

Elettronica In

www.elettronica.in.it

€ 6,00 - Anno XXVII - n. 252 - MARZO 2021



Monitoraggio ambientale con uChip



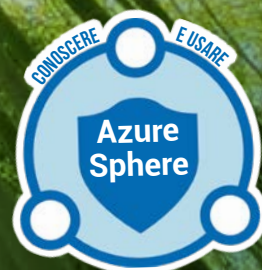
Sicurezza nell'IoT



Riconoscimento targhe con Google



- Smart sensor con web app
- Scheda a display, LED e pulsanti
- nvSRAM shield
- LoRa Edge LR1110 starter kit



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



La sicurezza prima di tutto

"Darcey Lee è una donna di 68 anni che vive sola con il suo gatto in un appartamento alla periferia di New York. Una sera di un'estate di qualche anno fa si vede piombare in casa dei poliziotti della CISA (Cybersecurity and Infrastructure Security Agency). Incredula e spaventata urla e chiede spiegazioni. I poliziotti non le rispondono e continuano a perquisire la casa alla ricerca del computer. Ma Darcey non ha il computer. Eppure Tracerouter parla chiaro, l'attacco DDoS (Distributed Denial of Service) è partito da

Basta un malware per dare modo agli hacker di infiltrarsi nei nostri dispositivi connessi

quell'abitazione. Ma come può una donna di 68 anni senza computer aver causato quell'attacco? Il router WiFi viene subito verificato. Tre i dispositivi connessi: un cellulare, uno smart speaker regalato dal figlio, e una

telecamera WiFi che Darcey ha comprato in offerta per poter controllare il gatto dal suo cellulare. Che sia...? Impossibile... E invece sì. "Qualcuno" ha usato la telecamera di Darcey per instradare l'attacco... a sua insaputa..."

Questa storia, ispirata da fatti realmente accaduti, mette in evidenza un aspetto molto trascurato: la sicurezza dei dispositivi connessi. Non è difficile trovare in rete storie di frigoriferi che inviano mail, telecamere accessibili da chiunque, router WiFi con backdoor ecc. Basta un malware infatti, per dare modo agli hacker di infiltrarsi nei nostri dispositivi connessi.

Ma a volte è sufficiente l'ingenuità. Guardate ad esempio questo sito www.insecam.org: raccoglie le telecamere pubbliche, ma anche quelle le cui credenziali non sono state modificate. Oppure provate a navigare nel sito www.shodan.io: con qualche semplice ricerca si possono trovare router, webcam, NAS e sistemi aperti e gestibili da remoto senza la necessità di inserire credenziali o con i dati di autenticazione predefiniti.

Il problema della sicurezza è spesso trascurato e con la grande disponibilità di moduli WiFi a basso costo è semplice per chiunque realizzare la propria applicazione, e spesso, spinti dall'entusiasmo di vedere il proprio progetto comunicare con il mondo, trascuriamo l'importanza di una comunicazione sicura. In questo numero analizziamo le principali modalità per rendere sicuri i vostri dispositivi IoT e iniziamo il corso di Azure Sphere, la soluzione di Microsoft nata per rendere invulnerabile la vostra applicazione. Articoli da leggere con attenzione.

ON LINE

elettronica.in

SI RINGRAZIA

Elenos
Futura Elettronica
Microchip
Mouser
Texas Instruments



In copertina:
Una piccola ma potente scheda Arduino-like, permette di rilevare dati ambientali e mostrarli graficamente su un display touch.

Boris Landoni

boris.landoni@elettronica.in

Elettronica In

Rivista mensile, anno XXVII n. 252

MARZO 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
Edizioni

Contenuti

01

RUBRICHE

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

Progetto di Copertina

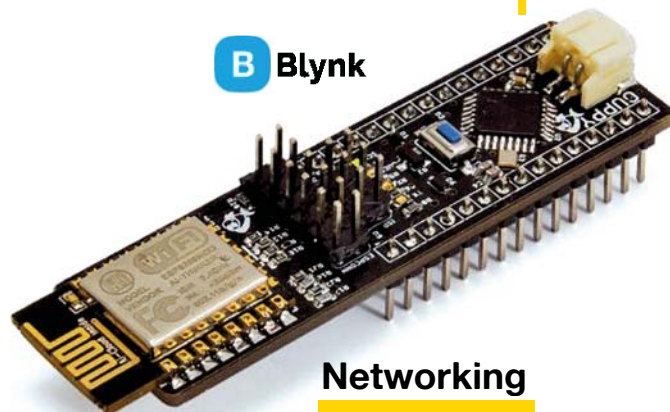
MONITORAGGIO AMBIENTALE CON UCHIP

55



31

B Blynk



Networking

Smart sensor
connesso a Blynk

19

ARTICOLI

WIRELESS

19 LoRa Edge LR1110 starter kit

Scopriamo LoRa Edge LR1110 di Semtech e il suo kit di sviluppo finalizzato alla produzione di dispositivi che coniugano le tecnologie LoRa, WiFi e GNSS per la geolocalizzazione.

NETWORKING

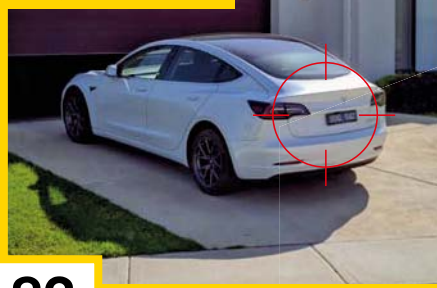
31 Smart sensor connesso a Blynk

Sviluppiamo un'applicazione IoT connettendo una scheda Fishino Guppy, equipaggiata con sensori di temperatura e umidità, alla piattaforma Blynk, che rende disponibili potenti widget per connettere il mondo dei sensori all'Internet of Things.

ARDUINO

45 nvSRAM board

Aggiungiamo una RAM "non volatile" dove scrivere e mantenere, grazie a una pila tampone, dati permanenti.

Wireless**LoRa Edge LR1110
starter kit****19****Arduino****nvSRAM board****45****Didattica****Sicurezza nell'IoT****99****Video****Riconoscimento
targa da web****83****Periferiche****Interfaccia utente
universale****71****Didattica****Corso
Azure Sphere****109****HOME PROJECT****55 Monitoraggio ambientale
con uChip**

Dotato di scheda SD ed RTC, rileva e memorizza temperatura, umidità e pressione barometrica mediante un sensore combinato, mostrando i dati graficamente su display touch.

PERIFERICHE**71 Interfaccia utente universale**

Integra 8 display a 7 segmenti, 8 led e 8 pulsanti, che possono essere gestiti in lettura e scrittura tramite bus bidirezionale a tre fili di tipo I²C.

VIDEO**83 Riconoscimento targa da web**

Usiamo il modulo ESP32-CAM per rilevare e riconoscere la targa di un veicolo ed azionare un attuttore con cui aprire un garage, sollevare una sbarra, attivare un allarme e altro ancora.

DIDATTICA**99 Sicurezza nell'IoT**

L'immediato futuro vede ogni genere di dispositivo connesso e se questo può essere fonte di una gran quantità di dati fruibili dalla Rete, l'esistenza di una gran quantità di nodi periferici rappresenta un'infinità di accessi disponibili per portare attacchi informatici. Analizziamo il problema e valutiamo le possibili soluzioni.

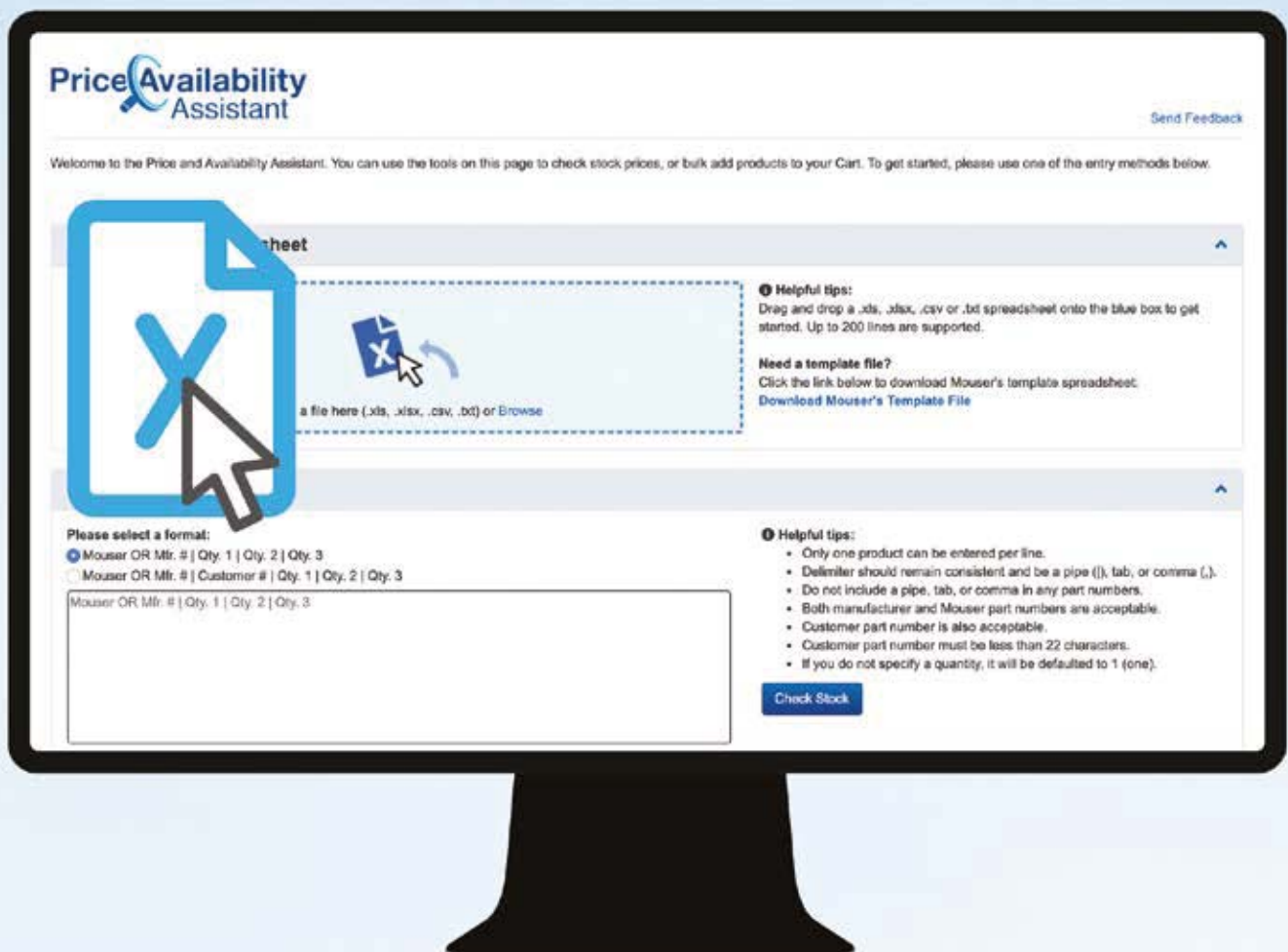
DIDATTICA**109 Conoscere e usare Azure Sphere**

Conosciamo la tecnologia di Microsoft per la creazione di applicazioni sicure, basate sull'interazione di hardware protetto e Cloud. Prima puntata: prepariamo la nostra scheda IoT.

Controllate facilmente il prezzo e la disponibilità di ogni articolo di cui avete bisogno

**PriceAvailability
Assistant**

STOCK • PRICE • BUY





Simulatore circuiti

Spesso di fronte alla realizzazione di un circuito elettronico veniamo dissuasi dalla prospettiva di realizzare una o più versioni del prototipo, con i problemi connessi alla realizzazione del circuito stampato (in casa o commissionato a uno dei tanti servizi on-line di preparazione PCB) dell'approvvigionamento dei componenti e via di seguito. Sicuramente un po' tutti sappiamo cosa stiamo progettando e confidiamo nel fatto che i nostri circuiti, una volta finiti, funzioneranno. Però avere a disposizione un simulatore che ci permetta di costruire virtualmente il nostro circuito e provarlo per vedere, prima di passare alla pratica, come il circuito funzionerà e se richiederà qualche ritocco. Un valido simulatore è quello reperibile on-line su www.falstad.com/circuit/ perché permette di creare i propri circuiti e modificarli in caso richiedano variazioni: consente infatti di avviare la simulazione del funzionamento, eventualmente applicando strumenti di misura in punti seleziona-

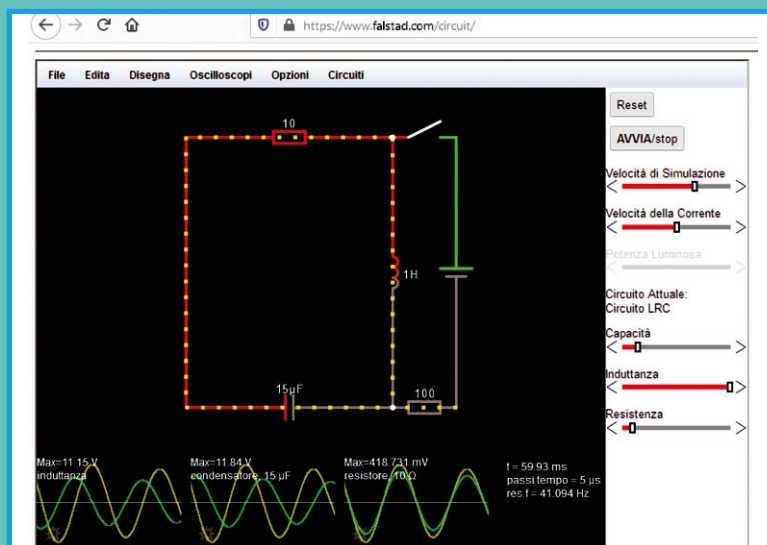
Tips & Tricks sono i nostri suggerimenti in pillole e i piccoli trucchi riguardanti sia l'hardware che il firmware, che sperimentatori, tecnici e progettisti elettronici vorrebbero conoscere per sviluppare meglio la propria passione: quella per l'Elettronica applicata. Ecco quelli che abbiamo preparato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

bili (funzione Oscilloscopio). Tutte le funzioni sono accessibili da pratici menu nella toolbar e soprattutto attraverso i menu contestuali che si aprono cliccando con il pulsante destro del mouse sull'area di lavoro. Con i cursori a destra di questa area è possibile variare alcuni parametri ed anche impostare la velocità della simulazione e dello scorrimento virtuale della corrente nei componenti.

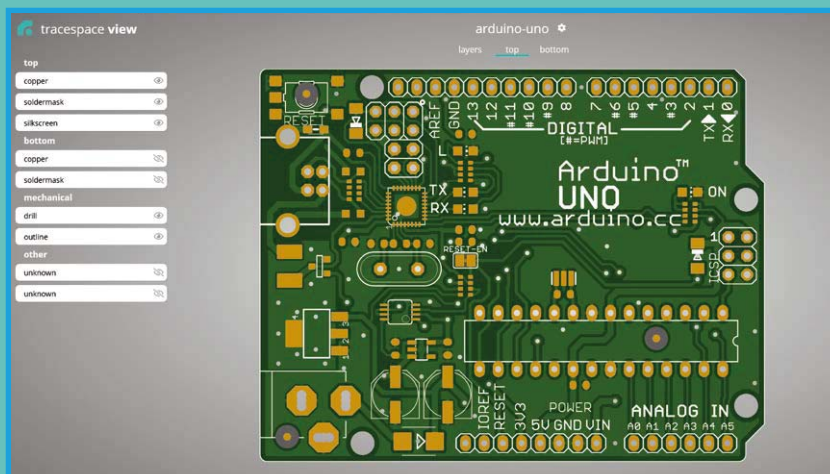
Verificare i file gerber

I file descrittivi dei circuiti stampati disegnati con i vari CAD per progettazione di PCB si chiamano gerber e riguardano le tracce lato rame, ma esistono anche altri layout utilizzati in produzione, come ad esempio la serigrafia (silk) il solder-resist e la foratura automatizzata con macchine CNC (drill).

Essendo disponibili molti programmi CAD, talvolta è difficile aprire i file creati da un software senza averlo, perché non sempre c'è compatibilità dei formati, per questo sarebbe comodo disporre di uno strumento universale in grado, se non di editare i gerber, quantomeno di visualizzarli. E questo strumento esiste, ovviamente sul web, che oggi offre di tutto; nello specifico si parla di Tracespace, accessibile da tracespace.io/view/. La pagina iniziale di questo sito consente di caricare file gerber di vari programmi e di visualizzarli con accanto i relativi attributi. Per caricare i file basta cliccare sul pulsante con il simbolo + in alto a destra nella pagina e si apre una finestra di selezione che permette di selezionare il percorso del computer o di rete da cui caricare il gerber; una volta selezionato e importato il file, in alto in centro alla pagina appare il suo nome con



↑ La pagina web del sito www.falstad.com che permette di simulare i circuiti.



sotto una toolbar che riporta le voci "layers" (per vedere gli eventuali strati interni del circuito stampato, se è un multilayer) "top" per vedere il lato componenti e "bottom" per il lato inferiore (tipicamente il lato saldature) del circuito stampato. Cliccando sulle rispettive voci è possibile vedere la traccia corrispondente. A sinistra si trovano invece i parametri delle tracce trovate, ossia top (traccia rame superiore) bottom (traccia inferiore) mechanical (riguardante l'eventuale piano di foratura incluso) e other (varie altre tracce). Vengono mostrate di volta in volta le sezioni corrispondenti al tipo di gerber, nel senso che se c'è una sola faccia appaiono le caselle corrispondenti.

