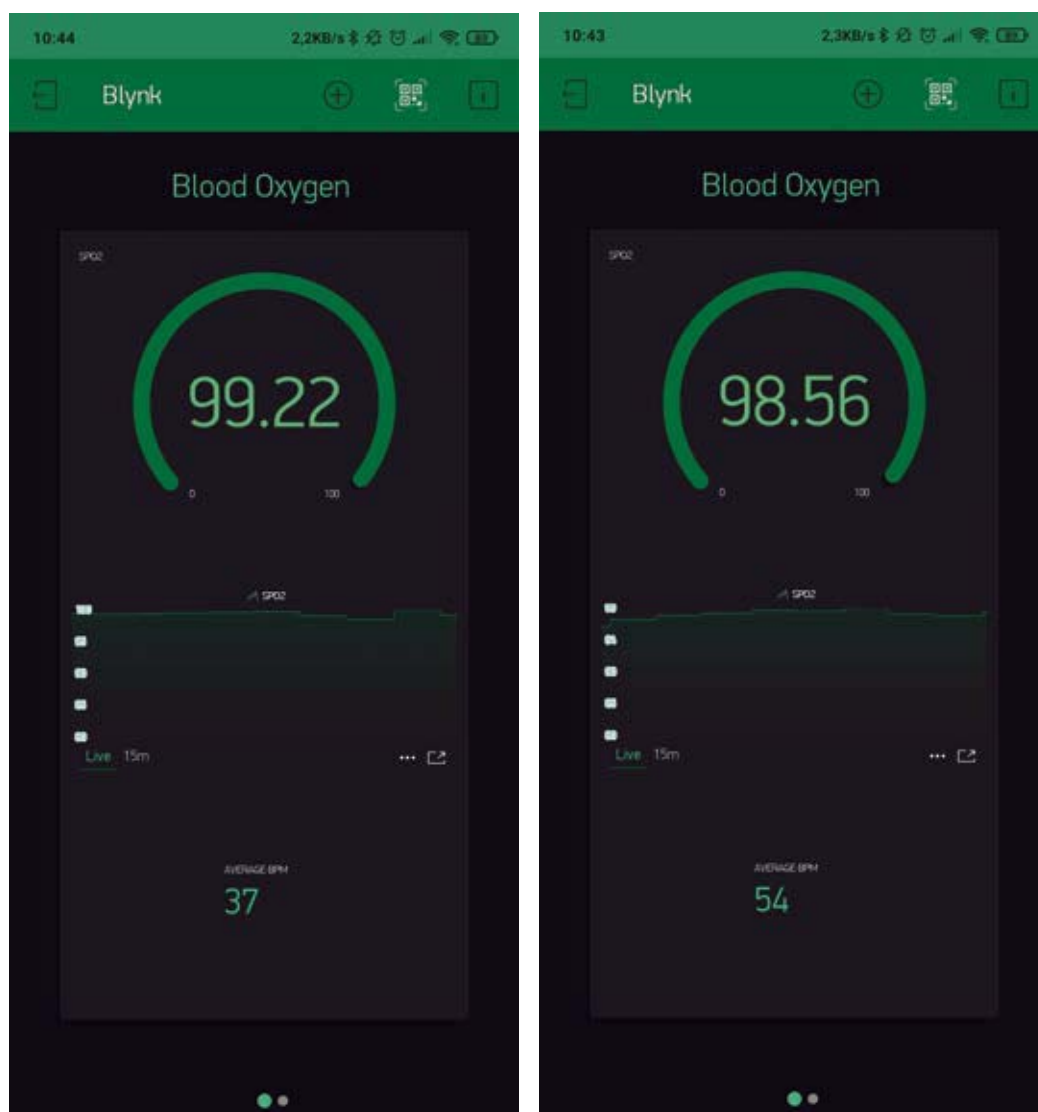
```
char auth[] = "u2JNsN7JOBRh-pKiNwXGaZY713IN1cyF"
```

In questo caso il token è l’insieme di caratteri alfanumerici compresi tra le virgolette e, come verrà spiegato più avanti, va introdotto nello sketch al punto indicato.

Nella parte iniziale sono definite le variabili in uso per la gestione del MAX30102; segue la parte di definizione del token, quindi quella di definizione delle credenziali di accesso alla rete WiFi cui collegheremo il nostro pulsiossimetro (// Your WiFi credentials) mediante la libreria di Blynk. La rete WiFi sarà quella dell’ambiente dove lo strumento verrà fatto funzionare e potrebbe, per esempio, essere anche quella creata dallo smartphone utilizzato



Fig. 5
Due esempi
dell'interfaccia
utente per
il pulsiossimetro:
vediamo dall'alto
verso il basso
il widget gauge
(quello che riporta
la percentuale di
saturazione) il
Superchart e l'LCD.



come Hot Spot, cosa che renderebbe il pulsiossimetro trasportabile ovunque ci sia quel telefono. Il firmware è strutturato in modo da inviare i dati al server Blynk con una periodicità tale da evitare il cosiddetto "flood error", che consiste "nell'inondare" di messaggi il server. Infatti se si inviano più di 100 valori al secondo al server, è possibile che si verifichi un "flood error" e l'hardware verrà disconnesso automaticamente dal server, perché considera tale comportamento simile ad una operazione di spamming, e quindi per evitare danni ad altri utenti, viene di fatto disconnesso quel collegamento. La parte di setup dello sketch contiene, oltre all'inizializzazione del sensore MAX30102, l'inizializzazione della libreria di comunicazione di Blynk, con `Blynk.begin(auth, ssid, pass)` che apre di fatto il canale WiFi, accede alla rete e invia il token di

riconoscimento al server.

Da questo punto in poi, se avete acceso il monitor seriale dell'IDE e abilitato l'istruzione `#define BLYNK_PRINT Serial` potete controllare se la scheda è stata autenticata e quindi è "nel sistema". Il firmware ora eseguirà il `void loop()` nel quale si vanno ad acquisire i dati dal sensore MAX30102 e poi si va ad eseguire la gestione della connessione Blynk. Il codice riportato nel **Listato 1** permette l'invio periodico dei dati letti dal sensore pulsiossimetro, che avviene ogni volta che `1/sampling` restituisce un valore multiplo; segue il tracciamento dei valori su Serial Plotter e l'invio su seriale. Nella seconda parte vengono stampati su seriale i valori letti dal MAX30102 riguardanti saturazione dell'emoglobina, temperatura del sensore, frequenza del battito, quindi si preparano le stringhe

▼ Listato 1

```
if ((i % SAMPLING) == 0) { //slow down graph plotting speed for arduino Serial plotter by thin out
  if ( millis() > TIMETOBOOT) {
    float ir_forGraph = (2.0 * fir - aveir) / aveir * SCALE;
    float red_forGraph = (2.0 * fred - avered) / avered * SCALE;

    //trancation for Serial plotter's autoscaling
    if ( ir_forGraph > 100.0) ir_forGraph = 100.0;
    if ( ir_forGraph < 80.0) ir_forGraph = 80.0;
    if ( red_forGraph > 100.0 ) red_forGraph = 100.0;
    if ( red_forGraph < 80.0 ) red_forGraph = 80.0;
    //      Serial.print(red); Serial.print(","); Serial.print(ir);Serial.print(".");

    if (ir < FINGER_ON) ESP02 = MINIMUM_SP02; //indicator for finger detached
    float temperature = particleSensor.readTemperature();

    Serial.print(" Oxygen % = ");
    Serial.println(ESP02);
    Serial.print(" Temp = ");
    Serial.println(temperature);
    draw(ESP02,beatAvg);

    Blynk.run();
    Blynk.virtualWrite(V4,ESP02 );
    Blynk.virtualWrite(V5,beatAvg );
  }
}
```

che vengono poi trasmesse in WiFi sfruttando i Virtual pin V4 e V5, cui sono inviate le variabili, rispettivamente ESP02 e beatAvg.

Una seconda porzione di codice sicuramente interessante è quella riportata nel **Listato 2**, che riguarda l'inizializzazione del display di bordo della scheda ESP32OLED (quindi il reset, il posizionamento del cursore, le font da utilizzare) e la visualizzazione su di esso (pertanto il monitor locale utile ad esempio quando l'app è off-line o si vuole vedere sul posto il risultato del monitoraggio) dei parametri forniti dalla breakout board del pulsiossimetro, vale a dire livello di saturazione dell'emoglobina e frequenza del battito cardiaco.

CONFIGURAZIONE DELL'IDE

Per poter caricare il codice nella scheda ESP32OLED che gestisce il pulsiossimetro è opportuno connettere il Personal Computer utilizzato per la programmazione con un cavo USB-A/microUSB alla ESP32OLED non ancora alimentata dalla batteria, aprire l'IDE Arduino per scrivere il programma e poi effettuare il download nella ESP32OLED. Ricordiamo che per poter programmare la scheda ESP32-OLED nell'IDE Arduino occorre prima caricare in quest'ultimo il driver di terze parti della scheda; allo scopo si apre l'IDE, si clicca *File > Preferenze* e nella finestra di dialogo cui si accede,

nell'apposita casella (URL addizionali per Gestore Schede) si scrive il seguente URL riferito al pacchetto ESP32:

https://resource.heltec.cn/download/package_heltec_esp32_index.json

Poi si impartisce il comando di menu *Strumenti > Scheda > Gestore schede* e dal relativo sottomenu, dove adesso apparirà anche la famiglia di schede cui appartiene la ESP32OLED, si sceglie quest'ultima. Per il resto si procede con il caricamento dello sketch come si farebbe con qualsiasi board Arduino



Cosa occorre?

Il materiale utilizzato in questo articolo è disponibile presso Futura Elettronica. La scheda con ESP32 e Oled (cod. ESP32OLED) è in vendita al prezzo di Euro 24,00, la breakout con il sensore MAX30102 (cod. MAX30102MOD) è in vendita al prezzo di Euro 7,50, la batteria al litio da 1100 mAh (cod. PLIB1100) è in vendita al prezzo di Euro 7,90. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)
Tel: 0331-799775 - <http://www.futurashop.it>



↓ Listato 2

```
void draw(double S,double B) {  
  
    Heltec.display->clear();  
    Heltec.display->setTextAlignment(TEXT_ALIGN_LEFT);  
    Heltec.display->setFont(ArialMT_Plain_16);  
    Heltec.display->drawString(0, 10, "SpO2");  
    Heltec.display->drawString(60, 10, (String)S);  
    Heltec.display->drawString(110, 10, "%");  
    Heltec.display->drawString(0, 40, "aBPM");  
    Heltec.display->drawString(60, 40, (String)B);  
    Heltec.display->drawString(110, 40, "");  
    Heltec.display->display();  
}
```

e una volta completato si può sconnettere la board dal Personal Computer.

Ricordiamo che è necessario andare a sostituire nello sketch che trovate nel nostro sito, alla riga corrispondente alla dichiarazione del token:

```
char auth[] = "u2JNsN7JOBRh-pKiNwXGaZY713IN1cyF"
```

il testo virgolettato (ossia il codice alfanumerico tra le virgolette) con il codice token che vi ha generato Blynk all'atto della creazione del progetto.

Una volta programmata la scheda ESP32OLED si può sconnetterla dal PC, quindi connetterla alla batteria al litio: dopo qualche istante il pulsiossimetro sarà pronto a funzionare.

CONCLUSIONI

Il progetto qui descritto rappresenta un'evoluzione del classico pulsiossimetro da dito, perché

non si limita a rilevare e visualizzare localmente i parametri vitali, ma ne consente l'invio in wireless a un dispositivo remoto come un tablet o uno smartphone, quindi permette il monitoraggio a distanza.

Oltretutto l'essersi appoggiati a un servizio come Blynk, che dispone di un proprio Cloud, permette anche di registrare i dati quando l'app è off-line (magari perché lo smartphone è fuori copertura dati di rete mobile o WiFi) e riceverli via notifica Push quando la si riporta on-line.

L'app, oltre che su smartphone, può essere creata anche su un tablet, per disporre a questo punto di uno spazio più ampio a schermo in cui, volendo, mettere più widget dei tre che vi abbiamo descritto in questo articolo.

Dunque, non ci resta che augurarvi buon lavoro, soprattutto se volete cimentarvi nella personalizzazione del progetto. ■

Elettronica In

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

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

COLLABORATORI HARDWARE della nostra Rivista

Realizzare un progetto per Elettronica In è molto semplice!

Dopo aver concordato con la redazione le linee guida del progetto, ti invieremo il materiale necessario con quale farai le prove e realizzerai un primo prototipo, anche su breadboard. Successivamente ci manderai il disegno dello stampato: alla sua realizzazione pratica provvederemo noi che ti rimanderemo la basetta ed i componenti per realizzare il prototipo finale, quello che poi andremo a fotografare e che presenteremo sulle pagine della rivista. Agli schemi elettrici definitivi provvederemo sempre noi mentre a te non rimarrà altro che descrivere il progetto nello stile della rivista.

Manda una mail a: redazione.hardware@elettronicaain.it

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

LED COB

LIGHTS

**100 LUMEN
WATT**



Realizza lettere, numeri per pannelli pubblicitari, illumina al meglio i prodotti nella tua vetrina e tanto altro con le barrette a LED COB. Ricoperte da gomma siliconica sono realizzate su supporto in alluminio per una migliore dissipazione di calore. Utilissime per svariati utilizzi!



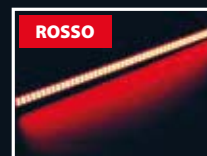
Cod. LEDCOB200B



Cod. LEDCOB200W



Cod. LEDCOB200V



Cod. LEDCOB200R

BARRETTE DI LED COB 12V-15W LUNGHEZZA 20 CM

Tipo di LED: COB chip
Angolo del fascio: 120°
Flusso luminoso: 100 lumen/watt
Vita media: 50.000 ore
Alimentazione: 12 VDC
Potenza: 15W
Consumo: 900 mA
Temperatura di funzionamento: da -20°C a +60°C
Dimensioni: 200x10x3 mm
Peso: 11 grammi

€5,90

A BARRETTA

**Collega
insieme più
barrette
di colori
differenti!**

Prezzi IVA inclusa.

Alimentale con:

**MAX 1
barretta di
LED COB**

ALIMENTATORE SWITCHING
12V - 15W - 1,25A

€5,00

Cod. POS15W12V



**MAX 2
barrette di
LED COB**

ALIMENTATORE
SWITCHING
12V - 36W - 3A

€8,50

Cod. ADL-40-12



**MAX 6
barrette di
LED COB**

ALIMENTATORE
SWITCHING
12V - 100W -
8,5A

€20,00

Cod. MW10012



Controlla tutto tramite App, Alexa, IFTTT e Google Assistant!



Interruttore Wi-Fi

Consente di attivare e disattivare un carico (luci o dispositivi elettronici) funzionante a 220 VAC con un assorbimento massimo di corrente di 10 A, come luci, ventilatori, stufe elettriche e tutto ciò che funziona con la rete domestica. Può essere controllato tramite smartphone o tablet, installando l'App "Smart Life" o "TUYA Life" per iOS o Android, oppure può anche tramite Amazon Alexa, Google Assistant e IFTTT.

€7,90

Cod. FR775

Installabile in scatole di derivazione o porta frutto



€19,90

Cod. FR776



Prezzi IVA inclusa.

€24,90

Cod. FR778



Telecomando Wi-Fi, IR

Dispositivo IR e Wi-Fi che permette tramite l'App "Smart Life o Tuya Life" di trasformare il vostro smartphone/tablet in un telecomando in grado di sostituire tutti i telecomandi IR utilizzati per controllare i dispositivi comunemente presenti in casa (condizionatori d'aria, TV, ventole, DVD e molti altri). Può essere controllato sia tramite App (Smart Life o Tuya Smart) sia tramite Amazon Alexa o Google Assistant.



WORKS WITH
amazon alexa

WORKS WITH
Google Assistant

WORKS WITH
IFTTT

Presa smart Wi-Fi

Controllabile con App, Amazon Alexa, Google Assistant e ifttt, per accendere/spegnere qualsiasi dispositivo ad essa collegato. Dispone di funzione timer, sistema per il monitoraggio del consumo di energia elettrica. La presa è in grado di supportare un carico complessivo di 3400W e una corrente massima di 16A.

Micro interruttore Wi-Fi a 1 canale

Modulo interruttore compatto Wi-Fi che consente di attivare e disattivare un carico (luci o dispositivi elettronici) funzionante a 220 VAC con un assorbimento massimo di corrente di 10 A. Può essere controllato sia tramite App (Smart Life o Tuya Smart) installata su smartphone, tablet, ecc. sia tramite Amazon Alexa, Google Assistant e IFTTT. Il modulo viene fornito completo di supporto rimovibile per guida DIN.

Disponibile anche a 2 canali con corrente massima di 5A per canale
cod. FR777 **€29,90**



€21,90
Cod. FR779

PI-HOLE, UN SERVER DNS INDOOR

di EMANUELE SIGNORETTA

Rendiamo più sicura la connessione domestica bloccando alcuni siti web inappropriati per i bambini, gli elementi traccianti e le pubblicità.

Sempre più spesso apprendiamo attraverso i tabloid o i telegiornali di giovanissimi che vengono adescati sui Social, partecipando a giochi potenzialmente letali (tipo la Blue Whale Challenge) o comunque pericolosi a livello psico-fisico. È uno dei numerosi rischi che si corrono lasciando "libero" accesso ad Internet a coloro che non sono ancora in grado di individuare una situazione di potenziale pericolo. Ma anche le persone più esperte non sono al sicuro quando navigano in rete, perché quotidianamente dei malintenzionati escogitano mezzi per "delinquere" informaticamente. Prima di incorrere in fraintendimenti è indispensabile specificare che il Web è uno strumento di vitale importanza e di cui non possiamo farne a meno, ma come per tutti i frutti dell'ingegno umano, è altresì indispensabile conoscerne le insidie e sapersi non solo difendere, ma anche prevenire certi rischi. Uno tra questi è evitare che i nostri dati personali vengano

Total queries (3 clients)

2,084

203

Percent Blocked

9.7%

Domains on Block

111,47

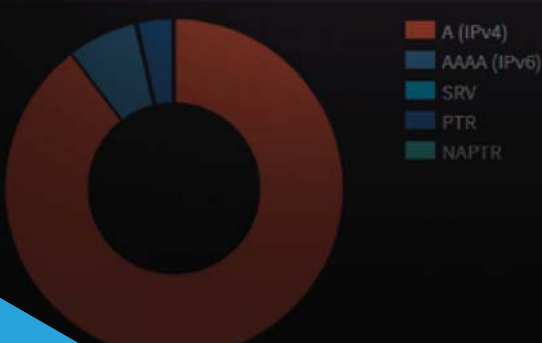
Total queries over last 24 hours



Client activity over last 24 hours



Query Types



Queries answered by

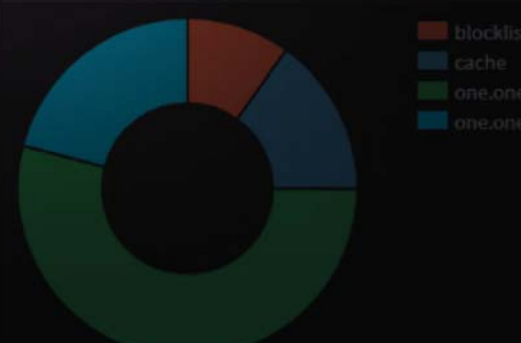
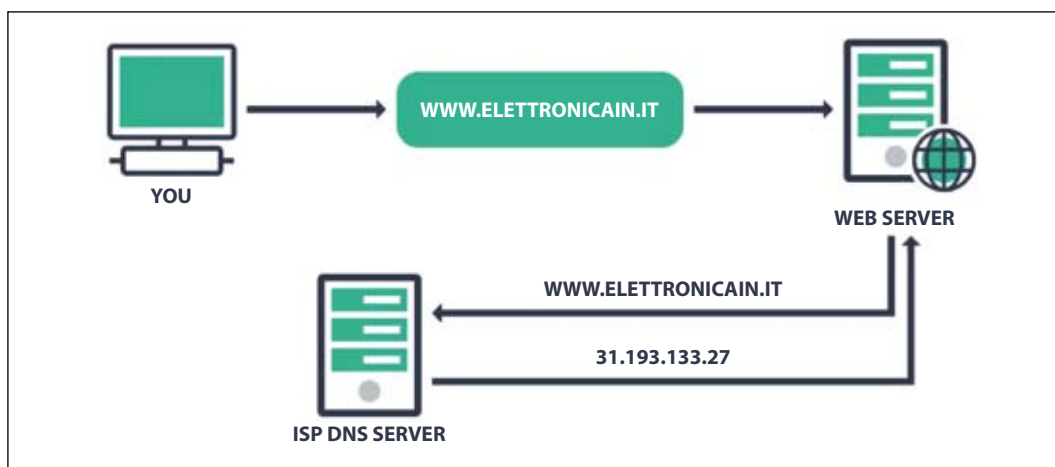


Fig. 1
Schema di funzionamento di un server DNS.



inconsapevolmente diffusi in rete e utilizzati per finalità di profilazione o, peggio ancora, per finalità illecite. Non è possibile sintetizzare in una manciata di pagine i metodi volti a mitigare la raccolta o il furto dei dati personali, ma comunque possiamo spiegare come bloccare gli elementi traccianti. Un

celebre proverbio infatti afferma: “Se un prodotto è gratis, allora il prodotto sei tu”. In effetti è vero: iscrivendoci gratuitamente a piattaforme Social Media, e-commerce, ecc. i nostri dati vengono raccolti ed elaborati da specifici algoritmi. Lo scopo finale di quest’ultimi è creare un “profilo” di

IL DOMAIN NAME SERVICE



Fig. 2
DNS utilizzato per il servizio virtual hosting.

Il Domain Name Service (acronimo di DNS) è un sistema ideato nel 1983 che permette di associare un nome agli host della rete. I vantaggi dell'utilizzo del DNS sono molteplici. In primis non è necessario ricordare mnemonicamente

l'indirizzo IP di un sito, ma semplicemente il nome. È anche possibile associare svariati servizi ospitati su più nomi allo stesso server. In caso di guasto o manutenzione è sufficiente cambiare i record DNS per indirizzare le richieste da un server ad un altro. Uno svi-

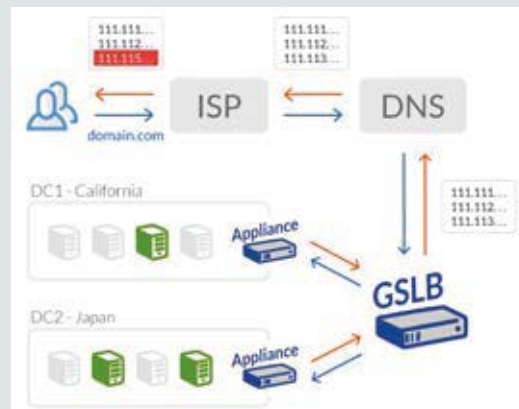


Fig. 3
DNS utilizzato per il load balancing.

luppatore può infatti realizzare ex-novo una nuova versione di un sito web su un altro server, e una volta completate le fasi di test può far entrare in funzione il nuovo sito senza causare un'interruzione del servizio. Il virtual hosting è un modo utilizzato per ospitare più siti

web, sullo stesso web-server, con la stessa interfaccia di rete, e quindi sullo stesso IP. Il processo è molto ingegnoso: il server DNS analizza l'header HTTP, individuando l'indirizzo alfanumerico e reindirizzando il client al sito “virtuale” per cui viene fatta la richiesta (Fig. 2).

ciascuno di noi in modo tale da poterci indirizzare una pubblicità mirata (basti pensare che l'86% dei proventi di Google arriva dalla pubblicità). In questo articolo spiegheremo come configurare la piattaforma open-source Pi-Hole al fine di bloccare elementi traccianti, pubblicità e siti dannosi all'interno di una rete locale.

COME FUNZIONA INTERNET

A titolo molto esemplificativo possiamo dire che la rete informatica è fatta da client e da server. Ogni client e ogni server, ma in generale ogni dispositivo, è identificato univocamente, all'interno della rete di appartenenza, da un indirizzo IP. Quando tentiamo la connessione ad un sito web (supponiamo sia *www.elettronica.in.it*), ci connettiamo all'indirizzo IP di quel sito (31.193.133.27). In automatico dunque, un meccanismo "traduce" il "nome" della risorsa richiesta nel corrispondente indirizzo IP. Il nome formale di questo processo è detto risoluzione DNS. Il DNS (o Domain Name System) è un registro universale, che contiene tutti i nomi di dominio

e li associa ai relativi IP pubblici; ad assumersi questo onere sono i server DNS. Ne esistono a migliaia, ma solo alcuni tra questi si distinguono per efficienza, velocità e stabilità: Google, Cloudflare, OpenDNS ecc. Ad ogni connessione, i server DNS ricevono la richiesta inviata dal client, interrogano il database e ci restituiscono l'indirizzo IP del server. A questo indirizzo il client tenterà la connessione. Uno schema riassuntivo di tale "meccanismo" è visibile in (Fig. 1).

CHE COS'È PI-HOLE

Pi-Hole è un software open source installabile sulla stragrande maggioranza delle distribuzioni Linux. Sul sito del progetto viene definito come un "buco nero per i server DNS". Concretamente, Pi-Hole è un normale server DNS configurato per dare dei falsi risultati per gli indirizzi inseriti in una blacklist; invece le richieste rivolte a un indirizzo non inserito nella blacklist verranno inoltrate al server DNS principale. Può essere installato anche su container Linux (creati con Docker).

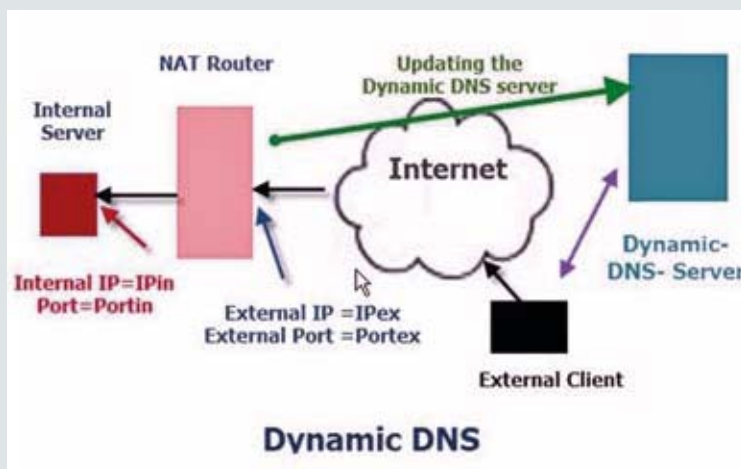
Un'altra tecnica che sfrutta i server DNS è il DNS Load Balancing. Questo sistema serve a distribuire un grosso carico computazionale su più server.

In caso di guasto di uno o più "nodi" di questo cluster, il sistema sopprime automaticamente la mancanza di quest'ultimi e trasferisce in modo più o meno omogeneo le richieste a quelli ancora in funzione (Fig. 3). Una delle funzionalità più usate dagli Hobbisti è senz'altro il DDNS (Dynamic DNS): la connessione domestica dispone di un indirizzo IP pubblico che, essendo dinamico, non rende semplice la gestione della connessione ai dispositivi smart. Immaginiamo di ospitare un web-server su un Raspberry Pi o su un Fishino.

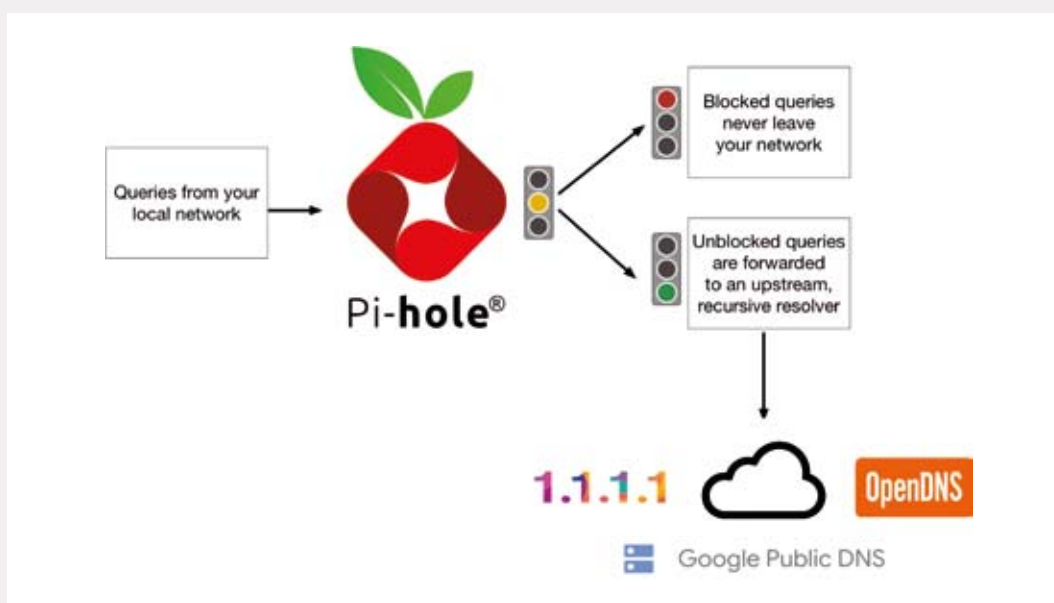
Il modo migliore per potersi connettere è aggiornare costantemente l'indirizzo IP di un normale DNS. Se l'indirizzo IP di casa è inizialmen-

te 1.2.3.4, e successivamente cambia in 2.3.4.5, noi possiamo sempre raggiungerlo all'indirizzo *test.example.com* (Fig. 4).

Fig. 4
Utilizzo del servizio DDNS.



➔ **Fig. 5**
Schema di
funzionamento
di Pi-Hole.



Per maggiori dettagli è possibile visitare il link Internet <https://docs.pi-hole.net/>. Lo schema di funzionamento del software è proposto nella **Fig. 5**.