Convertitore Base64 online

Non avete dimestichezza con l'espressione Base64 e avete un codice

in questo formato che volete, senza troppa fatica, decodificare? Potete allora utilizzare il servizio on-line di conversione accessibile dal sito www.base64decode.org. Base64 è una codifica che consente la traduzione di dati binari in stringhe di testo ASCII, rappresentando i dati sulla base di 64 caratteri ASCII diversi.

Per intendersi è qualcosa di utilizzato per la codifica dei dati binari nei messaggi di posta elettronica per convertire i dati nel formato ASCII, mostrandoli come caratteri. Si tratta quindi di dati nel formato a 8 bit.

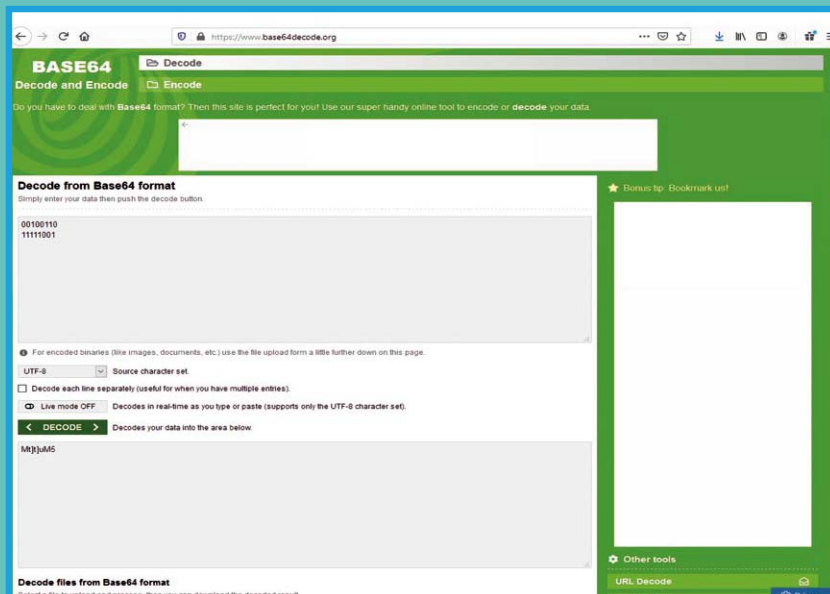
Il sito presenta due riquadri: in quello superiore si scrive o si carica il codice in formato Base64 e in quello inferiore appare il formato convertito (per la conversione bisogna cliccare sul pulsante verde DECODE). Mediante un menu a tendina è possibile impostare il set di caratteri sorgente (quello predefinito è l'UTF-8) per specificare al sito quale de-

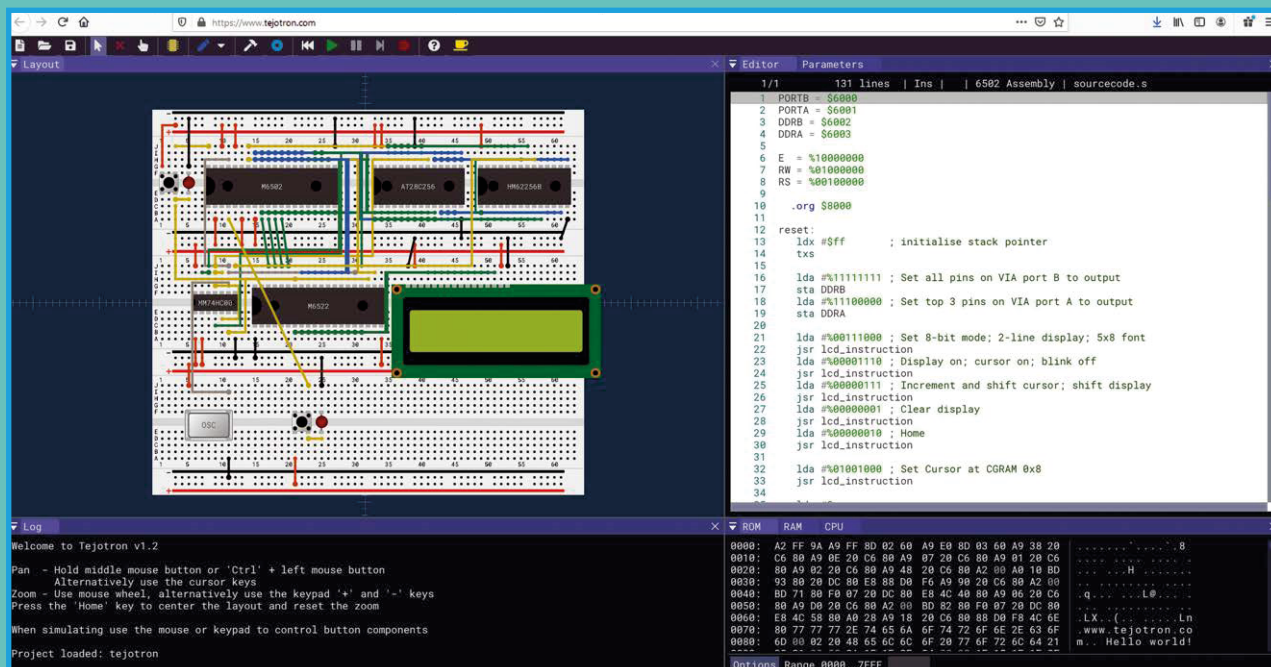
codifica debba utilizzare, ovvero lasciare che sia il convertitore a identificarlo (AUTO-DETECT). È possibile spuntare una casella per dire al sistema, nel caso sia inserito del codice composto da più linee, di decodificare ciascuna linea distintamente: senza spuntarla, il risultato della decodifica viene accodato su un'unica linea sotto forma di caratteri in fila, mentre spuntandola, i caratteri corrispondenti alle stringhe di ciascuna riga vengono messi ciascuno su una propria riga.

Provare diodi e BJT con il tester digitale

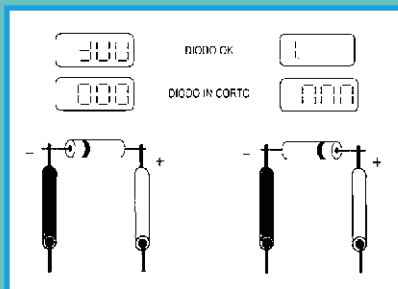
Quando in una porzione di circuito nella quale sono inseriti dei diodi non passa la corrente o non troviamo la tensione che ci aspettiamo, andiamo subito a pensare come verificare se i diodi sono interrotti o magari in cortocircuito. Per dirimere il dubbio possiamo utilizzare il tester predisposto per la verifica delle giunzioni PN: di norma esiste una specifica portata o una posizione del commutatore rotativo di selezione, ovvero una boccia dedicata (nei multimetri che non hanno il selettore); la prova si esegue comunque con una portata ohmmetrica, vale a dire con la misura della resistenza, perché prevede l'erogazione di una tensione e una piccola corrente attraverso i puntali dello strumento. Solitamente nei tester la prova delle giunzioni si esegue con la portata 2 kohm f.s. che in molti casi permette il test dello stato di diodi al silicio, in germanio e anche i LED (con il multimetro digitale); molti multimetri hanno, su questi, il simbolo del diodo proprio per ricordare questa possibilità. Altri strumenti, invece, hanno una posizione speciale del selettore dedicata alla prova diodi. In ogni caso, qualunque sia l'ambito o la funzione scelta, per verificare una giunzione è necessario rispettare la corretta polarità: inserire la spina del puntale nero (negativo) nella boccia COM e quella del puntale positivo (rosso) nella boccia V/Ω.

Successivamente si tocca l'anodo del diodo con il puntale rosso e il catodo con il nero: il terminale di catodo si riconosce perché è quello vicino alla banda colorata sul corpo del componente. Disponendo i puntali del tester in tale maniera, lo strumento deve indicare una resistenza dell'ordine di poche centinaia ohm, ma dipende dalla potenza del diodo (varia da diverse centinaia a meno di 100 ohm). Se si invertono i puntali (portando il diodo in polarizzazione





inversa) il quadrante dello strumento dovrà mostrare resistenza infinita (non misurabile) nel senso che nel multimetro analogico la lancetta andrà sul inizio scala e in quella digitale l'indicazione sarà 1. Se lo strumento continuerà a indicare una resistenza misurabile, vorrà dire che il diodo è danneggiato; lo stesso vale se la resistenza misurata tenendo il puntale rosso sull'anodo è vicino o uguale a zero.



In questo caso il diodo è praticamente in cortocircuito perché la sua giunzione di semiconduttore si è fusa per un'eccessiva dissipazione di potenza o forata per una sovratensione. Un diodo in cortocircuito, peraltro, presenta uguale resistenza invertendo i puntali del tester, cioè in tutte e due le polarizzazioni, giacché non fa più da semiconduttore ma è pari a una resistenza elettrica di bassissimo valore. Infine, un'indicazione da parte dello strumento di valore resistivo infinito anche in polarizzazione diretta (puntale rosso sull'anodo e nero sul catodo) indica che il diodo risulta interrotto. La verifica dell'integrità delle giunzioni descritta sinora per i diodi

può essere applicata anche ai transistor bipolari, giacché sono formati da due giunzioni PN, ciascuna delle quali singolarmente è un diodo. Per verificare l'integrità delle giunzioni consideriamo una giunzione alla volta utilizzando come riferimento il terminale che resta in comune, ovvero la base. Per il resto si procede considerando il transistor come un doppio diodo con l'anodo in comune se trattasi di un NPN e il catodo in comune se, invece, è un PNP. Durante i test bisogna ricordare di non toccare i terminali né il metallo dei puntali con le dita, altrimenti si inserisce la resistenza tra le dita e si otterranno indicazioni imprecise.

Simulazione e debug di firmware on-line

Creare un circuito logico programmabile a microcontrollore o microprocessore, con memorie e quant'altro, può presentare alcune difficoltà di progettazione. Quello accessibile da www.tejotron.com è un'interessante web app di simulazione software, che permette di cablare su delle breadboard virtuali circuiti digitali, editarne e simularne il firmware, verificando anche i codici in linguaggio macchina e il contenuto della memoria impiegata, così da fare tutte le valutazioni progettuali ed anche il debug in caso qualcosa non vada secondo le aspettative. Si tratta di un servizio didattico dedicato alla progettazione software, che fornisce il supporto necessario a chi deve svilup-

pare circuiti programmabili; ideale per gli sperimentatori, insegna i fondamenti attraverso pannelli che forniscono la situazione del programma in sviluppo o in fase di test.

Emulare Raspberry Pi con Windows 10

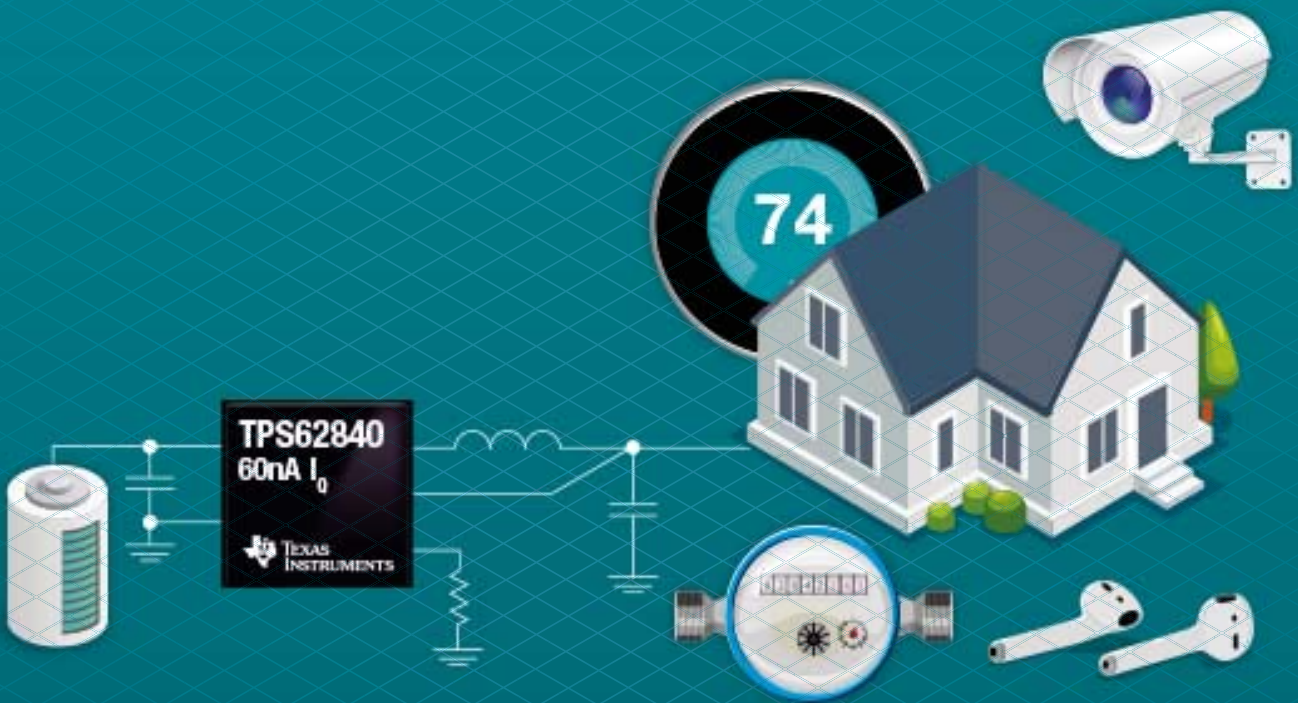
Come tutti i computer, anche Raspberry Pi (che è un single-board-computer) può essere virtualizzata e simulata su un sistema operativo in esecuzione su un Personal Computer; può quindi diventare una macchina virtuale, in esecuzione magari in ambiente Windows. Un esempio di ciò lo troviamo sul web all'indirizzo <https://www.qemu.org/> da dove si può scaricare l'emulatore di Raspberry Pi per Windows 10. Questo consente di simulare il sistema operativo Raspbian, in ambiente Windows 10 per verificare il codice scritto.

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

Date il benvenuto al regolatore switching con l' I_Q piu' basso al mondo.



Estendete la durata della batteria del vostro sistema grazie alla piu' alta efficienza possibile a carico minimo.

<http://www.ti.com/product/TPS62840>

TN1, produttore coreano di sistemi di monitoraggio industriale, moduli LCD embedded e dispositivi destinati all'IoT, sta entrando nel mercato worldwide con due smart panel dedicati all'IoT Industriale e professionale. La serie JUFT di TN1 costituisce una soluzione easy-touch di facile utilizzo che aiuta i progettisti a implementare facilmente la GUI (interfaccia grafica d'utente) attraverso un'interfaccia di comunicazione seriale e fornisce una facile integrazione con i sistemi esistenti. Il pannello JUFT M-A è destinato ad applicazioni come dispositivi medicali, macchine per la cura del corpo, stampanti 3D e siti industriali di varie dimensioni ed è attualmente

› L'IoT passa dallo smart panel



utilizzato in numerosi settori, inclusi siti industriali intelligenti (Smart Factory), sistemi di check-in senza personale e test di prototipi.

Nell'ambito della sua continua attività di ricerca e sviluppo, TN1 sta applicando la sua tecnologia SDC (Sensing Display Connectivity) per l'IoT alla tecnica di acquacultura nota come acquaponica, in cui si rivolge a tutti i lavoratori e gli ambienti dove occorre conoscere lo stato delle grandezze rilevate dai sensori e l'avanzamento del processo "produttivo".

Inoltre l'azienda coreana sta espandendo la propria presenza con un sistema di monitoraggio e controllo basato su cloud.

www.tn1.co.kr

› L'hockey su ghiaccio incontra la sensoristica

La NHL (National Hockey League) americana è andata ad implementare la tecnologia Puck and Player Tracking per le partite del campionato di hockey su ghiaccio. La tecnologia permette di tenere traccia di ogni movimento del disco (chiamato "puck" in gergo sportivo) e dei giocatori di ciascuna squadra durante una partita, con una frequenza fino a 200 volte al secondo e monitorando così la posizione con estrema precisione e in tempo reale. Il sistema Puck and Player Tracking può tracciare i puck a una velocità di 2.000 volte al secondo in tempo reale con una precisione a livello di pollice. Per raggiungere lo scopo, il sistema includerà 14-16 antenne installate nelle travi dell'impianto sportivo, quattro telecamere per supportare la funzionalità di tracciamento, un sensore posto sugli spallacci della tuta di ogni giocatore di ogni squadra e 40 dischi dotati di sensore interno basato su RFID.

La tecnologia è fornita dal Fraunhofer Institute tedesco e dalla sua controllata, Jogmo World Cup e deve affrontare sfide come la resistenza dell'elettronica al congelamento del disco e la grande velocità cui si può spostare, che arriva a superare i 180 km/h. I dati

ottenuti possono essere utilizzati per vari scopi: verificare l'entrata in rete del disco, analizzare i dati sullo spostamento dei giocatori e perfezionare le tattiche di gara, valutare la prestazione degli atleti ed altro ancora.

www.nhl.com/news



› Il Machine Learning descrive le immagini

Facebook ha lanciato una nuova modalità di AAT (testo alternativo automatico) che sfruttando l'Intelligenza artificiale migliora precisione e dettaglio delle descrizioni delle foto pubblicate sul diffusissimo Social e sul suo "cugino" Instagram, per aiutare i non vedenti e gli ipovedenti a capire i contenuti visivi.

Gli utenti del genere non possono comprendere il contenuto delle immagini a meno che non siano etichettate correttamente con un testo alternativo (anche detto "alt text") ma purtroppo molte foto vengono postate senza questo dato. Per ovviare a questo problema, nel 2016 Facebook introdusse la tecnologia AAT di visione artificiale per generare, in

modo automatico, le descrizioni di un'immagine, utilizzando parametri esistenti nel suo database. Questo lettore di schermo può descrivere il contenuto delle immagini utilizzando una voce sintetica, consentendo alle persone non vedenti o ipovedenti di "ascoltare" le foto nei loro feed di Facebook e di Instagram. Tale strumento è stato attualmente perfezionato incrementando il database e addestrando l'algoritmo di Intelligenza Artificiale tanto da riconoscere un numero 10 volte maggiore di elementi all'interno delle immagini (arrivando a 1200), distinguendo attività, simboli e tipologie di animali, fornendo descrizioni più dettagliate o indicando la possibilità che si tratti di questo o di quello.

› Il satellite conta gli animali grazie all'AI

Gli elefanti sono una specie da proteggere e per questo è opportuno osservarli, foss'anche impiegando i satelliti e i più sofisticati algoritmi di calcolo. Per questo nasce il sistema di osservazione basato sul satellite WorldView 3 della Maxar, che mediante una telecamera ad alta risoluzione e lunga portata permette di distinguere gli animali nelle immagini riprese ed anche di contarli.

L'impresa si deve alle Università di Oxford e di Bath del Regno Unito. In vero non è la prima volta che i satelliti vengono usati per contare e monitorare le specie animali: è stato già fatto per balene e pinguini, però era tecnicamente

più semplice in quanto lo sfondo delle immagini era abbastanza uniforme, perché si trattava del blu dell'oceano e del bianco del ghiaccio dell'Antartide; invece stavolta l'algoritmo è riuscito a distinguere e contare gli animali in uno sfondo eterogeneo che comprende praterie, arbusti e aree alberate.

Gli elefanti africani sono stati scelti perché sono il più grande animale terrestre e quindi il più facile da individuare, ma anche perché sono una specie vulnerabile, a causa della caccia cui sono soggetti per il valore dell'avorio ricavabile dalle loro zanne.

Lo studio sugli elefanti permetterà ai ricercatori di fare evolvere l'algoritmo,

che si spera possa presto individuare dallo spazio specie molto più piccole.

Per la sperimentazione sono state utilizzate le immagini fornite dal satellite WorldView-3 Maxar, situato in orbita a 617 km dalla Terra, che può fotografare un'area terrestre estesa per 5.000 km² in pochi minuti; proprio questa rapidità di scansione consente di ripassare sulla stessa area in un tempo sufficientemente breve da effettuare riconteggi consecutivi e confronti dei risultati, in grado di fornire un risultato preciso. Peraltro l'utilizzo del satellite facilita il controllo degli animali che si spostano da una nazione all'altra senza i vincoli che comporterebbe farlo da terra o con velivoli, ad esempio, costi, autorizzazioni per il valico delle frontiere (agli animali non si chiede il passaporto come agli uomini...) ed effetti di guerre e disordini. L'algoritmo che conta gli elefanti si basa su una rete neurale convoluzionale costruita grazie alle API TensorFlow Object Detection. TensorFlow è una libreria open source per machine learning sviluppata dal team Google Brain, mentre le API Object Detection sono state create allo scopo ed alla loro creazione hanno contribuito anche alcuni ricercatori della Nvidia; peraltro TensorFlow è una libreria che abbiamo utilizzato in alcuni progetti proposti di recente.

blog.maxar.com



Sempre grazie all'Intelligenza artificiale, l'AAT è ora in grado di fornire informazioni dettagliate su un'immagine specifica, compresa la posizione degli elementi nella fotografia (se si trovano in alto, al centro, in basso, a sinistra o destra) e la loro dimensione relativa all'immagine, quindi primo piano, sfondo ecc. Le descrizioni del testo alternativo sono attualmente generate con traduzione in 45 lingue e lo strumento è disponibile su Facebook per le foto nel NewsFeed, nei profili e nei gruppi; su Instagram invece, è accessibile per le foto nel Feed, in Esplora e nei profili.

www.facebook.com



May be an image of
1 person, standing,
and Machu Picchu

› Scooter elettrici IoT: possibile, con Ericsson



La diffusione della micro-mobilità urbana condivisa e quindi di biciclette e monopattini elettrici pubblici, implica la necessità della localizzazione, sia per comunicare agli utenti dove si trova il primo mezzo disponibile, sia per evitare furto e smarrimento.

In quest'ottica nasce il lavoro di Arkesa, che attraverso la piattaforma IoT Accelerator della Ericsson eroga connettività alle flotte di e-scooter Voi in

tutto il mondo: si tratta di connettività IoT da rete cellulare e soluzioni eSIM per facilitare la localizzazione dei mezzi e supportare l'implementazione delle flotte di monopattini elettrici Voi, consentendo sia di trovare i veicoli nelle vicinanze, sia di attivarli quando l'utente ne ha bisogno. Il tutto, contenendo i costi e garantendo ampia copertura del servizio grazie alla diffusione della rete supportata da Ericsson.

Voi è una società svedese di micro-mobilità che offre la condivisione di e-scooter in collaborazione con gli enti locali. Nei veicoli elettrici è inserita una SIM che può connettersi a diversi operatori e fornisce la connettività cellulare che grazie alla piattaforma Ericsson IoT Accelerator permette di gestire tutti i parametri operativi, assegnare i mezzi agli utenti che li hanno prenotati, localizzarli per indicare come raggiungerli, prelevarli in caso di guasto e recuperarli se vengono rubati.

Sempre più realtà sfruttano la tecnologia IoT che utilizza come veicolo dei dati la connettività cellulare, la quale è presente ormai dovunque e soprattutto nelle aree urbane; quindi è già disponibile e non richiede l'implementazione di reti wireless dedicate, come è per altre tecnologie IoT.

Ericsson, con la sua piattaforma e i suoi dispositivi, è tra i principali attori di un mercato che vedrà il numero di connessioni IoT da rete mobile raggiungere i 5,9 miliardi nel 2026.

Ericsson, tra i primi produttori di telefoni cellulari e reti telefoniche, con la sua piattaforma IoT consente raccolta e gestione dei dati.

www.ericsson.com

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.

Prezzi IVA inclusa.



cod. PCBPRODUCTION

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

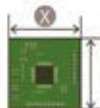
1 Layer

2 Layers

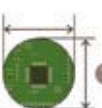
4 Layers

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

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.

Da Siemens un efficiente caricabatteria EV

Siemens Smart Infrastructure ha rilasciato il suo nuovo caricabatterie rapido Sicharge D destinato a soddisfare le esigenze della eMobility; è adatto a stazioni di ricarica rapida urbane e autostradali, parcheggi cittadini e centri commerciali ed offre una delle più alte efficienze di picco sul mercato (ben il 96%) potenza di ricarica scalabile e condivisione dinamica della potenza. Il nuovo caricatore pubblico di Siemens Sicharge D ha una potenza di ricarica scalabile fino a 300 kW, sia di fabbrica che tramite aggiornamenti plug-and-play. Inoltre supporta tensioni comprese tra 150 e 1.000 volt (V) e correnti fino a 1.000 ampere (A) su tutte le prese CC; ciò consente carichi a piena potenza per i futuri veicoli a 800 V, nonché i tassi di ricarica a bassa tensione richiesti dai veicoli tradizionali di oggi. La configurazione standard prevede due prese CC e una CA alloggiate in un design compat-

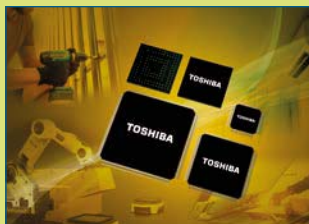


to che riduce al minimo la complessità dell'installazione. Il Sicharge D supporta la ricarica parallela dinamica, nel senso che tiene conto della richiesta di energia individuale di ciascuna auto connessa e adatta automaticamente il processo di ricarica alla tecnologia della batteria del veicolo elettrico in carica. Ciò garantisce che le auto connesse ottengano la mas-

sima potenza di cui hanno bisogno senza alcun intervento manuale aggiuntivo. Un display regolabile da 24 pollici integrato consente a chiunque di utilizzare facilmente il caricabatterie Sicharge D e può anche funzionare come strumento di intrattenimento e informazione durante le ricariche.

www.siemens.com/sicharge-D

Nuovi MCU a 32 bit Arm Cortex-ML



Toshiba aggiunge microcontrollori alla propria famiglia TXZ+ di dispositivi altamente integrati ottimizzati per i sistemi di controllo di motori, elettrodomestici, interfacce uomo-macchina e infrastrutture IoT. I dispositivi M3H sono basati sul core Arm Cortex-M3m operano fino a 120 MHz (150 DMIPS) e sono dotati di 512 KB di memoria Flash di codice, 32 KB di Flash dati e 4 KB di Flash utente separate, integrate con un ADC a 12 bit e due DAC a 8 bit. Includono una funzionalità avanzata di azionamento motori pro-

grammabile (A-PMD) e quella di pilotaggio degli LCD.

I dispositivi M4K e M4M si basano su Arm Cortex-M4 con FPU e operano fino a 160 MHz (203 DMIPS); hanno 256 KB di flash di codice, 32 KB di flash dati e 4 KB di Flash utente. Consentono il controllo FOC di un massimo di tre motori.

I microcontrollori M4M includono un controller CAN aggiuntivo e gli M4G e M4N sono ottimizzati per l'elaborazione di dati complessi fino a 254 DMIPS. I dispositivi hanno fino a 2 MB di memoria Flash di codice integrata e 32 kB di flash dati separata e integrano il blocco IP A-PMD per il controllo dei motori.

I dispositivi M4N includono inoltre un controllore OTG USB 2.0 full-speed con fino a 2 porte, un MAC Ethernet 10/100 e un controllore CAN.

toshiba.semicon-storage.com

Storage controller 24G SAS/PCIe Gen 4 Tri-mode da Microchip

Microchip ha messo in produzione i controller di archiviazione Smart Storage PCIe Gen 4 Tri-Mode SmartROC (RAID-on-Chip) 3200 e SmartIOC (I/O Controller) 2200 che includono il supporto all'interfaccia PCIe Gen 4 con tecnologia DirectPath per transazioni NVMe a bassa latenza e SAS 24G con Dynamic Channel Multiplexing (DCM) per un'aggregazione più efficiente di SAS o Dischi rigidi SATA su infrastruttura SAS 24G.

I dispositivi SmartROC 3200 e SmartIOC 2200 supportano interfacce host PCIe Gen 4 x8 e x16 e fino a 32 lane SAS/SATA/NVMe. I dispositivi supportano l'Universal Backplane Management (UBM) di SFF, Intel Virtual Pin Port (VPP) per la gestione intelligente del backplane e le specifiche PLDM (Platform Level

Data Model)/Redfish Device Enablement (RDE) basate su standard DMTF che semplificano l'implementazione della gestione fuori banda su MCTP (Management Component Transport Protocol)/BMC (Baseboard Management Controller). Microchip ha ampliato la sua esclusiva soluzione CBE (Controller-Based Encryption) maxCrypto per supportare SAS, SATA e, ora, NVMe.

www.microchip.com



Il laboratorio di misure in un tablet



PHYWE, azienda specializzata in piattaforme per la didattica in ambito tecnico e scientifico, propone Cobra SMARTlink, un tablet di misura, visualizzazione e acquisizione dati basato su CPU quad-core 2 GHz e dotato di touch screen da 10,1" Full-HD. E' stato sviluppato per il moderno insegnamento

delle scienze digitali e, oltre a sensori interni per accelerazione, luce, UV, umidità, temperatura (2 termocoppie tipo K per misurare fra 200 e 1.200 °C), corrente e tensione, consente la connessione da Bluetooth 5.0 o USB, di numerosi sensori esterni.

Il software di misura measu-

reAPP permette di acquisire, visualizzare, valutare, combinare, condividere più di 40 parametri di misura.

Cobra SMARTlink dispone anche di WiFi, memoria interna 64÷128 GB, 4 GB di RAM, fotocamere 2 e 5 MP, accelerometro e batteria 6.000 mAh.

www.phywe.com

NXP sblocca lo spettro a 6 GHz con il chipset tri-band WiFi 6E

NXP annuncia CW641, il suo primo SoC Tri-Band WiFi 6E per la banda 6 GHz, capace di estendere lo spettro disponibile fino a 1,2 GHz. Il SoC è progettato per access point e gateway di provider di servizi per consentire ai dispositivi di rete finale di sfruttare appieno lo spettro a 6 GHz. Il componente supporta una larghezza di banda del canale di 160 MHz con velocità di 4,8 Gbps e oltre 4 Gbps di throughput reale.

Con l'aumento della congestione nelle bande legacy da 2,4 GHz e 5 GHz, la FCC degli Stati Uniti ha approvato lo spettro di 1,2 GHz di libero utilizzo per la banda da 6 GHz; il SoC NXP nasce per utilizzare tale banda. L'aggiunta di funzionalità a 6 GHz alle piattaforme gateway offre ai fornitori di servizi la possibilità di suddividere in modo efficiente la larghezza di banda disponibile tra i dispositivi, per ottimizzare l'esperienza utente in un'ampia gamma di applicazioni. Le applicazioni mission critical, caratterizzate da larghezza di banda elevata e bassa latenza, come mesh back haul e cloud gaming, potrebbero migrare sui 6 GHz,



liberando le bande da 5 GHz e 2,4 GHz ad applicazioni che richiedono larghezza di banda inferiore. Il nuovo SoC Tri-Band WiFi 6E di NXP consente di sfruttare appieno lo spettro trasformativo a 6 GHz per aumentare le prestazioni delle reti mesh domestiche, streaming di musica e video ad alta risoluzione, giochi online, videochiamate, download digitali, contenuto web con molti dati e numerosi altri casi d'uso. Oltre alle applica-

zioni per punti di accesso, il SoC CW641 pone le basi per applicazioni WiFi 6E ad alte prestazioni in ambito consumer, automobilistico, industriale e Internet of Things (IoT). Le caratteristiche principali di CW641 includono transceiver WiFi 4x4 con 4 flussi spaziali, funzionamento a 6 GHz, 5 GHz e 2,4 GHz, banda del canale 160/80/40/20MHz, OFDMA uplink e downlink MU-MIMO.

media.nxp.com

FMBC EP: il filtro di rete trifase a doppio stadio

SCHURTER presenta la famiglia di filtri a doppio stadio FMBC EP per sistemi trifase. Grazie all'utilizzo di componenti di qualità (condensatori a film e induttanze avvolte su nucleo ad alta permeabilità) forniscono alta attenuazione e sono adatti ad applicazioni industriali dove si verificano elevate emissioni EMC. Le

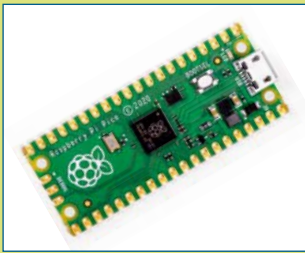
applicazioni tipiche includono macchine e sistemi per la produzione automatica, azionamenti per motori e macchinari per impianti industriali, ma anche inverter di frequenza per il controllo motore. I filtri sono progettati per correnti da 16A a 230A a una temperatura ambiente di 50°C e sono omologati sia ENEC che

cURus, a tensioni fino a 760 Vca. Le versioni standard sono adatte ad applicazioni industriali con correnti di dispersione <4 mA, mentre quelle speciali arrivano a dispersioni inferiori a 1 mA. I filtri di rete FMBC EP hanno morsetti per il cablaggio e la connessione di massa attraverso il case.

www.schurter.ch



Raspberry Pi diventa "Pico"



La Raspberry Pi Foundation lancia la nuova scheda Raspberry Pi Pico gestita da un Cortex-M0 + dual-core.

Raspberry Pi Pico è una piccola scheda, veloce e versatile costruita utilizzando il RP2040, un nuovissimo chip proprietario progettato dalla Raspberry Pi Foundation nel Regno Unito, che dispone di due core ARM Cortex-M0 operanti con clock di 133 MHz. Non mancano inoltre 264KB di RAM, nonché una porta microUSB. Ci sono anche 26 pin GPIO e un sensore per la temperatura. Per chi non lo sapesse, Pi Pico rappresenta il primo microcontroller prodotto da Raspberry.