L'HARDWARE CON CUI SPERIMENTIAMO

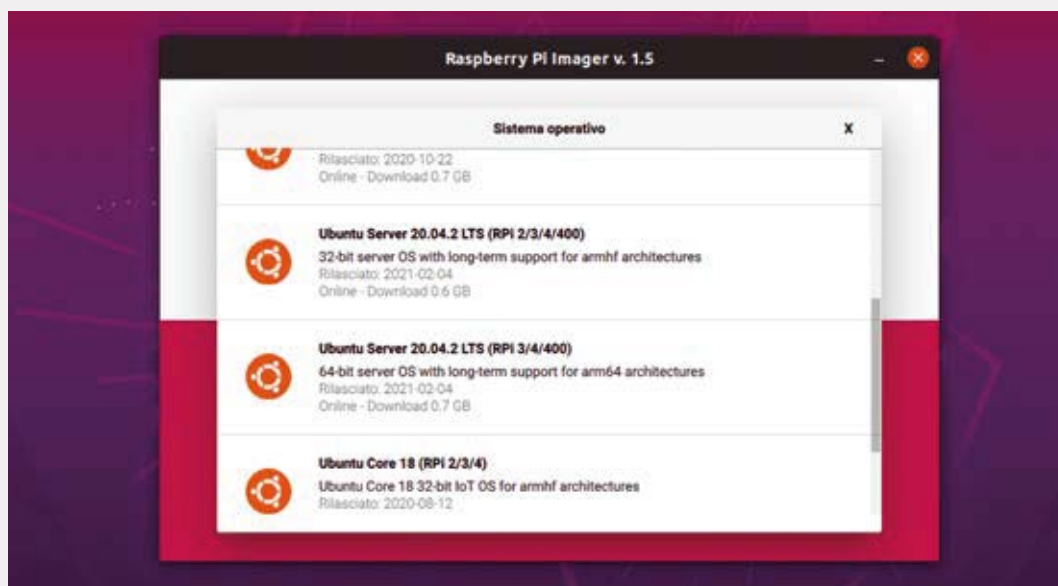
Per realizzare questo progetto è sufficiente utilizzare una Raspberry Pi 4 con 2 GB di RAM, connessa alla rete. A complemento della Raspberry Pi è richiesto altro hardware: i dissipatori, il case, una ventolina a 5V, una scheda microSD e l'alimentatore. Tutto il materiale è già disponibili in kit presso Futurashop (cod. RPI4FULLG2). Utilizzeremo anche

un cavo di rete di categoria 5e o superiore per collegare la Raspberry Pi alla rete LAN. È possibile collegarlo utilizzando la connessione WiFi, ma viste le esigue dimensioni occupate e il rischio interferenza pari a zero, optiamo per l'opzione cablata.

PREPARAZIONE DELLA SCHEDA SD

Prima di configurare Pi-Hole sulla Raspberry Pi dobbiamo preparare la scheda micro-SD che andrà inserita in essa. È consigliabile utilizzare una scheda da 32 GB, sebbene anche una da 16 GB sia già sufficiente. Inseriamola nel PC e avviamo il tool Raspberry Pi Imager (reperibile in Internet

➔ **Fig. 6**
Scelta del Sistema
Operativo da
flashare sulla
scheda SD, in
Raspberry Pi
Imager.



al link www.raspberrypi.org/software/ e disponibile per Windows, Linux e Mac). All'avvio, la schermata proposta riporterà i principali comandi e funzioni disponibili; qui facciamo dunque clic su Scegli SO→Other general purpose OS→Ubuntu e accederemo a una schermata simile a quella proposta in **Fig. 6**, dove troviamo le versioni di Ubuntu Linux disponibili.

Nel nostro caso abbiamo scelto Ubuntu Server 20.04.2 LTS a 64 bit, ma nulla ci vieta di selezionare una versione differente per architettura o per presenza dell'ambiente grafico. Una volta scelto il Sistema Operativo verremo riportati nella pagina principale. Selezioniamo la voce "Scegli scheda SD" e verremo rediretti in un menù che ci permetterà di scegliere la scheda SD su cui verrà installato il sistema operativo (**Fig. 7**). Una volta eseguita la scelta torneremo di nuovo nella schermata principale. Clicchiamo su "Scrivi" e nel menù contestuale che si aprirà clicchiamo su "Sì".

A questo punto il software scaricherà dalla Rete il file immagine corrispondente e provvederà ad installarlo sulla scheda.

Al termine dell'installazione, un apposito pop-up ci informerà che l'operazione è andata a buon fine e che possiamo rimuovere la scheda dal Personal Computer.

INSTALLAZIONE DEL SOFTWARE E PRIMA CONFIGURAZIONE

Una volta preparata la scheda SD, inseriamola nello slot dedicato della Raspberry Pi e colleghiamola al cavo di rete e al cavo di alimentazione. Poiché abbiamo scelto un'installazione senza ambiente grafico, non è necessario collegare un monitor. Per il primo avvio è necessario individuare l'indirizzo IP della Raspberry Pi. Per fare ciò è sufficiente collegarsi alla pagina di configurazione del proprio gateway oppure utilizzare appositi tool, quali Advanced IP Scanner, Fing, ecc.

Una volta che è stato determinato l'IP address possiamo collegarci in SSH al single board computer. È possibile scaricare tool da internet come ad esempio Putty, Teraterm, MobaXTerm, o, su dispositivi UNIX si può utilizzare il terminale nativo. Per il primo accesso nome utente e password coincidono: sono infatti "ubuntu". Quando connessi, ci verrà richiesto di cambiare la password e quando cambiata dovremo connetterci nuovamente. Come prima cosa lanciamo il comando `sudo apt-get update && sudo apt-get upgrade -y`. Questi comandi vengono utilizzati per aggiornare i pacchetti già presenti nell'installazione Ubuntu e portarli all'ul-

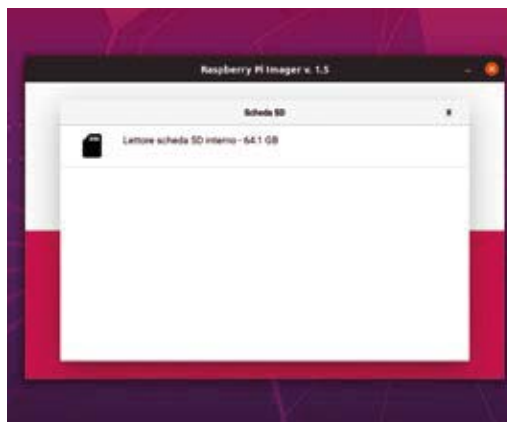


Fig. 7
Sceita del supporto su cui installare il Sistema Operativo.

tima versione. Lo step successivo è assegnare un indirizzo IP statico alla Raspberry Pi: deve essere raggiungibile allo stesso indirizzo, altrimenti ad ogni avvio dovremmo cambiare le impostazioni per il server DNS sul gateway.

ASSEGNAZIONE DELL'INDIRIZZO IP STATICO

È indispensabile che la Raspberry Pi sia raggiungibile sempre allo stesso indirizzo IP; è possibile assegnare un indirizzo IP statico nella pagina di configurazione del gateway oppure impostare la Raspberry Pi: noi opteremo per la seconda opzione. Connettiamoci in SSH alla Raspberry Pi e lanciamo il comando `ip link`. Ci verrà riportato un output come in **Fig. 8**, che mostra ciò che viene restituito dall'esecuzione del comando `ip link`.

Dobbiamo individuare i nomi delle interfacce utilizzate; quella che ci interessa è la ethernet, in questo caso nominata `eth0`. A questo punto lanciamo il comando:

```
sudo nano /etc/netplan/50-cloud-init.yaml
```

Si tratta di un file .yaml che riporta la configurazione di rete delle varie interfacce. Compiliamo il file come il codice presente nel **Listato 1**. Trattandosi di un file .yaml dobbiamo porre molta attenzione alle indentazioni, altrimenti la configurazione risulterebbe errata.

Una volta finito, premiamo CTR+X → Y → INVIO. Per convalidare la configurazione lanciamo il comando `sudo netplan apply`.

INSTALLAZIONE DI PI-HOLE

Una volta riavviato il dispositivo, ricolleghiamoci ad esso in SSH e avviamo l'installazione di Pi-Hole con il comando `sudo curl -sSL https://install.pi-hole.net | bash`.



Fig. 8
Dati restituiti
dall'esecuzione del
comando ip link.

```

enaselel@OMEN-dl-Enaselele:~$ ssh ubuntu@192.168.1.100
ubuntu@192.168.1.100's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-1028-raspi aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

System information as of Sat Feb 20 22:23:20 UTC 2021

System load:  0.15               Temperature:   32.1 C
Usage of /:   4.2% of 58.47GB    Processes:    144
Memory usage: 7%                Users logged in: 0
Swap usage:   0%                IPv4 address for eth0: 192.168.1.100

 * Introducing self-healing high availability clusters in MicroK8s.
   Simple, hardened, Kubernetes for production, from RaspberryPi to DC.
   https://microk8s.io/high-availability

2 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

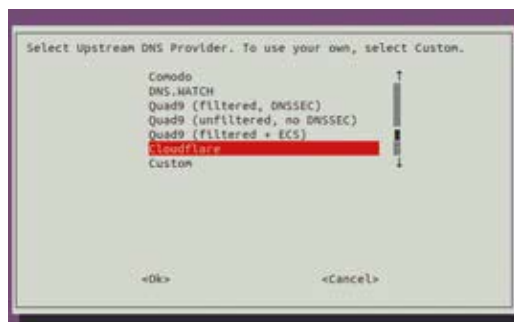
Last login: Wed Feb 17 22:43:24 2021 from 192.168.1.122
ubuntu@ubuntu:~$ ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT group default qlen 1000
    link/ether dc:a6:32:49:21:7c brd ff:ff:ff:ff:ff:ff
3: wlan0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN mode DEFAULT group default qlen 1000
    link/ether dc:a6:32:49:21:7d brd ff:ff:ff:ff:ff:ff
ubuntu@ubuntu:~$

```

Adesso verranno stampate su seriale una serie di stampe di debug, fino a quando non compariranno due schermate informative. Premiamo invio in entrambe per proseguire. Nella videata successiva (**Fig. 9**) ci verrà chiesto quali server DNS esterni vogliamo utilizzare: è a nostra discrezione scegliere uno piuttosto che un altro. Nel caso proposto dalla **Fig. 9** abbiamo scelto quelli di Cloudflare (1.1.1.1 e 1.0.0.1). Una volta premuto Invio, ci verrà chiesto se vogliamo utilizzare una lista già pronta di domini da inserire nella blacklist (**Fig. 10**); accettiamo questa "offerta" e proseguiamo. La schermata successiva, cui si arriva cliccando su OK, riguarda i parametri di rete, nello specifico l'indirizzo IP della Raspberry Pi e quello del gateway. Se i parametri coincidono con quelli impostati precedentemente confermiamo e andiamo avanti. Ora arriviamo allo step successivo in cui ci verrà chiesto se vogliamo installare la web GUI; accet-

tiamo l'impostazione predefinita, confermiamo e andiamo avanti. Il passo successivo è abbastanza delicato, in quanto dobbiamo scegliere se utilizzare il Web Server suggerito (Lighttpd) oppure utilizzarne uno a nostra discrezione (Apache, Nginx e via di seguito) come mostrato in **Fig. 11**. Qualora fossimo sicuri di utilizzare il solo Pi-Hole che si appoggia al Web Server, possiamo confermare la scelta predefinita; in caso contrario è opportuno negare il consenso e installare manualmente un altro Web Server. Effettuata la scelta possiamo passare a scegliere se salvare i dati delle query: confermiamo l'intenzione di farlo lasciando selezionato On (Recommended (se non volessimo farlo, cliccheremo su Off) che è l'impostazione predefinita. Nell'ultimo step ci verrà chiesto quali dati vogliamo salvare sul server locale e quali visualizzare: come ci propone la **Fig. 12**, è possibile effettuare la scelta tra quattro "livelli", da 0 a 3. Il livello 0 permette di salvare tutti i dati e mostrare le statistiche. Il livello 1 esclude i domini, sia quelli più cercati, sia quelli più bloccati. Il livello 2 nasconde tutto ciò già incluso nel livello 1 e anche i client. Il livello 3 disabilita tutti i dati e mostra esclusivamente statistiche anonime. Una volta confermata la nostra scelta, a video comparirà un riepilogo dei dati di configurazione relativi alle impostazioni che abbiamo fatto, con la password d'accesso alla Web GUI su cui andremo a lavorare.

Fig. 9
Scelta dei server
DNS esterni da
utilizzare.



WEB GUI ADLIST E BLOCKLIST

Completata l'installazione, è possibile accedere alla web GUI digitando nella barra degli indirizzi del browser: `IP_DEL_RASPBERRYPI/admin`.

Inseriamo la password e ci apparirà una schermata che sarà come quella proposta dalla **Fig. 13**.

La grafica è ben curata e riporta tutti i parametri essenziali: in alto a sinistra vengono riportati dati quali la temperatura e il carico del processore e la memoria occupata.

Risaltano all'occhio le statistiche sul numero di query, quante di quest'ultime sono state bloccate (anche in percentuale) e il numero di domini bloccati. Prendono spazio anche una serie di "timeline" che indicano la distribuzione dell'attività nelle ultime 24 ore e due istogrammi a torta che rappresentano i tipi di queries effettuate e le risposte: se sono state indirizzate alla blocklist, a uno dei due server DNS esterni o se erano già presenti nella cache locale.

Adesso dobbiamo preoccuparci di impostare le blacklist desiderate; e come si fa ciò?

Semplice: è necessario recarsi su Group Management → Adlists, allorché verrà caricata una pagina come quella visibile nella **Fig. 14**. Per caricare un'adlist (sostanzialmente un elenco aggiornato di domini "proibiti") è sufficiente inserire i link delle varie liste. Sono facilmente reperibili in rete, ma per comodità vi rimandiamo ad un link GitHub: <https://github.com/search?q=pihole&type=topics>. È possibile caricare manualmente i domini (e/o i relativi sottodomini) da censurare o meno; per inserire un dominio nella blacklist basta selezionare la voce "Blacklist".

La schermata che ci comparirà sarà del tutto simile

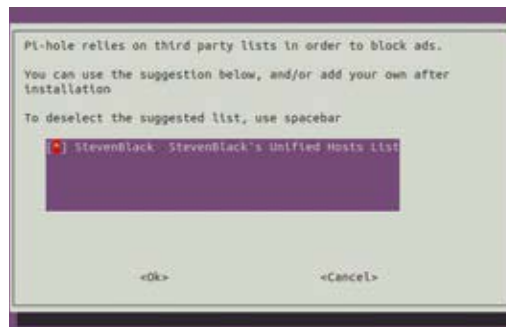


Fig. 10
Aggiunta di un'adlist pre-configurata

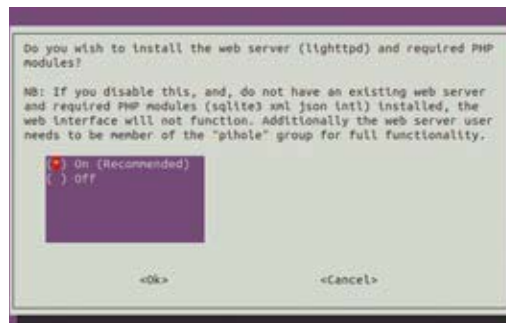


Fig. 11
Installazione del web-server.

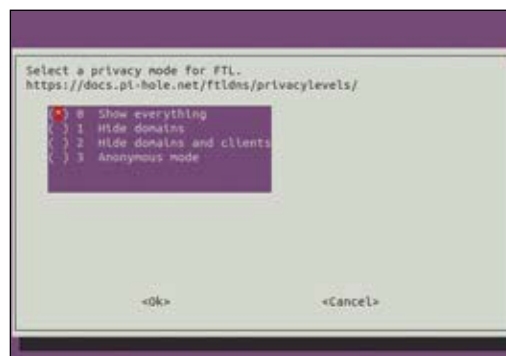
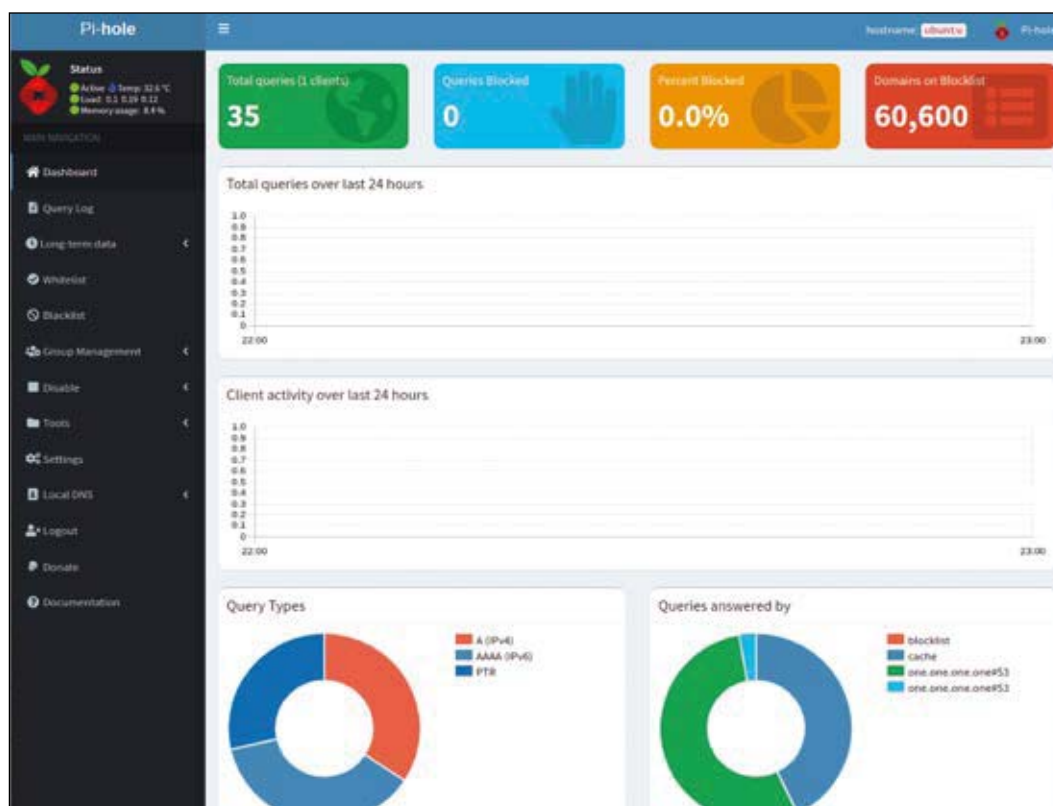


Fig. 12
Livelli di privacy del salvataggio dei dati.

↓ Listato 1

```
# This file is generated from information provided by the datasource. Changes
# to it will not persist across an instance reboot. To disable cloud-init's
# network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
  ethernets:
    eth0:
      dhcp4: no
      addresses:
        - 192.168.1.100/24
      gateway4: 192.168.1.1
      nameservers:
        addresses: [1.1.1.1, 8.8.8.8]
      optional: true
  version: 2
```

Fig. 13
Primo avvio
della Web-GUI.



a quella illustrata precedentemente, durante la configurazione del nostro software Pi-Hole. Nel campo "Domain" inseriamo il dominio che vogliamo bloccare e spuntiamo la voce "Add domain as wildcard" se vogliamo bloccare tutti i relativi sottodomini. Clicchiamo su "Add to Blacklist" e il gioco è fatto! Per la prima configurazione è necessario fare in modo che Pi-Hole legga i domini bloccati tra le varie liste. Andiamo dunque su "Tools" → "Update Gravity" → "Update": in pochissimi minuti il sistema avrà scaricato in locale i dati delle varie liste. Inoltre il sistema programma in automatico un task affinché queste liste vengano aggiornate periodicamente, una volta alla settimana, nella notte tra Sabato e Domenica.

SETTAGGIO DEL SERVER DNS SUL GATEWAY

Bene, adesso la nostra Raspberry Pi è pronta a fare da "buco nero" per la nostra connessione; manca solo un ultimo step, che consiste nell'impostare il suo indirizzo IP come quello del server DNS. Quest'ultimo step ci permetterà di impostare la Raspberry Pi come server DNS di tutta la rete locale. Potremmo impostarlo manualmente su tutti i dispositivi, ma la scelta risulterebbe poco pratica; invece, impostandolo sul gateway, estenderemo la

configurazione automaticamente a tutti i dispositivi connessi (salvo eventuali modifiche manuali) e in caso di nuovi dispositivi connessi anch'essi beneficeranno di questa impostazione. Non approfondiremo nel dettaglio questa procedura in quanto varia per ciascun gateway, ma comunque forniremo delle informazioni di massima. È necessario entrare nella pagina di configurazione del gateway (normalmente all'IP 192.168.1.1 oppure 192.168.0.1) e una volta effettuato il login bisogna individuare il menù di configurazione della rete locale (di norma è indicato con "Rete locale"). Nell'impostazione di cui alla voce "Server DNS primario" inseriamo l'indirizzo IP della nostra Raspberry Pi e in corrispondenza della voce "Server DNS secondario" inseriamo l'indirizzo di un ulteriore server DNS (noi abbiamo scelto 1.1.1.1). Sconsigliamo vivamente di lasciare l'ultimo campo vuoto poiché qualora la Raspberry Pi non fosse raggiungibile (per attività di manutenzione o per qualsiasi altro motivo) il gateway non avrebbe alcun server alternativo, rendendo vani tutti i tentativi di connessione. Segnaliamo infine che alcuni gateway forniti dagli operatori non permettono di inserire indirizzi di server DNS esterni diversi da quelli propri.

UTILIZZARE IL SERVER DNS ANCHE SU RETE MOBILE

È possibile filtrare la connessione mobile, ma la procedura è più complicata in quanto non è possibile impostare “nativamente” i server DNS della rete mobile sugli smartphone. È perciò necessario installare il sistema Pi-Hole su una VPS (Virtual Private Server) personale (AWS ne mette alcune a disposizione gratuitamente) a cui verrà successivamente affiancato un servizio VPN (Virtual Private Network). Uno dei più noti è OpenVPN: è una soluzione open source ed è installabile in modalità “server” sulla distribuzione Ubuntu. Uno dei vantaggi è anche quello di ottenere una connessione criptata per aumentare il livello di sicurezza e privacy. Si tratta comunque di un argomento troppo complesso per poter essere trattato in questo articolo, perciò vi rimandiamo al link <https://docs.pi-hole.net/guides/vpn/openvpn/overview/>.

CONCLUSIONI

Bene, si chiude qui la presentazione di questo semplice ma altrettanto potente software open-source che nello specifico abbiamo installato su Raspberry Pi per affidarle il compito di controllore. Oltre a quelle che abbiamo incontrato e analizza-

to, esistono numerose altre funzionalità che (per ragioni di spazio e per non essere dispersivi) non abbiamo potuto trattare; tali funzionalità sono tutte di facile apprendimento e se sfruttate adeguatamente vi permetteranno di personalizzare ulteriormente il servizio.

Ancora una volta abbiamo avuto prova di ciò che può realizzare un piccolo gruppo di sviluppatori per migliorare la vita di un folto numero di utilizzatori e di come la comunità dell’open source si dimostri prolifica e prodiga di soluzioni utili a tutti. ■



Cosa occorre?

Per la realizzazione di questo progetto consigliamo gli starterkit per Raspberry Pi che contengono tutto il necessario per configurare immediatamente distribuzioni come Pi Hole. In particolare per questo progetto consigliamo lo starter kit con Raspberry Pi da 2GB (cod. RPI4STARTERG2) in vendita a Euro 75,00. Sul sito [futurashop.it](http://www.futurashop.it) sono disponibili kit anche per Raspberry Pi da 4GB e da 8 GB. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:

**Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)
Tel: 0331-799775 - <http://www.futurashop.it>**

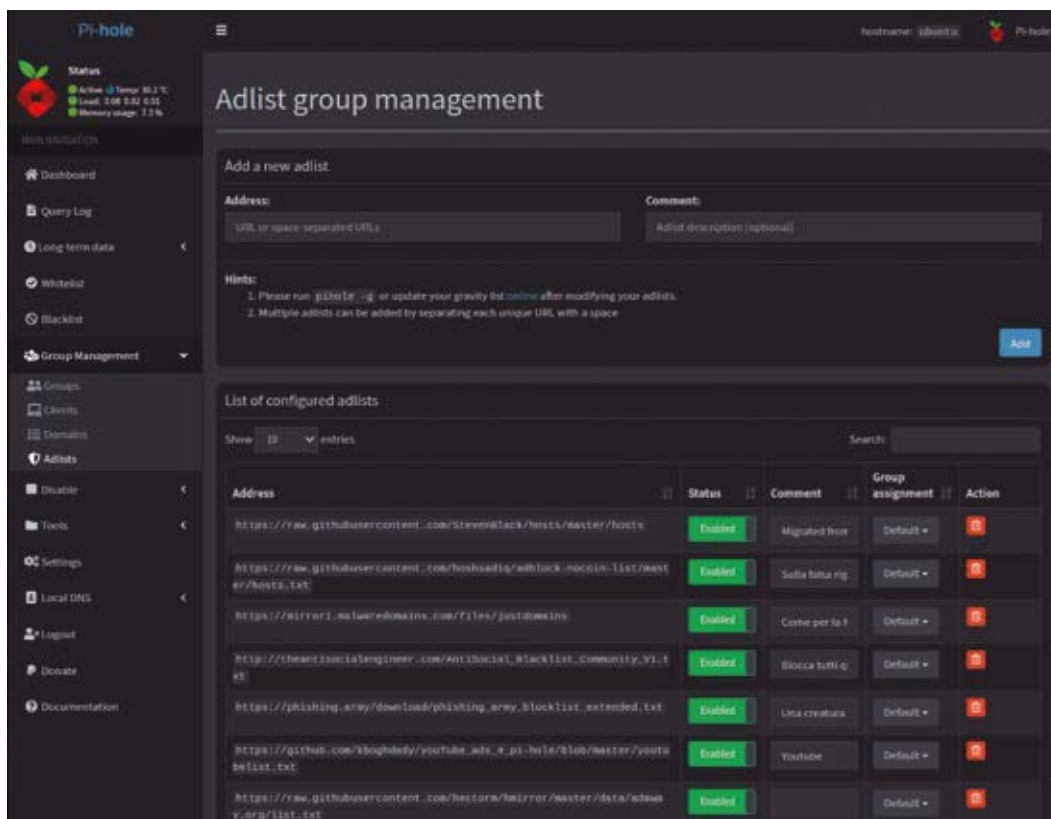


Fig. 14
Caricamento
delle adlists.

Strumenti di misura per veri professionisti!



Rilevatore da parete 3 in 1 (legno, metalli, cavi elettrici)

Rileva la presenza di cavi elettrici sotto tensione (con passaggio di corrente), strutture in legno (telai, travetti, barre, ecc), oggetti metallici (travi, telai, tubi, tondini, perni, barre in metallo). Dispone di ampio display retroilluminato, buzzer di segnalazione, pulsante di calibrazione automatica. Si spegne automaticamente dopo 3 minuti di inutilizzo. Alimentazione con batteria da 9 V (acquistabile separatamente), dimensioni 156x70x26 mm.

€29,50

Cod. FR760

Rilevatore da parete professionale (legno, metalli, cavi elettrici)

Rileva la presenza di cavi elettrici sotto tensione (con passaggio di corrente), strutture in legno (telai, travetti, barre, ecc), oggetti metallici (travi, telai, tubi, tondini, perni, barre in metallo). Dispone di ampio display LCD retroilluminato, buzzer di segnalazione, spegnimento automatico dopo 5 minuti di inutilizzo. Alimentazione tramite batteria al litio (integrata) da 3,7 volt 1500 mAh. Dimensioni 195x90x35 mm.



- ✓ Professionale
- ✓ Batteria al litio ricaricabile integrata da 3,7 V - 1500 mAh con autonomia fino a 10 ore
- ✓ Case robusto
- ✓ Prestazioni migliorate

€78,00

Cod. FR801

Misuratore di distanza fino a 80 m con LASER verde e messaggi vocali.

Metro LASER multifunzione compatto con display LCD retroilluminato, messaggi vocali, menu a icone, livella digitale, memoria interna (in grado di memorizzare fino a 100 misurazioni) e batteria ricaricabile agli ioni di litio. È in grado di effettuare le seguenti misurazioni: distanza (da 20 cm a 80 metri), area, volume, misurazione pitagorica, misurazione verticale e molte altre. Visualizza lo stato di carica delle batterie; si spegne automaticamente se inattivo (laser dopo 30 secondi, strumento dopo 2 minuti). Grado di protezione IP54. Dimensioni 120x50x26 mm.



€79,00

Cod. FR798

Misuratore di distanza LASER da 20 cm a 30 m

Compatto metro LASER con display LCD retroilluminato. Permette di misurare in modo estremamente facile e veloce la distanza (da 20 cm a 30 metri), l'area e il volume. Dispone di spegnimento automatico del LASER e del dispositivo; visualizza anche lo stato di carica delle batterie. Alimentazione con 2 batterie mini stilo da 1,5 volt tipo AAA (incluse). Dimensioni 105x34x20 mm.



€27,00

Cod. FR797

Prezzi IVA inclusa

TRASMETTITORE DCF77



di **STEFANO PURCHIARONI**

Produce un segnale orario ricavato da un RTC molto preciso e lo irradia localmente sui 77,5 kHz con protocollo DCF77. Ideale per avere un riferimento orario o testare ricevitori.

L'idea di sincronizzare un orologio con un riferimento remoto e affidabile è quasi contemporanea alla nascita della Radio: ai tempi di Guglielmo Marconi, l'ingegnere Sir Howard Grubb propose il concetto di sincronizzazione in un discorso tenuto alla Royal Dublin Society già nel 1898. Dopo pochi anni erano già stati attivati servizi di diffusione via radio del segnale orario. Quello che vi proponiamo in queste pagine nasce in quest'ottica, perché è un trasmettitore radio di segnale orario generato localmente; opera sulla stessa frequenza della notissima stazione DCF77 (ma a potenza tanto ridotta da coprire un appartamento) in modo che l'orario irradiato possa essere ricevuto dagli orologi e radiosveglie presenti nel suo raggio di azione e in grado di sincronizzarsi con il DCF77. Prima di entrare nel merito del progetto è opportuna una panoramica sulla radiodiffusione del segnale orario.

Fig. 1
Struttura del
segnale orario
Six-Pip.



Iniziò la Marina USA nel 1903, dalla sede del UNSO (U.S. Naval Observatory) a Washington DC, seguito l'anno successivo dal Navy Yard di Boston.

Entro il 1913 tutto il territorio USA e Canadese venne raggiunto dal servizio.

Intanto, anche in Europa prese il via la diffusione del segnale orario con trasmissioni su lunghezza d'onda di 2 km dalla cima della Tour Eiffel, da parte del French Bureau of Longitudes.

L'inglese BBC, nei primi anni 20 iniziò l'emissione regolare del classico "Six-Pip" ancora oggi ricevibile. Lo scopo di tali trasmissioni era prettamente militare, e mirava alla sincronizzazione degli orologi sulle navi, ma l'interesse sconfinò presto nell'ambito civile, dalle ferrovie all'orologeria.

Il metodo di sincronizzazione era manuale e prevedeva uno schema da seguire durante l'ascolto, per interpretare una serie di "beep" da 1 kHz emessi nella forma di segnali simil-Morse.

Si mossero quindi i primi passi per automatizzare la sincronizzazione degli orologi: Il Daily Sketch del 4 ottobre 1912 segnala che il londinese F. O. Read inventa un sistema di trasmissione del segnale generato dal passaggio del pendolo di un orologio di riferimento su una bobina. L'utente finale doveva quindi disporre di un orologio simile dove l'oscillazione del pendolo veniva forzata a seguire il ritmo ricevuto via radio. Tuttavia questo ed i molti altri brevetti depositati negli anni a seguire, non trovarono sbocchi commerciali.

Il 1928 vide l'invenzione del primo orologio a pendolo elettro-meccanico capace di interpretare il Six-Pip (**Fig. 1**) generato a Greenwich.

L'accuratezza osservata era di circa 2 secondi l'anno. Pur trattandosi di una evidente innovazione, neanche questo orologio ebbe larga diffusione.