L'RP2040 è dotato di un processore Arm Cortex-M0 + dual-core con 264 KB di RAM interna e supporta fino a 16 MB di Flash. Dispone di diversi GPIO che includono I²C, SPI e altre funzioni speciali. Per quanto potente e focalizzata su applicazioni impegnative, la nuova scheda del lampone è rivolta a tutti: principianti ed esperti dell'elettronica. I suoi impieghi sono molteplici: dal controllo di un elettrodomestico al funzionamento di un display luminoso, Raspberry Pi Pico soddisfa le esigenze di ogni sviluppatore ed è programmabile in C e MicroPython. Gli utenti più esperti possono sfruttare il ricco set di periferiche di Raspberry Pi Pico, tra cui SPI, I²C e I/O programmabili (PIO) per la gestione di periferiche personalizzate. Questa nuova scheda non ha il supporto né al Wi-Fi né al Bluetooth, ma viene offerta a un prezzo ridottissimo, che è intorno ai 4 \$ USA.

www.raspberrypi.org

Nuove soluzioni GNSS da STM

STMicroelectronics propone un nuovo integrato per protezione ESD e adattamento dell'impedenza che semplifica la progettazione di ricevitori GNSS portatili. Il front-end RF BPF8089-01SC6 nasce per i ricevitori Global Navigation Satellite System consente di risparmiare spazio. Il BPF8089-01SC6 fornisce un'interfaccia a 50 Ω tra l'antenna del ricevitore e l'amplificatore a basso rumore (LNA) ed è pronto per il plug-and-play con gli LNA STA8089 e STA8090 della ST. Questo dispositivo compatto e integrato in genere sostituisce una rete di corrispondenza contenente fino a cinque condensatori, resistori e induttori, oltre a due dispositivi di protezione discreti, con un ingombro molto ridotto. La protezione ESD fornita è conforme alla norma IEC 61000-4-2 ($C = 150$ pF, $R = 330$ Ω) e supera il livello 4: 8 kV per scarica del contatto e 15 kV per scarica in aria. Il dispositivo resiste anche a una tensione di impulso di 2kV (conformità MIL STD 883C ($C = 100$ pF, $R = 1.5$ k Ω)). Il BPF8089-01SC6 è adatto per l'uso in ricevitori satellitari portatili per costellazioni GPS, Galileo, GLONASS, BeiDou e QZSS. Il BPF8089-01SC6 è contenuto in un package SOT23-6L.



www.st.com

Nuovi componenti per Macro BTS da Renesas

Quattro nuovi dispositivi arricchiscono l'offerta di circuiti integrati Renesas per le stazioni base ricetrasmittenti (BTS) Macro della rete cellulare. Si tratta degli amplificatori a guadagno variabile (VGA) F4482/1 TX a quattro canali e della famiglia F011x di amplificatori a basso rumore (LNA) a due canali.

Il nuovo set di dispositivi include anche l'amplificatore driver RF F1471 ad alta potenza con P1dB di oltre 1/2W e lo switch RF F2934 per il percorso di feedback della predistorsione digitale (DPD). In nuovi dispositivi vanno incontro alle esigenze dei progettisti che devono affrontare la transizione verso il 5G.

I nuovi dispositivi possono essere combinati con l'amplificatore RF ad alto guadagno F1490 e l'F0443 RX VGA altamente integrato per implementare una completa catena del segnale RF. Il portafoglio di soluzioni per Macro BTS include il VGA TX quad-path F4482/1 altamente integrato, con Balun per gamma di frequenza RF da 400 MHz a 2.800 MHz, filtro passa basso, amplificatore e attenuatori digitali a gradino integrati su singolo chip.

Il dispositivo F4482/1 beneficia della tecnologia Renesas Zero Distortion, per una migliore qualità del servizio con estesa gamma dinamica, e della tecnologia Glitch-Free per proteggere l'assemblaggio PA e semplificare l'implementazione DPD. Il secondo dispositivo è l'LNA di primo stadio F011x a doppio canale (LNA bilanciato o LNA a due canali) caratterizzato da gamma di frequenza RF da 650 MHz a 2.700 MHz, bassa figura di rumore di 0,55 dB a 2.600 MHz e bassa perdita di ritorno (ingresso -27 dB, uscita -23 dB).

L'amplificatore driver RF F1471 ad alta linearità, ha una gamma di frequenza RF da 400 MHz a 4.200 MHz con guadagno di 17dB, OIP3 fino a + 38dBm e OP1dB di 28,5dBm a 2.600 MHz. Prestazioni OIP3 e polarizzazione CC sono regolabili. Infine lo switch RF SP2T F2934 ha una gamma di frequenza RF da 50 MHz a 6.000 MHz, isolamento di 70dB a 1 GHz e 2 GHz, 74dB a 3 GHz e 67dB a 4 GHz e sfrutta la tecnologia Renesas Kz a impedenza costante per mantenere il VSWR durante la transizione di commutazione.

www.renesas.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.it/abbonamenti

10 NUMERI
A SOLI

€39

ANZICHÉ € 60,00

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

10
NUMERI
A SOLI

€45

ANZICHÉ € 60,00

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



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**



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.

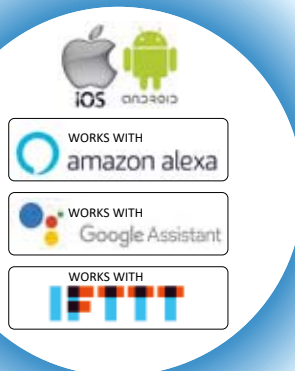
Presa smart Wi-Fi

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.



€21,90

Cod. FR779



LoRa Edge LR1110 STARTER KIT



di PIER CALDERAN

Scopriamo LoRa Edge LR1110 di Semtech e il suo kit di sviluppo finalizzato alla produzione di dispositivi che coniugano le tecnologie LoRa, WiFi e GNSS per la geolocalizzazione.

Semtech, storico produttore di soluzioni di connettività LoRa, ha recentemente sviluppato e reso disponibile la propria piattaforma LoRa Edge, nata come soluzione di geolocalizzazione per le applicazioni e i dispositivi IoT. La piattaforma, basata sul nuovissimo chip LR1110, nasce per lo sviluppo di applicazioni e dispositivi per geolocalizzazione in ambito IoT e integra un ricetrasmittitore (transceiver) LoRa con la tecnologia wireless WiFi e quella di scansione GNSS, più una serie di servizi di geolocalizzazione basati su cloud, creando un'architettura unica che offre un ottimo equilibrio tra precisione della posizione rilevata e ridotto consumo energetico. In queste pagine presenteremo la piattaforma di sviluppo integrata partendo dall'analisi e descrizione di quello che ne è il cuore, vale a dire l'integrato LR1110 concepito da Semtech per le applicazioni IoT e la connessione RF.



LoRa Edge™ LR1110



Fig. 1
Il kit di
valutazione
LR1110 di
Semtech.

L'INTEGRATO LR1110

Il primo chip LoRa Edge creato per la geolocalizzazione da rete satellitare GNSS (Global Navigation Satellite System, ossia la combinazione di GPS e altri sistemi di ausilio e localizzazione) si chiama LR1110 ed è ampiamente documentato e supportato sul sito ufficiale Semtech, <https://www.semtech.com/products/wireless-rf/lora-transceivers/lr1110>. Le sue caratteristiche principali sono:

- ricetrasmittitore RF Half-Duplex LoRa/(G) FSK ad alta sensibilità a bassa potenza;
- bande di frequenza ISM nella gamma 150 ÷ 960 MHz;

- Low Noise Figure RX a basso rumore per una maggiore sensibilità LoRa/(G) FSK;
- percorso PA ad alta potenza +22 dBm;
- percorso PA ad alta efficienza +15 dBm;
- selettore di alimentazione del regolatore PA integrato per semplificare la doppia alimentazione +15/+22 dBm;
- in grado di supportare BOM multi-regione in tutto il mondo, il circuito si adatta alla rete di corrispondenza per soddisfare i limiti normativi;
- pienamente compatibile con la famiglia SX1261/2/8 e lo standard LoRaWAN;
- scansione a bassa potenza GNSS (GPS/BeiDou);
- scansione passiva WiFi a bassissimo consumo 802.11b/g/n;
- gamma di sintesi di frequenza continua 150-2700 MHz;
- ADC RX a larghezza di banda elevata (fino a 24 MHz DSB);
- elaborazione digitale in banda base;
- motore crittografico;
- package QFN32 5x5 mm.

KIT DI VALUTAZIONE LR1110

Per questo prestante e versatile componente, Semtech ha reso disponibile il kit di valutazione LoRa Edge, configurato per la banda di frequenza europea 868 MHz. Il kit basato su LR1110 contiene due diversi moduli RF:

- modulo RF singolo per antenna GNSS passiva;
- modulo RF singolo per antenna GNSS attiva.

Il kit LR1110 è costituito da una scheda con tre connettori SMA: uno per l'antenna Wi-Fi, uno per l'antenna GNSS (Global Navigation Satellite System) e uno per l'antenna LoRa (**Fig. 1**). Oltre alle funzionalità di geolocalizzazione Wi-Fi e GNSS, supporta le modulazioni LoRa e (G)FSK ed è completamente compatibile con le generazioni precedenti di LoRa.

È in grado di trasmettere fino a +22 dBm sul PA ad alta potenza o fino a +15 dBm sul PA a bassa potenza e supporta un funzionamento continuo a bassa potenza nelle bande ISM 150 MHz – 960 MHz.

L'EVK viene dato con un'applicazione demo che consente di dimostrare le capacità di geolocalizzazione dell'LR1110.

LA DEMO DI GEOLOCALIZZAZIONE LR1110

La scheda LR1110 è montata come fosse uno shield su una scheda di prototipazione STM32 Nucleo (**Fig. 2**) che gestisce il software demo e il

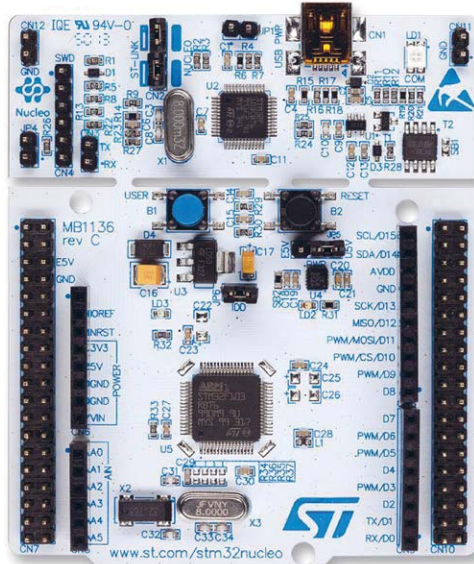


Fig. 2
La scheda STM32
Nucleo.

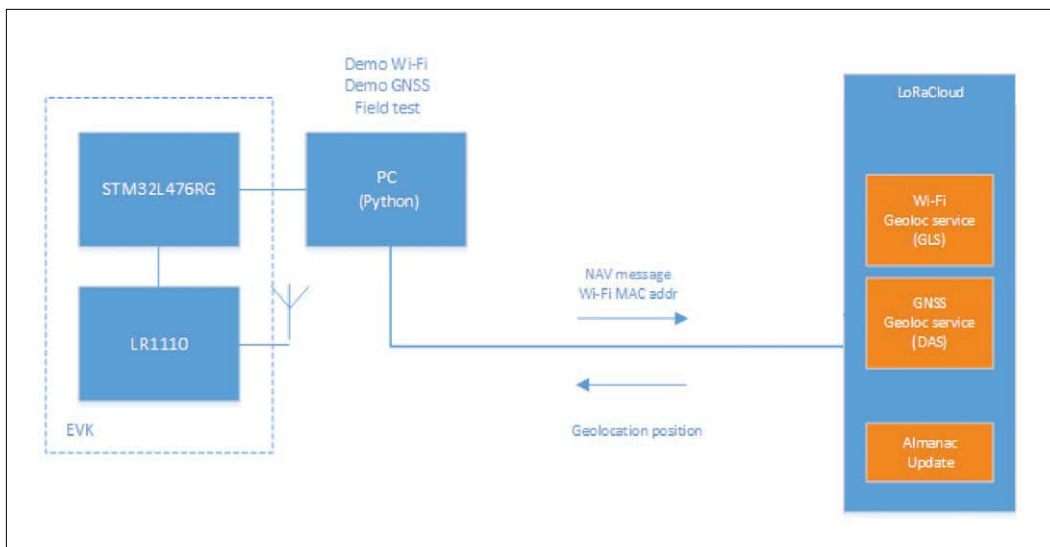


Fig. 3
Architettura
della demo di
geolocalizza-
zione LR1110.

display touch TFT, montato a sua volta sopra la scheda LR1110.

La demo di geolocalizzazione LR1110 viene eseguita sull'hardware EVK LR1110, con il display TFT collegato. Richiede un file eseguibile caricato nella scheda STM32 Nucleo. Nel caso non fosse precaricato bisogna scaricarlo dal GitHub (vedi più avanti). Le antenne GNSS e WiFi devono essere entrambe collegate agli appositi connettori SMA. La demo si concentra sulla dimostrazione della geolocalizzazione GNSS e WiFi. Il collegamento LoRa, per il momento, non viene utilizzato, quindi l'antenna LoRa non è necessaria.

La geolocalizzazione richiede che il server calcoli la geolocalizzazione del dispositivo in base ai risultati della scansione WiFi o GNSS. Viene inoltre implementata una geocodifica inversa, al fine di visualizzare l'indirizzo della geolocalizzazione calcolata sul display del kit di valutazione.

L'architettura demo complessiva della geolocalizzazione LR1110 è schematizzata nel diagramma a blocchi proposto nella **Fig. 3**.

Come si può vedere nella figura, la comunicazione con il server di geolocalizzazione è assicurata da un'applicazione host Python in esecuzione su un computer Windows o Linux. La comunicazione tra il kit di valutazione LR1110 e il computer è una comunicazione seriale tramite USB.

Come vedremo, la connettività con i server di geolocalizzazione può essere abilitata anche grazie a un link LoRa o LoRaWAN nell'applicazione finale di un Network Server.

Un output KML (Keyhole Markup Language) consente di visualizzare i risultati della geolocalizzazione su una mappa, sia per la geolocalizzazione WiFi che GNSS.

Il pacchetto software LR1110 EVK e il file eseguibile possono essere scaricati dalla pagina Semtech LR1110, al link seguente: www.semtech.com/products/wireless-rf/lora-transceivers/lr1110 e facendo clic sul link "LR1110 Geolocation Demo Software Package". Si aprirà il repository [https://github.com/Lora-net](https://github.com/Lora-net/lr1110_evk_demo_app) da cui scaricare il file https://github.com/Lora-net/lr1110_evk_demo_app

Il contenuto dell'archivio è il seguente:

- **lr1110_evk_vx.y.z.bin**: è il file eseguibile binario che dev'essere caricato nella scheda Nucleo.
- **lr1110-evk_vx.y.z.tar.gz**: è il pacchetto di installazione dell'applicazione host Python.
- Una volta installato, il pacchetto dell'applicazione host contiene le seguenti applicazioni:
- **AlmanacUpdate**: aggiorna i dati Almanac.
- **FieldTestPost**: invia i risultati del test al server di geolocalizzazione per la post-elaborazione.
- **Lr1110Demo**: demo di geolocalizzazione.
- **Lr1110FieldTest**: applicazione di test.

```

Amministratore: cmd
C:\Users\User>Lr1110Demo --version
2.1.0

```

Fig. 4
La versione di
LR1110Demo.



Fig. 5
L'output di
UsbConnection-
Check.

```

Amministratore: cmd - UsbConnectionCheck
Got message from embedded. Sending question...
Got response from embedded. Communication ok.
Error on USB line. Communication KO!
Got message from embedded. Sending question...
Got message from embedded. Sending question...
Got response from embedded. Communication ok.
Error on USB line. Communication KO!
Got message from embedded. Sending question...
Got message from embedded. Sending question...
Got response from embedded. Communication ok.
Error on USB line. Communication KO!

```

- **UsbConnectionCheck**: utility per verificare la connessione USB tra l'applicazione host Python e l'EVK.

Per motivi di spazio, ci limiteremo a spiegare l'applicazione Lr1110Demo e l'utility UsbConnectionCheck.

INSTALLAZIONE DELL'HOST PYTHON

Per utilizzare l'applicazione Lr1110Demo, è necessario installare un host Python. Accertarsi di poter eseguire Python dalla riga di comando, ovvero che sia nel path delle variabili d'ambiente. Bisogna avere installato almeno una versione di Python 3.5 o superiore. Assicurarsi anche di poter eseguire Pip dalla riga di comando.

A questo punto, il processo di installazione qui

descritto presuppone che abbiate scaricato il pacchetto di distribuzione del software host dal GitHub.

In genere il pacchetto di distribuzione viene denominato "lr1110evk- <version_number> .tar.gz". Navigare nella cartella che contiene il file di distribuzione di origine e digitare:

```

cd <cartella del repository clonato>/host
python ./setup.py sdist # compila l'archivio di
distribuzione dei sorgenti
python -m pip install dist/<BUILT_SOURCE_DIST>.
tar.gz # Installa la distribuzione dei sorgenti creata
in precedenza.

```

Il pacchetto di distribuzione in realtà installa diversi programmi. Per verificare l'installazione, si può controllare che, per esempio, l'applicazione Lr1110Demo sia installata.

Per fare ciò, digitare nel terminale:

Lr1110Demo – version

Deve restituire un numero di versione, per esempio 2.1.0, ovvero la versione che stiamo usando per questo articolo (**Fig. 4**).

I software installati devono comunicare con la scheda, per cui, supponendo che il comando *Lr1110Demo --version* abbia restituito un numero di versione corretto, si può procedere con la verifica USB, lanciando dalla riga di comando l'utility UsbConnectionCheck. Dovrebbe apparire una finestra simile a quella proposta nella **Fig. 5**.

ESECUZIONE DELLA DEMO DI GEOLOCALIZZAZIONE LR1110

Una volta controllato il collegamento USB con l'EVK, si vedrà apparire nel display TFT qualcosa di simile alla **Fig. 6**. Il pulsante "START" della schermata di avvio apre il menu principale con due pulsanti:

- RADIO TEST MODE, un'applicazione per testare il modem LoRa e GFSK
- DEMONSTRATIONS, due demo di geolocalizzazione (**Fig. 7**).

Per ora ci concentreremo sulla demo di geolocalizzazione LR1110, che esegue tre test:

- WiFi Scan, per la geolocalizzazione WiFi;
- GNSS autonomous, per la geolocalizzazione esterna con GNSS autonomo;
- GNSS assisted, per la geolocalizzazione esterna con GNSS assistito.

Fig. 6
La schermata di
avvio della demo
LR1110 2.1.0.



Si noti che le due modalità GNSS richiedono che l'applicazione host Python sia in esecuzione per essere attive. Il pulsante GO! di GNSS autonomous e GNSS assisted sono disattivati e devono essere accesi in blu. I due pulsanti GO! saranno spenti (grigi) fintantoché l'applicazione host Python non si sarà avviata. In qualsiasi menu della demo, il pulsante BACK consente di tornare al menu precedente.

SCANSIONE PASSIVA WI-FI

Nella schermata principale dell'applicazione WiFi Scan, premendo il pulsante CONFIG si può configurare il tipo e i canali WiFi, oppure eseguire una scansione WiFi passiva utilizzando la configurazione predefinita.

Apparirà la schermata con il messaggio PRESS START TO BEGIN e, premendo il pulsante START, la scansione si avvierà fino a quando non viene rilevato un indirizzo MAC.

Al termine del processo di scansione WiFi, sullo schermo viene visualizzato il numero di indirizzi MAC trovati sui canali configurati e, premendo SEND, apparirà la geodecodifica inversa con l'indirizzo della posizione (**Fig. 8**).

Il pulsante START attiva una nuova scansione WiFi, mentre il pulsante SEND invia i risultati della scansione all'applicazione host Python in esecuzione sul PC. Il software Python crea e invia una richiesta al server di geolocalizzazione e attende la sua risposta. La risposta dal server di geolocalizzazione viene quindi propagata all'EVK. La demo di geolocalizzazione mostra la posizione risolta del dispositivo, convertita dalle coordinate GPS calcolate dal risolutore mediante geocodifica inversa.



Fig. 7
Il menu principale (a sinistra) e la schermata DEMONSTRATIONS.

Il pulsante RESULT mostra gli indirizzi MAC rilevati per canale e tipo WiFi, con il valore RSSI corrispondente.

Le frecce sullo schermo consentono di navigare tra le diverse pagine dei risultati per analizzare i dati di geolocalizzazione del dispositivo. Nel nostro esempio, le coordinate corrispondono alla location di chi scrive.

Attenzione: per poter rilevare la posizione mediante il servizio di geolocalizzazione WiFi sono necessari almeno due MAC Address, altrimenti sullo schermo apparirà un messaggio di errore di questo tipo:

Response:
{“result”: null, “warnings”: [“WiFi location estimation requires at least two entries in \”wifiAccessPoints\””]}

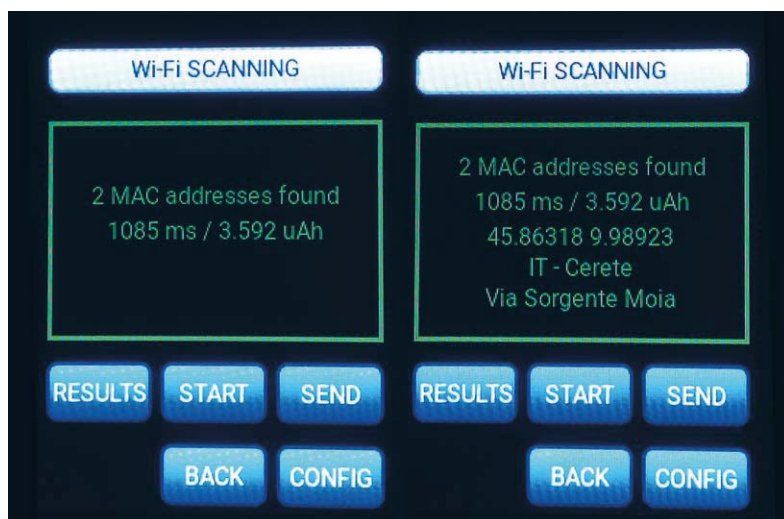


Fig. 8
Il risultato della scansione WiFi (a sinistra) e della geodecodifica (a destra).

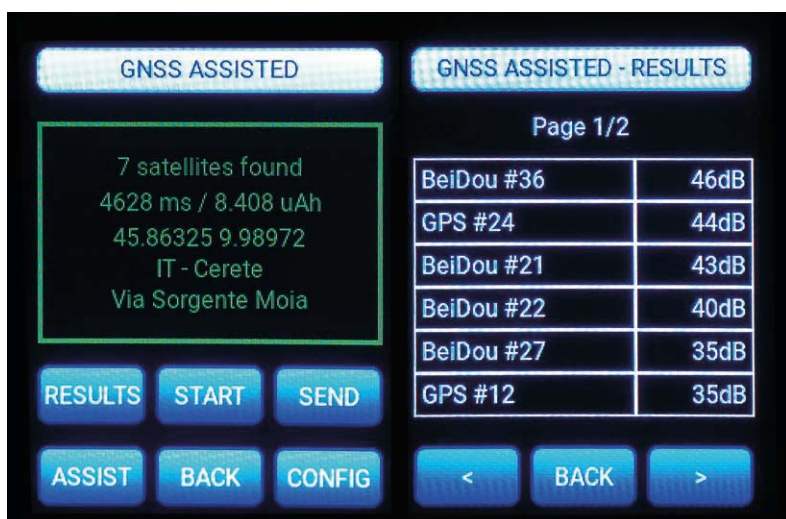


Fig. 9
I risultati della
geolocalizzazione
GNSS assisted.

Se tutto è configurato correttamente e la geolocalizzazione ha successo, apparirà un messaggio di questo tipo:

```
Result from server:
- Coordinate: Coordinate<lat:45.863226, long:9.989113, alt:0.0>
- Reverse geo-coding: Via Sorgente Moia,Cerete,IT
```

Prima di passare all'argomento successivo, diamo alcune informazioni sui due servizi di Semtech offerti per i dispositivi e le applicazioni LoRa Cloud (chiamati con l'acronimo DAS che sta per Device and Application Services) e sui servizi per la geolocalizzazione LoRa Cloud (chiamati GLS che invece sta per Geolocation LoRa Cloud Service).

SERVIZI PER DISPOSITIVI E APPLICAZIONI LORA CLOUD

DAS sta per Device and Application Services e si tratta di servizi per dispositivi e applicazioni LoRa Cloud che comprendono una serie di funzioni di gestione per i dispositivi basati su LoRaWAN. Queste funzionalità includono la telemetria completa del dispositivo, la configurazione di dispositivi e applicazioni, la sincronizzazione dell'orologio e i servizi avanzati di trasporto dei dati, con robustezza configurabile contro la perdita di pacchetti e la frammentazione dei dati. Lo scopo di questi servizi è quello di semplificare il processo di sviluppo di soluzioni endpoint gestite e di rendere le tecnologie LoRaWAN più accessibili agli sviluppatori. Come vedremo più avanti, si potrà accedere ai servizi DAS con un account gratuito sul portale LoRa Cloud di Semtech.

SERVIZI PER LA GEOLOCALIZZAZIONE LORA CLOUD

GLS sta per Geolocation LoRa Cloud Service e comprende servizi per la geolocalizzazione sotto forma di API cloud che possono essere facilmente integrate in una rete LoRaWAN o in una applicazione server, per consentire la stima della posizione di qualsiasi dispositivo basato su LoRa su una rete LoRaWAN. Quando i gateway ricevono un pacchetto di dati trasmesso da dispositivi basati su LoRa, generano anche metadati come la potenza del segnale ricevuto, il rapporto segnale/rumore e l'ora di arrivo. Quando questi metadati dai gateway vengono assemblati in una query e trasmessi tramite una richiesta post HTTP, il servizio è in grado di calcolare la posizione del dispositivo.

APPLICAZIONE HOST PYTHON

L'applicazione host Python LR1110Demo consente la connettività con il server di geolocalizzazione per inviare i risultati della scansione e recuperare la geolocalizzazione WiFi. Fornisce inoltre le informazioni di assistenza del dispositivo LR1110, come posizione e ora approssimative, in modalità assistita GNSS. Per eseguire l'applicazione host Python, basta digitare il comando e gli argomenti seguenti in una finestra del terminale:

```
Lr1110Demo -d <COM_PORT> -r <COORDINATES_ASSISTED_LR1110> <COORDINATES_EXACT> <TOKEN_WIFI_GLS> <TOKEN_GNSS_DAS>
```

dove:

- d <COM_PORT>: è il numero di porta COM della scheda Nucleo. COMx su Windows, /dev/ttyAC-Mx su Linux.
- r: parametro per la geocodifica inversa (opzionale). Visualizza l'indirizzo risolto della posizione.
- <COORDINATES_ASSISTED_LR1110>: posizione approssimativa entro 150 km per la scansione assistita GNSS.
Il formato del comando è <DECIMAL_LATITUDE>, <DECIMAL_LONGITUDE>, <DECIMAL_ALTITUDE> senza spazio tra le virgole.
Per esempio, le coordinate di Bergamo sono:
latitudine 45° 41' 56" 04 N = 45,415604
longitudine 09° 40' 12" 00 E = 9,401200
altitudine = 0 metri (sul livello del mare)
Per cui si scriverà: 45.415604, 9.401200,0
- <COORDINATES_EXACT>: coordinate reali per consentire al risolutore di calcolare e visualizzare l'errore di geolocalizzazione. Il formato è lo stesso di prima.

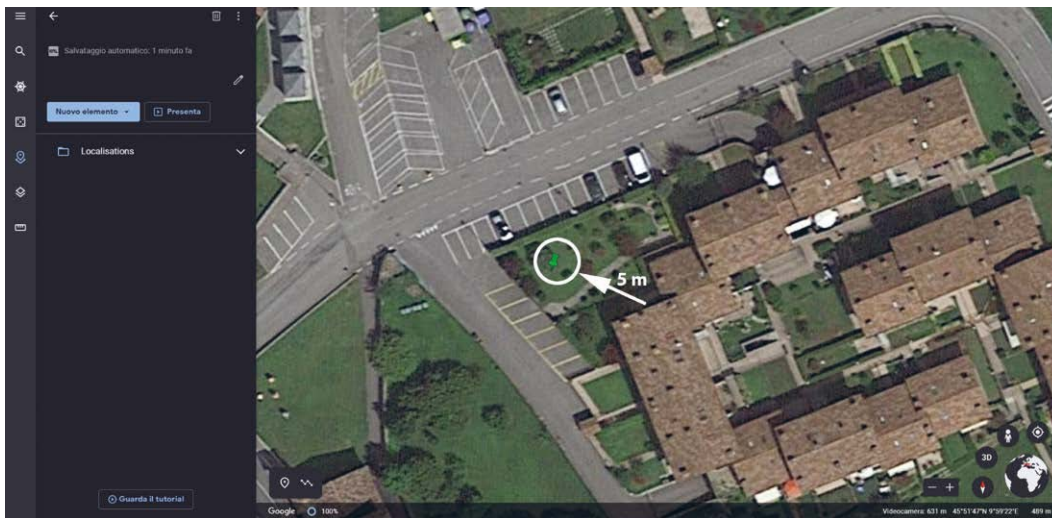


Fig. 10
La mappa
satellitare di
Google Earth.

- <TOKEN_WIFI_GLS>: token di intestazione HTTP dal LoRa Cloud GLS, per autenticare le richieste. Può essere ottenuto andando alla pagina https://www.loracloud.com/portal/geolocation/token_management (vedi più avanti).
- <TOKEN_GNSS_DAS>: token di intestazione HTTP DAS, per autenticare le richieste. Può essere ottenuto andando alla pagina https://www.loracloud.com/portal/device_management/tokens (vedi più avanti).

Si tenga presente che si può usare solo un token alla volta, quindi un esempio di riga di comando per attivare correttamente l'applicazione host Python per il server DAS è la seguente, supponendo che la porta COM utilizzata sia la 40:

```
Lr1110Demo -d 40 -r 45.415604, 9.401200,0 45.415604,
9.401200,0 AQEA7xmT6kz8edrLzDfUXsjcoBfQJKPm/yP2Am/
LY5U0/s3UwiuT
```

Per il server GLS è la seguente:

```
Lr1110Demo -d 40 -r 45.415604, 9.401200,0 45.415604,
9.401200,0 AQEAJy+kgpO+9eNaEgPyvakZMN1XiDQVh0hjJJ/
SOFQlmYnRiB1
```

Ovviamente i token DAS e GLS qui sopra sono solo di esempio e vanno recuperati dal proprio account LoRa Cloud, come vedremo fra poco.

SCANSIONE GNSS

Come già detto, sono implementate due modalità di scansione GNSS. Di seguito descritte

- La modalità autonoma GNSS (GNSS autono-

mous) ricerca e decodifica il segnale forte dai satelliti. Questa modalità non richiede dati di assistenza. Può essere utilizzata per il rilevamento interno/esterno. Richiede la connettività con l'applicazione host Python per fornire il tempo di acquisizione corrente con una precisione di 1 secondo.

- La modalità assistita GNSS (GNSS assisted) consente la ricerca di tutti i satelliti raggiungibili. Richiede la connettività con l'applicazione host Python per fornire informazioni di assistenza, ovvero il tempo di acquisizione corrente con una precisione di 1 secondo e una posizione approssimativa con la precisione di 150 km. Richiede inoltre la connettività con il server di geolocalizzazione per il calcolo della posizione del dispositivo. I dati dell'Almanac aggiornati consentono di ottenere un tempo di acquisizione minimo.

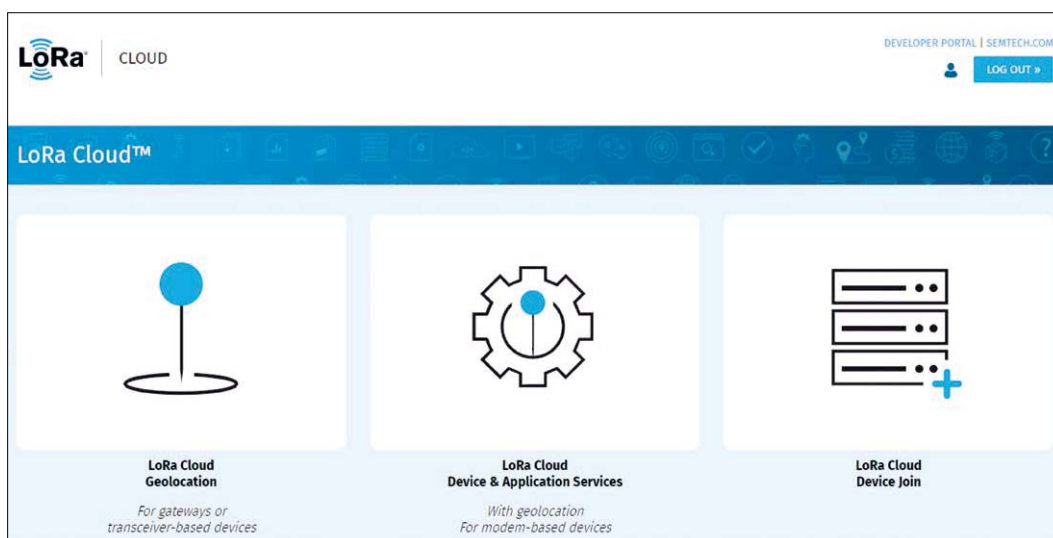
Processo di aggiornamento Almanac

Gli Almanac sono un insieme di parametri utilizzati per stimare la posizione dei satelliti visibili, nel momento e nel luogo della scansione. I dati dell'Almanac non aggiornati implicano un aumento del tempo di acquisizione e dell'energia spesa per la scansione. I dati di LR1110 Almanac devono essere aggiornati ogni tre mesi, sebbene si consiglia di aggiornarli ogni mese per ottenere prestazioni di scansione ottimali.

Durante l'esecuzione di una scansione GNSS, viene visualizzato un messaggio di avviso sulla schermata EVK se i dati di Almanac risalgono a prima di 31 giorni.

Questo messaggio di avviso non impedirà l'esecuzione della scansione GNSS se i dati Almanac

Fig. 11
La home page del
portale LoRa Cloud.



hanno meno di 3 mesi, ma la scansione GNSS non verrà eseguita se i dati Almanac hanno più di 3 mesi. Gli Almanac sono distribuiti mensilmente dal server Semtech LoRaCloud. Fare riferimento alla documentazione disponibile al link <https://www.loracloud.com/documentation>.

Visualizzare la geolocalizzazione GNSS

Il pulsante START attiva una nuova acquisizione GNSS, mentre il pulsante SEND invia i risultati della scansione al server di geolocalizzazione, tramite l'host Python, e visualizza la posizione del dispositivo grazie alla geocodifica inversa. Il LED sniffing sulla scheda è acceso durante il processo di scansione. Se il risultato è positivo, ovvero quando l'operazione di sniffing ha trovato almeno 4 o 5 satelliti validi, la richiesta al server darà una risposta di questo tipo:

Result from server:

```
- Coordinate: Coordinate<lat:45.86353, long:9.98933, alt:567.92>
- Reverse geo-coding: Via Sorgente Moia,Cerete,IT
```

Il pulsante RESULTS visualizza l'elenco dei satelliti rilevati, con il corrispondente nome e forza del segnale (**Fig. 9**).