Fu IBM a diffondere dal 1956 i primi orologi

radio-controllati nel mercato "Consumer" sulla west coast, basati sul segnale orario emesso dalla stazione radio WWVH delle isole Hawaii (**Fig. 2**) su frequenze che vanno da 2,5 a 15 MHz. La struttura del segnale è molto articolata e contiene i campi anno, minuto, ora e ordinale del giorno nell'anno. Lo schema veniva trasmesso in determinati orari della giornata, assieme a voce e musica.

IL SEGNALE ORARIO AI NOSTRI TEMPI

Dopo questo sintetico excursus storico e geografico, spostiamoci ai nostri luoghi e tempi, e vediamo chi diffonde oggi un segnale orario, e come lo fa. La sorgente di riferimento è ormai un orologio atomico, che garantisce errori al di sotto del nanosecondo l'anno. Il metodo di trasmissione si è assestato sulla LF (bassa frequenza fra 40 e 80 kHz) vista la sua capacità di raggiungere grandi distanze, ed anche per l'economicità dei ricevitori. Gli USA adottano oggi una trasmissione con portante a 60 kHz.

In Europa il segnale orario viene diffuso da Gran Bretagna, Svizzera, Germania e Francia.

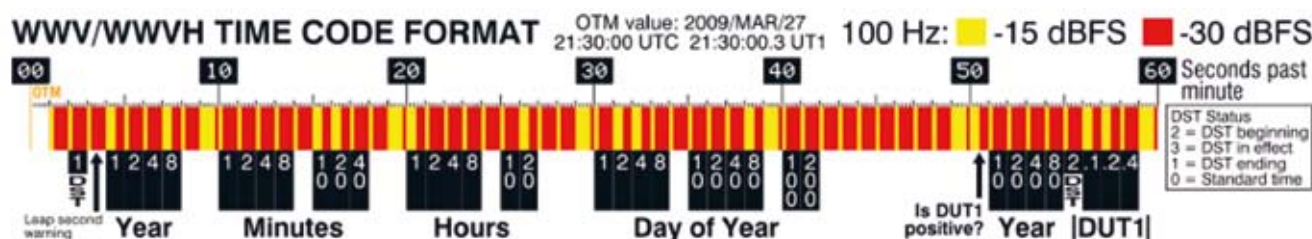
L'Italia aveva una sua trasmissione ad orari predefiniti, emessa dalla RAI fino al dicembre 2016 con il dato fornito dall'orologio atomico dell'Istituto Nazionale di Ricerca Meteorologica (Torino) che prosegue a tutt'oggi ma con un dato auto-generato e non affidabile. Il segnale di riferimento ricevibile costantemente in Italia è ad oggi unicamente il tedesco DCF77, nato nel 1959 ma assestato nella forma attuale solo dal 1973. Esso è trasmesso dal *Physikalisch-Technische Bundesanstalt* (PTB) su una portante di 77,5 kHz emessa con 50 kW di potenza dall'antenna operativa di una coppia che comprende una antenna di riserva, presso la città di Mainflingen, a 25 km da Francoforte.

La sorgente del dato è costituita da un insieme coordinato di tre orologi atomici al cesio.

È possibile acquisire il segnale con un semplice ricevitore radio dotato di antenna in ferrite accordata sulla frequenza di trasmissione.

Il contenuto informativo è organizzato con uno

Fig. 2
Struttura del
segnale orario
americano WWVH.



standard che prevede un abbattimento di 6db (ovvero del 75%) dell'ampiezza della portante all'inizio di ogni secondo, per una durata di 100 o 200 ms, rappresentanti rispettivamente i bit "0" e "1". Nel secondo 59 l'abbattimento si arresta, dando modo agli orologi che ricevono il segnale di sincronizzarsi. Il segnale copre l'Italia, e il resto d'Europa (**Fig. 3**). Durante il giorno i disturbi provocati da fonti esterne, e dalla instabilità della ionosfera, causano problemi in ricezione ma, durante la notte lo "strato di Heaviside" della ionosfera si addensa aumentando la riflessività e di conseguenza la copertura del segnale. In tali condizioni, il ricevitore ha una maggior probabilità di acquisire il segnale e decodificarlo. Il segnale modulante è proposto nella **Fig. 4**. Nella **Fig. 5** vedete invece la codifica dei 59 impulsi trasmessi ogni minuto: in grigio, i dati non trattati dalla nostra applicazione (valore trasmesso "0" per ogni bit).

La codifica delle informazioni orarie segue lo standard BCD, dove i due nibble di 4 bit del byte codificato contengono rispettivamente le decine e le unità del valore di input, con il bit meno significativo per primo. Per esempio il minuto "37" verrà codificato nella forma **1110 1100**. La decodifica giova di diverse possibilità di verifica dell'integrità del dato, per rendere minima la possibilità di errore. In particolare sono stati previsti tre bit di parità, per i gruppi "Minutes", "Hours", ed infine per l'intero blocco di 22 bit rappresentante la data. In aggiunta, il ricevitore effettua anche l'analisi dei contenuti rispetto all'intervallo di riferimento di ogni elemento, per esempio l'ora deve essere contenuta nel range 0-23.



Fig. 3
Copertura della trasmissione del segnale DCF77 nativo.

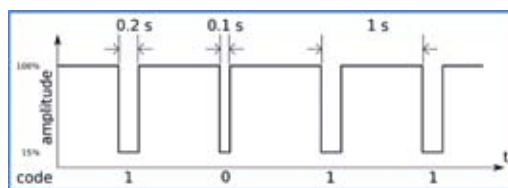


Fig. 4
Ampiezza della portante e suo abbattimento durante la modulazione.

I primi 14 bit che seguono il bit 0 di inizio del minuto codificano informazioni meteorologiche e allerte fornite da servizi civili, crittografate e con licenza d'uso rilasciata dalla tedesca HKW-Elektronik. Nel progetto proposto in queste pagine, non viene generata alcuna informazione relativa a questa sezione e quindi i ricevitori che la trattano e ne

Fig. 5
Codifica dei dati DCF77.

Bit	Nome	Peso BCD	Descrizione	Bit	Nome	Peso BCD	Descrizione	Bit	Nome	Peso BCD	Descrizione
00	START		Inizio minuto (sempre 0)	20	STIME		Inizio Time (sempre 1)	40		10	
01	METEO			21	MINUT	1		41		20	Giorno (segue)
02				22		2		42	WEKDY	1	
03				23		4		43		2	Giorno settimana 01-07 dal Lunedì
04				24		8	Minuti 00-59	44		4	
05				25		10		45	MONTH	1	
06				26		20		46		2	
07				27		40		47		4	Mese 01-12
08			14 bit "MeteoTime"	28 (P1)			Parità minuti	48		10	
09				29	HOUR	1		49		20	
10				30		2		50	YEAR	1	
11				31		4		51		2	
12				32		8	Ore 00-23	52		4	
13				33		10		53		8	Anno 00-99 nel secolo
14				34		20		54		10	
15	CALL		Anomalia	35 (P2)			Parità ore	55		20	
16	CHANG		Annuncio cambio stagione	36	DAY	1		56		40	
17	CEST		Orario estivo	37		2		57		80	
18	CET		Orario invernale	38		4	Giorno 01-31	58 (P3)			Parità bit 36..58 inclusi
19	LEAP		Annuncio Leap Second	39		8		59	PAUSE		Nessuna modulazione



scartando segnali molto disturbati o di lunghezza non conforme con quanto atteso. Inoltre, prima di convalidare l'orario ricevuto, la maggior parte degli orologi attendono almeno due campioni validi. Alla fine è molto raro, anche se non impossibile, che un orologio radio-controllato mostri informazioni errate.

IL TRASMETTITORE DCF77

Prima di entrare nel dettaglio dell’hardware vediamo perché un trasmettitore con segnale simile al DCF77 è utile:

- ricezione scarsa o impossibile del vero segnale DCF77;
- necessità in loco di un riferimento temporale non standard;
- impostazione automatica di tutti gli orologi radio-controllati della propria abitazione;
- verifica di orologi dei quali si sospetta un mal-funzionamento;
- test su ricevitori DCF77 autocostruiti.

Il circuito prevede una catena analogica, disegnata in giallo nella **Fig. 6**, che produce un segnale sinusoidale con frequenza di 77.500 Hz, irradiato tramite una antenna in ferrite dopo una piccola amplificazione. Tale segnale viene modulato per abbattimento tramite impulsi generati dalla sezione digitale, rappresentata in azzurro. La sezione digitale si basa su un microcontrollore PIC 18F2620, e comprende periferiche di I/O per monitorare lo stato di funzionamento dell’apparecchio e per le impostazioni. Per garantire una forte stabilità del dato orario, facciamo uso di un RTC evoluto modello DS3231 prodotto dalla Maxim, adottato nel pratico formato break-out board, con batteria tampone per non perdere le impostazioni in caso di black-out. La precisione ottenibile non è certo paragonabile a quella dell’orologio atomico sul quale si basa il segnale DCF77, ma è accettabile, come si evince dalla **Tabella 1**. Il software a bordo del microcontrollore, oltre a produrre il segnale codificato DCF77, permette di generare delle anomalie intenzionali allo scopo di

verificare il funzionamento sia di orologi commerciali, ma soprattutto per la diagnostica di quelli auto-costruiti. Il circuito è alimentato da un adattatore esterno da 12V/1A, la cui tensione viene direttamente utilizzata solamente dallo stadio di amplificazione. Per alimentare l’oscillatore, l’RTC ed il display viene derivata una tensione di 5V ottenuta da un regolatore lineare 78L05. L’alimentazione per il microcontrollore è derivata dai 5V per interposizione di un diodo Schottky D1 a bassa tensione di soglia, per evitare di alimentare le altre parti del circuito durante la programmazione In-Circuit dal connettore ICSP. Vari condensatori livellano e rimuovono disturbi dannosi per il funzionamento delle parti. Il microcontrollore PIC18F2620 della Microchip è un generoso “Mid range” ad 8 bit. Verrà programmato per fornire sul pin MOD un segnale digitale destinato a modulare il segnale prodotto dall’oscillatore, prima dell’immissione dello stesso nell’amplificatore. Dopo lo stadio di amplificazione basato su un TDA2030, il segnale è inviato ad una antenna in ferrite, realizzata avvolgendo 170 spire di filo di rame smaltato del diametro di 0,25 mm su una ferrite cilindrica lunga 14 cm del diametro di 1 cm, opzionalmente accoppiata con un condensatore non polarizzato da 2 nF (**Fig. 7**). Possono essere sperimentati altri tipi di antenna da collegare al circuito, per esempio esterne collegate tramite un doppino, o coassiale ad uso audio per tratte più lunghe. La potenza di uscita non supera i 2 watt, e si scoraggia l’impiego di amplificatori di potenza superiore, per evitare conseguenze legali. Il prototipo ha regolato orologi situati nel raggio di una decina di metri dal trasmettitore.



Fig. 6
Schema a blocchi del trasmettitore.

DISPOSITIVO	PRODUTTORE	PREZZO MEDIO	DERIVA MASSIMA
Miniature Atomic Clock	Microsemi	2000\$	0.3 ms / anno [1]
DS3231	Maxim Integrated	2\$	1 min / anno [2]
MCP79400	Microchip	2\$	2 min / anno [2]
DS1307	Maxim Integrated	2\$	8 min / anno [2]
PCF8563	NXP Semiconductors	2\$	15 min / anno [2]

Tabella 1
Confronto tra alcuni orologi elettronici, il primo dei quali è atomico. [1] dichiarato dal produttore, [2]: benchmark di Switchdoc Labs.



Il principio fisico alla base delle oscillazioni di un orologio atomico sfrutta la frequenza di risonanza degli atomi, ed in particolare del Cesio, elemento preferito ed usato nella moderna definizione del Secondo. Nel 1967 l'ufficio internazionale pesi e misure di Parigi ha infatti definito il secondo come "La durata di 9.192.631.770 periodi della radiazione corrispondente alla transizione tra due livelli iperfini, da ($F=4$, $M_F=0$) a ($F=3$, $M_F=0$), dello stato fondamentale dell'atomo di cesio-133". La struttura iperfine degli atomi è cosa nota già dalle osservazioni di Michelson del 1881, strutturate in seguito da Pauli nel 1924. Senza entrare nella sua complessa trattazione fisico-matematica, che può essere interessante materia di approfondimento, proviamo a sintetizzarne le basi. Sia gli elettroni in moto attorno al nucleo, che il nucleo stesso, hanno un momento di spin. Si viene allora a creare

un dipolo magnetico, interpretabile come l'effetto del moto delle cariche elettriche. I due spin, elettronico e nucleare, tendono ad allinearsi creando varie possibili posizioni reciproche di minor energia potenziale nelle quali l'atomo si assesta. Così come accade al livello più grossolano degli orbitali atomici, anche tale separazione "iperfine" degli stati energetici può dar luogo a transizioni, ottenibili inviando energia al sistema atomo. Negli orologi atomici, un magnetron contenente un gas di Cesio-133, eccita gli atomi con onde radio stazionarie, su una frequenza generata esternamente e sintonizzabile tramite un anello di retroazione, principio base dei fenomeni oscillatori. Gli atomi assorbiranno le onde aventi esattamente la frequenza di oscillazione propria della transizione iperfine cercata. Un sistema di regolazione provvede quindi a mantenere stabile l'oscillatore ester-

no sulla frequenza di risonanza. Una copia di questo segnale costituisce infine il riferimento temporale per il conteggio, per fornire il risultato con una accuratezza estrema. Molti gli studi e le evoluzioni, nonché le ricadute commerciali legate alla produzione di riferimenti temporali integrati sempre più piccoli, precisi ed economici. La miglior precisione ad oggi raggiunta prevede un errore massimo di un secondo su 15 miliardi di anni.



L'oscillatore prevede l'uso di un quarzo da 77.500 Hertz, un po' difficile da reperire, ma non impossibile. La scelta di impiegare un oscillatore al quarzo è legata alla strettissima banda passante dei ricevitori. Una deriva di soli 100 Hertz può infatti far scartare il segnale in ricezione. L'oscillazione parte da un multivibratore astabile realizzato con due porte di un hex-inverter CMOS HEF4096. L'onda quadra viene in seguito modificata per renderla "quasi" sinusoidale con due filtri RC in cascata. Il segnale ottenuto dal gruppo oscillatore viene modulato per abbattimento tramite un transistor NPN BC337, pilotato dalla linea "MOD" proveniente dalla sezione digitale del circuito. Benché il segnale nativo preveda un abbattimento del livello del segnale del 75%, nel nostro circuito ci spingiamo al 90%. Questo per evitare che orologi vicini al trasmettitore vadano in saturazione anche durante la modulazione. Da prove sul campo il segnale così ottenuto appare gradito agli orologi radio-

controllati commerciali, ed ancor più da quelli auto-costruiti utilizzando i moduli ricevitori con antenna in ferrite reperibili sui siti di e-Commerce. Lo stadio di amplificazione utilizza un componente normalmente destinato alla banda audio, ma nel nostro caso la frequenza da trattare rientra nei limiti degli amplificatori monolitici. La scelta è caduta su un TDA2030, che richiede poca componentistica aggiuntiva. Sarebbe possibile ottenere un'elevata efficienza tramite amplificatori in classe "D" integrati, ma la potenza di uscita sarebbe troppo elevata. A parte l'esigenza di usare a questo punto un alimentatore con amperaggio maggiore, si rischia di infastidire il vicinato e di incorrere in sanzioni. La sezione digitale del circuito è costruita attorno al microcontrollore PIC18F2620, programmato con il firmware che potete scaricare dal nostro sito, il cui sorgente viene descritto più avanti. Le periferiche hardware collegate alla MCU (Micro-Controller Unit) sono:

- un modulo RTC pre-assemblato inserito su strip-line maschio, modello DS3231 con interfaccia I²C-Bus;
- un display LCD modello Hitachi HD44780, pilotato con 4 linee dati, inserito su strip-line femmina;
- due pulsanti per le impostazioni;
- un LED bicolore verde-rosso a catodo comune, per segnalare il bit in corso di trasmissione;
- un piccolo altoparlante per la stessa segnalazione in forma di toni acuti e gravi.

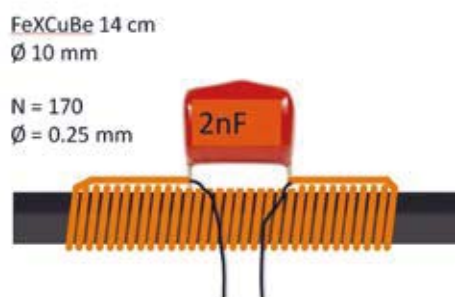


Fig. 7
L'antenna in ferrite.

Il LED e l'altoparlante hanno lo scopo di segnalare il bit in corso di trasmissione: lo zero logico è indicato da un suono grave ed un lampeggio di luce rossa. Un suono acuto e un lampeggio verde indicano invece un bit ad 1. In tal modo è possibile, anche solo ascoltando i suoni, trascrivere il treno di impulsi in corso di trasmissione: per esempio durante il test di un orologio autocostruito. Sul display LCD sono mostrati data ed ora correnti, nonché il giorno della settimana, l'uso o meno dell'ora legale (che viene gestita automaticamente), l'abilitazione alla trasmissione, e due parametri di alterazione intenzionale della qualità del dato trasmesso. Quest'ultima possibilità permette di testare le politiche di correzione degli errori in un orologio auto-costruito introducendo due tipi di disturbi:

- impulsi in posizioni casuali che invertono il segnale per 6 ms, fino a 9 volte al secondo;
- alterazione della durata dei bit fino a +/- 9 ms rispetto a quella standard di 100/200 ms.

In una seconda pagina di visualizzazione, il display mostra direttamente il treno di bit in trasmissione e un'etichetta con la tipologia di dato correntemente generato. I testi sono disponibili in Italiano ed in Inglese, ma la scelta della lingua è fatta tramite una flag che richiede la ricompilazione del sorgente. Il firmware pronto per il caricamento a bordo della MCU è predisposto con le etichette in Italiano.

IL FIRMWARE

Il programma è scritto in linguaggio C, adatto per il compilatore MikroC 8.2. Come accennato precedentemente, l'impiego di versioni più recenti dello stesso compilatore, o di compilatori diversi come XC, Hi-Tech C, C18, Mplab X, ecc, richiederà una fase di Porting, per adattarlo alle specifiche idiosincrasie del dialetto scelto. Questo vale solo nel caso si voglia modificare il codice, perché esso è già stato compilato mettendo a disposizione direttamente il file binario HEX pronto per l'upload nel microcontrollore, tramite programmatore con zoccolo, o tramite la porta ICSP offerta sul retro del circuito stampato.

Analizziamo ora la logica adottata per gestire il dispositivo. I commenti sono auto-esplicativi e facilitano molto la comprensione dei singoli passaggi. Nel nostro sito è disponibile il sorgente completo, mentre in queste pagine ci limiteremo a una descrizione sommaria dei suoi blocchi principali. La prima sezione riguarda i flag di compilazione: commentando la definizione della flag di compila-

zione **USERTC** si esclude il modulo RTC.

In tal caso verrà usato il quarzo da 16 MHz impiegato come clock per il microcontrollore, ma con una deriva annua di 10-20 minuti anziché di 1 minuto ottenibile con l'RTC. Il flag **USEMONITOR** abilita la porta seriale, che permette di vedere lo stream completo in trasmissione. È stata usata durante il debugging e si consiglia di lasciarla commentata se non per scopi di studio. Gli ultimi due flag stabiliscono la lingua per le etichette mostrate sul display LCD (**Listato 1**).

Dopo queste sezioni troviamo la funzione di caricamento da EEPROM dei parametri di lavoro: prevede l'impostazione del valore predefinito alla prima lettura (**Listato 2**).

Funzioni di conversione della data in giorni o secondi, e viceversa

Si basano su algoritmi noti e facilmente reperibili sul web. Le due prime funzioni calcolano i giorni trascorsi da quello di riferimento detto "Epoch", o viceversa da tale numero ricostruiscono la data. Queste sono poi impiegate nelle due successive funzioni **sec2date** e **date2sec**, che finalmente eseguono la conversione completa fornendo il numero di secondi trascorsi da "Epoch" alla data indicata, in una variabile a 32 bit senza segno, e viceversa. Da osservare che Epoch vale 01/01/1970, e con 32 bit possiamo arrivare fino al 7 febbraio 2106. Nei sistemi che ancora usano variabili con segno, si avrà invece un date-bug per overflow già il 19 gennaio 2038. La penultima funzione **CalcDate** aggiunge alla struttura passata il valore del Giorno della settimana, in convenzione **ISO 8601** da 1 per il lunedì, a 7 per la Domenica, coerente con la codifica DCF77 (gli USA adottano un formato non standard). L'ultima funzione **CalcChangeDays** determina le date di cambiamento della stagione legale, sempre in secondi da Epoch. L'algoritmo le imposta in accordo, con quanto previsto, alle ore 2 dell'ora solare della prima domenica dopo il 25 dei mesi di marzo ed ottobre.

La segnalazione "**ANN**" è mostrata sul display in luogo del valore di Aggiunta di millisecondi al tempo di impulso, durante l'annuncio del cambiamento di stagione, da un'ora prima ad un minuto dopo della transizione. Durante questo periodo non si può modificare il campo Aggiunta (**Listato 3**). Completiamo la descrizione con **GetTime** e **SetTime** che, tramite le funzioni precedenti, calcolano o impostano l'orario corrente. GetTime interroga dapprima l'RTC ottenendo l'ora solare, e successivamente determina la stagione legale



↓ Listato 1

```
// Compilation choices
#define USERTC                // DS3231 usage. Comment if device
is not used
//#define USEMONITOR          // Serial UART monitor. Comment
if not used
#define ITALIAN                // Use Italian labels
//#define ENGLISH              // Use English labels
```

corrente, la necessità di annunciarne il cambiamento, ed infine calcola l'ora locale aggiungendo un'ora durante l'estate legale. SetTime opera al contrario, sottraendo l'eventuale ora legale per ottenere l'ora solare (**Listato 4**).

ISR e Main

La funzione **interrupt** costituisce la Interrupt Service Routine, chiamata laddove un evento inneschi un interrupt nel microcontrollore opportunamente programmato per tale scopo. Nel main viene impostato Timer0 avviando il conteggio secondo un preset su 16 bit tale da ottenere un interrupt per evento overflow ogni millisecondo, tenendo conto anche dell'overhead introdotto dal codice stesso della funzione. Scopo della funzione è fondamentalmente incrementare la struttura dati *LocTime* per far avanzare i campi ora e data, pilotare il pin MOD di output con il livello di segnale previsto per il bit corrente, inquinandolo intenzionalmente con il rumore programmato dal main, e pilotare LED e altoparlante per le segnalazioni opto-acustiche corrispondenti al bit in trasmissione. Inoltre, la funzione interrupt imposta alcune flag per ordinare la gestione di eventi specifici nel main. Viene altresì generata un'onda quadra da 500 Hz sul pin PROBE, per il tuning di Timer0 possibile solo modificando il preset e ricompilando. La durata delle operazioni eseguite nella procedura di interrupt non supera mai 0,2 ms, ben entro i limiti concessi per scongiurare la sovrapposizione di un interrupt

durante la gestione del precedente (**Listato 5**). La funzione principale del programma, chiamata **main**, all'accensione del dispositivo provvede ad inizializzare l'impianto di I/O e le periferiche collegate: display LCD, modulo RTC, linee di input con resistenza di pull-up per i pulsanti, linea seriale opzionale se abilitata a livello di compilazione, ed infine avvia il Timer0 per la chiamata alla ISR ogni millisecondo, in modo da rendere indipendente l'avanzamento dei valori orari ed il triggering delle azioni da eseguire nel main. Al messaggio di benvenuto "*Trasmettitore DCF-77*" segue il recupero dei parametri di lavoro da EEPROM e la lettura del contenuto dei registri del modulo RTC, con le informazioni orarie, ed inizia finalmente il ciclo principale di presentazione e trasmissione dell'orario. Facciamo subito una importantissima premessa: l'orario mostrato dal display è quello in corso di trasmissione ma, una volta acquisito da un orologio radio-controllato, questi lo presenterà solo in corrispondenza del passaggio al minuto successivo, ovvero al fronte di salita del primo impulso (al secondo zero). Quindi l'orario mostrato sul display sarà sempre un minuto avanti a quello mostrato dagli orologi che hanno acquisito il segnale orario. Si sarebbe potuto scegliere di mostrare l'orario che mostrerebbero gli apparecchi riceventi, ma non sarebbe stato coerente con il treno di impulsi in corso di trasmissione. È sufficiente tenerne conto quando si imposta l'orario del trasmettitore. Il loop principale del main ha in carico il compito di popolare l'array **dcf** contenente i bit da trasmettere, codificandoli secondo le regole previste dal protocollo DCF77. Tale azione si ripete all'inizio di ogni minuto. Inoltre, sempre ad inizio minuto, viene compilato l'array **noise** tramite generazione di numeri casuali, per stabilire il ms di inizio di ogni picco da usare per la generazione dei disturbi richiesti dall'utente. Sempre il main provvede ad aggiornare il display con le informazioni data, ora, e parametri di lavoro correnti, facendo lampeggia-

↓ Listato 2

```
// Load parameters from EEPROM (set defaults at first reading)
void LoadParams() {
    OnAir      = RetrieveByte(1); if (OnAir==0xFF)    {OnAir=TRUE;      StoreByte(OnAir, 1);}
    NoiseLev   = RetrieveByte(2); if (NoiseLev==0xFF) {NoiseLev=0;      StoreByte(NoiseLev, 2);}
    SeasonChange = RetrieveByte(3); if (SeasonChange==0xFF) {SeasonChange=TRUE; StoreByte(SeasonChange, 3);}
    Dt         = RetrieveByte(4); if (Dt==0xFF)        {Dt=9;           StoreByte(Dt, 4);}
    Spkr       = RetrieveByte(5); if (Spkr==0xFF)       {Spkr=TRUE;      StoreByte(Spkr, 5);}
}
```

Listato 3

```
void days2date(unsigned long nd, struct TTime *) {
// Compute the Date fields using the given number of days from Epoch
// Year is set in the 00-99 interval, assuming we are in the 21th century
long za, zb, zc, zd, ze, zf, zx;
za=nd+2472632;
zb=(4*za+3)/146097;
zc=((-zb*146097)/4)+za;
zd=(4*zc+3)/1461;
ze=((-1461*zd)/4)+zc;
zf=(5*ze+2)/153;
zx=((-(153*zf+2)/5)+ze+1);
t->day=(unsigned short)zx;
zx=(-zf/10)*12+zf+3;
t->month=(unsigned short)zx;
zx=(zb*100+zd-4800+(zf/10))-2000;
t->year=(unsigned short)zx;
}

unsigned long date2days(struct TTime *) {
// Compute the number of Days from Epoch to the given date
// The year is assumed to be in the 00-99 range (21th century)
static unsigned int z, y, m, j;
// Calculate the number of Days
z=(14-(t->month))/12;
y=t->year+6800-z; // ex 4800
m=t->month+12*z-3;
j=((153*m+2)/5)+(t->day)+y*365+(y/4)-(y/100)+(y/400)-2472633;
t->ndays=j; // Store result in the time structure
return j;
}

void sec2date(unsigned long ns, struct TTime *) {
// Compile Date and Time fields using the given number of Seconds from Epoch
long zx;
// Calculate Time fields and reduce seconds to Days from epoch
zx = ns % 60; ns/=60;
t->s = (unsigned short)zx;
zx = ns % 60; ns/=60;
t->m = (unsigned short)zx;
zx = ns % 24; ns/=24;
t->h = (unsigned short)zx;
days2date(ns, t); // Compute and compile remaining Date fields
}

unsigned long date2sec(struct TTime *) {
// Calculate the number of seconds from Epoch to the given date
long zx;
zx=date2days(t); // Convert date to days from epoch
zx*=86400; // Convert days to seconds from epoch
zx+=3600*(long)t->h;
zx+=60*(t->m);
zx+=t->s;
t->ns = zx; // Store result in the time structure
return zx; // Also return the number of seconds from epoch
}

void CalcDate(struct TTime *) {
// Calculate the number of days from Epoch, the WeekDay and the days remaining to next sunday
date2days(t); // Calculate the n. of days from epoch (result stored in *t)
t->wd = ((t->ndays)-4)%7+1; // Calculate the Week Day (1..7=Mon..Sun)
t->dom = 7-(t->wd); // Calculate the n. of days to reach next Sunday
}

void CalcChangeDays(int y) {
// Calculate the day in the y year where Winter<-->Summer Time Changes occur
// Set a temporary time-date structure with the 25/03 of specified year
TTime tmpTime; tmpTime.year=y; tmpTime.month=3; tmpTime.day=25; tmpTime.h=0; tmpTime.m=0; tmpTime.s=0;
CalcDate(&tmpTime); // Calculate the n. of days from epoch to 25/3/y
tmpTime.ndays+=tmpTime.dom; // Shift to first sunday
Sec1 = tmpTime.ndays*86400+7200; // Winter to Summer change second
tmpTime.month=10; // Set the temporary date-time to october
CalcDate(&tmpTime); // Days from epoch to 25/10/y
tmpTime.ndays+=tmpTime.dom; // Shift to first sunday
Sec2 = tmpTime.ndays*86400+7200; // Summer to Winter change second
}
```

↓ Listato 4

```
void GetTime() {
// Get solar date and time from the RTC device if used
// Calculate the additional needed information: Day of week, CET/CEST transition
// Calculate and set the Local time according with the Legal time change settings
unsigned short dummy;
#ifdef USERTC
// If the RTC module is used, query it to update the solar date & time
SolTime.s = Bcd2Dec(DS3231_Read(SecREG));
SolTime.m = Bcd2Dec(DS3231_Read(MinREG));
SolTime.h = Bcd2Dec((0x3F & DS3231_Read(HouREG)));
SolTime.day = Bcd2Dec(DS3231_Read(DayREG));
SolTime.month = Bcd2Dec((0x3F & DS3231_Read(MonREG)));
SolTime.year = Bcd2Dec(DS3231_Read(YeaREG));
#endif
date2sec(&SolTime); // Calculate number of past Days and Seconds from epoch
// Determine the Winter to Summer (Sec1) and Summer to Winter (Sec2) times for solar year
CalcChangeDays(SolTime.year);
// Season determination
if ((SolTime.ns>=Sec1) && (SolTime.ns<=Sec2)) IsSummer = TRUE; else IsSummer = FALSE;
if (SeasonChange) { // Season change is activated by user
// Change time announcement determination 59 minutes before change up to 1 minute after change
if (((SolTime.ns>=Sec1-3540)&&(SolTime.ns<=(Sec1+60)))
|| ((SolTime.ns>=Sec2-3540)&&(SolTime.ns<=(Sec2+60)))) Change = TRUE; else Change = FALSE;
} else Change=FALSE; // No Change announcement if Season change is disabled
LocTime.ns = SolTime.ns; // Set the local time (seconds from Epoch field)
LocTime.ns += SeasonChange*IsSummer*3600; // If the Season Change is asked, add 1 hour during Summer
LocTime.ndays = LocTime.ns/86400; // Update the number of Days from epoch in the Local time
sec2date(LocTime.ns, &LocTime); // Calculate the Date and Time fields in the Local time
CalcDate(&LocTime); // Calculate the day number and the current WeekDay
}

void SetTime() {
// Set the current time values in the DS3231 module
SecondBegun = FALSE; // Suspend Time update during save
LocTime.s = 0; // Reset seconds
mss = 0; // Reset millisecond
// Calculate the local seconds from epoch using the Date, updated by buttons
date2sec(&LocTime);
// Calculate the solar time subtracting 1 hour from Local time during Summer, if change is required
sec2date(LocTime.ns-SeasonChange*IsSummer*3600, &SolTime);
#ifdef USERTC
// If the RTC module is present, update it
DS3231_Write(SecREG, (Dec2Bcd(SolTime.s)));
DS3231_Write(MinREG, (Dec2Bcd(SolTime.m)));
DS3231_Write(HouREG, (0x3F & (Dec2Bcd(SolTime.h))));
DS3231_Write(DayREG, (Dec2Bcd(SolTime.day)));
DS3231_Write(MonREG, (0x3F & (Dec2Bcd(SolTime.month))));
DS3231_Write(YeaREG, (Dec2Bcd(SolTime.year)));
#endif
SecondBegun = TRUE; // Trigger time update
}
```

re il dato che eventualmente è in corso di modifica. Alla modifica segue, dietro triggering da parte dell'automa a stati, l'aggiornamento dell'RTC o della EEPROM. Il main provvede alla gestione dei due pulsanti utilizzabili per cambiare pagina di visualizzazione, o per le impostazioni, tramite una logica basata su un automa a stati finiti (**Listato 6**).