Google Earth

Dato che viene prodotto un file KML ad ogni rilevazione, possiamo visualizzare la posizione del dispositivo dai satelliti tramite Google Earth (<https://earth.google.com>). Dalla home page di Google Earth basta creare un nuovo progetto e importare dal disco locale il file KML.

Il risultato della vista satellitare di Google Earth della nostra location è visibile nella **Fig. 10**. Per dovere di informazione, il puntaspilli nella foto indica il punto del dispositivo LR1110 a circa 5 metri dal terrazzo in cui è posizionata l'antenna.

LORA CLOUD

Semtech ha creato un suo portale che ha chiamato LoRa Cloud (<https://www.loracloud.com>) che offre i già citati servizi GLS e DAS.

Per accedere al portale è necessario creare un account gratuito e scegliere dalla home page un servizio fra quelli illustrati nella **Fig. 11**. Le tre icone permettono di accedere ai servizi seguenti:

- LoRa Cloud Geolocation: For gateways for transceiver-based devices
- LoRa Cloud Device & Application Services: With geolocation For modem-based devices
- LoRa Cloud: Device Join

LoRa Cloud Geolocation

Senza dilungarci troppo, diciamo solo che, aprendo questa pagina si possono gestire i token GLS, ovvero quelli che dobbiamo usare nella riga di comando dell'host Python, vista prima.

Facendo clic su Manage Tokens del menu laterale si aprirà una pagina (**Fig. 12**) in cui sono visibili i token GLS. È sufficiente copiare il token primario (Primary Token) con il tasto Copy.

Fatto questo, si può tornare alla finestra del terminale ed incollare il token nella riga di comando, che ricordiamo essere questa:

```
Lr1110Demo -d 40 -r 45.415604, 9.401200,0
45.415604, 9.401200,0
```

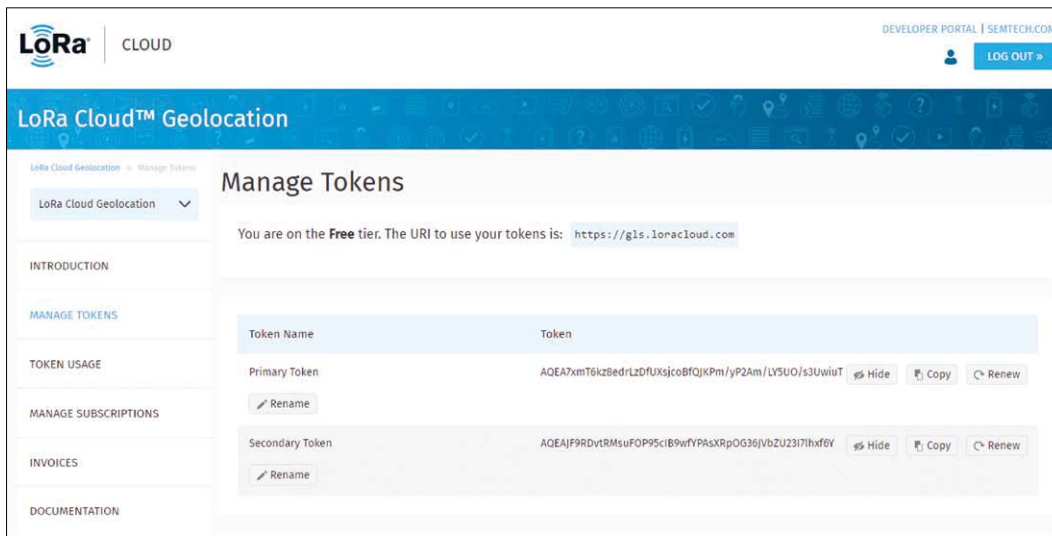


Fig. 12
La pagina
Manage
Tokens GLS.

AQEA7xmT6kz8edrLzDfUXsjcoBfQJKPm/yP2Am/LY5UO/s3UwiuT

Le coordinate devono essere approssimativamente quelle della vostra location. Una volta avviato l'host Python è possibile eseguire la demo WiFi Scan, come abbiamo visto in precedenza.

LoRa Cloud Device & Application Services

In questa pagina si possono gestire i token DAS, da usare nella riga di comando dell'host Python. Facendo clic su Manage Tokens, si aprirà una pagina (Fig. 13) in cui sono visibili i token DAS. È sufficiente copiare il token primario (Primary Token) con il tasto Copy, quindi si torna alla fine-

stra del terminale e si incolla il token nella riga di comando:

```
Lr1110Demo -d 40 -r 45.415604, 9.401200,0 45.415604, 9.401200,0
AQEAJy+kgp0+9eNaEgPyvakZMN1XiIDQVh0hjJJ/SOFQLmYnRIb1
```

Le coordinate riportate devono essere approssimativamente quelle della vostra location (in modalità assisted è sufficiente un'approssimazione di 150 km). Una volta avviato l'host Python si può eseguire la demo GNSS autonomous o GNSS assisted, come abbiamo visto prima.

LoRa Cloud Device Join

Questo servizio consente di effettuare il claim (possiamo considerarlo come l'attestazione) dei

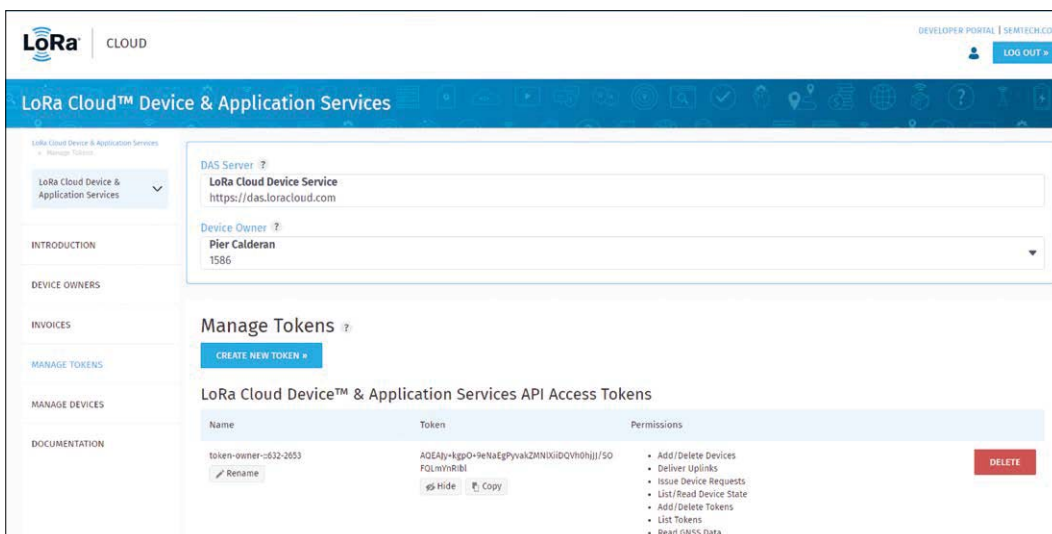
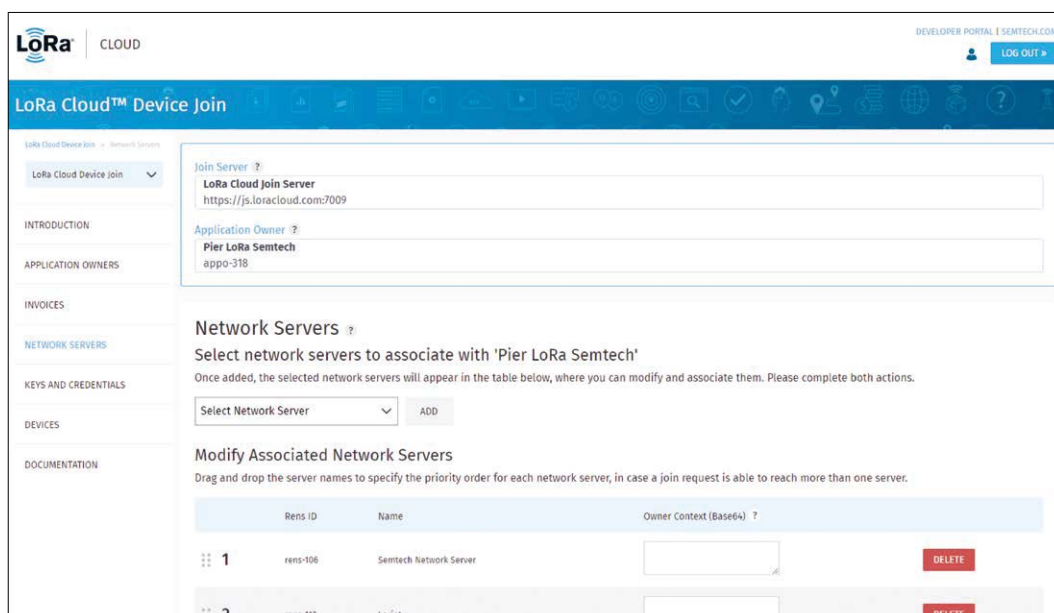


Fig. 13
La pagina
Manage
Tokens DAS.

Fig. 14
La pagina Network
Server.



dispositivi basati su modem LoRaWAN e collegati a un Network Server fra quelli disponibili, elencati di seguito:

- Actility ThingPark Wireless DEV1;
- The Things Industries Cloud Hosted (staging);
- The Things Industries Cloud Hosted (staging);
- Syniverse Hub;
- Semtech Network Server;
- Orbiwise;
- Senet;
- Loriot.

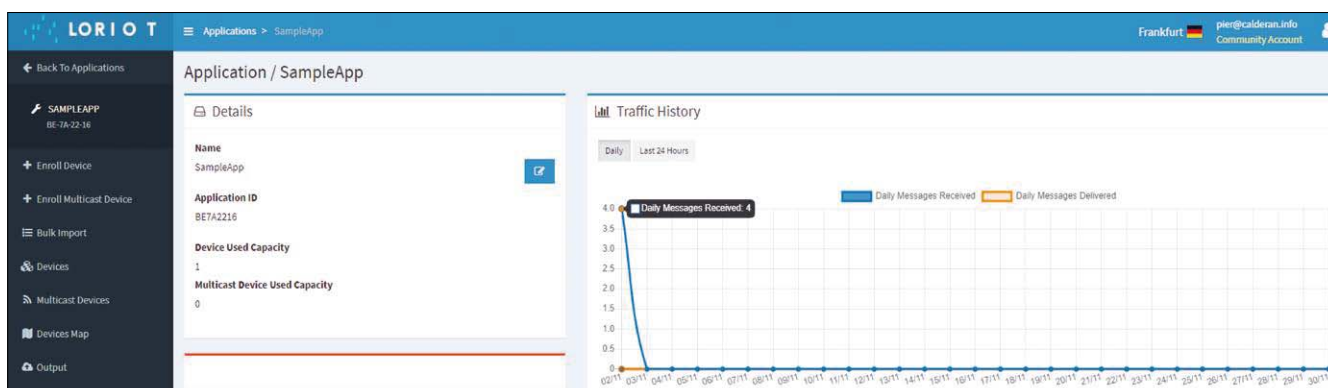
La procedura di registrazione del device LR1110 è comune a tutti i dispositivi LoRaWAN. Aprendo la pagina LoRa Cloud Device Join, è sufficiente scegliere la voce Network Server dal menu e, dalla pagina che si apre, sceglierne uno dell'elenco proposto (**Fig. 14**).

Per il nostro esperimento abbiamo creato un account gratuito sul cloud Loriot (<https://www.loriot.io>). Dopo l'accesso al cloud ci siamo collegati a un server europeo fra quelli proposti dal portale (Frankfurt-De, per l'esattezza). Dalla pagina Gateway di Loriot abbiamo quindi creato un gateway con un concentratore iC880A della Wireless Solutions ed una scheda Raspberry Pi, oltre che con un'applicazione (**Fig. 15**) basata su di un dispositivo con le specifiche LoRaWAN dell'EVK LR1110.

I parametri da impostare per il dispositivo su Loriot li abbiamo ricavati dalla schermata EUI dell'EVK: DEV EUI e JOIN EUI.

Nella pagina Devices di LoRa Cloud (**Fig. 16**) abbiamo creato un dispositivo eseguendo il claim con i dati della schermata EUI dell'EVK, ossia CHIP EU, DEV EU e PIN. Per il software di gestio-

Fig. 15
La pagina
Application di
Loriot.



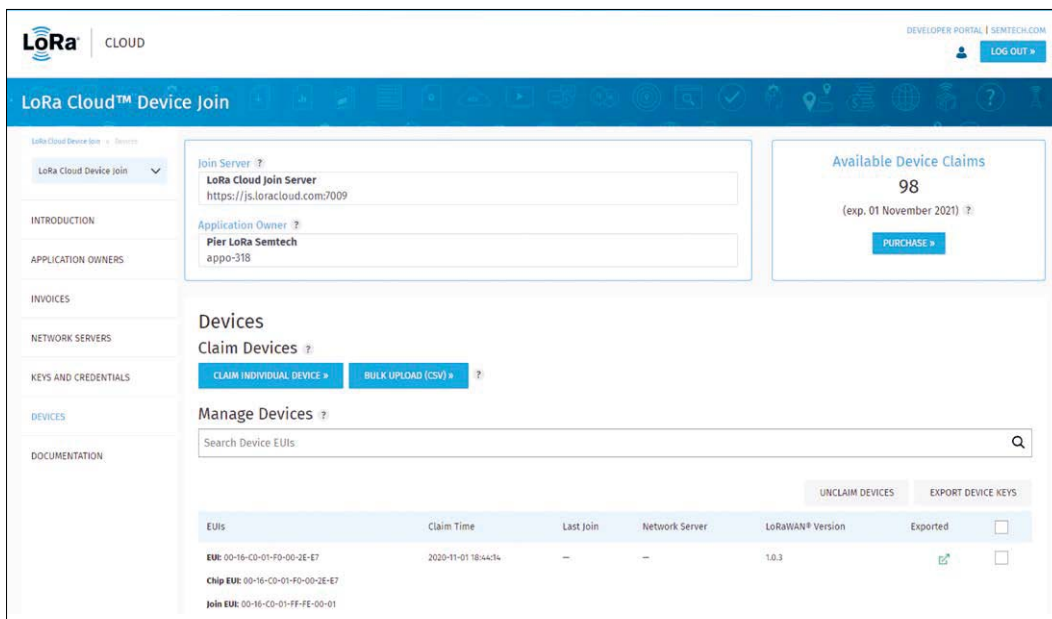


Fig. 16
La pagina
Devices di
LoRa Cloud.

ne, ovvero del Packet Forwarder di Lorient, basta seguire le istruzioni di installazione alla pagina Software di Lorient.

Qui ci sono anche le istruzioni specifiche per Raspberry Pi e iC880A. Una volta avviato il gateway su Raspberry Pi, si vedrà il messaggio connected nella pagina gateway di Lorient. A questo punto, avviando l'applicazione RADIO TEST MODES dall'EVK, basta mandare pacchetti LoRaWAN con le applicazioni disponibili (**Fig. 17**): TX Cont Wave, Packet Error Rate e Ping Pong.

CONCLUSIONI

Il kit di valutazione LR1110 è una valida proposta per gli sviluppatori che vogliono prendere la direzione della tecnologia LoRa e LoRaWAN nel campo della geolocalizzazione.

La scelta della scheda STM32 Nucleo come hardware per la gestione dell'EVK potrebbe essere molto limitante per tanti sviluppatori abituati a schede più "facili". Ci rammarichiamo del fatto che, a questo proposito, il supporto per lo sviluppo di applicazioni specifiche sulla scheda Nucleo è molto scarso, per non dire inesistente. Ma questo lo possiamo imputare a un "errore di gioventù" della community e del progetto STM Nucleo. ■



Cosa occorre?

Per chi volesse approfondire l'argomento LoRa e LoRaWAN consigliamo il libro disponibile presso Futura Elettronica "Capire e usare LoRa e LoRaWAN" (cod. S5597) disponibile a Euro 17,90. Per poter sperimentare con la tecnologia LoRa e seguire gli esempi riportati nel libro, è disponibile il kit EndDevice (cod. SC102) a Euro 25,00. 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>

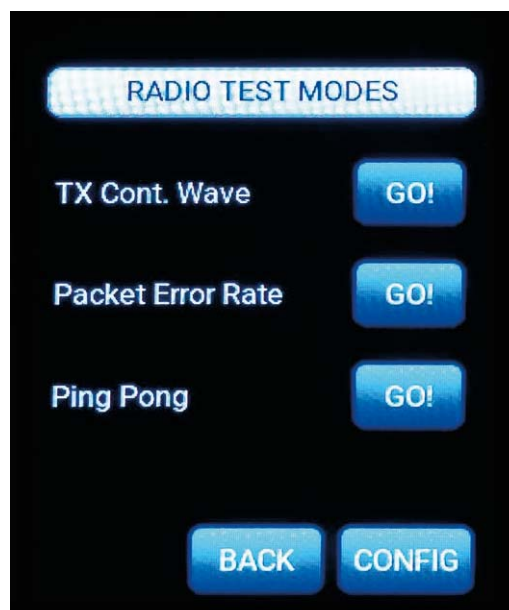


Fig. 17
La schermata
RADIO TEST
MODES.



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



BOX TRASPARENTE PER FISHINO UNO
cod. FISHINOUNOBOX
€ 4,90

cod. FISHINOUNO
€ 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 3,3V 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 (5Vdc e 3,3Vdc)
- Compatibile con shield e schede millefori
- Peso 37 g
- Dimensioni 101,5 x 53,5 x 15 mm

BOX TRASPARENTE PER FISHINO MEGA
cod. FISHINOMEGABOX
€ 5,90

Fishino MEGA



cod. FISHINOMEGA
€ 55,00

Fishino32



BOX TRASPARENTE PER FISHINO32
cod. FISHINO32BOX
€ 4,90

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

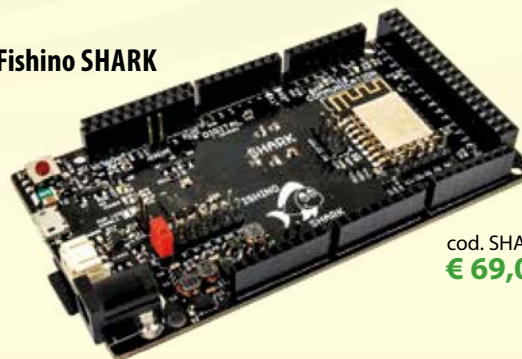
Fishino GUPPY



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



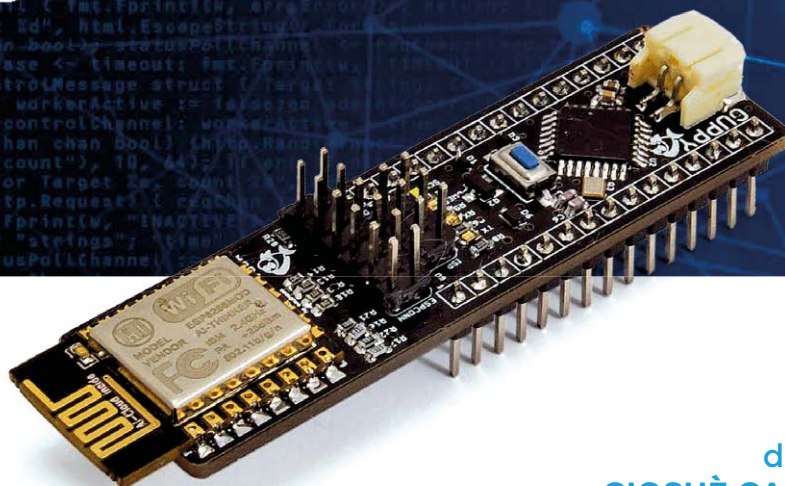
cod. PIRANHA
€ 49,00



- 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.

SMART SENSOR CONNESSO A Blynk

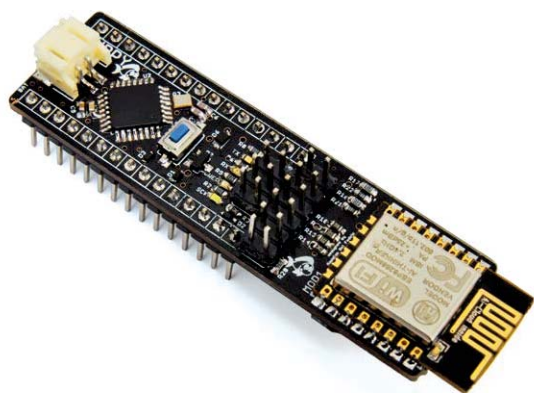


dell'Ing.
GIOSUÈ CALIANO

Sviluppiamo un'applicazione IoT connettendo una scheda Fishino Guppy, equipaggiata con sensori di temperatura e umidità, alla piattaforma Blynk, che rende disponibili potenti widget per connettere il mondo dei sensori all'Internet of Things.

Negli ultimi anni abbiamo assistito al proliferare di oggetti, all'interno delle nostre case, al lavoro e nella vita di tutti i giorni, resi "intelligenti" e connessi alla Rete. Sono passati più di 20 anni da quando è stata coniata l'espressione "Internet of Things" e l'IoT nasce proprio dall'idea di portare nel mondo digitale gli oggetti della nostra vita quotidiana. In questi anni le tecnologie IoT si sono moltiplicate e sviluppate, così come si sono profondamente evoluti i numerosi ambiti applicativi: casa intelligente e edifici intelligenti (Smart Building), telelettura dei contatori energia elettrica, gas e acqua (Smart Metering), Smart Factory (il cuore del sistema Industria 4.0) ed altro, con una rapidità incredibile ed un'espansione senza limiti. L'Internet of Things è un paradigma che non conosce, potenzialmente, confini applicativi: dall'autovettura che dialoga con l'infrastruttura stradale per prevenire incidenti, agli elettrodomestici di casa che si coordinano per ottimizzare

- 100% compatibile con Arduino Nano.
- Modulo WiFi a bordo.
- Connettore per microSD.
- Tripla alimentazione: batteria LiPo, porta USB e connettore di alimentazione esterno (6,5÷20 volt).
- Stadio di alimentazione switching ad elevata efficienza, con step-up per ottenere i 5 volt dalla batteria e step-down per ricavarli dall'ingresso esterno.
- Sezione 3,3 volt potenziata per gestire il modulo WiFi, la microSD ed eventuali ulteriori componenti esterni.
- Caricabatteria a bordo con indicazione dello stato di carica.



l'impegno di potenza, dagli impianti di produzione che scambiano dati con i manufatti per la gestione del loro ciclo di vita; dai dispositivi medicali che si localizzano nel presidio di un pronto soccorso, agli sci che inviano informazioni sullo stato della neve o sulla severità di una caduta. Se è vero che tutti gli oggetti possono diventare "intelligenti" connettendosi alla rete e scambiando informazioni, è altrettanto vero che questo processo non avviene in tutti gli ambiti con la stessa velocità: ciò dipende dall'esistenza di soluzioni tecnologiche consolidate, dagli equilibri competitivi in un determinato mercato e, in definitiva, dal bilancio tra il valore dell'informazione e il costo di creazione della rete di oggetti intelligenti.

In questo articolo vogliamo proporvi un'applicazione IoT riguardante gli "smart sensor", ossia sensori intelligenti in grado di rilevare le grandezze elettriche, fisiche o chimiche e di rielaborare le informazioni raccolte per poi trasmetterle sotto forma di segnale digitale all'esterno, per esempio ad uno smartphone. Il cellulare sarà in grado di gestirli, visualizzarli e in alcuni casi di prendere delle decisioni, ovvero di controllare il processo o semplicemente osservare l'andamento nel tempo delle variabili misurate. Per ottenere semplicemente queste funzioni useremo

una scheda piccola ma potente come la Fishino Guppy, connessa a due sensori, uno di temperatura (il classico DS18B20) e uno di umidità (DHT11). Ovviamente è solo un esempio, perché alla Fishino possono essere collegati sensori di svariata natura e, come vedremo, anche in numero elevato. In generale ogni sensore che può essere connesso ad una scheda Arduino-like può essere controllato da uno smartphone ad essa collegata. Quindi la descrizione che seguirà è relativa ad uno degli esempi possibili; i più intraprendenti tra voi potranno facilmente integrare i sensori che occorrono alla loro specifica applicazione, utilizzando le informazioni e gli esempi riportati nell'articolo.

La modalità di gestione della visualizzazione e controllo dei dati sullo smartphone è delegata alla piattaforma Blynk. Blynk (www.blynk.io), come ama definirsi, è una piattaforma IoT indipendente dall'hardware con app mobili "white label", cioè senza marchio, personalizzabile a piacimento, caratterizzata da un cloud privato, dalla gestione dei dispositivi (sono circa 400 le piattaforme hardware gestibili), con analisi dei dati e perfino un esempio di "machine learning".

COME FUNZIONA BLYNK

Blynk è stata progettata per l'Internet delle cose; può controllare l'hardware da remoto, visualizzare i dati dei sensori, può memorizzare i dati e fare molte altre cose interessanti.

Le componenti principali della piattaforma Blynk sono tre:

- le Blynk App, ossia le app che consentono di creare le interfacce per i vostri progetti utilizzando gli svariati widget che sono 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, per il massimo della riservatezza, qualora dovesse servire. E' open-source e potrebbe facilmente gestire migliaia di dispositivi e può anche essere lanciato su un 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.

Ora occorre uno sforzo di immaginazione: ogni volta che si preme un pulsante nell'app Blynk sul cellulare, il messaggio viaggia verso il Blynk Cloud, dove

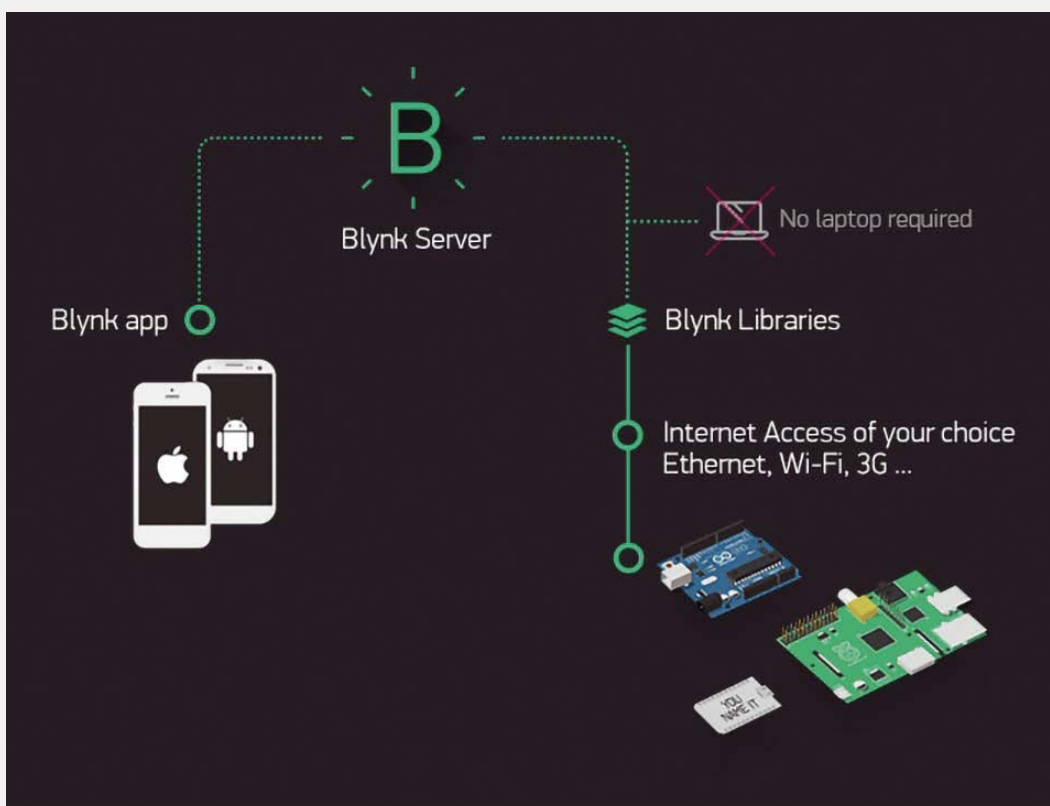


Fig. 1
Le
componenti
di base della
piattaforma
Blynk.

“magicamente” trova la sua strada verso l’hardware prescelto. Funziona nello stesso modo nella direzione opposta e tutto avviene praticamente istantaneamente (**Fig. 1**).

Per iniziare a lavorare con Blynk occorrono un hardware completo di interfaccia wireless e uno smartphone o tablet Android (o iOS).

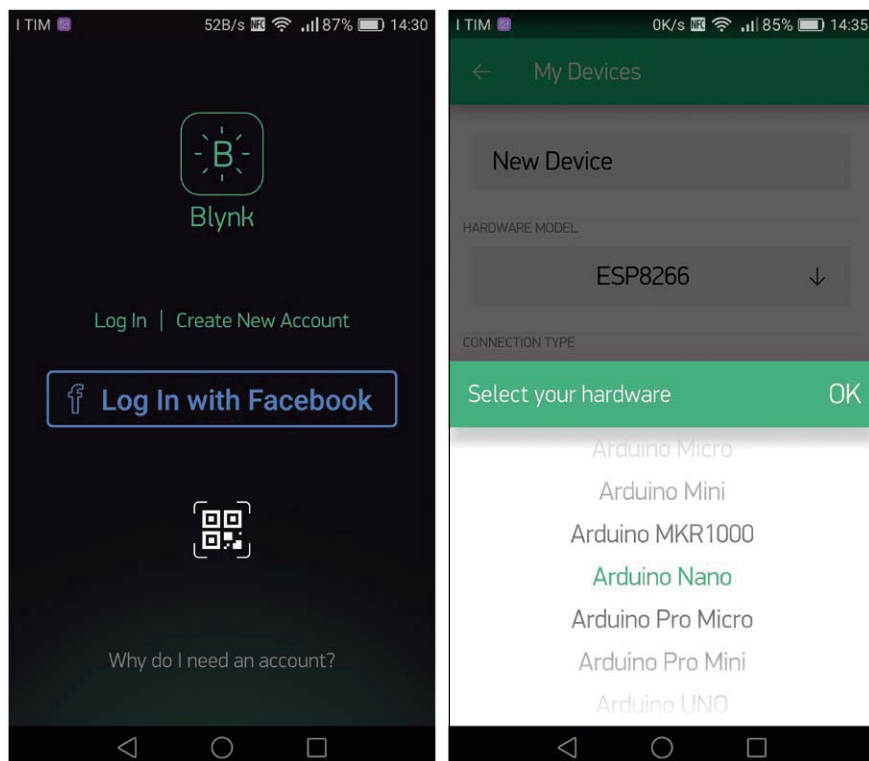
L’hardware può essere una scheda Fishino (o Arduino o quello che avete a disposizione). Blynk lavora su Internet; ciò significa che l’hardware scelto dovrebbe essere in grado di connettersi alla rete. Alcune delle schede, come Arduino Uno, avranno bisogno di uno shield Ethernet o WiFi per comunicare; altre, come quelle appartenenti alla gamma Fishino, sono già abilitate per Internet, avendo il WiFi a bordo. Ma anche se non si dispone di uno shield, è possibile collegarlo tramite USB al computer portatile o desktop (è un po’ più complicato per i neofiti, ma si può fare). Nello smartphone verrà eseguita l’app Blynk che è un generatore di interfaccia ben progettato che funziona sia su iOS che su Android. A questo punto siamo quasi pronti: occorre dapprima scaricare l’app Blynk per lo smartphone mediante i servizi classici (Google Play per Android e App Store per iOS).

L’app non è altro che un “costruttore di applicazioni”

per lo smartphone e permette di creare l’interfaccia, come poi vedremo. Occorre poi scaricare la libreria Blynk per Arduino (ora alla versione 0.6.1) dal sito <https://github.com/blynkkk/blynk-library/releases/tag/v0.6.1>, e installarla nell’IDE di Arduino, come al solito (se non sapete come si fa, leggete il sito <https://www.arduino.cc/en/guide/libraries>).

Dopo aver scaricato l’app Blynk, dovrete creare un nuovo account Blynk (**Fig. 2**); consigliamo di utilizzare un indirizzo di e-mail reale perché così le cose si semplificheranno. Avere un account è necessario per salvare i progetti e avere accesso ad essi da più dispositivi da qualsiasi luogo ed è anche una misura di sicurezza. È sempre possibile impostare il proprio private Blynk Server e avere il pieno controllo. Dopo aver effettuato correttamente l’accesso al proprio account, si può iniziare a creare un nuovo progetto. Selezionare quindi il modello di hardware da utilizzare. A questo punto viene reso disponibile il cosiddetto Token di autenticazione. Il Token di autenticazione è un identificatore univoco necessario per collegare l’hardware allo smartphone: è la targa dell’accoppiata app+hardware che stiamo per realizzare. Ogni nuovo progetto creato avrà il proprio token di autenticazione. Si otterrà automaticamente il token di autenticazione sul messaggio di posta

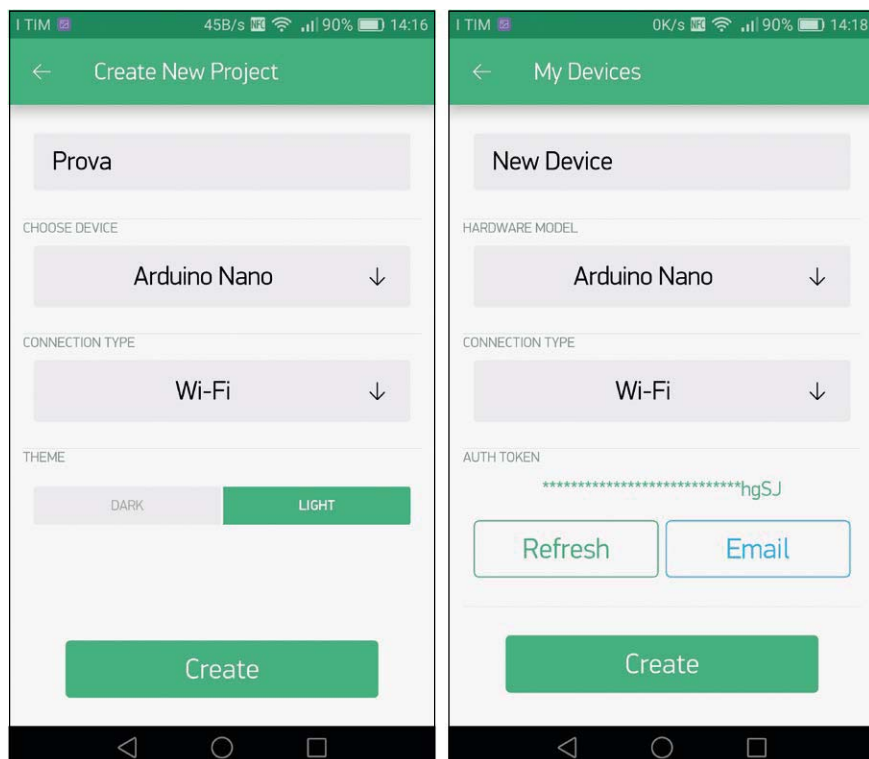
Fig. 2
Creare un account
all'avvio di Blynk e
scegliere l'hardware
associato.