INTERFACCIA UTENTE

Il display LCD da due righe di 16 caratteri, i due pulsanti P1 e P2, il LED verde-rosso e l'altoparlante costituiscono i dispositivi di interfaccia per

l'utente. Una volta acceso il dispositivo, sul display appare il messaggio di benvenuto (**Fig. 8**).

Se l'inizializzazione del modulo RTC va a buon fine, il display mostra la Pagina principale, con l'ora e la data correnti, e lo stato dei parametri di lavoro (**Fig. 9**). Dopo la **data**, con l'anno espresso nel formato breve (il trasmettitore funziona solo per questo secolo), vediamo il **giorno della settimana**, calcolato con un algoritmo a partire dai campi della data stessa, seguito dalla **stagione legale** in corso (EST: Estate, INV: Inverno). La seconda riga riporta l'**ora**, l'**aggiunta** da applicare



alla lunghezza standard degli impulsi di 100/200 ms, per un massimo di +9 o -9 ms. Per default avremo +0, cioè impulsi standard. Seguono altri quattro parametri: l'abilitazione dell'**altoparlante** (simbolo speaker attivo o barrato), l'applicazione dell'**ora legale** con i valori possibili S (Stagionalità)

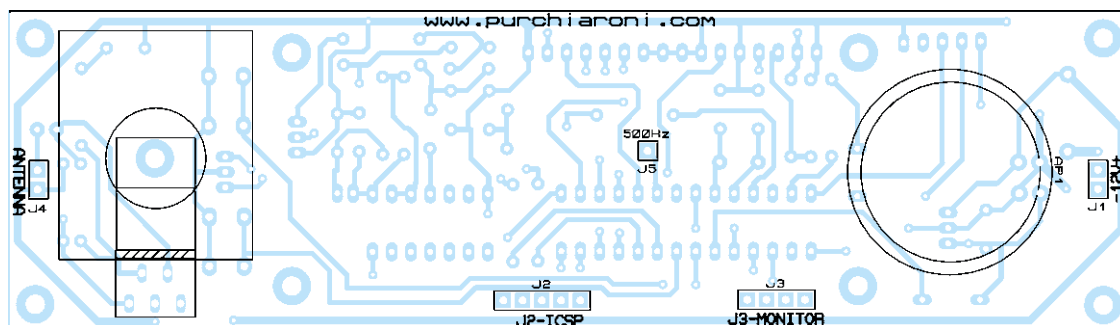
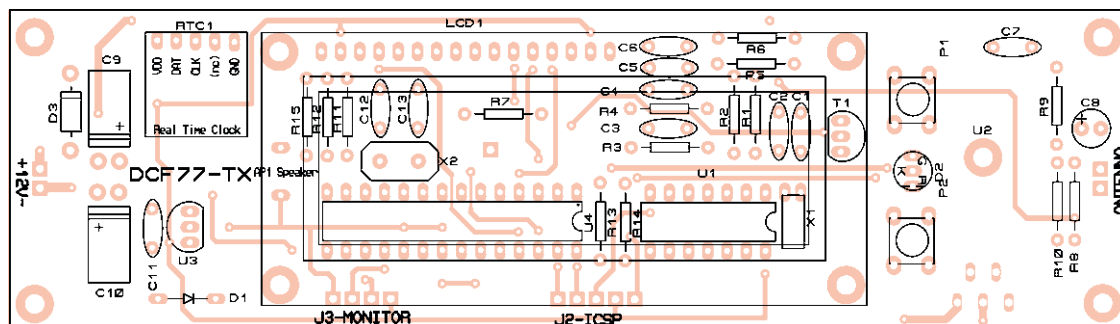
o "-" (non applicata), l'abilitazione alla trasmissione (simbolo antenna accesa o barrata), ed infine il numero di picchi di rumore da 6 ms, posizionati a caso in modo diverso per ogni secondo. In questo caso possiamo generare fino a 9 impulsi, che invertono il valore del livello di modulazione

Listato 5

```

void interrupt() {
// Interrupt Service Routine. Called on any interrupt (TMR0 enabled)
// ----- TMR0 ----- : Manage Timer0 overflow
if (INTCON.TMR0IF) {
// Prepare the timer for the next interrupt
TMR0H = Timer0_Preset >> 8; // Preset for Timer0 MSB register
TMR0L = Timer0_Preset & 0xFF; // Preset for Timer0 LSB register
INTCON.TMR0IF = FALSE; // Clear TMR0 interrupt flag
PORTA.F4 ^= 1; // To probe pin (500 Hz expected)
// Update the current Local time
msCur+=Timer0_Tic; // Current total millisecond update
mss+=Timer0_Tic; // Update Millisecond in current second
if (mss==500) HalfSecond=TRUE; // Set half-second marker
if (mss >= 1000) { // Current second is over
mss = 0; // Reset current millisecond counter
SecondBegun = TRUE; // Set "Second has begun" flag to trigger actions in main
IncrementTime(&LocTime); // Increment the Local time of 1 second
#ifdef USERTC
IncrementTime(&SolTime); // Increase also the Solar time when the RTC is not used
#endif
}
if (OnAir) { // If the transmission flag is enabled...
// Determine the output level
if ((mss==0)&&(LocTime.s<59)) { // Millisecond 0 - Turn on the output except for last second
lev = 1;
ridx = 0; // Reset noise peak index
if (dcf[LocTime.s]==0) {
LD0_PIN=1; LD1_PIN=0; // Set the 0 signal led
} else {
LD0_PIN=0; LD1_PIN=1; // Set the 1 signal led
}
}
if ((mss==(100+Dt-9))&&(dcf[LocTime.s]==0)) { // Millisecond 100 - Turn off output for "0" bit
lev = 0;
LD0_PIN=0; // Reset the 0 signal led
}
else if ((mss==(200+Dt-9))&&(dcf[LocTime.s]==1)) { // Millisecond 200 - Turn off output for "1" bit
lev = 0;
LD0_PIN=0; LD1_PIN=0; // Reset both the signal led
}
// Determine if we are in a noise peak
noi = 0;
if ((NoiseLev>0) && (ridx<NoiseLev)) {
if ((mss >= noise[ridx]) && (mss <= noise[ridx]+NOISE_DUR)) { // We are in a noise pulse interval
noi = 1; // Set the noise signal
ridx++; // Step the index for the next peak
}
}
OUT_PIN = lev ^ noi; // Set the Output PIN mixing standard level and noise
// Emit Bip Bop sounds depending on current bit being transmitted
if ((Spkr)&&(OnAir)&&(mss<200)&&(LocTime.s<59)) {
if (dcf[LocTime.s]==0) SND_PIN = (mss/2) % 2;
else SND_PIN = (mss ) % 2;
} else
SND_PIN = 0;
}
else OUT_PIN = 1; // Lay down the signal if OnAir user param is off
}
}
}

```



corrente. Il valore predefinito è 0, ossia nessun rumore. Ricordiamo che la data e l'ora si riferiscono all'ora Locale, ricavata dall'ora Solare memorizzata nel modulo RTC con l'aggiunta di 1 ora durante l'Estate legale. E, come già spiegato, sarà 1 minu-

to avanti rispetto all'orario mostrato dagli orologi radio-controllati che hanno ricevuto e trattato l'informazione trasmessa. La **Fig. 10** propone un esempio con impostazioni personalizzate. In questo caso avremo impulsi allungati di 3 ms,

Elenco Componenti:

R1: 470 Kohm
R2: 4,7 Mohm
R3,R4: 1 Kohm
R5,R8: 1 Mohm
R6: 100 ohm
R7: 2,7 Kohm
R9: 1,5 kohm
R10: 47 Kohm
R11: 18 Kohm
R12: 2,2 Kohm
R13,R14,R15: 220 ohm
C1,C2: 47 pF ceramico
C3,C4: 1 nF multistrato
C5,C6,C7: 1 µF multistrato
C8: 22 µF 25 VL elettrolitico
C9: 220 µF 35 VL elettrolitico
C10: 220 µF 16 VL elettrolitico
C11: 100 nF multistrato
C12,C13: 22 pF ceramico
D1: SB140 Shottky
D2: LED bicolore Verde-Rosso c.c.
D3: 1N4007
U1: HFE4069UBP
U2: TDA2030
U3: 78L05
U4: PIC18F2620
LCD1: Display Hitachi 2x16
RTC1: DS3231-MODULE
T1: BC337
X1: Quarzo 77,5 KHz
X2: Quarzo 16 MHz
AP1: SPEAKER 8-32 ohm 25-35 mm
J1, J4: Connettore JST 2 Poli M 2 mm
J2: strip femmina 5 poli
J3: strip femmina 4 poli
J5: connettore 1 polo

Varie

- : strip femmina 5 poli per RTC1
- : dissipatore per U2
- : Circuito stampato S1550 (149x44)

l'altoparlante è muto (speaker barrato), nessuna applicazione dell'ora legale ("-"), trasmissione inattiva (antenna barrata), 6 impulsi di rumore introdotto casualmente. Premendo brevemente **P1** si entra nella Seconda pagina di visualizza-



Fig. 8
Schermata di avvio.

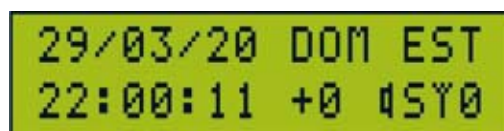


Fig. 9
Pagina principale.

zione, con il dettaglio del treno di impulsi inviati in trasmissione (**Fig. 11**). Il bit in Reverse indica quello in corso di trasmissione, facente parte del tema indicato alla fine della seconda riga, abbreviato a 6 caratteri e coerente con quello riportato nella tabella di codifica DCF77 presentata prima. Solo l'ora viene mostrata in questa pagina. Per tornare alla Pagina principale si deve premere nuovamente **P1**. La modifica dei campi data, ora, aggiunta, stagionalità, trasmissione e rumore avviene premendo **P2** fino a raggiungere il campo desiderato, che poi modificheremo premendo **P1** unitariamente per ogni singolo incremento o a lungo, per avanzare con continuità prima lenta e poi più rapida. La pressione lunga di **P2** fa uscire subito dalle impostazioni, cosa ottenibile anche continuando ad avanzare tra i campi fino alla fine. Fuori dalle impostazioni è possibile resettare il dispositivo premendo a lungo **P1**. LED ed altoparlante indicano visivamente e acusticamente il bit in trasmissione, rendendo possibile seguire in dettaglio il treno di modulazione, o trascriverlo senza guardare il dispositivo, grazie all'altoparlante, se abilitato. La codifica è luce Verde e suono Acuto per i bit "1", e Rossa/Grave per gli "0".

REALIZZAZIONE PRATICA

Passiamo adesso alla costruzione, che richiede un circuito stampato realizzabile a partire dalle tracce lato rame scaricabili dal nostro sito www.elettronica.it. La faccia inferiore ospita l'altoparlante, il circuito integrato TDA2030 con dissipatore, ed i connettori strip-line per i collegamenti esterni. Sul lato frontale del PCB vanno montati tutti gli altri componenti e i connettori per l'RTC e per il Display LCD, compatibile sia con modelli con retroilluminazione che senza. Bisogna prevedere un distanziale

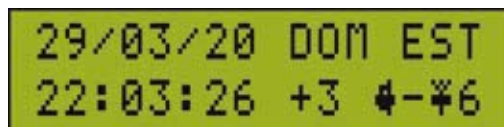
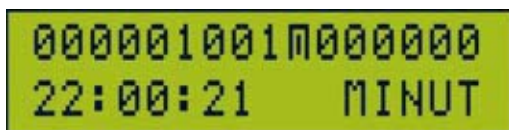


Fig. 10
Impostazioni personalizzate.

Fig. 11
Seconda pagina.



(va benissimo una guarnizione idraulica da 1/2" in cartone pressato) tra il dissipatore del TDA2030 ed il circuito stampato, per non toccare le saldature dei componenti circostanti. L'altoparlante può essere fissato con del biadesivo o colla a caldo. Si realizza un cavetto per collegare l'antenna, e per collegare un Jack femmina al connettore 12V nel verso corretto. Il diodo D3 sull'alimentazione, protegge la circuiteria da eventuali inversioni. Il layout del circuito stampato per il trasmettitore prevede un connettore a due poli a passo mil per l'alimentazione, che si è scelto di lasciare lontana per evitare interferenze, ed un altro per l'antenna in ferrite da posizionare internamente o al di fuori del mobile che si intende costruire. A tal proposito si raccomanda, nel caso si decidesse di ospitare l'antenna all'interno del mobile, di utilizzare plastica o altri materiali trasparenti alle onde radio. Un mobile in metallo farebbe da schermo, impedendo la trasmissione del segnale.

COLLAUDO

Una volta acceso il dispositivo, premete i pulsanti per regolare data ed ora su un tempo di test; la trasmissione parte immediatamente. Se avete un oscilloscopio collegatelo ad una antenna in ferrite posta nei paraggi e potrete visualizzare il segnale, ma non è fondamentale. Potete resettare un oro-

logio radio-controllato per osservare la ricezione in corso, mostrata di solito tramite un'icona significativa che lampeggia sul suo display. Tali orologi attendono due o tre minuti prima di convalidare il dato ricevuto. A sincronizzazione avvenuta vedrete un orario che sarà un minuto indietro rispetto a quello mostrato dal trasmettitore, ma ciò è normale alla luce di quanto detto nell'articolo. Se state costruendo un vostro orologio radio-controllato, questo dispositivo vi sarà prezioso durante i test, grazie alla sua capacità di introdurre disturbi ai quali vorrete sicuramente porre rimedio, ideando e sperimentando ingegnose politiche di correzione. È doveroso ricordarvi che la generazione di un segnale DCF77 che coinvolga uno stretto ambito domestico non pone problemi legali ma, laddove tale segnale superasse i confini della propria abitazione, si rischierebbe di incorrere in sanzioni. Tenetelo ben presente nell'utilizzo. □



Cosa occorre?

I componenti utilizzati in questo progetto sono di facile reperibilità. Il microcontrollore da programmare (cod. PIC18F2620-I/SP) è disponibile a Euro 9,00, mentre il PCB può essere realizzato utilizzando il servizio PCBPRODUCTION disponibile sul sito www.futurashop.it. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:

Futura Elettronica srl, Via Adige 11, 21013 Gallarate (VA)
Tel: 0331-799775 - <http://www.futurashop.it>

La scheda e la relativa antenna trasmettente in ferrite.



Scopri il mondo di ARDUINO e le sue infinite potenzialità!

Progetta, costruisci e divertiti con i moduli per Arduino predisposti per essere inseriti sui mattoncini della LEGO®. Puoi utilizzarli per robotica, con Arduino o Raspberry Pi, ecc. Nella confezione sono comprese, oltre ai sensori e agli accessori, anche una board Arduino Uno compatibile e una scheda Shield sperimentale dove puoi collegare in tutta semplicità i moduli sensori e attuatori. Una dispensa a corredo vi accompagna con gli sketch per testare i vari moduli con Arduino e la Sensor Shield.

*29 MODULI PER ARDUINO
COMPATIBILI CON LEGO®*

€ 79,00

Cod. ARDULEGOKIT

I MODULI PRESENTI NELLA CONFEZIONE:

- Modulo con display 7 segmenti a 4 cifre
- Modulo con sensore di fiamma
- Modulo con LED RGB 10 mm
- Modulo sensore di temperatura e umidità
- Modulo touch sensor
- Scheda sensore umidità terreno
- Scheda sensore livello liquidi
- Modulo con LED rosso
- Modulo con LED verde
- Modulo con LED giallo
- Modulo con sensore effetto di Hall
- Modulo con LED RGB SMD
- Modulo con sensore luce ambiente
- Modulo con pulsante rosso
- Modulo con pulsante giallo
- Modulo con interruttore a pulsante
- Modulo con buzzer senza elettronica
- Modulo sensore di vibrazione
- Modulo con sensore di temperatura
- Modulo con potenziometro
- Modulo con fotoresistenza
- Modulo con interruttore reed
- Modulo con ricevitore IR a 3 pin
- Modulo con buzzer con elettronica
- Modulo con encoder rotativo
- Modulo sensore scala di grigi
- Modulo con microswitch

Prezzi IVA inclusa



Starter kit V6 con Arduino Uno REV3



**IL PIÙ
COMPLETO!**

€ 65,00

Cod. ARDUKITV6

Acquista anche il kit
completo con tutti i
componenti necessari per
realizzare gli esperimenti
descritti nel libro "l'ABC
di ARDUINO".

Where are you?

Localizzatori GPS/GSM/GPRS per tutte le esigenze in grado di monitorare in tempo reale l'auto, la moto, la bicicletta, i bagagli e tanto altro.



Prezzi IVA inclusa.

LOCALIZZATORE GPS/GSM/GPRS CON ASCOLTO AMBIENTALE

Segui in tempo reale, tramite browser e APP gratuita, il percorso di veicoli e persone! Oltre a visualizzare i movimenti del tuo veicolo in tempo reale, potrai controllare la velocità, la direzione e i punti di sosta. Il localizzatore permette anche di gestire una flotta di autoveicoli con un unico account. Dispone di batteria ricaricabile agli ioni di litio da 6.000 mA che fornisce un'autonomia prolungata nel tempo (fino a 30 giorni in standby). Grazie al potente magnete di cui è dotato, è possibile fissarlo su superfici metalliche.



cod. FR763

€ 74,00



cod. TK102

€ 54,00

MICRO LOCALIZZATORE SATELLITARE GPRS/GSM

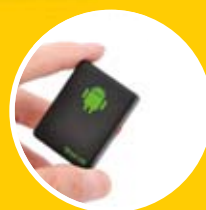
Tutto sotto controllo con il nuovo micro localizzatore GPRS/GSM. Per attivarlo basta inserire una SIM di qualsiasi operatore; inviando un SMS potrai attivare l'ascolto ambientale oppure ricevere un link con le coordinate che indicano la posizione in tempo reale del localizzatore. Dispone di batteria agli ioni di litio ricaricabile. Ideale per tenere sotto controllo la posizione di autoveicoli, persone, cose, animali domestici, bagagli, ecc. La confezione comprende: il micro localizzatore, il cavetto USB-micro USB per la ricarica della batteria e le istruzioni in italiano.

Dimensioni: 43,2x32x14 mm.
Peso: 20 grammi.



cod. A8TRACK

€ 14,00



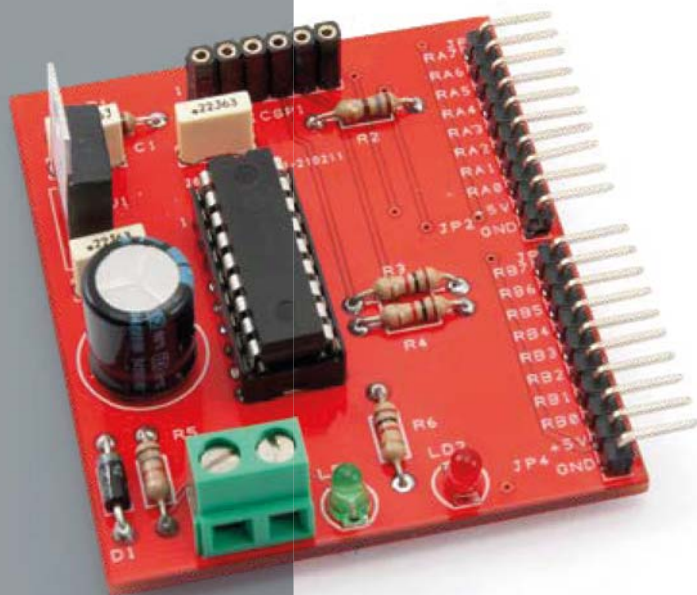
LOCALIZZATORE TASCABILE GPS/GSM/GPRS

Ideale per la localizzazione personale e veicolare. Può inviare le proprie coordinate (latitudine e longitudine) via SMS verso telefoni cellulari, oppure, tramite tecnologia GPRS, ad un computer opportunamente configurato. Dispone di microfono integrato e batteria ricaricabile.

OROLOGIO... DA LABORATORIO

di CRISTIANO RUGGERI

Mostra giorno, data, ora, è completamente programmabile e dispone di una sveglia e di un sensore di luce ambiente per regolare automaticamente la luminosità del display.



N

el numero scorso vi abbiamo introdotto la scheda LedKey, un'interfaccia utente con display e pulsanti ideale per implementare numerosi apparati che richiedono un dispositivo di input/output. Tra questi c'è sicuramente un orologio elettronico digitale, nel quale chiaramente occorre un visualizzatore

dell'ora ed eventuale data, nonché una serie di pulsanti per le impostazioni. Abbiamo quindi deciso di proporvi questo progetto, che possiamo considerare un evergreen, realizzando una scheda a microcontrollore da interfacciare alla LedKey e aggiungendo un qualcosa in grado di fornire ora e data stabili che verranno sia visualizzate, sia utilizzate per implementare delle sveglie. Ora e data sono generate e mantenute da una breakout board che integra un Real Time Clock modello DS1302, una piccola batteria al litio con relativo portabatterie, oltre a qualche componente di contorno.

➔ **Fig. 1**
La breakout board
DS1302RTC.



La scheda RTC (venduta dalla Futura Elettronica, codice DS1302RTC) solleva il microcontrollore dall'onere di usare l'orologio di sistema, perché infatti il micro si limiterà ad acquisire le stringhe di dati fornite dall'RTC e ad elaborarle sia per la funzione ora e data, sia per implementare la sveglia.

IL MODULO RTC

RTC significa Real Time Clock, ossia orologio in tempo reale, autonomo e capace di proseguire nel suo conteggio perché dotato batteria in grado di mantenerlo in funzione anche in assenza dell'alimentazione principale.

Il modulo utilizzato (**Fig. 1**) ha come cuore l'integrato DS1302, il quale contiene una RAM statica da 31 byte, che viene letta e/o scritta per ricavare o impostare l'ora, più una serie di contatori che realizzano un efficiente e completo orologio-calendario il quale fornisce secondi, minuti, ore, giorno della settimana, data, mese e anno con correzione per gli anni bisestili fino al 2100. Può operare con formato 12/24 ore e con indicazione AM/PM, supporta la scrittura/lettura a singolo

byte oppure di tutta la memoria in una volta (burst mode). Dispone di una batteria di backup con cui continua ad operare (anche per molti anni, visto il consumo bassissimo di solamente $1\mu W$) quando viene a mancare la fonte primaria di alimentazione, e comunica col microprocessore tramite una seriale sincrona SPI di tipo MISO/MOSI a 4 o 3 fili. Può essere alimentato con una tensione compresa tra 2 e 5,5 V, il che lo rende utilizzabile praticamente con qualsiasi piattaforma a microprocessore. Giusto per completezza, diciamo che dispone anche di un efficiente sistema di ricarica della batteria di backup, chiamato Trickle-Charge, che però in questo caso non viene utilizzato poiché la batteria di backup è una litio non ricaricabile.

Nel nostro progetto, come vedrete più avanti, lo utilizzeremo in modalità a tre fili; il principio di funzionamento della seriale sincrona è identico a quello descritto nel fascicolo n° 252.

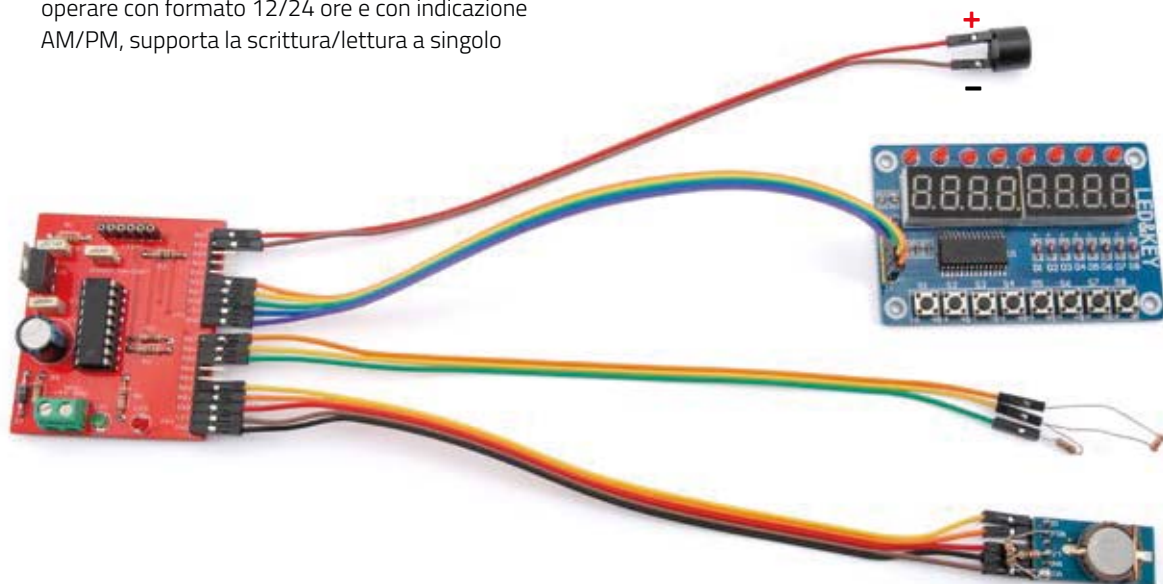
IL CIRCUITO

Esaurito l'argomento "RTC" passiamo a descrivere l'orologio-sveglia che vi proponiamo, che è l'insieme delle parti visibili nella **Fig. 2**, nonché nello schema di cablaggio che trovate nelle prossime pagine; il tutto è coordinato dalla scheda a microcontrollore (MCU) della quale proponiamo e descriviamo lo schema elettrico, prima di passare agli aspetti costruttivi.

SCHEMA ELETTRICO SCHEDA MCU

E' basata su un PIC16F88 a 18 pin, ma al posto di questo può essere utilizzato qualsiasi PIC che abbia la stessa pin-out (ce ne sono parecchi).

➔ **Fig. 2**
La scheda a
micro-controllore
(a sinistra) collegata
alla LedKey, all'RTC,
al fotoreistore e al
cicalino.



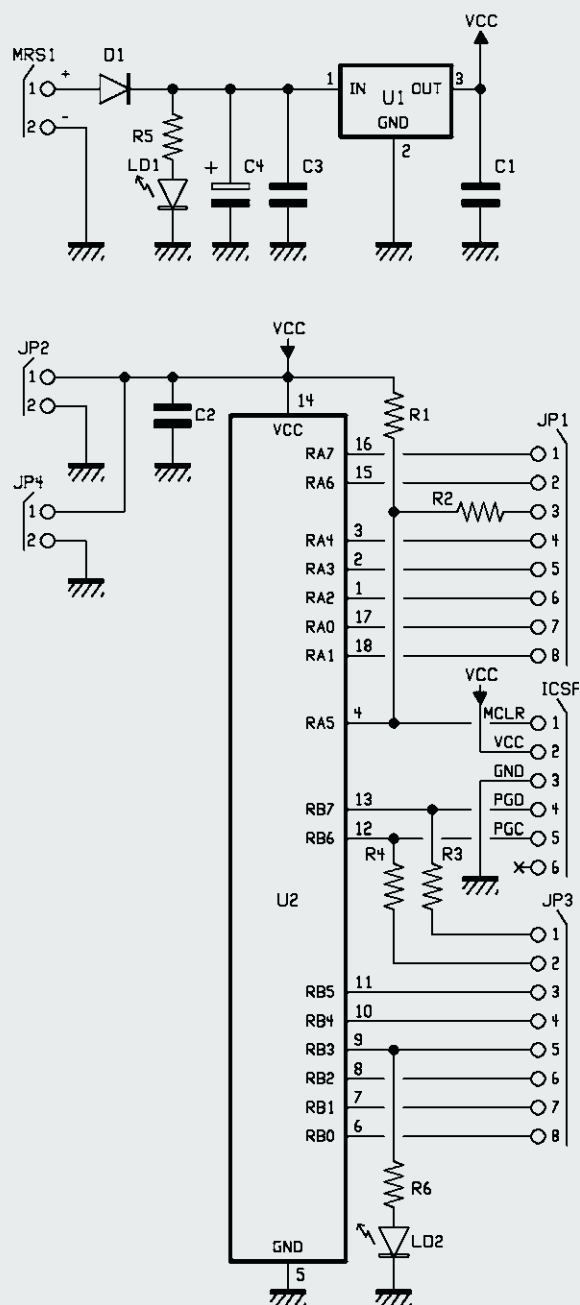
Tutti i pin di I/O sono stati portati all'esterno sui due connettori JP1 e JP3, entrambi dei pin strip a 8 pin. Ad essi sono stati affiancati i connettori JP2 e JP4, a due pin, che forniscono due alimentazioni a 5 V e due masse. Gli I/O sono disposti in ordine, da RB0 a RB7 e da RA0 a RA7. Poichè i pin RB7 e RB6 svolgono anche la funzione di dati (PGD) e clock (PGC) per l'in circuit serial programmer (connettore ICSP, cui si può collegare un pickit2, 3, 4 o compatibile), sono stati disaccoppiati dall'uscita attraverso due resistenze (R3 e R4) da 1kohm, le quali proteggono eventuali utilizzatori dalle transizioni determinate dal programmatore.

Abbiamo portato esternamente anche la linea RA5 (-MCLR/VPP) attraverso una resistenza da 180 ohm (R2) in quanto tale linea può essere utilizzata sia come reset che come input. Se usata come reset, la resistenza da 180 ohm è sufficientemente bassa da garantirne il funzionamento corretto quando tirata a massa. Se usata come ingresso, invece, sarebbe meglio portare R2 ad almeno 470 ohm, valore adatto a proteggere una eventuale uscita logica collegata ad essa, quando durante la programmazione la tensione sul pin RA5 sale fino a 12V. In pratica, il discorso è analogo a quanto visto per RB7 e RB6.

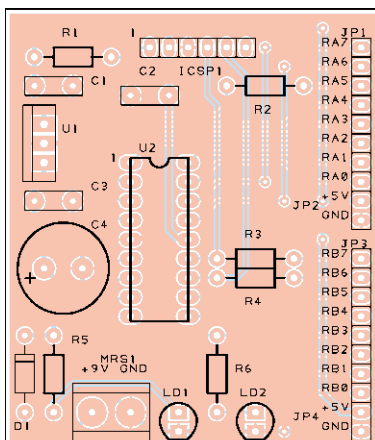
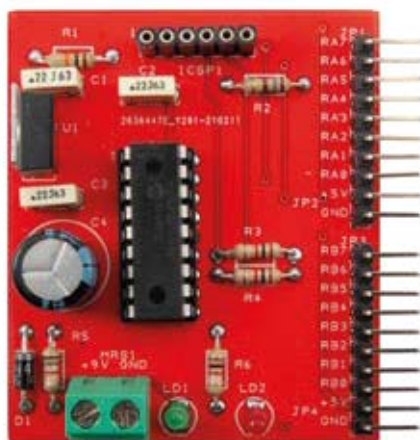
Per alimentare il micro ed eventuali utilizzatori aggiuntivi (nel nostro caso scheda LedKey e RTC) abbiamo anche inserito un alimentatore stabilizzato a 5V, secondo lo schema che ormai utilizziamo senza problemi da anni. Partiamo da sinistra con la morsettieria MRS1, cui applichiamo la tensione di alimentazione di 9÷12VDC; segue il diodo D1, un 1N4004 o 1N4007 o simile, che protegge tutta la scheda da inversioni di polarità accidentali in ingresso. Proseguendo troviamo i condensatori C3 e C4 (100nF e 470 µF 25V) all'ingresso di U1, il classico regolatore di tensione 7805. All'uscita di U1 troviamo C1 e C2, entrambi da 100nF, uno a filtraggio dell'uscita del regolatore, e l'altro collegato tra GND e VCC del microcontrollore U2 (il nostro 16F88). Completano il circuito la resistenza di reset R1 da 4,7 kohm/10 kohm, il led di segnalazione di presenza di alimentazione (R5 da 1,8 kohm e LD1 verde da 3 mm) ed un led rosso di attività (o altro, LD2), collegato direttamente alla porta RB3 tramite la resistenza R6 da 1 kohm.

REALIZZAZIONE PRATICA

La scheda di controllo del sistema, quindi quella a microcontrollore, è un circuito piuttosto semplice, pertanto la realizzazione artigianale con press'n'peel, fotoincisione, CNC o altro è alla



portata di chiunque. Potete anche chiedere alla Futura Elettronica di fornirvi il circuito stampato già pronto tramite il servizio PCBPRODUCTION disponibile sul sito www.futurashop.it. Quale che sia il modo in cui lo avete ottenuto, non ci sono regole precise per montarlo: i componenti sono veramente pochi, più o meno tutti della stessa altezza, quindi montateli come meglio credete, rispettando polarità e valori. Noi consigliamo di montare lo



Elenco Componenti:

D1: 1N4007	U2: PIC16F88 (o altro con medesima pinout) + zoccolo 9+9 DIL
LD1: led verde 3mm	
LD2: led rosso 3mm	
C1, C2, C3: 100nF÷220nF, poliestere 5mm	Varie: ICSP1: pin strip femmina passo 2,54mm 6pin JP1+JP2, JP3+JP4: pin strip maschio (dritto o 90°) passo 2,54
C4: 470uF elettrolitico, 25V	MRS1: morsettiera 2 morsetti Passo 5,08
R1: 4,7kohm÷10kohm 1/4W	
R2: 180÷470 1/4W	
R3,R4,R6: 1kohm 1/4W	
R5: 1,8kohm 1/4W	
U1: uA7805	

zoccolo per primo e il regolatore e il condensatore elettrolitico per ultimi. Sistemata la scheda a microcontrollore possiamo collegare tra loro le varie parti che compongono questo bizzarro progetto didattico: per fare ciò occorre riferirsi allo schema di cablaggio. Si tratta di semplici collegamenti realizzabili con spezzoni di flat-cable (o cavetti da laboratorio), ricordando che l'RTC richiede due resistenze di pull-up da 1 kohm sulle linee SCLK e I-O e che al pin. Inoltre la fotoresistenza si connette fra RB5 ed RB6 e tra questo pin ed RB5 va posta una resistenza da 10 kohm.

IL FIRMWARE

Abbiamo scritto delle "librerie" per facilitare l'uso dell'RTC nei progetti PicBasic, limitandoci al minimo indispensabile, ossia la definizione delle variabili necessarie, contenute nel file RTC_DS1302_VARS.PBP:

```

RTC_YEAR Var byte 'variabili data e ora da RTC
RTC_DAY Var byte '
RTC_MONTH Var byte '
RTC_DATE Var byte '
RTC_HOUR Var byte '
RTC_MIN Var byte '
RTC_SEC Var byte '
RTC_COMMAND Var byte '

```

e le uniche due subroutine per leggere e scrivere l'RTC in burst mode, ossia tutto in una volta, contenute nel file RTC_DS1302_SUBS.PBP (Listato 1). Avremmo anche potuto creare delle routine per impostare la trickle charge, oppure il modo operativo a 12/24 ore, piuttosto che la scrittura/lettura a singolo byte, ma perchè privarvi del piacere della sperimentazione? Noi abbiamo costruito un punto di partenza e voi potrete divertirvi ad evolverlo e

perchè no anche migliorarlo. Non dimentichiamoci che il nostro scopo è la didattica, ed il soggetto di queste pagine a noi pare molto didattico. L'utilizzo di questa mini libreria è estremamente semplice: come già visto per l'articolo relativo al LedKey, dopo aver definito l'assegnazione dei pin di IO alla scheda (nel nostro progetto le righe di codice sono quelle che seguono)

```

symbol IO = PORTb.0 'dati RTC
symbol SCLK = PORTb.1 'clock RTC
symbol RST = PORTb.2 'reset RTC
symbol STROBE = PORTa.0 'strobe per il display
symbol SCL = PORTa.1 'clock per il display
symbol SDA = PORTa.2 'dati per il display

```

occorre includere il file RTC_DS1302_VARS.PBP, all'inizio del vostro progetto, dove avete definito tutte le variabili, facendo attenzione al percorso (path).

```

INCLUDE "modedefs.bas"
INCLUDE "LED_KEY_VARS.PBP" 'variabili ledkey
INCLUDE "RTC_DS1302_VARS.PBP" 'variabili per RTC

```

Analogamente, le routine verranno inserite in coda al codice sorgente della vostra applicazione.

```

INCLUDE "LED_KEY_SUBS.PBP" ' routine per LEDKEY
INCLUDE "RTC_DS1302_SUBS.PBP" ' routine per RTC

```

Per comodità, il punto ideale dove inserire le librerie è la stessa cartella che include in vostro codice sorgente. L'utilizzo delle due routine è estremamente semplice:

- se si vuole conoscere tutto il pacchetto data-ora si deve invocare la routine GET_TIME (gosub GET_TIME) e successivamente andare a

- leggere le variabili RTC_YEAR, RTC_DAY...
- se si vuole impostare data-ora, occorre scrivere i nuovi valori nelle variabili RTC_YEAR, RTC_DAY... quindi invocare la routine SET_TIME (gosub SET_TIME).

Si deve solamente prestare attenzione ad un piccolo particolare: tanto in lettura quanto in scrittura, tutti i parametri sono impostati in modo "decimale" dal punto di vista dell'interpretazione umana, ma in variabili "esadecimale".

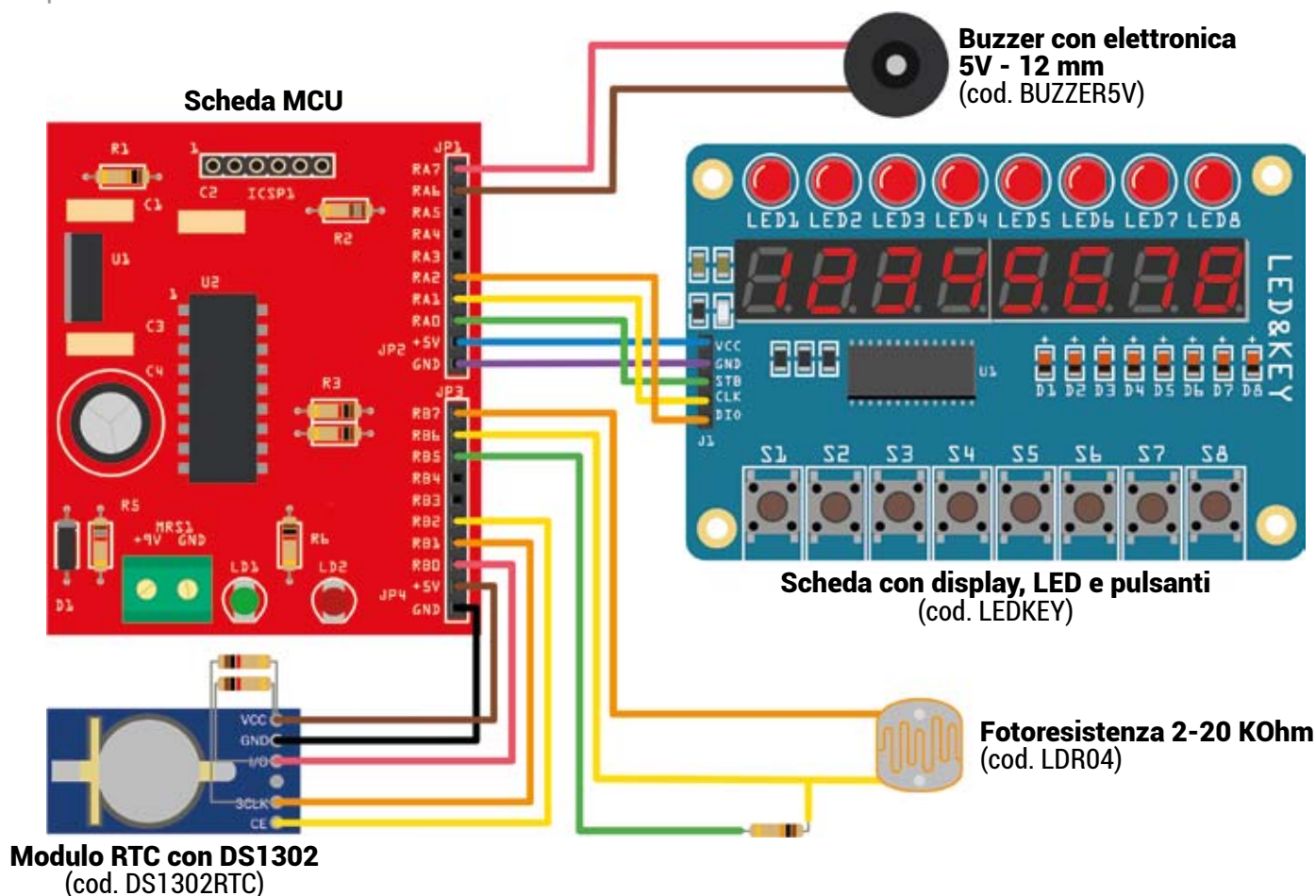
Non vi preoccupate: la cosa è più facile di come sembra e alcuni semplici esempi vi aiuteranno. Immaginiamo di voler impostare 08.30.51 di LUNEDÌ 22 novembre 2047 (valori casuali, non verificate che sia una data valida), non dobbiamo fare altro che scrivere queste righe di codice:

```
RTC_YEAR = $47 ' anno 2047
RTC_MONTH = $11 ' novembre
RTC_DATE = $22 ' giorno 22
RTC_DAY = $01 ' lunedì
RTC_HOUR = $08 ' ore 8
RTC_MIN = $30 ' 30 minuti
RTC_SEC = $51 ' e 51 secondi
```

quindi invocare la SET_TIME con "gosub SET_TIME". In pratica, il nibble basso contiene sempre le unità ed il nibble alto sempre le decine, ma l'escursione massima di ogni nibble è espressa solamente con i primi 10 caratteri (da 0 a 9) del set esadecimale. Potremmo anche scrivere

```
RTC_YEAR = $FA ' in che anno siamo???
RTC_HOUR = $38 ' in un giorno abbiamo 38 ore?
```

schema di CABLAGGIO



↓ Listato 1

```
*****
' Subroutine che scrive RTC
SET_TIME:
    RST = 1                    ' pronto per il trasferimento
    Shiftout IO, SCLK, LSBFIRST, [$8e, 0]    ' abilita scrittura
    RST = 0                    ' Reset RTC
    RST = 1                    ' pronto per il trasferimento
    ' Scrive gli 8 registri dell'RTC in burst mode
    Shiftout IO, SCLK, LSBFIRST, [$be, RTC_SEC, RTC_MIN, RTC_HOUR, RTC_DATE, RTC_MONTH, RTC_DAY, RTC_YEAR, 0]
    RST = 0                    ' Reset RTC
    Return

*****
' Subroutine che legge da RTC
GET_TIME:
    RST = 1                    ' pronto per il trasferimento
    Shiftout IO, SCLK, LSBFIRST, [$bf]        ' comando che inizia la lettura.
    ' Legge gli 8 registri dell'RTC in burst mode
    Shiftin IO, SCLK, LSBPRE, [RTC_SEC, RTC_MIN, RTC_HOUR, RTC_DATE, RTC_MONTH, RTC_DAY, RTC_YEAR, RTC_COM-
MAND]
    RST = 0                    ' Reset RTC
    Return

*****
```

ma questo genererebbe solamente dei conteggi sballati e privi di significato. Col tempo il DS1302 si riallineerebbe in automatico, però il contatore degli anni impiegherebbe "anni" per riallinearsi, così come quello delle ore impiegherebbe "ore" appunto. Pertanto è meglio evitare la scrittura erranea di dati omettendo l'identificatore di esadecimale. Se poi vorrete saperne di più a riguardo di come noi abbiamo gestito la conversione da esadecimale a decimale visualizzata, potrete analizzare il codice sorgente, fornito gratuitamente come sempre. Naturalmente se riuscirete (e siamo certi che ce la farete) a migliorarlo e ottimizzarlo, ne saremo molto felici: il nostro scopo è quello di trasferire ciò che con gli anni e con l'esperienza abbiamo imparato a chiunque ne voglia godere. Se per caso qualcuno dovesse fare meglio di noi, beh, vorrebbe dire che siamo stati bravi nel trasferire le nostre conoscenze.

UTILIZZO IN PRATICA

Per prima cosa dovete programmare il micro con il codice HEX che avrete preventivamente scaricato dal nostro sito www.elettronica.in.it. Potrete usare il programmatore e il microcontrollore che preferite (dopo aver adattato il codice al vostro scopo);

noi abbiamo usato il consolidato ed affidabile PIC16F88 insieme ad un pickit2. Potete programmare il PIC a circuito completamente montato, così come prima del montaggio. Giusto per avere la certezza di aver fatto tutto correttamente, potreste alimentare a 9V il circuito del micro ed attendere qualche secondo che il led rosso inizi a lampeggiare a circa 4 Hz, indicando che sia la scheda che il codice sono a posto. Doveste vedere solo il led verde acceso, qualcosa non va da qualche parte, ma siamo certi che non sarà questo il caso. Spegnete tutto, ultimate i collegamenti, quindi accendete il sistema che deve partire immediatamente. All'accensione il display è completamente acceso ad una luminosità intermedia. Dopo qualche secondo compare la scritta "Elett.In" che gradualmente viene portata alla luminosità massima per 5 secondi. Quindi il sistema inizia a visualizzare data, ora e giorno analizzando le unità di secondi con questa logica:

- secondo 0 e 1: mostra il giorno della settimana
- secondo 2 e 3: mostra la data
- secondi da 4 a 9: mostra l'ora

Naturalmente, la scheda RTC potrebbe contenere dei dati sballati, specie se si tratta della prima ac-



censione. Per questa ragione il software si accorge se si tratta di una prima accensione in assoluto, quindi programma l'RTC con una data predefinita (ore 07.29.50 del 01.01.2021, Domenica, sveglia ore 07.30). Tenete presente che se avete verificato in precedenza il corretto funzionamento del micro senza aver collegato le varie periferiche, questa operazione di prima accensione in assoluto è già stata fatta, quindi occorrerà riprogrammare il micro da spento e riaccenderlo con tutte le periferiche collegate affinché essa venga ripetuta. Altrimenti potrete procedere con l'impostazione manuale di tutto l'orologio: vediamo come. Per prima cosa capiamo come interpretare i led e le scritte, ma più i led in alto, visto che le scritte sono abbastanza autoesplicative.

- il primo LED a sinistra in alto indica se la sveglia è attiva oppure no. Se acceso, la sveglia è attiva.
- I successivi 7 led indicano il giorno della settimana, da lunedì a domenica.

Poichè i display sono a 7 segmenti, i numeri sono ben chiari così come la maggior parte delle lettere, però alcune, quelle particolarmente composte, sono state approssimate e necessitano di un minimo di interpretazione, ma siamo certi non avrete difficoltà. Vediamo ora come sono stati utilizzati gli 8 tasti.

TASTO 1 (primo a sinistra): in funzionamento normale attiva o disattiva la sveglia. In modalità programmazione non ha alcuna funzione.

TASTO 2: entra in modalità impostazione allarme. Esce per timeout dopo 20 secondi di inattività.

TASTO 3: entra in modalità impostazione data ora. Esce per timeout dopo 20 secondi di inattività.

TASTO 4: in modalità impostazione, seleziona il campo da variare. Quando si imposta l'allarme, esso, alterna tra ora minuti e visualizzazione ora. Se in impostazione data ora, scansiona i sette campi modificabili, partendo dal giorno della settimana per terminare con i secondi.

TASTO 5: incrementa di 1 i giorni della settimana e di 10 ogni altro campo.

TASTO 6: incrementa di 1 ogni campo tranne il giorno della settimana, dove non ha effetto.

TASTO 7: salva ed esce. Mostra la scritta lampeggiante "Fatto".

TASTO 8: in impostazione esce senza salvare. In modalità normale spegne/accende il display. Riportiamo alcune note di funzionamento che potrete verificare sia sperimentalmente che leggendo il codice sorgente.

- Se spegnete il display per il notturno con il tasto 8, esso rimarrà spento fino a quando non lo

riattiverete con lo stesso tasto oppure fino a che non si verificheranno degli eventi precisi che lo porteranno automaticamente all'accensione. Essi sono la sveglia (tasto 1 con led 1 acceso) e l'entrata in programmazione (tasti 2 e 3).

- La fotoresistenza rileva la luce ambientale ed il micro, che ne controlla in tempo reale lo stato, decide di impostare la luminosità della LedKey tra tre livelli, basso, medio e alto. I livelli che abbiamo provato sperimentalmente con la nostra fotoresistenza sono scritti nelle costanti:

<code>LIGHT_LOW</code>	con 500	' luce bassa
<code>LIGHT_HIGH</code>	con 800	' luce alta

ma sono validi solo per la nostra di fotore-sistenza. Con altri modelli potrebbe essere necessario ritoccare tali valori.

- Durante l'impostazione dei vari campi, il sistema tiene conto delle relative escursioni massime quindi impedisce di impostare valori differenti. Tiene inoltre conto dei giorni per ogni mese e del 29 febbraio per l'anno bisestile. Ricordiamo che un anno è bisestile quando è divisibile per 100 o per 4 con resto 0 (2000, 2004, 2008..., 2024..., 2100).

ACCORGIMENTI FIRMWARE

Volendo evitare di collegare più fili agli stessi pin, come ad esempio il +5 e la massa, ci siamo inventati un sistema poco convenzionale, ma non per questo non utilizzabile o pericoloso, per ricavare un +5V o una massa funzionali (massa e +5V funzionale, ossia qualcosa che pur non essendo massa o a +5V ha potenziale di 0 volt o 5V).

Abbiamo utilizzato questo sistema per collegare il buzzer e il partitore della fotoresistenza. Come visibile dallo schema di cablaggio, il positivo del buzzer è collegato direttamente al pin RA7, mentre il negativo al pin RA6; poichè questo buzzer è in grado di funzionare da 3 a 12 V, con un consumo di corrente di pochi mA, portando a livello logico 1 (circa 5V) RA7 e a livello logico 0 RA6 (circa 0V rispetto a massa) esso risulterà alimentato e quindi suonerà, pur non risultando elettricamente collegato ad alcuna massa e +5V, bensì solo a due pin del micro. Il codice per fare ciò è molto semplice. Innanzitutto definiamo i pin RA7 e RA6

<code>symbol BUZZER</code>	<code>= PORTa.7</code>	' buzzer sveglia
<code>symbol LOW_BUZZER</code>	<code>= PORTa.6</code>	' buzzer sveglia

quindi in un punto qualsiasi dell'avvio portiamo a



▼ Listato 2

```
ADCIN 5, DAY_LIGHT          ' legge luce ambiente
                              ' e seleziona luminosità su tre livelli.

if DISP_STATUS = IS_OFF then ' se display spento
    DISP_BRIGHT = BRIGHT_0   ' Spegnimento
else                          ' se invece acceso
    if DAY_LIGHT <= LIGHT_LOW then DISP_BRIGHT = BRIGHT_1 ' luce molto bassa
    if DAY_LIGHT > LIGHT_LOW then
        if DAY_LIGHT <= LIGHT_HIGH then
            DISP_BRIGHT = BRIGHT_3 ' luce media
        endif
    endif
    if DAY_LIGHT > LIGHT_HIGH then DISP_BRIGHT = BRIGHT_7 ' luce molto forte
endif
gosub DISPLAY_BRIGHT        ' aggiorna display.
```

zero logico le due linee di comando per spegnere il cicalino:

```
low LOW_BUZZER          ' negativo buzzer a 0
low BUZZER              ' comando buzzer a 0
```

A questo punto, per far suonare il buzzer sarà sufficiente portare a 1 o 0 la linea BUZZER, senza toccare più la linea LOW_BUZZER.

Lo stesso principio vale per i segnali della lettura della fotoresistenza:

```
SYMBOL POS_PHOTO = portb.7
' Positivo partitore, lato fotoresistenza
SYMBOL NEG_PHOTO = portb.5
' Negativo partitore, lato resistenza 10k
```

Qui però occorre anche dire al sistema che RB6 è ingresso analogico (AN5, posizionato su RB6), e lo facciamo con le linee di codice:

```
TRISB = %01000000          ' RB6 ingresso
ANSEL = %00100000          ' AN5 analogico
```

Ora impostiamo "l'alimentazione" del partitore con le linee di codice:

```
low    neg_photo          ' imposta il negativo
high   pos_photo          ' imposta il positivo
```

Quindi è sufficiente leggere in runtime l'ingresso AN5 ed agire in conseguenza al valore letto (**Listato 2**).

CONCLUSIONI

Con questo articolo abbiamo voluto mostrarvi uno dei tanti utilizzi di un'interfaccia utente (UI User Interface in inglese) tanto comoda quanto versatile come LED Key. L'abbinamento ad un semplice microcontrollore come il PIC16F88 permette di gestirla facilmente lasciando anche diversi PIN liberi per poter collegare altri tipi di sensori. Non vi resta che sperimentare. □



Cosa occorre?

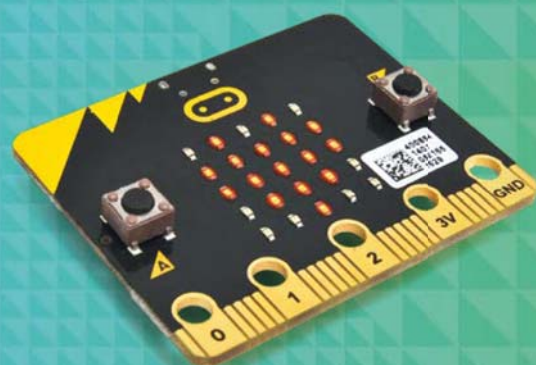
I componenti utilizzati in questo progetto sono disponibili presso Futura Elettronica. La scheda con display, led e pulsanti (cod. LEDKEY) è disponibile a Euro 5,90. Il microcontrollore PIC16F88 (cod. PIC16F88-I/P) è in vendita a Euro 6,50, la breakout con DS1302 (cod. DS1302RTC) costa Euro 4,80. Per chi volesse realizzare il PCB del controllo del display, è disponibile il servizio PCBPRODUCTION sul sito www.futurashop.it. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)
Tel: 0331-799775 - <http://www.futurashop.it>



TOBBIE II



di GRETA GALLI

Sperimentiamo la robotica con un piccolo automa governato dalla board micro:bit interfacciata ad uno smartphone.

Avete mai valutato l'idea di avere un robot come amico? No, non ci stiamo referendo a qualche cartone animato o a qualche serie TV futuristica, ma di farlo nella realtà. Ebbene, se la vostra risposta propende per il sì, ci teniamo quindi a presentarvi Tobbie II, un robot interamente programmabile per fare tutto ciò che vogliamo o per tenerci semplicemente compagnia e distrarci. La caratteristica principale di questo robot è che oltre ad essere il fratello maggiore di Tobbie, un robot a sei zampe in grado di interagire con l'ambiente circostante tramite dei programmi già predisposti che sfruttano dei sensori infrarossi, Tobbie II possiede al suo interno una scheda programmabile micro:bit, la quale gli permette di essere più avanzato del

predecessore, dato che è possibile programmarlo a piacimento per fargli svolgere numerosi compiti e attività.

UNO SGUARDO ALL'INTERNO DEL KIT

Tobbie II viene venduto in un kit tutto da montare come se fosse un modellino, quindi i pezzi sono tutti uniti a delle grandi basi di plastica pronti per essere staccati, grazie all'ausilio di un tronchesino, e incastrati tra di loro senza la necessità di usare della colla, inoltre all'interno del kit troviamo anche due motori che consentono i movimenti principali di Tobbie II: la camminata e la rotazione del busto. Le istruzioni di montaggio, fatte ad immagini, consentono una costruzione molto facile. Si parte innanzitutto dalla base di Tobbie II, ovvero dai due pezzi contenenti i motori. Infatti Tobbie visto da davanti e dall'esterno è un robot quasi del tutto simmetrico, ma all'interno del busto le posizioni degli ingranaggi sono diverse in base al movimento che dovrà compiere quel motore.

Possiede sei gambe divise in due gruppi alternati tra di loro, ovvero durante il movimento tre gambe sono appoggiate a terra mentre le altre tre avanzano in aria per poi appoggiarsi a terra e consentire alle altre tre di alzarsi e di ritornare successivamente a terra.

Il motore, che compie il movimento rotativo, è collegato ad un perno che rimane sempre fisso, mentre la rotazione del busto avviene nella parte finale del perno consentendo così al robot di girare nella direzione esatta senza mai sbagliare.

Una volta montato il corpo il passaggio successivo sarà quello di dotare Tobbie II di un cervello, ovvero di micro:bit. Questa piccola scheda a microcontrollore (inclusa nel kit) è inserita in un apposito shield che consente alla board di gestire i motori e i sensori. La meccanica è strutturata in modo che i due pulsanti presenti su micro:bit vengano premuti azionando due perni posti sui lati della testa di Tobbie II. Sulla sommità del casco abbiamo un'apertura per permettere il collegamento di micro:bit tramite un cavo micro USB (utile per programmare la scheda velocemente senza doverla rimuovere

dal robot). L'alimentazione è data da un vano batterie di quattro pile AAA accessibile dall'orecchio sinistro. Nella parte anteriore della testa si trovano poi due sensori ad infrarossi per creare diverse attività e infine, collegato alla scheda troviamo anche un buzzer che permette di emettere vari suoni. Una volta strette tutte le viti a stella, il vostro Tobbie II è pronto per essere programmato!

LA SCHEDA PROGRAMMABILE MICRO:BIT

Micro:bit non è altro che una board a microcontrollore, tipo Arduino per intenderci, ma più facile da utilizzare; è stato sviluppato dalla BBC proprio per i bambini ed è programmabile a blocchi. A differenza di Arduino possiede dei pin d'uscita che possiamo considerare "invisibili", infatti sulla parte inferiore troviamo una sezione color oro con cinque fori e i 24 pin disponibili sono tutti lì, collegabili tramite dei cavi con un connettore a coccodrillo o espandibili in stile Arduino (pin) grazie ad un apposito shield; invece i cinque fori sono dedicati all'utilizzo di soli 3 pin (0, 1, 2) più VCC (3V) e GND, collegabili grazie a dei cavi con morsetto a banana.

Sulla parte anteriore si trovano 25 led rossi, disposti a matrice, programmabili per ottenere scritte, faccine o immagini e inoltre ci sono i due pulsanti denominati A e B già saldati sulla scheda.

Nella parte posteriore invece troviamo i vari componenti di micro:bit, un codice ID identificativo della scheda, una porta per il cavo micro USB, un tasto per il reset e un connettore per un'eventuale alimentazione esterna.

Infine la scheda è dotata di un modulo Bluetooth, un accelerometro per riconoscere i vari movimenti, come ad esempio lo scuotimento, una bussola per capire l'orientamento del robot e un sensore di temperatura.

IL CONTROLLO DI TOBBIE II

Per controllare Tobbie II esistono due metodi differenti: il primo passa attraverso l'applicazione gratuita scaricabile sia su Android che su iOS, mentre il secondo è tramite la programmazione fatta sul sito <https://microbit.org>.

Fig. 1
Una volta connessa, la scheda di Tobbie II dovrebbe apparire tra le risorse del computer come unità a disco MICROBIT.



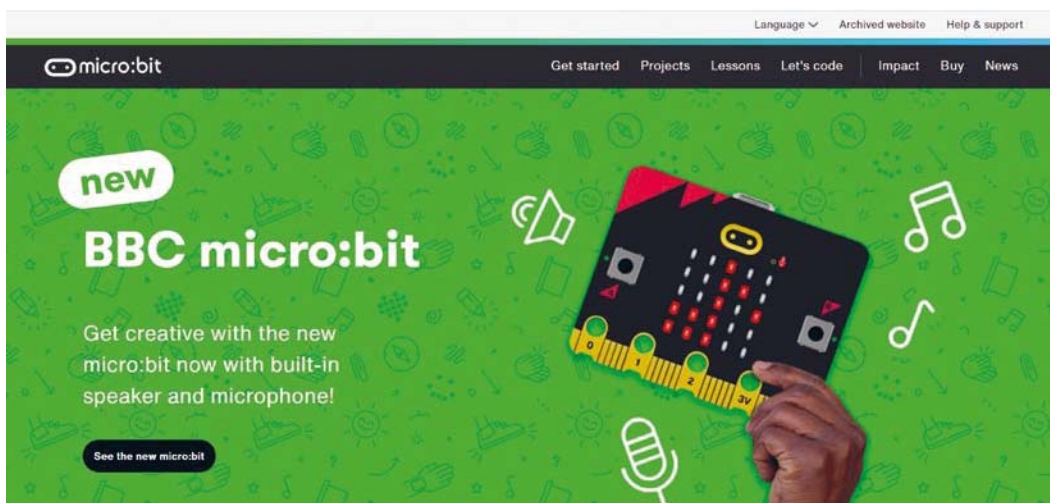


Fig. 2
Sito di
micro:bit.

Tobbie II è fornito con dodici programmi pronti all'uso per imparare ad utilizzarlo (trovate il relativo file .zip direttamente nel sito futurashop.it).

In questi programmi, tra i più interessanti, troviamo un "Follow me", ovvero un programma per far sì che il nostro "robotto" ci segua, un classico "Sasso, carta e forbice", oppure, contenuto nel dodicesimo file, denominato "APP remote control" troviamo il programma che permette di comandare da remoto Tobbie II.

Per memorizzare un programma su micro:bit bisogna collegare la scheda tramite cavo USB al computer e poi trascinare il programma .hex nella sezione dedicata sui dispositivi connessi al computer (Fig. 1).

LA PROGRAMMAZIONE

Per programmare Tobbie II è disponibile la piattaforma online gratuita di micro:bit (Fig. 2) rag-

giungibile dall'indirizzo <https://microbit.org> nella sezione "Let's code" e qui potrete scegliere in quale linguaggio programmare: se a blocchi, in Javascript o in Python (Fig. 3). Il bello della piattaforma è che supporta vari linguaggi, tra i quali sicuramente quello a blocchi è il più semplice per dei principianti. Nella sezione blocchi selezioniamo "nuovo progetto" per poi andare su Avanzati>Estensioni e cerchiamo "Tobbie" nella barra delle ricerche per aggiungerla all'IDE. Adesso nella sezione dei blocchi disponibili ci sarà quella dedicata a Tobbie II. I blocchi riportati sono suddivisi in apposite sezioni differenziate dai colori: i blocchi azzurri, denominati "fondamentali", servono per l'accensione dei led seguendo dei disegni già prefissati o per la creazione dei nostri disegni o delle nostre scritte; i blocchi viola sono dedicati agli input di micro:bit che, come visto prima, possono essere i due pulsanti o la posizione della scheda; i blocchi rossi servono per

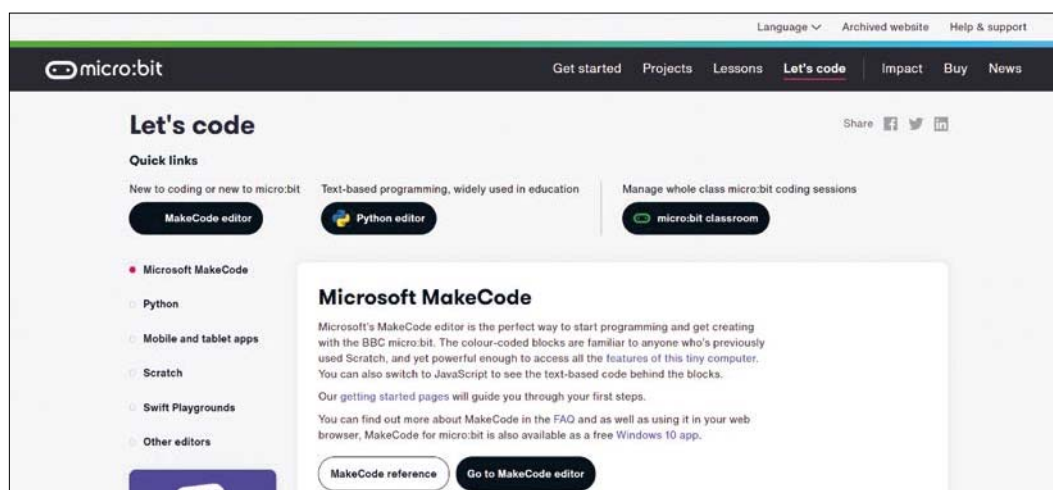
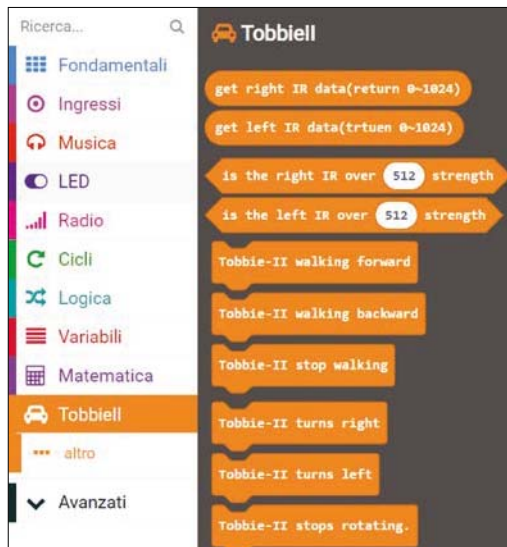


Fig. 3
Sezione del
sito micro:bit
nella quale
scegliere
con quale
linguaggio
programmare.

Fig. 4
Blocchi di
programmazione
predisposti per
Tobbie II.



la musica, ovvero per i suoni che il nostro buzzer può emettere; quelli verdi sono dedicati ai cicli inseribili nei vari programmi (while, do while, for); infine quelli azzurri contengono istruzioni come l'if. La parte di color porpora è dedicata alla creazione delle nostre variabili, mentre la parte viola serve per eventuali operazioni matematiche fatte con le nostre variabili e, infine, i blocchi arancioni sono dedicati a Tobbie (Fig. 4), infatti contengono i suoi movimenti e i suoi input tramite i due sensori ad infrarossi posti sotto la visiera.

La cosa interessante della programmazione a blocchi tramite questo sito è che mentre si sta programmando a blocchi il programma viene automaticamente convertito in JavaScript e viceversa, così da permettere anche ai meno esperti di avvicinarsi

ad un nuovo linguaggio di programmazione.

I programmi disponibili per Tobbie II sono quelli elencati di seguito.

- 00.** Assembly-Checker.hex, che serve per il primo avvio.
- 01.** Explorer.hex: esplora l'ambiente circostante.
- 02.** Follow_Me.hex: il robot vi seguirà ovunque andrete.
- 03.** Singing-Dancing.hex: il robot inizierà a cantare e ballare.
- 04.** Light-Tracker.hex, per seguire una luce.
- 05.** Thermometer.hex diventa un termometro digitale.
- 06.** Calculator.hex: Tobbie II diventa una calcolatrice in grado di fare delle operazioni elementari tramite i due pulsanti.
- 07.** Rock_Paper_Scissors.hex: corrisponde al gioco sasso carta forbice.
- 08.** Dice.hex estrarrà un numero a caso da 1 a 6 come se fosse un dado.
- 09.** Notice-Board.hex vi farà praticamente da post it ricordandovi qualcosa (inseribile modificando il programma) quando passerete davanti a lui.
- 10.** Compass.hex trasformerà Tobbie II in un compasso.
- 11.** Bowling.hex: il robot diventerà una palla da bowling che camminerà verso il bersaglio in base a quanto avrete fatto giusto il movimento di lancio (non lanciatelo!).
- 12.** APP Remote Control.hex programma da installare per poter usare l'applicazione da telefono.

PROGRAMMA FOLLOW ME

Questo programma, già predisposto dai creatori di



Tobbie II, sfrutta i due sensori a infrarossi del robot per permettergli di seguire una direzione imposta da un oggetto, ad esempio la nostra mano, posto davanti a lui.

I sensori ad infrarossi, simili a quelli che si utilizzano con Arduino, sono dei sensori che sfruttano dei raggi ad infrarossi ($\lambda=700\text{nm}$) per rilevare la distanza di un oggetto.

Per esaminare il codice possiamo semplicemente aprire il file .hex nella sezione di micro:bit dedicata alla programmazione.

Una volta aperto, noteremo che per tutta la durata del programma vengono rilevati costantemente i valori dei sensori di destra e di sinistra, e in base ai valori rilevati Tobbie II eseguirà una determinata funzione.

Facciamo però attenzione al fatto che i sensori funzionano in modo contrario a quello a cui siete probabilmente abituati, infatti più il valore numerico è alto più significa che l'oggetto è vicino al sensore, più è basso invece, più l'oggetto è lontano, ma i valori oscillano comunque tra 0 e 1.023, dove un 900 può essere considerato come il suo massimo.

Se i due sensori sono entrambi ad un valore maggiore di 310, Tobbie II camminerà all'indietro perché significherà che la vostra mano sarà vicino alla faccia del robot, se uno dei due sensori sarà ad un valore maggiore di 400 Tobbie II si muoverà verso quel sensore, per attuare i movimenti di destra e di sinistra mentre, nell'ultimo caso, se entrambi i sensori hanno un valore maggiore di 50, Tobbie II camminerà in avanti, ma attenzione a non allontanarvi troppo perché Tobbie II potrebbe smettere di seguirvi se supererete una distanza maggiore di quella massima rilevabile.

SASSO, CARTA, FORBICE

Questo programma dimostrativo consente di giocare insieme a Tobbie all'omonimo gioco che molti di voi conosceranno.

Nella memoria della micro:bit sono memorizzate le tre icone del gioco e, una volta che vi sarete avvicinati ad uno dei due sensori, Tobbie II vi mostrerà le sue icone per poi iniziare a girare su se stesso, durante quest'ultima azione il codice prenderà a caso un numero tra 1 a 3 e finito il giro visualizzerà l'immagine corrispondente al numero scelto, che corrisponde poi ad una delle icone: sasso, carta o forbice. È un programma molto facile e divertente con cui passare del tempo insieme al vostro nuovo amico cibernetico.

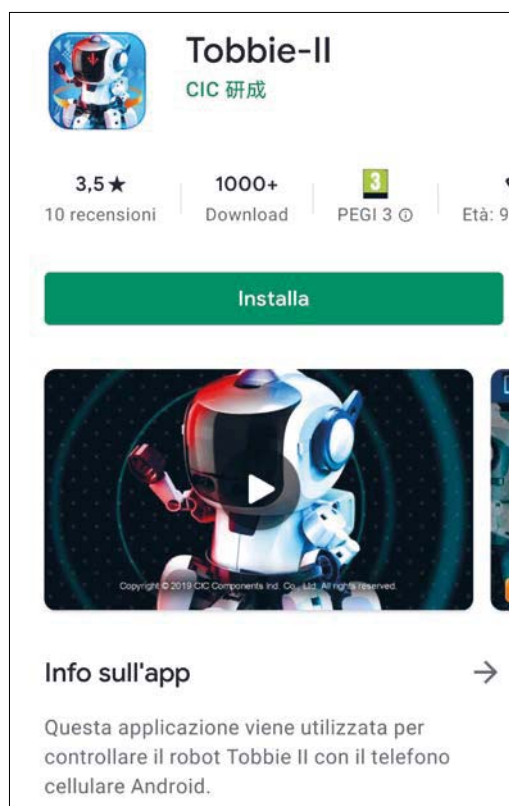


Fig. 5
Applicazione
di Tobbie.

EXPLORER

Questo programma, invece, è l'ideale per consentire a Tobbie II di muoversi orientandosi nell'ambiente; infatti una volta cliccato il pulsante A, Tobbie II inizierà a muoversi in avanti fino a quando non troverà un ostacolo, per poi fermarsi, cambiare direzione e ripartire, fino a quando non cliccherete il pulsante B.

APPLICAZIONE PER IL CONTROLLO REMOTO

Per poter utilizzare questa applicazione abbiamo innanzitutto bisogno di avere programmato micro:bit con il programma *APP Remote Control.hex*. Dobbiamo inoltre aver installato l'app sul nostro smartphone (**Fig. 5**). Fatto questo attiviamo il Bluetooth e colleghiamo il nostro Tobbie tramite il pulsante di scansione.

Una volta connesso avremo a disposizione un joystick per il movimento, delle faccine per cambiare la disposizione dei LED e una cella di inserimento del testo per mandare qualche messaggio scorrevole: la **Fig. 6** propone l'interfaccia dell'applicazione con a sinistra il pad per il movimento, in centro le icone da mostrare sulla faccia di Tobbie II e in fondo una casella di testo per fare scrivere dei messaggi scorrevoli sulla faccia del robot.

Nel caso non si volesse controllare Tobbie II col

Fig. 6
Interfaccia utente
dell'app.

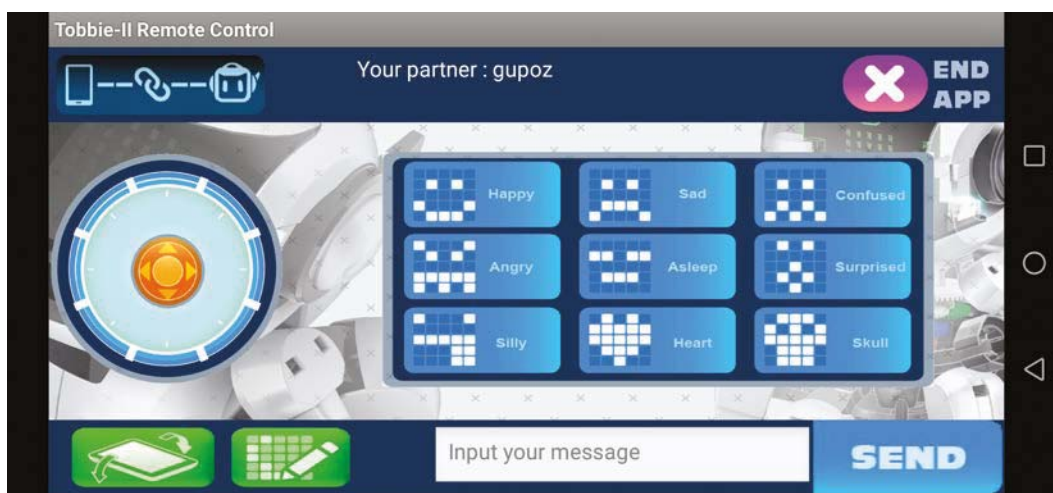
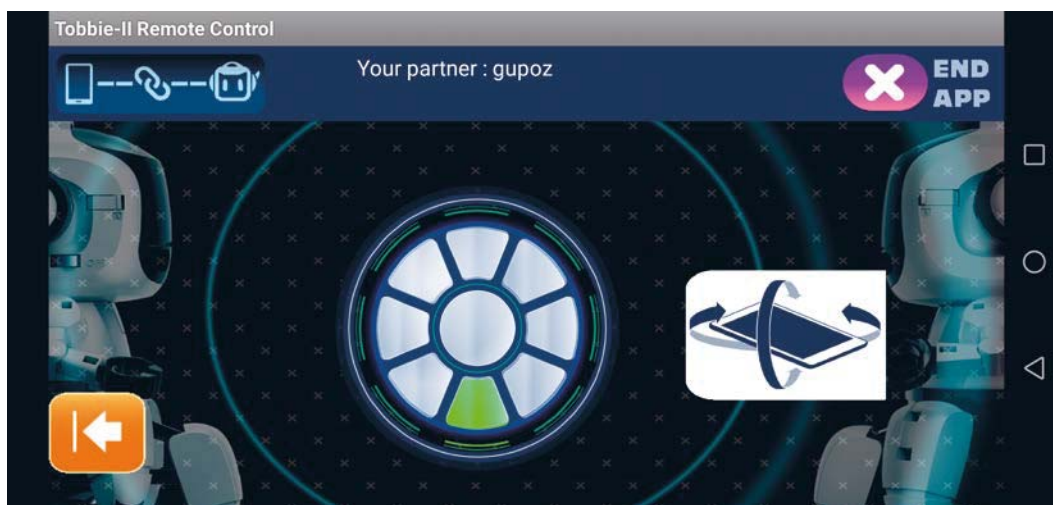


Fig. 7
Modalità joystick
dell'app.



joystick è possibile utilizzare anche uno smartphone o tablet come sensore di orientamento; a riguardo, in **Fig. 7** è proposto il joystick che fa muovere Tobbie II in base all'orientamento dello smartphone, se piegato in avanti, indietro o verso destra e sinistra. Questa funzione sfrutta l'accelerometro interno al telefono, normalmente sfruttato per la funzione di orientamento automatico dello schermo.

CONCLUSIONI

Tobbie II è il robot giusto se state cercando qualcosa di avanzato ma non troppo difficile da utilizzare e programmare, con il quale potete anche imparare un secondo linguaggio di programmazione nel caso non lo conoscestes. Non vi rimane che provare tutti i dodici programmi forniti e dare sfogo alla vostra fantasia per creare quello che più vi piace con il linguaggio che vorrete. Buon lavoro, ma soprattutto ... buon divertimento.

Cosa occorre?

Il robot programmabile con micro:bit Tobbie II (cod. KSR20) è disponibile presso Futura Elettronica a Euro 89,00. Esiste una versione semplificata non programmabile chiamata Tobbie (cod. KSR18) in vendita a Euro 37,00. Per chi volesse sperimentare con micro:bit Futura Elettronica mette a disposizione un kit di sperimentazione (cod. MICROBITKIT) a Euro 39,90. E' disponibile anche un libro per programmare la scheda micro:bit in Python (cod. S5556) a Euro 14,90. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)
Tel: 0331-799775 - <http://www.futurashop.it>

UN TEAM DI PROFESSIONISTI PER IL TUO PROGETTO



oltre **1500** progetti sviluppati
26 anni di attività
più di **15** progettisti



L'area R&D di Futura Group studia, progetta e realizza prototipi per differenti settori utilizzando tecnologie innovative. Nel team di Futura Group fa parte il personale interno e un numero elevato di collaboratori esterni, ciascuno specializzato nel proprio settore (Disegno CAD – Raspberry Pi – Arduino – Audio – Sviluppo firmware – Linux – ecc). La disponibilità di oltre 5000 prodotti elettronici e componenti consente di ridurre i tempi di test e sviluppo e di realizzare facilmente il primo prototipo. La decennale esperienza nel settore e la necessità di ottimizzare i tempi e i costi, ha portato Futura Group a selezionare una serie di terzisti per la produzione di piccola o grande serie.

I SERVIZI CHE TI OFFRIAMO



Progetto Completo

Partendo dalla tua idea progettiamo per te la soluzione.



Sviluppo PCB

Ottimizziamo il tuo prototipo disegnando un PCB compatto secondo le tue specifiche.



Scrittura Codice

Scriviamo il codice del core per dare vita alla tua applicazione.



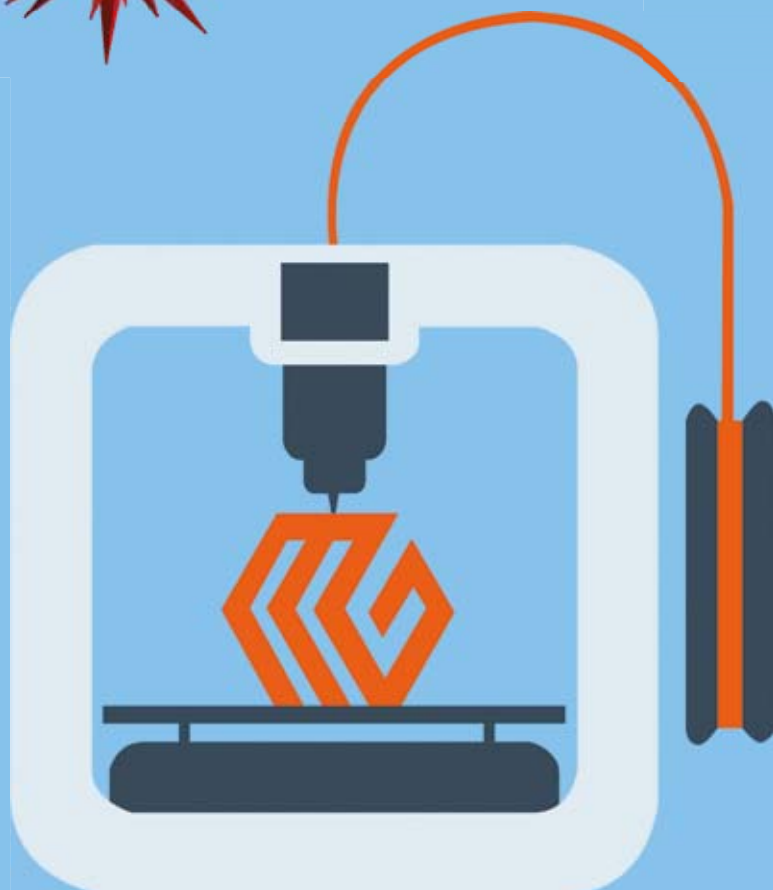
Produzione Progetto

Portiamo il prototipo alla fase di produzione seguendo per te tutte le fasi.

Trasformiamo le tue idee in realtà!

Servizio taglio laser e incisione

Futura Elettronica ti offre il servizio di taglio laser ed incisione per realizzare pannelli, parti meccaniche, mascherine, scritte, decori ed altro ancora in plexiglass (o metacrilato).



Servizio stampa 3D

Futura Elettronica mette inoltre a disposizione le proprie stampanti 3D FDM, per chi vuole realizzare un oggetto tridimensionale con dimensioni fino a 40x40x40 cm.

Informazioni dettagliate su
www.futurashop.it



Prepariamo la connessione della nostra scheda al Cloud Azure IoT Central e configuriamo l'interfaccia di programmazione. Seconda puntata.

CONOSCERE

E USARE

Azure Sphere

2

di PIER CALDERAN

Lo scopo dei dispositivi Azure Sphere è quello di connettersi in modo sicuro al Cloud Azure. Per soddisfare tale esigenza, Microsoft, ha semplificato la connessione dei dispositivi IoT creando un portale dedicato per poter visualizzare in modo più semplice i dati di telemetria provenienti dai dispositivi connessi e/o per controllarli da remoto. Il portale (servizio) dedicato a tali applicazioni si chiama **Azure IoT Central** ed è disponibile sul web all'indirizzo: <https://apps.azureiotcentral.com>. Al mondo di Azure Sphere abbiamo dedicato questo corso, iniziato descrivendo tale ambiente Cloud Microsoft e le sue potenzialità nell'ambito dell'IoT, introducendo altresì (nella prima puntata) la scheda Azure Sphere MT3620 della Futura Elettronica (cod. FEAZURESPH), la piattaforma hardware che utilizzeremo nelle varie esercitazioni,

la quale si basa sul SoC MT3620 della Mediatek montato a bordo di un apposito modulo di Avnet. Dopo aver visto come configurare questa piattaforma e scrivere ed eseguire il canonico programma Hello World, in questa puntata spieghiamo come preparare la connessione della nostra scheda al cloud Azure IoT Central, con cui comunicherà per inviare i dati ambientali rilevati. Faremo anche una parentesi su un argomento cui abbiamo accennato nella prima puntata: la programmazione dell'interfaccia USB necessaria alla programmazione, basata sul chip FTDI FT4232, la cui prerogativa è che permette di configurare il modo di funzionamento delle periferiche di cui dispone.

AZURE IOT CENTRAL

Iniziamo dunque con Azure IoT Central, che abbrevieremo da ora in poi come "IoT Central": si

tratta di un'offerta Software as a Service (SaaS) di Microsoft, nata per creare applicazioni online senza sforzo, dato che le configurazioni sono sempre guidate. Come vedremo più avanti in questo corso, c'è una procedura di certificazione dei dispositivi Azure Sphere per la connessione sicura. Intanto ci dedichiamo alla spiegazione di come accedere al portale Azure IoT Central, di come creare un'applicazione e i dispositivi virtuali che ospiteranno i nostri dispositivi reali.

ACCOUNT AZURE

Per prima cosa è necessario avere un account Azure.

Se non si possiede un account Azure è possibile ottenerne uno gratuito per una prova di 12 mesi, visitando l'indirizzo seguente: <https://azure.microsoft.com>

Dalla pagina di benvenuto basta seguire la procedura guidata per creare un account e avere diritto ai seguenti benefici:

- Più di 25 servizi sempre gratuiti e i servizi più popolari utilizzabili gratis per 12 mesi;
- 170 € di credito da usare nei primi 30 giorni;
- Nessun addebito automatico.

Esaurito il credito, verrà chiesto se si vuole continuare con il pagamento in base al consumo. In tal caso, si pagherà solo se si usa una quantità di servizi superiore a quella gratuita.

CREARE UN'APPLICAZIONE PERSONALIZZATA SU IOT CENTRAL

Una volta effettuato l'accesso con il proprio account Azure, si può accedere al portale Azure IoT Central all'indirizzo <https://apps.azureiotcentral.com>, che permette di accedere alla home page del portale stesso; prima di iniziare vi consigliamo di leggere tutte le informazioni a riguardo e capire come funziona e cosa fa il portale. Senza dilungarci nei dettagli, diremo solo che IoT Central è una piattaforma per centralizzare le applicazioni che consentono la connessione sicura ai dispositivi Azure Sphere remoti.

Ovviamente, le applicazioni che vengono create in IoT Central "consumano" il credito del vostro account Azure, ma per rendere le cose un po' più facili, soprattutto ai neofiti, suggeriamo di iscriversi gratuitamente a una versione di prova per 7 giorni prima di deciderne l'acquisto. Dopo i 7 giorni l'applicazione viene eliminata. Si può comunque crearne una nuova, sempre in prova per 7 giorni, e continuare così fino a quando non vi decidete di utilizzare l'account a pagamento. Facendo clic su "Compilazione" del menu laterale a sinistra, si entra in una pagina simile alla Fig. 1.

A questo punto, con un clic sul pulsante "App personalizzate..." si entra nella pagina dedicata alla creazione di un'applicazione personalizzata (Fig. 2).

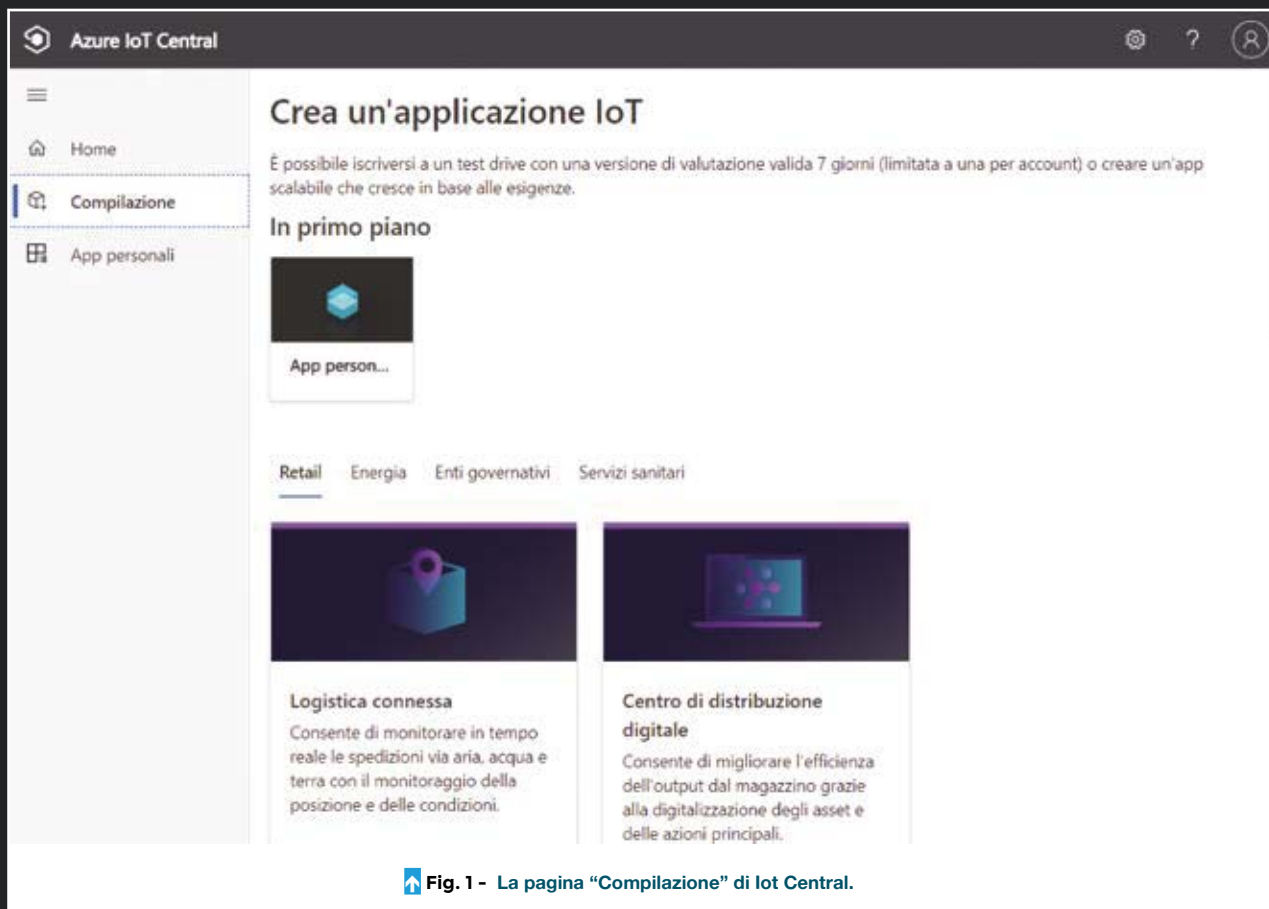


Fig. 1 - La pagina "Compilazione" di IoT Central.

Azure IoT Central

Compilazione > Nuova applicazione

Nuova applicazione Personalizzato

È sufficiente rispondere ad alcune domande e l'app sarà operativa a breve.

Informazioni sull'app

Nome dell'applicazione * ⓘ
Mia applicazione

URL * ⓘ
mia-applicazione-futura.azureiotcentral.com

Modello di applicazione * ⓘ
Applicazione personalizzata

Piano tariffario *
☒ **Gratuito**
 Prova per 7 giorni senza impegno
 5 dispositivi gratuiti
☐ **Standard 0**
 Per i dispositivi che inviano pochi messaggi al giorno
 2 dispositivi gratuiti 400 messaggi/mese

[Crea](#) [Annulla](#)

Sicurezza predefinita

Prezzi

Non sono previsti costi di risoluzione. Si paga solo per le risorse necessarie. [Visualizzare i dettagli sui prezzi](#)

Sicurezza

È possibile proteggere i prodotti connessi con la sicurezza IoT end-to-end predefinita e mantenere il controllo sui dati con funzionalità di privacy come l'accesso in base al ruolo e l'integrazione con le autorizzazioni di Active Directory.

Fig. 2 - La pagina "Nuova applicazione" di IoT Central.

Nel form "Nuova applicazione" dovete compilare i campi nel modo descritto qui di seguito.

Nome dell'applicazione: un nome deve essere univoco in tutto Azure poiché il nome dell'applicazione verrà usato anche per l'URL.

URL: mettere un nome univoco. Se si fornisce un nome già utilizzato, apparirà un messaggio di errore: nel nostro caso abbiamo scelto: "mia-applicazione-futura" per l'URL e "Mia applicazione" per il nome. Verrà creato un indirizzo di terzo livello di questo tipo:

mia-applicazione-futura.azureiotcentral.com

Piano tariffario: selezionare una delle quattro opzioni:

- **Gratuito**, per 7 giorni senza impegno. 5 dispositivi gratuiti.
- **Standard 0**, per i dispositivi che inviano pochi messaggi al giorno, 2 dispositivi gratuiti e 400 messaggi/mese;
- **Standard 1**, per i dispositivi che inviano pochi messaggi all'ora, 2 dispositivi gratuiti, 5.000 messaggi/mese;
- **Standard 2** (scelta più diffusa), per i dispositivi che inviano messaggi ogni pochi minuti, 2 dispositivi gratuiti e 30.000 messaggi/mese.

Anche se si può scegliere subito un piano tariffario a pagamento, è comunque consigliabile iscriversi con un abbonamento gratuito, per provare se tutto funziona.

Si potrà sempre migrare l'applicazione gratuita a un piano tariffario a pagamento in un secondo tempo.

Alla scadenza del periodo di prova di 7 giorni, è possibile convertire l'applicazione in un account a pagamento senza perdita dei dati oppure rinunciare, perdere tutto e ricominciare daccapo.

Al termine, premere il pulsante "Crea".

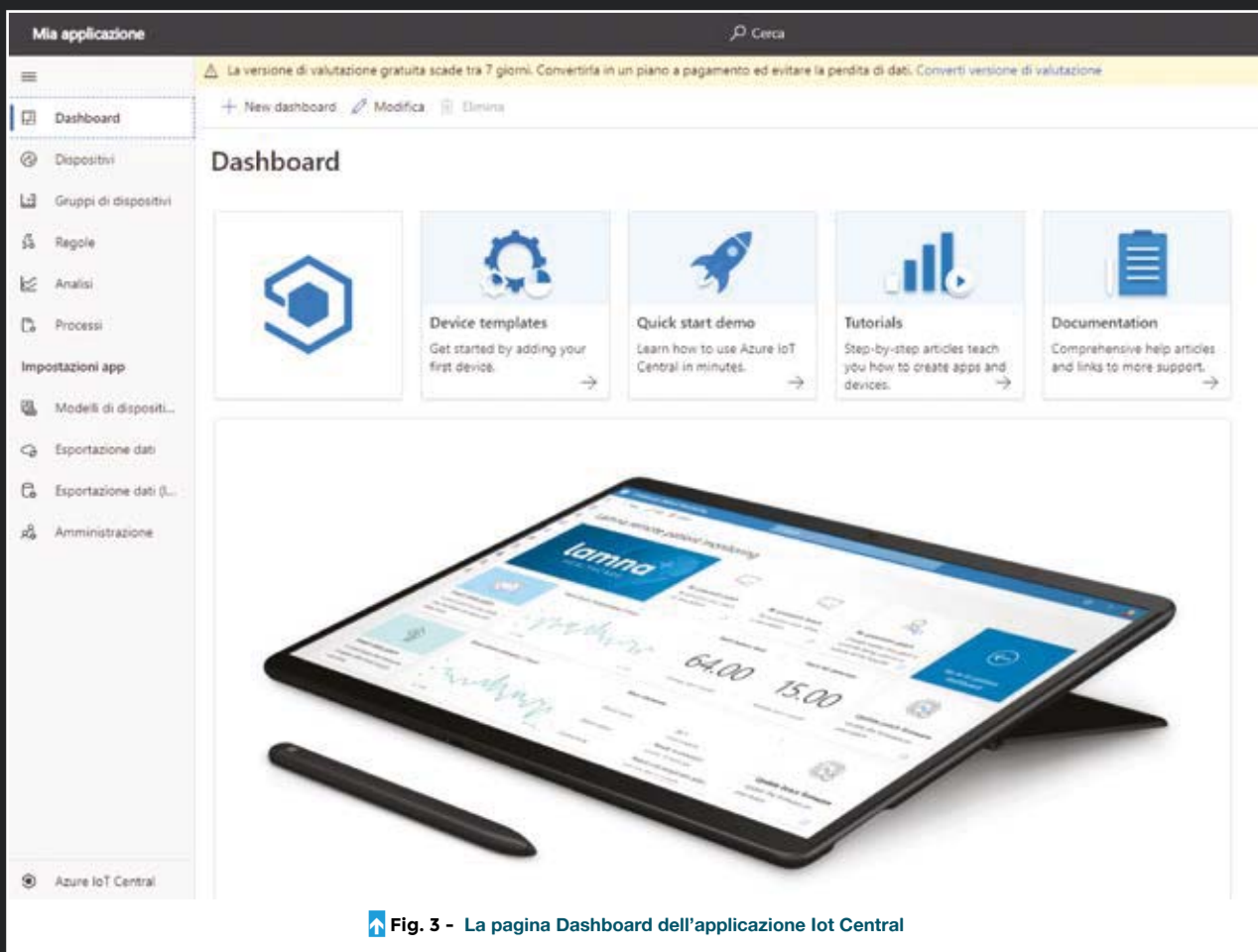
Verrà eseguito il provisioning dell'applicazione. Il termine tradotto sarebbe qualcosa come "approvvigionamento", ma lo lasceremo così com'è. Il provisioning altro non è che la registrazione dell'applicazione nel portale Azure.

Dopo qualche secondo apparirà una schermata simile alla Fig. 3, con il menu e la dashboard dell'applicazione.

Si fa notare che, se avete scelto il piano tariffario gratuito, apparirà sempre l'avviso seguente: "La versione di valutazione gratuita scade tra 7 giorni. Convertirla in un piano a pagamento ed evitare la perdita di dati". Nella pagina principale dell'applicazione ci sono varie voci nel menu a sinistra che spiegheremo più avanti nel corso. Per il momento ci concentreremo su "Modelli di dispositivo" e "Dispositivi".

MODELLO DI DISPOSITIVO IOT CENTRAL

Prima di creare un dispositivo all'interno della vostra applicazione, la prima cosa da fare è creare un "Modello di



 **Fig. 3 - La pagina Dashboard dell'applicazione IoT Central**

dispositivo". Qui si devono definire tutte le informazioni e le interazioni che saranno ereditate dal dispositivo reale. Ecco la procedura passo passo.

1. Fate clic su "Modelli di dispositivo" del menu a sinistra.
2. Nella pagina "Modelli di dispositivo" selezionate "+ Nuovo".
3. Apparirà una schermata per "Selezionare il tipo" (Fig. 4); qui dovete cliccare il riquadro "Dispositivo IoT".
4. Fate clic sul pulsante in basso "Successivo: Personalizza", il che farà apparire la schermata "Personalizza".
5. Assegnate un nome al modello, per esempio "Futura MT3620" (ignorare la casella "Dispositivo gateway", perché il dispositivo non è un gateway, allorché vi apparirà una schermata con i pulsanti "Avanti: rivedi" e "Indietro").
6. Se le informazioni sono corrette, fate clic su "Avanti: rivedi" e apparirà così una schermata "Visualizza" (Fig. 5).
7. Fate clic sul pulsante "Crea" per creare il modello: nella pagina "Crea un modello" cui accederete potrete selezionare le opzioni "Modello personalizzato" o "Importa un modello".
8. Selezionare l'opzione "Modello personalizzato" allorché apparirà una successiva schermata (Fig. 6) con le opzioni seguenti: **Aggiungi funzionalità al componente**

predefinito, Aggiungi un'interfaccia ereditata" e Aggiungi un componente.

9. Selezionare l'opzione "Aggiungi un'interfaccia ereditata".
10. Apparirà una schermata in cui scegliere un'interfaccia "Personalizzata" oppure "Interfaccia importazione". Selezionare un'interfaccia "Personalizzata" per proseguire, dopodiché a video apparirà una schermata "Interfaccia" in cui bisogna aggiungere una funzionalità. Aggiungere una funzionalità con il tasto "+ Aggiungi funzionalità".
11. Si aprirà un pannello Funzionalità (Fig. 7) con i campi da compilare come spiegato nel prossimo paragrafo.

AGGIUNGERE FUNZIONALITÀ

Le funzionalità del dispositivo sono, per esempio, i dati di telemetria, i comandi o le proprietà.

Per la visualizzazione dei dati telemetria del nostro primo esempio applicativo aggiungeremo gli elementi di telemetria "Temperature" e "Humidity", dato che useremo il sensore di temperatura e umidità DHT22, incorporato nella demoboard.

Più avanti parleremo del codice che manderà i dati di telemetria tramite la funzione SendTelemetry all'interno dell'applicazione.

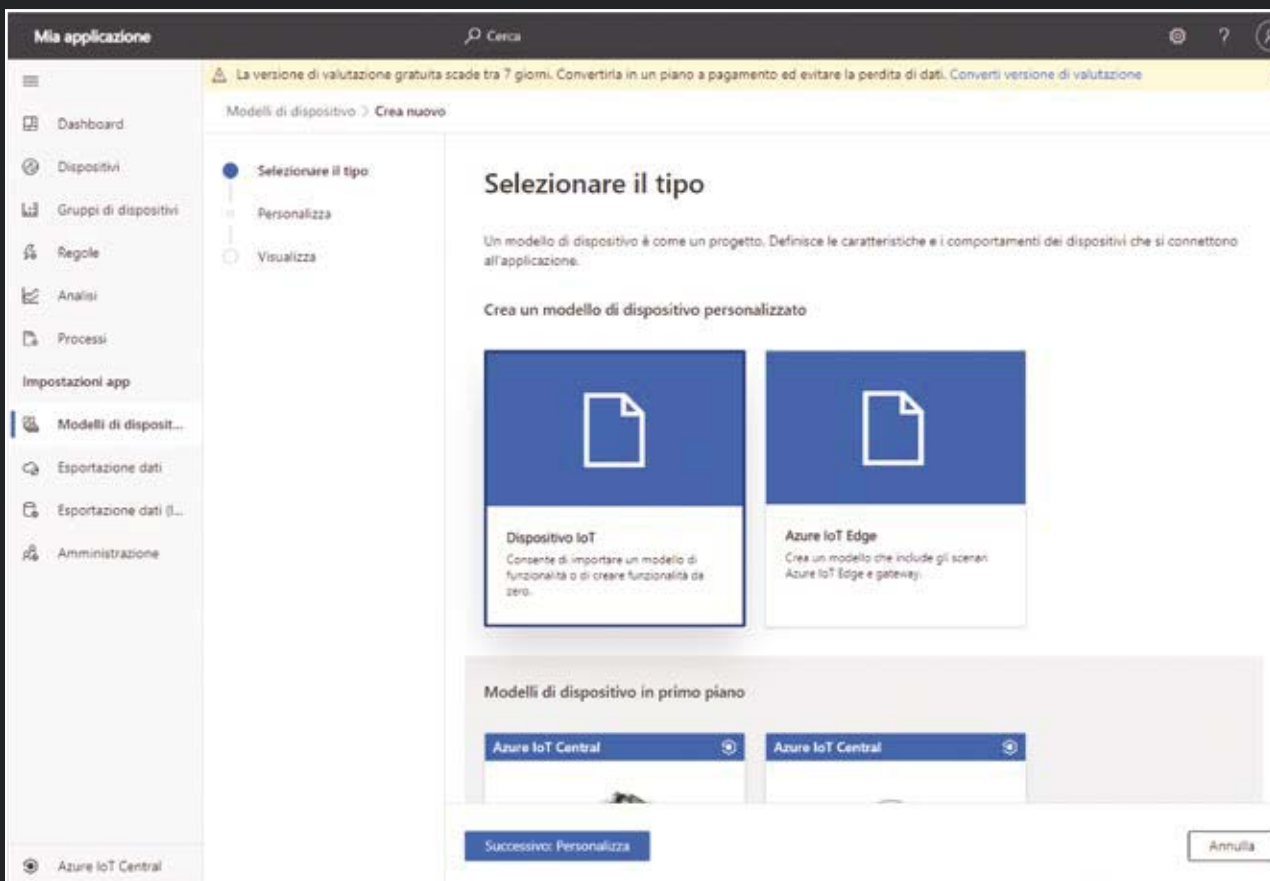


Fig. 4 - La pagina "Selezionare il tipo".

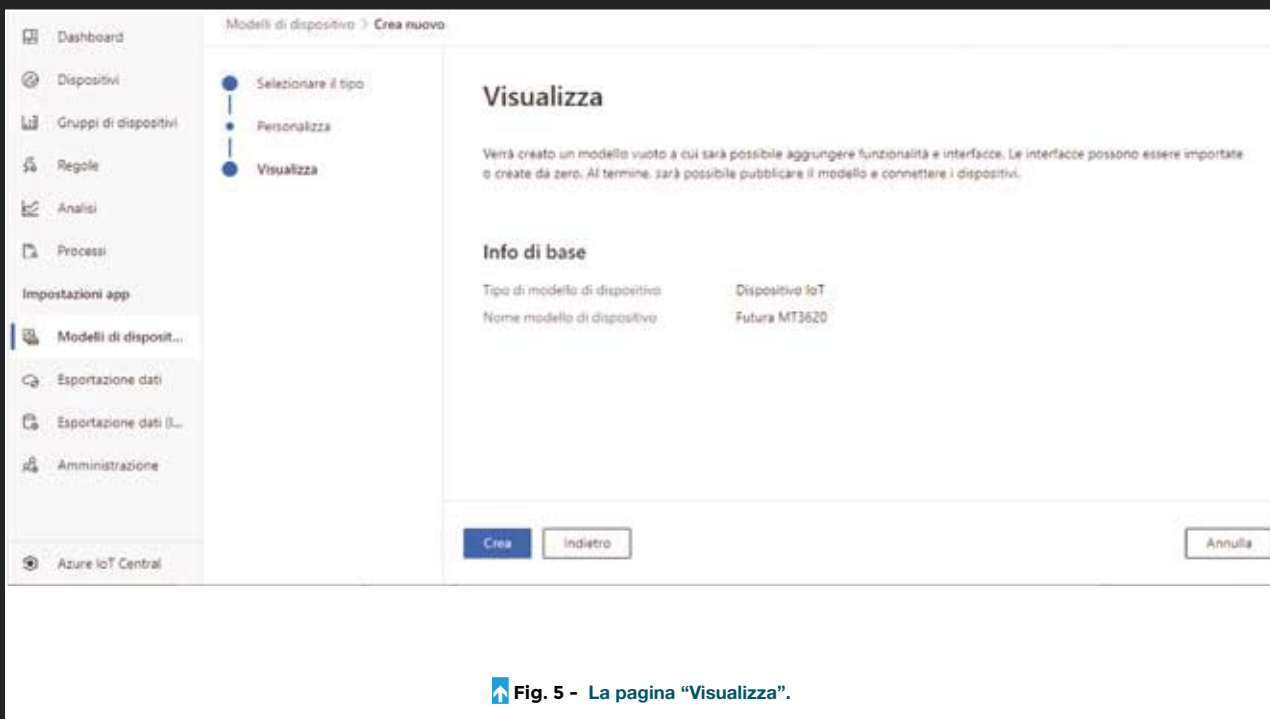


Fig. 5 - La pagina "Visualizza".

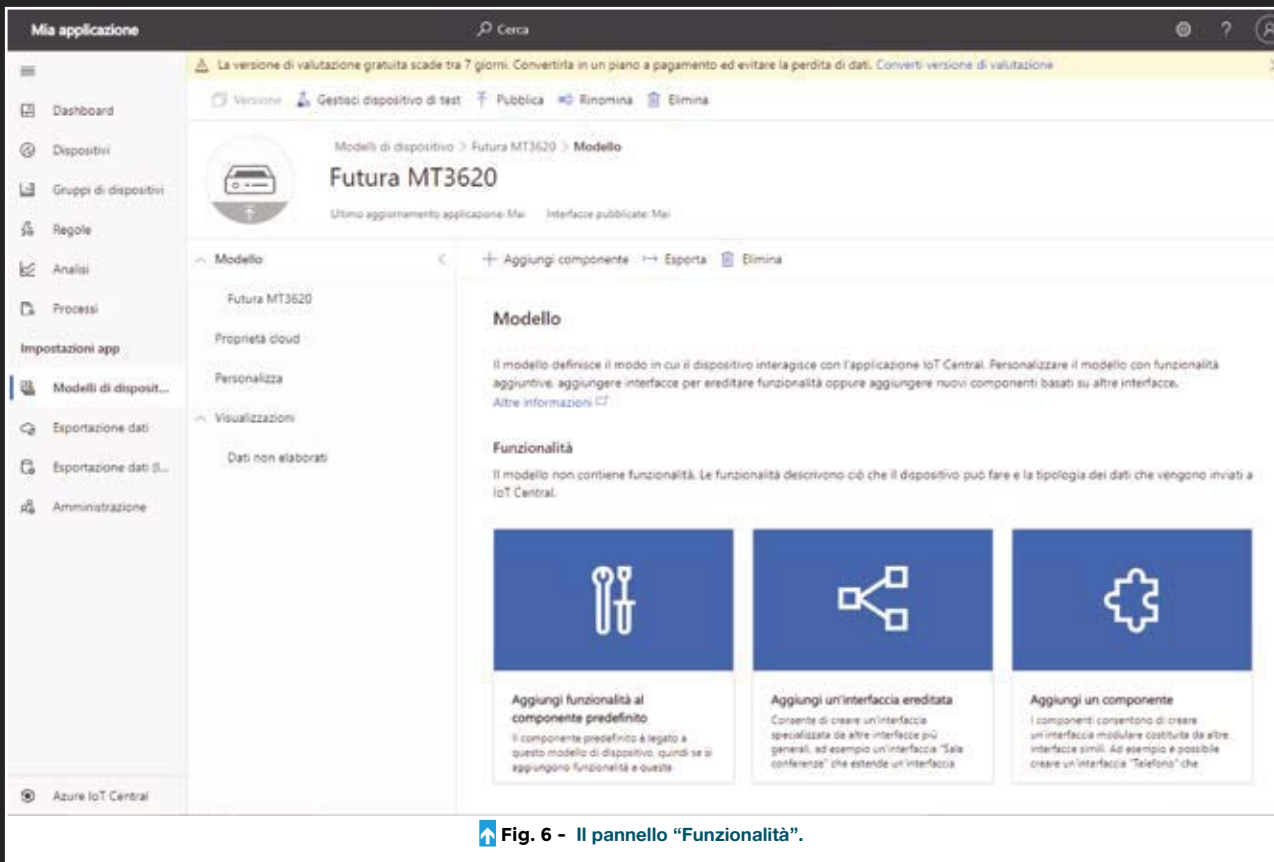


Fig. 6 - Il pannello "Funzionalità".

Compilare i campi in questo modo:

- **Nome visualizzato:** per esempio, "Temperature" (o qualsiasi altro testo);
- **Nome:** "Temperature" (deve essere lo stesso utilizzato nella funzione SendTelemetry);
- **Tipo di funzionalità:** Telemetry;
- **Tipo semantico:** Temperature;
- **Schema:** Double;
- **Unità:** °C;
- **Unità di visualizzazione:** °C;

- **Commento:** un testo qualsiasi;
- **Descrizione:** un testo qualsiasi.

Ripetere l'operazione per la telemetria dell'umidità facendo clic su "+ Aggiungi funzionalità". Compilare i campi in questo modo:

- **Nome visualizzato:** per esempio, "Humidity" (o qualsiasi altro nome che desiderate);
- **Nome:** "Humidity" (deve essere lo stesso usato nella funzione SendTelemetry);



Fig. 7 - La pagina "Funzionalità".

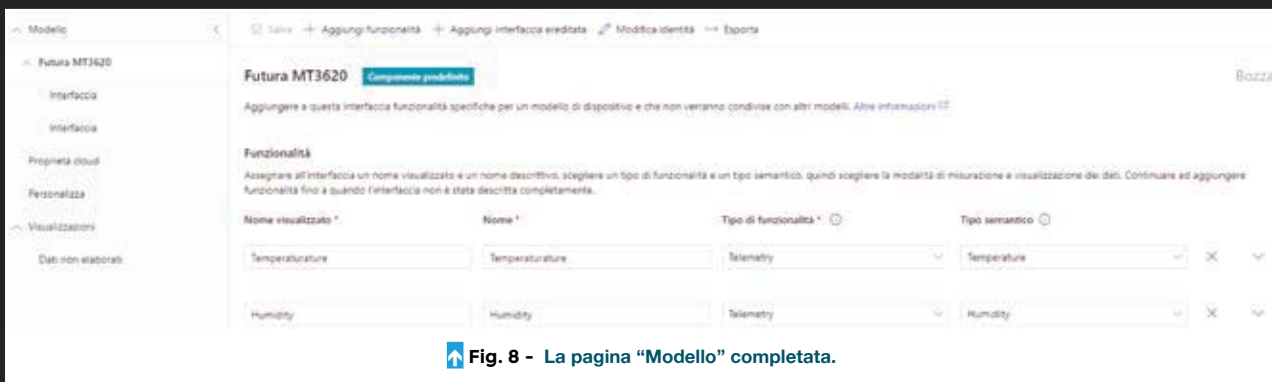


Fig. 8 - La pagina "Modello" completata.

- **Tipo di funzionalità:** Telemetry;
- **Tipo semantico:** Humidity;
- **Schema:** Double;
- **Unità:** %;
- **Unità di visualizzazione:** %;
- **Commento:** un testo qualsiasi;
- **Descrizione:** un testo qualsiasi.

Una volta terminata l'aggiunta delle funzionalità, si deve salvare il modello del dispositivo tramite l'icona "Salva" che farà apparire a video di simile a quanto mostrato nella **Fig. 8**.

AGGIUNGERE VISUALIZZAZIONI

Una volta create le funzionalità è possibile aggiungere dei riquadri per visualizzare in tempo reale i dati di telemetria

provenienti dai sensori.

Per aggiungere la visualizzazione della telemetria della temperatura seguire la procedura di seguito descritta.

1. Selezionate "Visualizzazioni" dal menu a sinistra.
2. Nella finestra per la selezione del tipo di visualizzazione che apparirà, cliccate su Selezionare "Visualizzazione del dispositivo".
3. Si aprirà la pagina "Modifica visualizzazione" in cui compilare i campi in questo modo:
 - **Nome visualizzazione:** per esempio, "Vista DHT".
 - **Telemetria:** Selezionare "Temperature" dal menu a discesa.
4. Fate clic su "Aggiungi riquadro" e verrà aggiunto il riquadro "Temperature (°C)" come mostrato in **Fig. 9**.
5. Fate clic su "+ Telemetria".

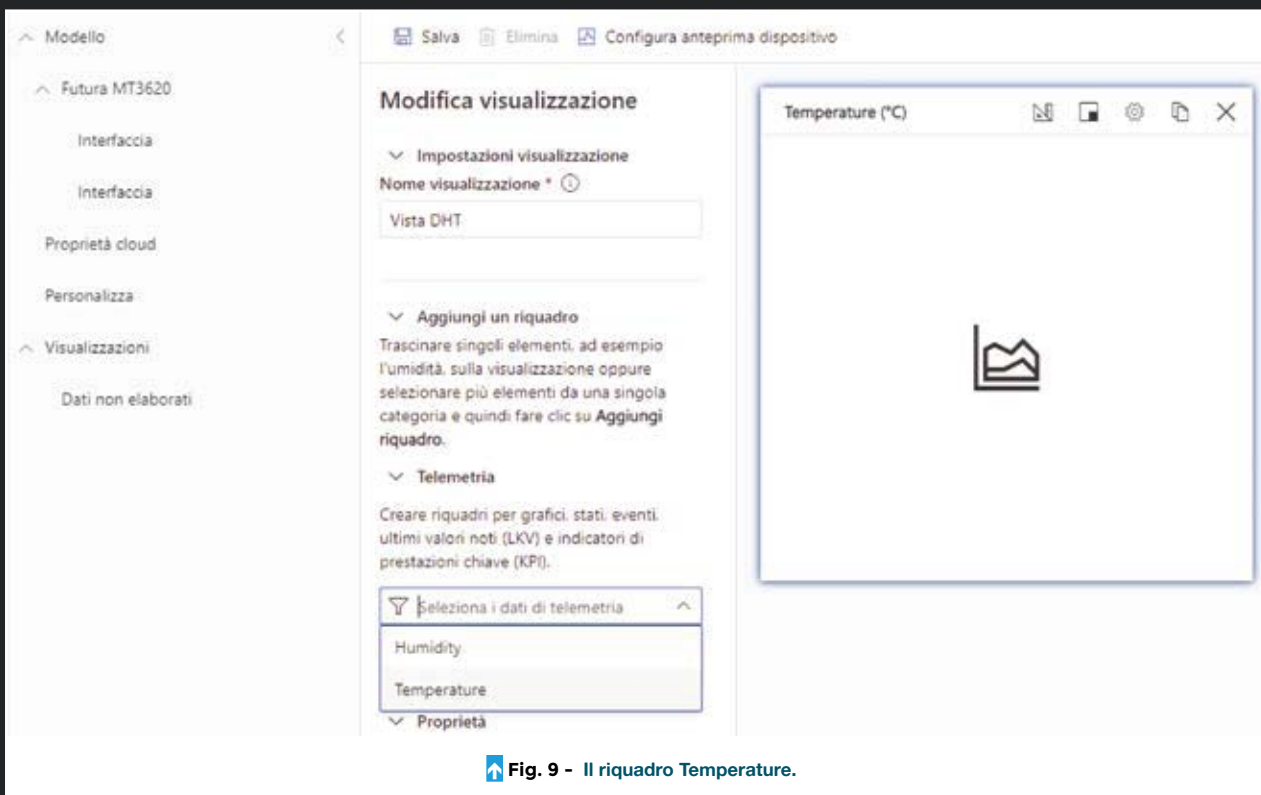


Fig. 9 - Il riquadro Temperature.

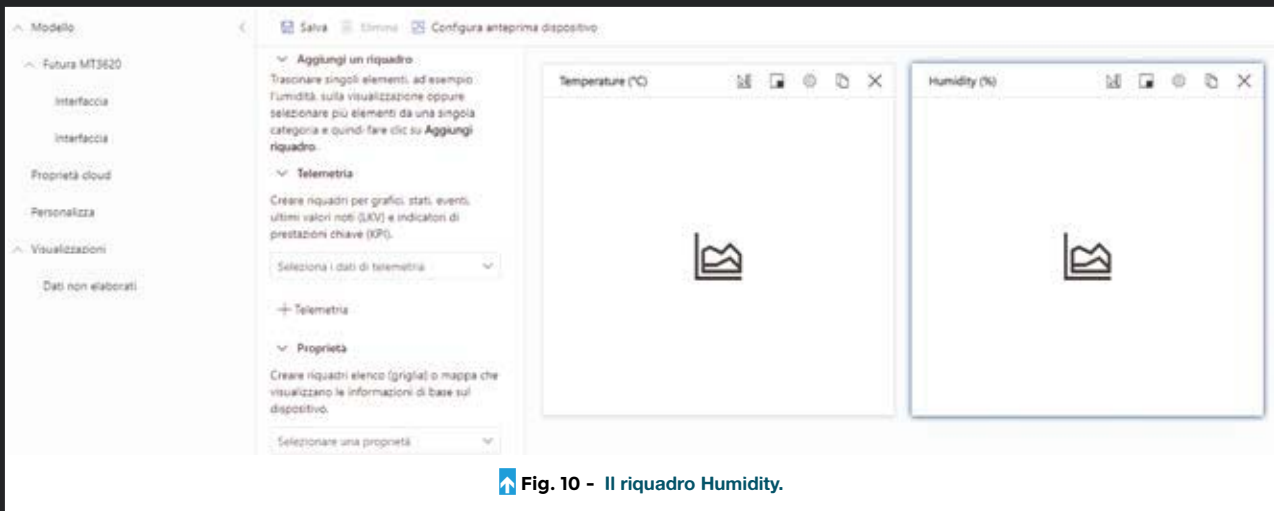


Fig. 10 - Il riquadro Humidity.

6. Selezionate "Humidity" dal menu "Telemetria".
7. Fate clic su "Aggiungi riquadro" e verrà aggiunto il riquadro "Humidity (%)" come in Fig. 10.

Questa operazione crea due distinti riquadri di "Temperature" e "Humidity". Se invece volete aggiungere un solo riquadro con entrambi i dati di "Temperature" e "Humidity", dovete procedere in questo modo:

1. Selezionate "Temperature" dal menu "Telemetria".
2. Fate clic su "+ Telemetria".
3. Selezionate "Humidity" dal menu "Telemetria".
4. Fate clic su "Aggiungi riquadro".
5. Verrà aggiunto il riquadro combinato con "Temperature (°C), Humidity (%)" di Fig. 11.

A questo punto dovete clic su "Salva" per salvare la visualizzazione del modello di dispositivo. Attenzione! Il modello deve essere pubblicato perché diventi disponibile per il/i dispositivo/i; allo scopo dovete fare clic sull'icona "Pubblica" che vedete evidenziata dalla freccia in alto, sopra il riepilogo del dispositivo (Fig. 12).

CREARE UN DISPOSITIVO

A questo punto il modello di dispositivo è pronto e quindi potete passare alla creazione del dispositivo vero e proprio, che si baserà su tale modello. Ecco come fare.

1. Selezionare la pagina "Dispositivi" dal menu a sinistra, allorché apparirà una pagina vuota con nessun dispositivo (Fig. 13).

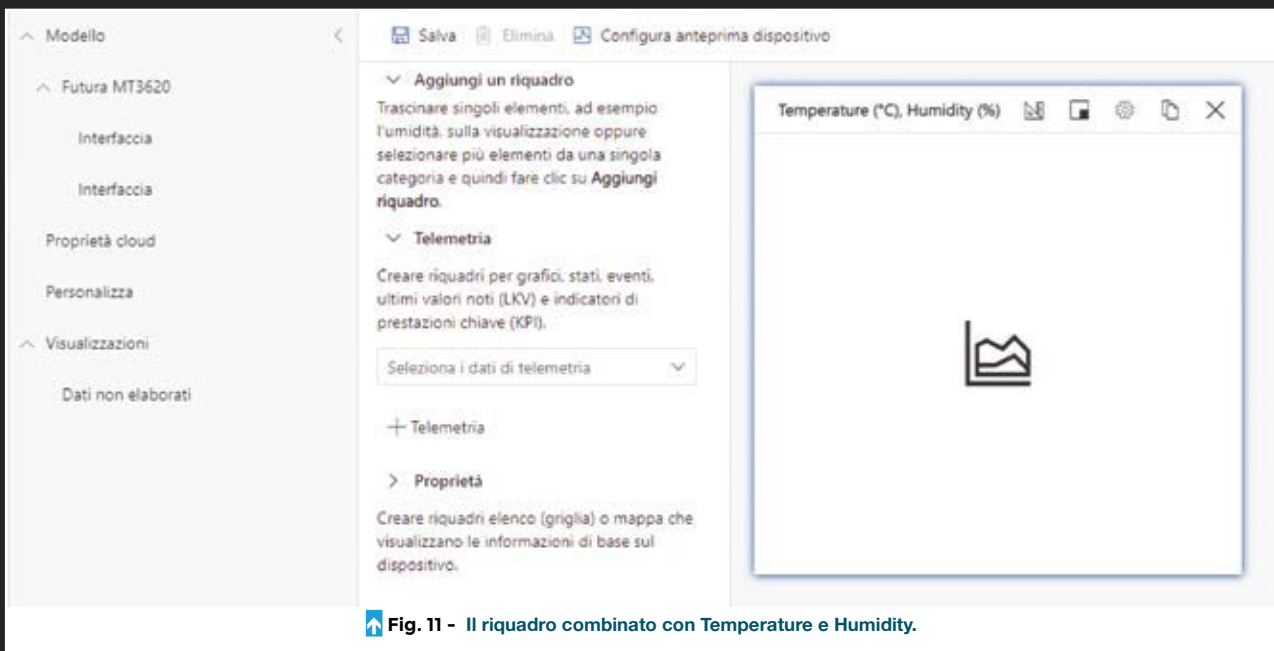


Fig. 11 - Il riquadro combinato con Temperature e Humidity.

2. Selezionare “+ Nuovo dispositivo” e nella finestra che apparirà andrete a compilare i campi in questo modo (**Fig. 14**):
 - **Modello di dispositivo:** selezionare il modello creato al passo precedente, per esempio “Futura MT3620”.
 - **Nome del dispositivo:** un nome qualsiasi, per esempio, “Futura MT3620 Device 1”.
 - **ID dispositivo:** questo ID viene assegnato automaticamente dal sistema.
 - **Simulare questo dispositivo?:** Lasciare selezionata l'opzione “No”.
3. Fare clic sul tasto “Crea”.
4. Verrà creato un dispositivo con il nome “Futura MT3620 Device 1”.

PROGRAMMARE L'INTERFACCIA USB

Come abbiamo visto nella prima puntata di questo corso, la scheda FEAZURESPH di Futura Elettronica va programmata con una breakout board esterna, che monta l'integrato FTDI FT4232H (cod. FT1479K). Questa soluzione serve a ottimizzare spazio e costi. Infatti, basta usare lo stesso programmatore per tutte le schede senza spese aggiuntive. Quando le schede Futura MT3620 vengono programmate, possono essere alimentate direttamente tramite la presa USB-C incorporata.

La breakout board consente di gestire da PC, mediante una sola porta USB, fino a 4 porte seriali virtuali COM. È in grado di supportare, a livello fisico, comunicazioni RS232, RS422

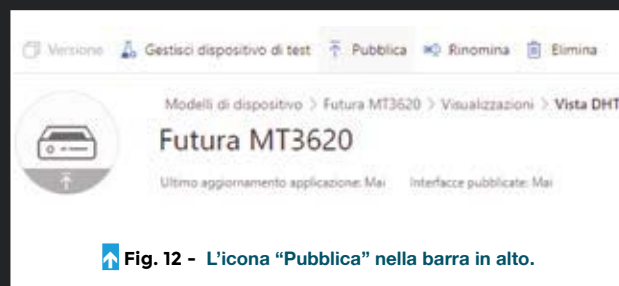


Fig. 12 - L'icona “Pubblica” nella barra in alto.

ACCESSO AD IOT CENTRAL

Una volta creato il dispositivo, bisognerà creare una connessione consentita per l'accesso a IoT Central (spiegheremo la procedura nella terza puntata); così la vostra scheda diventerà accessibile solo da voi e da nessun altro.

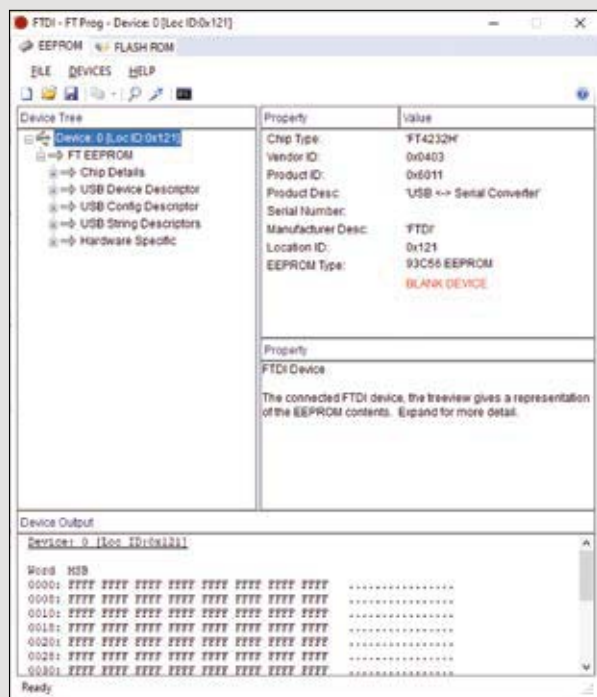
e RS485. Viene fornita con i componenti SMD e la porta USB-C già saldati, ad eccezione dei due strip a 13 vie che vanno saldati a mano. Per l'utilizzo come programmatore della nostra scheda Futura MT3620 occorre configurare la breakout board ad hoc. Per prima cosa, bisogna scaricare il programma FT_PROG dal sito FTDI al seguente indirizzo: <https://ftdichip.com/utilities>.

Questo programma consente di configurare la EEPROM della scheda in base alle proprie necessità. Per facilitare le cose abbiamo pensato che fosse utile fornire un template da caricare in FT_PROG in modo da configurare la scheda secondo le specifiche.

Attenzione! Per poter scrivere nella EEPROM della scheda è necessario ponticellare con una goccia di stagno le due piazzole contrassegnate con “J3V3IO”.

Una volta installato FT_PROG, ecco la procedura spiegata in pochi passi.

1. Collegare la presa USB-C a una porta USB del PC.
2. Dalla finestra principale (**Fig. Box**), fare clic sull'icona lente d'ingrandimento per leggere la EEPROM.
3. Apparirà il device 0 con la dicitura BLANK DEVICE.
4. Con il tasto destro del mouse fare clic sul device 0 per aprire il menu contestuale.
5. Selezionare la voce “Apply template from file”.
6. Caricare il nostro template “ftdi_template.xml”.
7. Apparirà un messaggio “the template was successfully applied”.
8. Con il tasto destro del mouse aprire il menu e selezionare la voce “Program Device”.
9. Dopo alcuni secondi apparirà in basso la scritta “Ready”.



Per verificare la configurazione, andate in “Gestione dispositivi” del computer e assicuratevi che ci siano tre porte COM installate. Facciamo notare che è stata disabilitata la porta B del chip FT4232H e che la descrizione del dispositivo è “MSFT MT3620 Std Interface”.

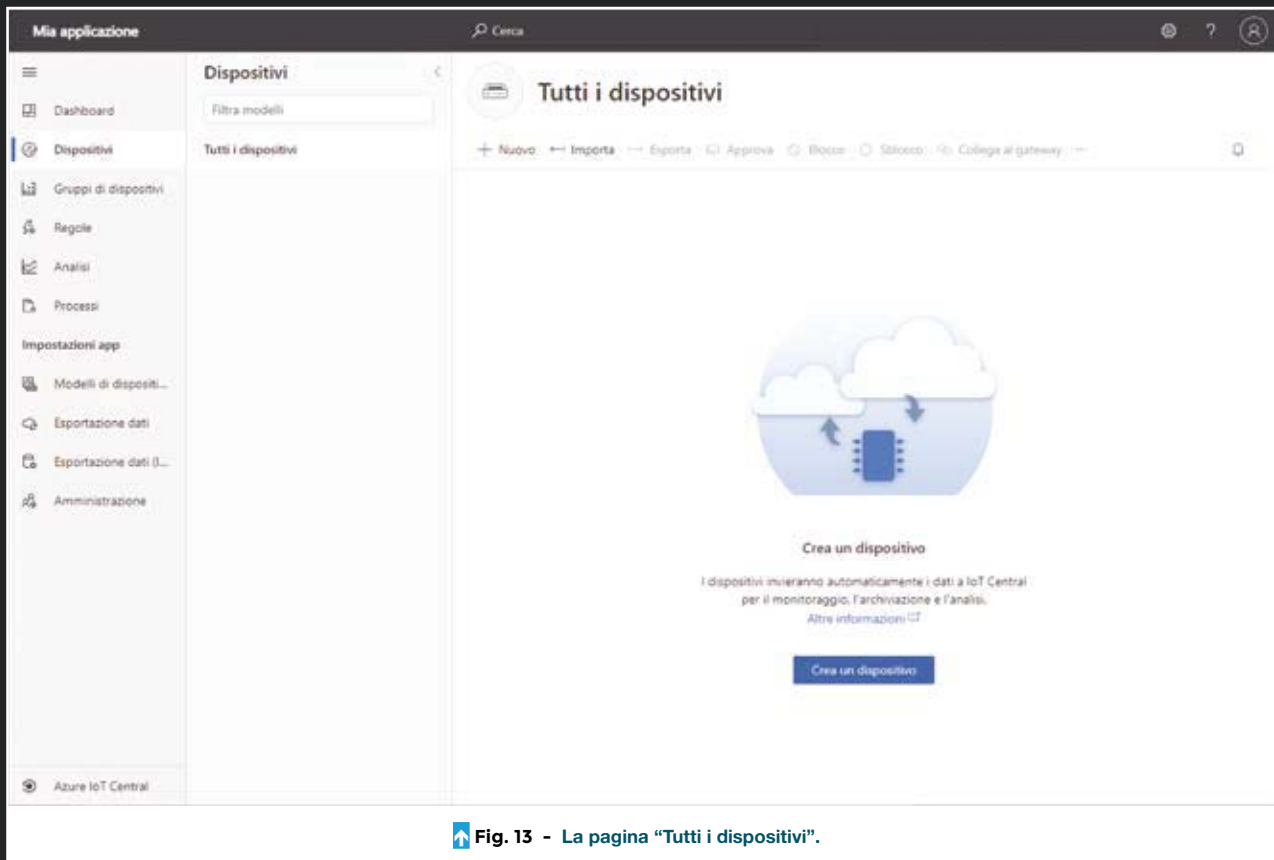


Fig. 13 - La pagina "Tutti i dispositivi".

CONCLUSIONI

Bene, per questa puntata abbiamo terminato. Vi abbiamo mostrato come poter registrare la scheda FEAZURESPH su IoT Central di Microsoft e come configurare l'ecosistema passo passo. Con la speranza di essere stati sufficien-

temente chiari, nell'esporsi la configurazione, vi diamo appuntamento alla prossima, nella quale approfondiremo la materia con nuovi esempi applicativi per la nostra demoboard IoT e altri aspetti di Azure Sphere e spiegheremo la registrazione del dispositivo IoT. □

Fig. 14 - La finestra "Crea un dispositivo".



Cosa occorre?

La demoboard di Futura Elettronica con MT3620 (cod. FEAZURESPH) è disponibile a Euro 128,00 e include i componenti base per poter sperimentare con la tecnologia Azure Sphere. Dal sito futurashop.it è possibile acquistare anche le breakout board correlate alla demoboard. La board USB/TTL in grado di creare quattro porte seriali virtuali (cod. FT1497K) è in vendita a Euro 35,00, mentre la board adattatrice necessaria per programmare la demoboard (cod. FT1498K) è disponibile a Euro 25,00. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:

Futura Elettronica, Via Adige 11, 21013 Gallarate (VA)
Tel: 0331-799775 - <http://www.futurashop.it>

SCEGLI LA TUA STAMPANTE 3D

Affidati a chi le progetta e le sviluppa

STAMPANTE 3D 40x40x40 cm in KIT

stampante 3D FDM (Fused deposition modeling)
capace di produrre stampe di **40x40x40 cm** (64.000 cm³)

€ **999,00**

Cod. **3D4040**

- ✓ Struttura interamente in alluminio
- ✓ Area di stampa: X 40 cm, Y 40 cm, Z 40 cm
- ✓ Diametro filamento: 1,75 mm
- ✓ Tipo di filamento: ABS, PLA, NYLON ed altri ancora
- ✓ Diametro ugello fornito: 0,4 mm
- ✓ Diametro ugelli opzionali: da 0,3 mm a 0,8 mm
- ✓ Velocità di stampa massima: 300 mm/s (in funzione dell'oggetto da stampare)
- ✓ Piatto di stampa fisso: vetro temperato da 6 mm
- ✓ Riscaldatore per piatto di stampa: 40 x 40 cm – 12V/240 W con adesivo 3M (opzionale)
- ✓ Controllabile da PC o da modulo LCD (opzionale)
- ✓ Alimentazione tramite modulo switching 220 VAC/12 VDC 350 W
- ✓ Istruzioni di montaggio in italiano con illustrazioni



Versione montata
cod. 3D4040/M
€ 1.299,00 IVA inclusa.



STAMPANTE 3DRAG IN KIT

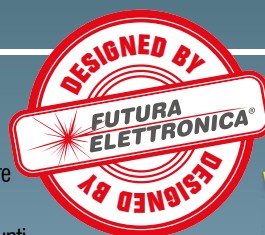


€ **499,00**

Cod. **3DRAG/K**

- ✓ Struttura in alluminio per combinare rigidità e leggerezza
- ✓ Assemblaggio semplificato con giunti metallici e parti in ABS stampate a iniezione
- ✓ Piano di stampa con movimentazione X/Y
- ✓ Estrusore movimentato sul solo asse Z
- ✓ Volume di stampa massimo: 20 x 20 x 20 cm
- ✓ Risoluzione: X e Y - 0,015 mm; Z - 0,39 micron
- ✓ Estrusore con ugello da 0,5 mm per filo da 3 mm in PLA e ABS
- ✓ Velocità di stampa tipica: 120 mm/s
- ✓ Piatto riscaldato

Versione montata
cod. 3DRAG/M
€ 699,00 IVA inclusa.

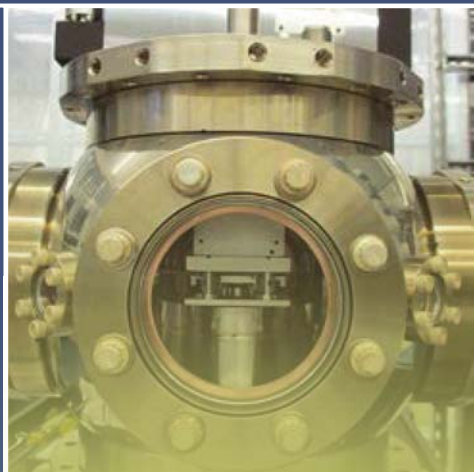


VASTA GAMMA DI ACCESSORI!
SCOPRILI TUTTI SU
www.futurashop.it

**E CON LA STAMPANTE 3DRAG
PUOI ANCHE...
STAMPARE
CON IL CIOCCOLATO**

**Set estrusore Choco
3DRAG - in kit**
Cod. 3DCHOCO - € 178,00





JAQAL: l'assembly per i Quantum Computer

L'informatica quantistica non sarà elaborazione a tutti gli effetti fino a quando non ci sarà un software quantistico a corredo dell'hardware. Un progetto di computer quantistico open source sviluppato ai Sandia National Laboratories di Albuquerque (New Mexico) mira a superare il problema con un linguaggio assembly adattato al calcolo quantistico e testato su QSCOUT (Quantum Scientific Computing Open User Testbed); questo è costituito da una trappola che tramite un campo elettromagnetico sospende una catena di 171 atomi di itterbio ionizzati che levitano in una camera a vuoto. Lampi di luce laser UV fanno ruotare questi atomi, eseguendo algoritmi scritti nel codice assembly quantistico chiamato Just Another Quantum Assembly Language o JAQAL. Sebbene Google, IBM e altri abbiano costruito macchine quantistiche più grandi e prodotto i propri linguaggi di programmazione, QSCOUT offre vantaggi a coloro che desiderano esplorare questa frontiera dell'informatica. I gate superconduttori come quelli delle macchine Google e IBM, sono veloci ma anche instabili: perdono coerenza (e dati) in meno di un secondo. Grazie alla tecnologia di intrappolamento ionico, QSCOUT può mantenere la coerenza del calcolo per 10 secondi. Ma il vero vantaggio di QSCOUT è la capacità che offre all'utente di controllarne il funzionamento aggiungendo operazioni nuove o modificate. All'architettura del set di istruzioni di base della macchina, JAQAL aggiunge comandi per inizializzare gli ioni come qubit, ruotarli individualmente o insieme in vari stati, intrecciarli in sovrapposizioni e leggere i loro stati finali come dati di output.

www.sandia.gov

Fuel cell ad ammoniaca

Un gruppo di ricercatori del Fraunhofer Institute sta partecipando ad un consorzio internazionale per sviluppare celle a combustibile funzionanti utilizzando l'ammoniaca come combustibile, destinate alla propulsione di future navi con motori elettrici.

L'idrogeno (H_2) è da tempo considerato il combustibile ideale sia per i motori endotermici a gas, sia per la produzione di elettricità tramite fuel-cell da impiegare per la propulsione elettrica; questo anche in ambito nautico.

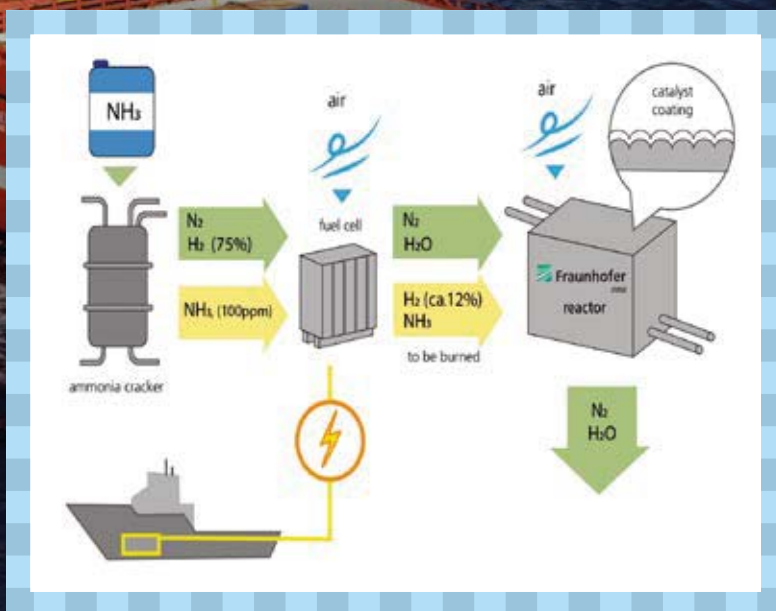
Sfortunatamente l'idrogeno occorre alle fuel-cell dev'essere, a parte alcune tipologie, puro e produrre idrogeno puro costa molto. Questo lo sanno bene al Fraunhofer Institute for Microengineering and Microsystems (IMM) di Mainz (Germania) che sta lavorando per alimentare le fuel-cell con dell'ammoniaca in stato gassoso (NH_3). Nell'ambito del progetto ShipFC, l'istituto tedesco sta collaborando con 13 partner per sviluppare la prima fuel-cell alimentata ad ammoniaca destinata al trasporto marittimo, un settore che, insieme a quello aereo, è tra i principali produttori di gas ad effetto serra

(attualmente è responsabile di circa il 2,6% delle emissioni globali di CO_2).

La tecnologia in sviluppo mira a creare una propulsione priva di emissioni e al tempo stesso sicura, affidabile e facile da utilizzare anche su navi di grandi dimensioni e su lunghe tratte. Il progetto è coordinato da NCE Maritime Cleantech dalla Norvegia e i ricercatori di Fraunhofer sono responsabili dello sviluppo del convertitore catalitico del sistema.

L'ammoniaca è di facile ed economica produzione e può anche funzionare come vettore energetico di alta qualità, concorrenziale rispetto all'idrogeno. Infatti mentre per avere una discreta autonomia l'idrogeno deve essere immagazzinato a $-253^\circ C$ allo stato liquido o a pressioni di circa 700 bar allo stato gassoso, l'ammoniaca allo stato liquido può essere conservata, invece, a una temperatura ragionevole di $-33^\circ C$ a pressione atmosferica e $+20^\circ C$ a 9 bar. Ciò rende lo stoccaggio e il trasporto di questo vettore energetico notevolmente più semplice e sicuro.

Il processo per produrre elettricità dall'ammoniaca non è ovviamente di-



per il trasporto marino

retto: l' NH_3 viene immessa in un cracker (reattore) e suddivisa in azoto (N_2) e idrogeno in un processo di reforming. Una piccola quantità di ammoniaca non viene convertita e rimane nel flusso di gas. Azoto e idrogeno vengono immessi nella cella a combustibile, introducendo aria e consentendo così all' H_2 di sviluppare elettroni che viaggiano nell'elettrolita generando corrente elettrica e formando acqua come residuo della reazione elettrochimica. L'idrogeno non è completamente convertito nella cella a combustibile: circa il 12% del vettore e dell'ammoniaca residua lasciano la cella a combustibile ed entrano nel convertitore catalitico sviluppato da Fraunhofer IMM. Qui è introdotta altra aria e il residuo viene a contatto con

un foglio di metallo ondulato rivestito da un catalizzatore a base di particelle di platino, il quale una reazione chimica che produce acqua e azoto. Il team dell'IMM sta anche sviluppando un reattore che funziona passivamente. Un piccolo prototipo iniziale sarà pronto entro la fine del 2021, per essere seguito da un prototipo di dimensioni reali entro la fine del 2022. Nella seconda metà del 2023, la prima nave con una fuel cell ad ammoniaca prenderà il largo: si tratta della Viking Energy, battello della compagnia norvegese Eidesvik già protagonista di altre sperimentazioni "verdi" come la propulsione a gas naturale liquefatto.

www.fraunhofer.de



Tecnologia Siemens per Bus sostenibili

La città tedesca di Ratisbona vede per le proprie strade autobus elettrici già dal 2017 ed oggi si affida a Siemens Smart Infrastructure per incrementare l'elettificazione attraverso l'allestimento, nel deposito degli autobus nella Markomannenstrasse, di un'infrastruttura di ricarica e della tecnologia di distribuzione dell'energia sviluppata da Siemens, dimostratasi ideale per integrarsi nel deposito degli autobus esistente, in modo semplice ed economico.

Le stazioni di ricarica Sicharge UC 200 della Siemens riforniranno 22 nuovi autobus elettrici in 23 parcheggi con una potenza di ricarica di 150 kW ciascuno. La ricarica avverrà durante la notte o durante altre interruzioni delle operazioni utilizzando cavi e connettori di ricarica. 16 dei 23 parcheggi avranno uno speciale avvolgicavo che si avvolge automaticamente al termine della ricarica, lasciando libere e percorribili le corsie carrabili. I veicoli saranno alimentati con elettricità verde prodotta a livello locale e rinnovabile da REWAG, il fornitore di energia del gruppo di Ratisbona.

Per collegare il deposito alla rete elettrica pubblica e per distribuire l'elettricità nell'edificio, la Siemens sta installando anche trasformatori e un quadro di bassa tensione S8 Sivacon. Le singole stazioni di ricarica nel deposito verranno quindi alimentate tramite un sistema di condotti sbarre Sivacon 8PS. Ciò che rende unica questa forma di distribuzione di energia sono gli speciali connettori che adattano facilmente il sistema di condotti elettrici alle particolari caratteristiche dell'edificio esistente. Questa soluzione è più flessibile dei cavi, più facile e veloce da installare. È possibile aggiungere carichi aggiuntivi tramite unità di derivazione secondo necessità.

www.siemens.com/smart-infrastructure

Ricarica wireless a distanza per cellulari

Oppo ha annunciato il suo rivoluzionario sistema di ricarica wireless a distanza, che dovrebbe esordire con il concept smartphone Oppo X 2021.

Al Mobile World Congress di Shanghai, Oppo ha annunciato l'implementazione e futura diffusione nei suoi smartphone di ultima generazione del sistema di ricarica wireless a distanza, che rispetto a quelli già visti si distingue perché non è necessario appoggiare l'apparato a una piastra contenente l'antenna irradiante, ma è sufficiente che sia nelle vicinanze.

La sua tecnologia di ricarica flash VOOC, utilizzata da altre aziende e una soluzione di ricarica wireless a distanza con cui sfiderà l'analoga soluzione Xiaomi.

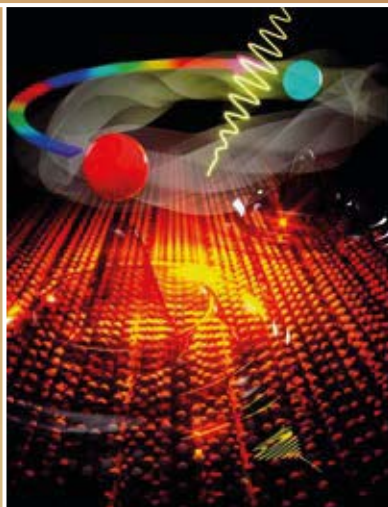
Questa tecnologia, basata sulla risonanza di un circuito elettromagnetico,

dovrebbe iniziare a caricarsi quando si trova entro 10 cm dal suo speciale tappetino di ricarica restando però utilizzabile, a patto di non allontanarsi troppo dalla postazione di ricarica.

Mentre Mi Air Charge, l'analoga soluzione di Xiaomi, supporta solo la ricarica da 5 W al momento, Oppo afferma che la sua versione può salire a 7,5 W; la gara è ora per vedere chi riuscirà a integrare per prima tecnologie di questo tipo in un dispositivo consumer.

Con The Flash Initiative, Oppo mira a integrare la ricarica VOOC in una gamma di prodotti di altri produttori partner, tra cui si ipotizza che figureranno Anker, Volkswagen ed NXP Semiconductors.

events.oppo.com/en



Transistor topologici: veloci e a consumo zero

I futuri transistor potrebbero basarsi su "materia topologica" in cui l'elettricità fluisce solo attraverso le superfici, quasi senza dissipazione di energia. Recenti scoperte suggeriscono che i materiali topologici potrebbero un giorno trovare impiego nell'elettronica ad alta velocità e bassa potenza e nei computer quantistici. Alcuni scienziati hanno rivelato di poter attivare queste proprietà elettroniche esotiche usando esplosioni di raggi laser. I ricercatori hanno studiato il pentatellururo di zirconio (ZrTe₅), che possiede straordinarie proprietà elettroniche in base alla sua topologia. La topologia è la branca della matematica che indaga su quali aspetti delle forme possono sopravvivere alla deformazione. Ad esempio, un oggetto a forma di ciambella può deformarsi a tazza ma non potrebbe assumere una forma che non abbia un foro. I ricercatori hanno sviluppato isolatori topologici elettronici dove gli elettroni che pulsano lungo i bordi o le superfici sono "topologicamente protetti", il che significa che resistono a qualsiasi disturbo che potrebbe ostacolare il loro flusso. Nel pentatellururo di zirconio, noto come semimetallo di Dirac, gli elettroni si muoverebbero a velocità di circa un trecentesimo di quella della luce su distanze fino 10 micron.

www.ameslab.gov

Immagini panoramiche da Marte

Atterrata con successo il 18 febbraio, la sonda Rover Perseverance inviata dalla NASA su Marte si sta già dando da fare, esplorando la superficie del Pianeta rosso e inviando sulla Terra stupende immagini panoramiche riprese dalla sua Mastcam a 360° e registrazioni audio. Il rover Mars 2020 Perseverance ha gettato il suo primo sguardo ad alta definizione intorno alla sua "casa" nel cratere Jezero il 21 febbraio, dopo aver ruotato la sua "testa", di 360 gradi, consentendo alla Mastcam-Z di catturare il suo primo panorama marziano. Mastcam-Z è un sistema a doppia fotocamera dotato di una funzione di zoom, che consente alle telecamere di ingrandire, mettere a fuoco e acquisire video ad alta definizione, nonché immagini panoramiche a colori e 3D. Il sistema di telecamere può rivelare

dettagli da 3 a 5 millimetri vicino al rover e da 2 a 3 metri nei pendii lontani lungo l'orizzonte. Con questa capacità, l'astrobiologo robotico può fornire un esame dettagliato di oggetti vicini e distanti. Le telecamere aiuteranno gli scienziati a valutare la storia geologica e le condizioni atmosferiche del cratere Jezero, nonché a identificare rocce e sedimenti degni di un'analisi più attenta da parte degli altri strumenti del rover. Le telecamere aiuteranno anche il team della missione a determinare quali rocce il rover dovrebbe campionare e raccogliere per l'eventuale ritorno sulla Terra. Ottenuto componendo 142 immagini, il panorama rivela il bordo del cratere e la parete rocciosa di un antico delta di fiume in lontananza.

www.nasa.gov

Nuovo allarme evoluto WiFi-GSM/GPRS



€119,00

Cod. FR783



Controllabile con:



Smart Life



Tuya Smart



Amazon Alexa



Google Home

Centralina antifurto wireless programmabile, controllabile da remoto tramite Smartphone sfruttando la rete GSM/GPRS o Wi-Fi e direttamente con Amazon Alexa o Google Home. Le App "Smart Life" o "Tuya Smart" permettono di configurare e gestire il funzionamento dell'allarme in ogni suo dettaglio. È possibile aggiungere sensori ed accessori opzionali per coprire fino a 99 zone. La confezione comprende: la centralina, due radiocomandi, un sensore wireless per porte/finestre, un sensore PIR wireless, due chiavi TAG RFID, la batteria di backup e l'alimentatore di rete. Manuale di utilizzo in italiano.

Prezzi IVA inclusa



Accessori
opzionali:

Sensore PIR
wireless



€14,00

Cod. FR783PIR

Sensore
porte/finestre
wireless



€11,00

Cod. FR783MAGN

Radiocomando
supplementare



€9,00

Cod. FR783RAD

Transponder
a portachiavi



€5,00

Cod. FR783TAG

Sirena wireless
da interno



€29,00

Cod. FR783SIR

Sirena wireless
da esterno con
pannello solare



€65,00

Cod. FR783SIREXT

I tuoi PCB a partire da

1€

Con il servizio PCB di Futura Elettronica, potrai realizzare facilmente il circuito stampato del tuo progetto!

Vai su www.futurashop.it, inserisci nella ricerca il codice "PCBPRODUCTION" e carica il tuo Gerber scegliendo le dimensioni e le caratteristiche preferite.

COLORE SERIGRAFIA



BIANCA



NERA

La serigrafia nera è disponibile solamente per PCB bianchi e gialli.

COLORE SOLDER



VERDE



ROSSO



GIALLO



BLU



BIANCO



NERO

SPESSORE (mm)



0.2

0.4

0.6

0.8

1.0

1.2

1.6

2.0

2.4

2.8

3.2

NUMERO DI LAYERS

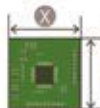
1 Layer

2 Layers

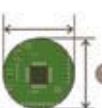
4 Layers

6 Layers

DIMENSIONI (cm)



Single Size



Pannel Size

- 5x5 - 5x10 - 5x15 - 5x20 - 5x25 - 5x30
 - 10x10 - 10x15 - 10x20 - 10x25 - 10x30
 - 15x15 - 15x20 - 15x25 - 15x30
 - 20x20 - 20x25 - 20x30
 - 25x25 - 25x30
 - 30x30



cod. PCBPRODUCTION

ECCO QUALCHE ESEMPIO:

PCB con 1 layer
 da 5x5 cm
 spessore 1,6 mm
 solder verde
 serigrafia bianca
€ 1 cad.

PCB da 2 layer
 da 10x10 cm
 spessore 1,6 mm
 solder verde
 serigrafia bianca
€ 6 cad.

Prezzi IVA inclusa.

www.futurashop.it

Futura Group srl
 Via Adige, 11 • 21013 Gallarate (VA)
 Tel. 0331/799775

**FUTURA
 ELETTRONICA®**