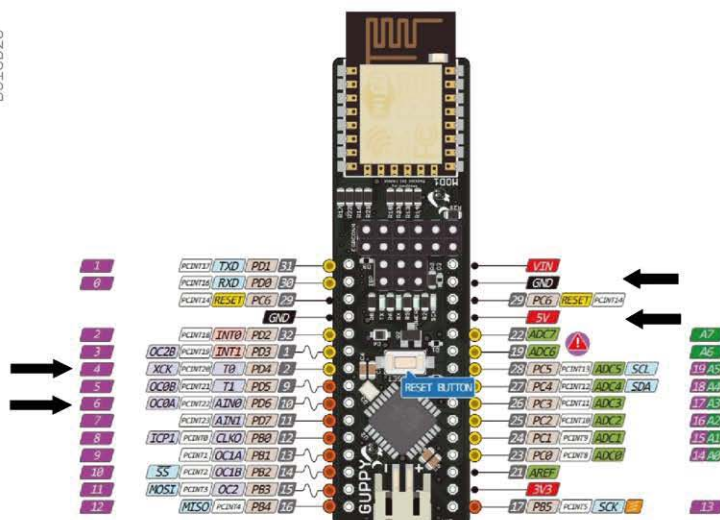
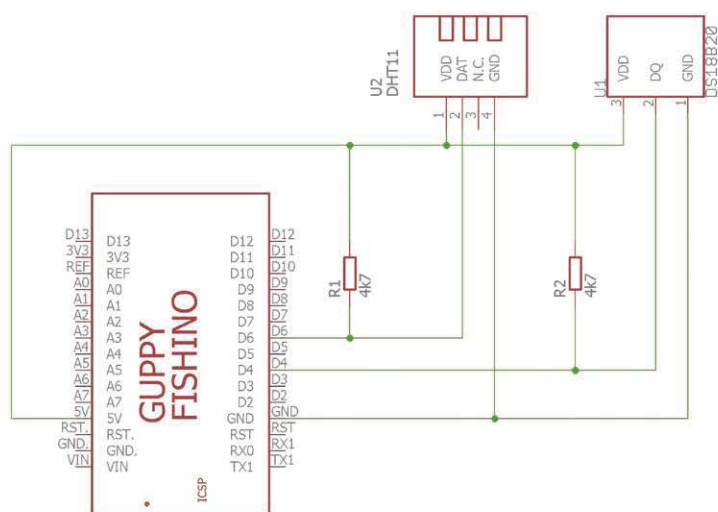


elettronica dopo la creazione del progetto. È inoltre possibile copiarlo manualmente. Fare clic sulla sezione dispositivi e sul dispositivo richiesto sele-

zionato (**Fig. 3**). Ovviamente è bene non condividere il token di autenticazione con nessuno, a meno che non si desideri che qualcuno abbia accesso all'har-

Fig. 3
Creare un nuovo
progetto scegliendo
il titolo, il device
hardware (Arduino
Nano) e il tipo di
connessione alla
rete (WiFi); subito
verrà rilasciato il
Token.





HARDWARE E CIRCUITO IN PRATICA

Lo schema elettrico della nostra applicazione è davvero semplice, come potete vedere nella **Fig. 4** insieme al pin-out del Fishino Guppy e nella **Fig. 5** nella versione cablata su breadboard. La scheda Fishino Guppy acquisisce i dati ambientali utilizzando per la temperatura un sensore DS18B20 (U1) e per l'umidità un DHT11 (U2): occorre aggiungere solo due resistenze di pull-up da 4,7 kΩ sulle linee dati che fanno capo ai due sensori e che risultano collegate all'ingresso 4 del Guppy per il sensore di temperatura U1 (il DS18B20) e all'ingresso 6 per U2 (DHT11).

In **Fig. 5** vediamo lo schema di cablaggio che segue lo schema elettrico. Ovviamente possiamo integrare altri sensori utilizzando ulteriori porte di ingresso della Fishino Guppy, oppure utilizzare il bus One-Wire in uso per il sensore DS18B20 collegando altri sensori che sfruttano lo stesso bus. Ognuno potrà sviluppare lo smart-sensor che occorre alla propria applicazione specifica, senza grossi problemi. I componenti necessari per la realizzazione di questa esperienza sono sostanzialmente pochi

Fig. 4
Schema di cablaggio del circuito (a sinistra) e piedinatura del Guppy (a destra).

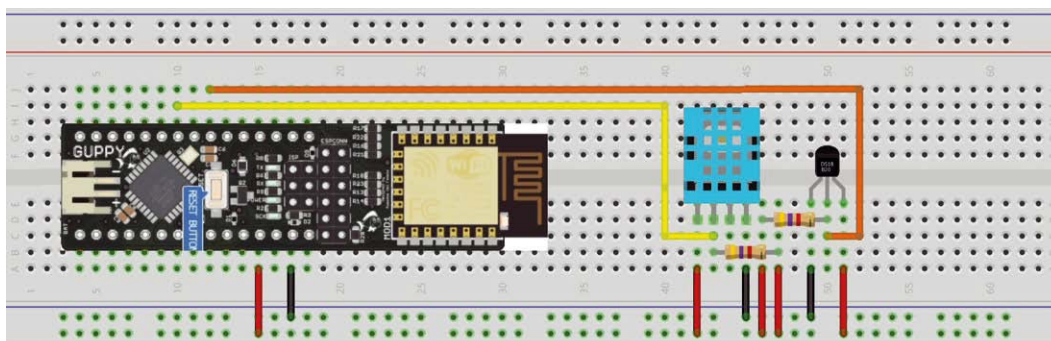


Fig. 5
Schema di montaggio su breadboard.

Il DHT11 è un sensore di umidità e temperatura digitale a basso costo. Per la misura dell'umidità utilizza un sensore di tipo resistivo, mentre per la misura della temperatura dell'aria circostante un termistore NTC. I sensori sono connessi a un microcontrollore a 8 bit contenuto nel package. Ogni DHT11 è calibrato in laboratorio e i coefficienti di calibrazione sono memorizzati nella memoria OTP del microcontrollore a bordo per essere usati nella misura delle due quantità fisiche. Entrambe le misure sono disponibili in uscita dal componente con un segnale digitale seriale sul pin dati. È abbastanza semplice da usare, ma richiede una accurata tempistica per leggere i dati disponibili. L'unico vero svantaggio di questo sensore è che la misura è ottenibile una volta ogni 2-6 secondi.

Caratteristiche del DHT11:

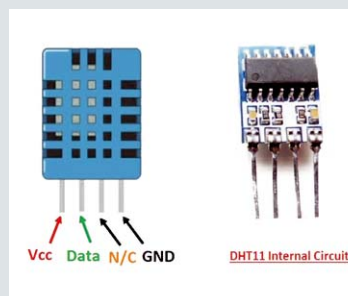
- compensazione sull'intera gamma di temperatura;
- misura dell'umidità relativa (20% - 90%) e della temperatura (0 - 50 °C);
- accuratezza massima per la misura di umidità: $\pm 4\%$ RH; risoluzione $\pm 1\%$ RH;
- accuratezza massima per la misura di temperatura: $\pm 2\text{ }^{\circ}\text{C}$; risoluzione $\pm 1\text{ }^{\circ}\text{C}$;
- segnale digitale calibrato;
- eccezionale stabilità a lungo termine;
- non sono necessari componenti extra;
- lunga distanza di trasmissione (fino a 20 metri).
- basso consumo energetico (3-5.5 VDC, max 2.5 mA durante la misura, 0.15 mA in standby).

Il protocollo che utilizza il componente per trasferire i dati è interessante. Tutte le letture del sensore vengono inviate utilizzando un bus a filo singolo

che riduce il costo e allunga la distanza di trasmissione. Il bus dati a filo unico usa una resistenza di pullup a VCC. Il formato di comunicazione può essere suddiviso in tre fasi: 1) Request; 2) Response; 3) Data Reading.

- 1) Request: per richiedere le letture dal DHT11, occorre inviare una richiesta che è rappresentata da un segnale low sul bus per almeno 18 ms e poi da un segnale high per almeno 40 μS .
- 2) Response: dopo la fase di Request, il DHT11 risponde in automatico indicando che ha ricevuto la richiesta, con un segnale di almeno 54 μS low e almeno 80 μS high.
- 3) Data reading: dopo la fase di Response, il DHT11 invia i dati misurati dai sensori. I dati sono compressi in un pacchetto di 5 segmenti di 8 bit ciascuno. In totale $5 \times 8 = 40$ bit. I primi due segmenti sono l'umidità letta (intero e decimale). I seguenti due sono la temperatura letta (intero e decimale) e l'ultimo segmento è il Check Sum che è la somma dei primi 4 segmenti. Se il valore di Check Sum non è uguale alla somma dei primi 4 segmenti significa che i dati ricevuti non sono corretti.

Ogni bit inviato è costituito da un segnale 54 μS low e da 24 μS a 70 μS high a seconda del valore del bit da inviare: Bit '0': ~ 54 μS low e ~ 24 μS high; Bit "1": ~ 54 μS low e ~ 70 μS high.



e tutti acquistabili direttamente sul sito di Futura Elettronica (www.futurashop.it). Di seguito riportiamo le specifiche dei prodotti interessati in questa realizzazione, unitamente al codice:

- Fishino Guppy (7305-GUPPY);
- DHT11 (8300-DHT11);
- DS18B20 (1310-DS18B20);
- Breadboard 400 contatti con base autoadesiva, dimensioni 82x55 mm (7300-BB400).

Serve poi, per realizzare le connessioni elettriche, una confezione con jumper maschio-maschio 7300-JUMPER65.

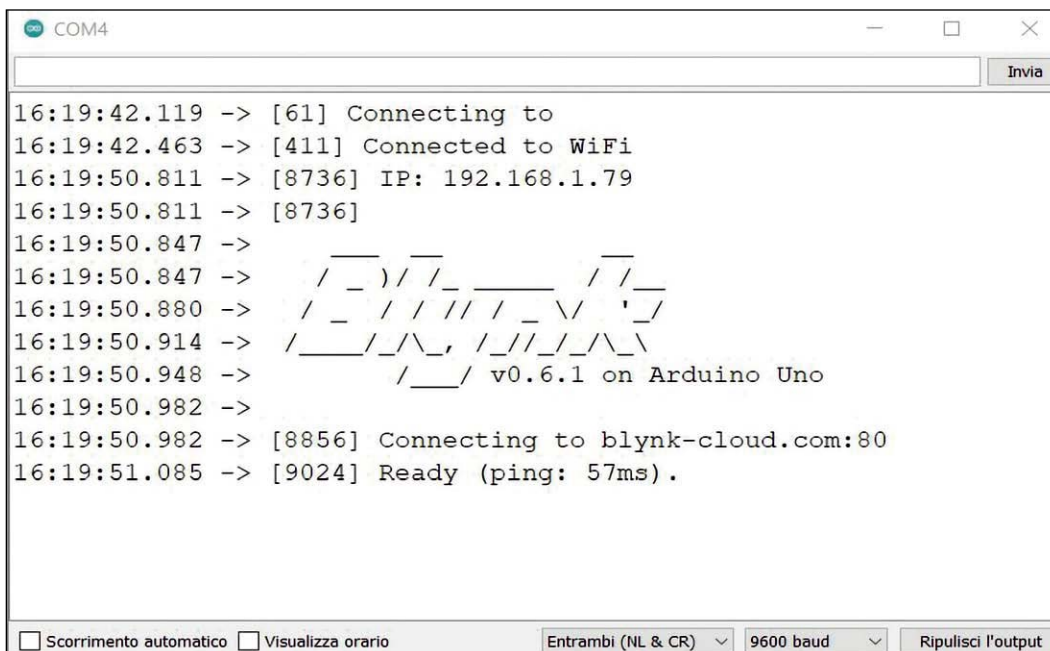
IL FIRMWARE PER LA GUPPY

Siamo giunti finalmente alla parte probabilmente più interessante dell'articolo, ovvero la descrizione del firmware. Come già accennato, tutto il software scritto per questo progetto è prelevabile dal sito della rivista e può essere fonte di ispirazione per altre vostre idee.

Il progetto si compone di un unico file da caricare sul Guppy mediante l'IDE di Arduino, oltre all'app da sviluppare sullo smartphone con Blynk, di cui ci occuperemo dopo. Il firmware da caricare sulla piccola Fishino è visibile nel **Listato 1**. Oltre ad includere la libreria propria di Fishino e le librerie dei sensori utilizzati, occorre includere la libreria di Blynk dedicata alle schede Fishino, *BlynkSimpleFishino.h* che trovate nella directory `... \Arduino \libraries \Blynk \src \`, una volta installata la libreria. Tale libreria ci 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 è introdotto nel firmware con la variabile `auth[]`. Nella parte iniziale sono definite le variabili in uso per la gestione dei due sensori.

Subito dopo la definizione del token, occorre introdurre il nome del vostro WiFi SSID con relativa password, per permettere al Guppy di accedere alla rete mediante la libreria di Blynk.

A questo punto è definita la prima delle importanti variabili di Blynk, il timer di cui occorre brevemente parlare. Si definisce come "BlynkTimer timer" dove timer è il nome (che può essere ovviamente variato). A cosa serve un timer? È importante inviare i dati al server periodicamente, con una frequenza non elevata e mantenere il **void loop()** il più pulito possibile. Infatti la programmazione con Blynk prevede un forte utilizzo di funzioni che non sono all'interno del **void loop()**, anzi talvolta tale parte del programma è praticamente costituita da pochissime righe di codice che inizializzano solo l'applicazione, a diffe-



```
COM4
16:19:42.119 -> [61] Connecting to
16:19:42.463 -> [411] Connected to WiFi
16:19:50.811 -> [8736] IP: 192.168.1.79
16:19:50.811 -> [8736]
16:19:50.847 ->
16:19:50.847 ->  / _ ) / / _ _ _ _ _ / / _ _
16:19:50.880 ->  / _ _ / / / / / _ _ \ / ' _ /
16:19:50.914 ->  / _ _ / _ / \ , / _ / _ / \ _ \
16:19:50.948 ->           / _ _ / v0.6.1 on Arduino Uno
16:19:50.982 ->
16:19:50.982 -> [8856] Connecting to blynk-cloud.com:80
16:19:51.085 -> [9024] Ready (ping: 57ms).
```

☐ Scorrimento automatico ☐ Visualizza orario Entrambi (NL & CR) 9600 baud Ripulisci l'output

Fig. 6
Monitor
seriale
dell'IDE di
Arduino che
mostra la
connessione
al server
Blynk.

renza della programmazione classica dell'Arduino. BlynkTimer consente quindi di inviare i dati al server Blynk periodicamente a determinati intervalli (definibili) senza interferire con le routine della libreria di Blynk, e questo è davvero importante per 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. Ecco anche il motivo per il quale il **void loop()** è praticamente svuotato, poiché data la velocità di esecuzione del microcon-

trollore a bordo della scheda, un possibile invio di una variabile potrebbe avvenire centinaia di migliaia di volte al secondo, che oltre ad essere inutile (pensiamo ad una temperatura che varia certamente con maggiore lentezza) è considerata dal server simile ad un'operazione di spamming, quindi per evitare danni ad altri utenti, viene di fatto disconnesso quel collegamento.

In questa fase è importante prevedere due ope-

DICE SMART E DICE CARE: IoT PER LA SMART HOME

Arriva sul mercato DICE CARE, un dispositivo per Smart Home che controlla le attività domestiche dei propri cari e notifica situazioni anomale tramite app; attraverso un'Intelligenza Ambientale rileva se la persona in casa sta riposando, se è uscita o se non compie le attività consuete, notificando la situazione. Il dispositivo interpreta tutti i valori provenienti dall'abitazione e li trasforma in azioni dirette o in notifiche e alert personalizzati. Il dispositivo può essere abbinato a DiCE SMART, hub tecnologico di facile installazione e utilizzo (realizzato anch'esso da DiCEworld - www.diceworld.it) che



si connette alla rete di casa tramite Wi-Fi, legge i consumi elettrici in tempo reale grazie al sensore D-ENERGY installato sul contatore, controlla e gestisce i livelli di consumo degli elettrodomestici connessi con prese smart e comanda anche lampadine in wireless. L'app DiCEhome permette di attivare DiCE SMART anche da remoto, impostando soglie massime e visualizzando su smartphone i consumi elettrici in caso di cambiamenti.

↓ Listato 1

```

/*****
This example shows how to use Fishino Guppy with a temperature sensor and an
humidity sensor, to connect your project to a Blynk app.
This requires Fishino Software libraries: http://fishino.it/en/download/
Change WiFi ssid, pass, and Blynk auth token to run :)
Feel free to apply it to any other example. It's simple!
Author: G. Caliano
Blynk library is licensed under MIT license. This example code is in public domain.
*****/

/* Comment this out to disable prints and save space */
// #define BLYNK_PRINT Serial

#include <SPI.h>
#include <Fishino.h>
#include <BlynkSimpleFishino.h>
#include <OneWire.h>
#include <FishinoDallasTemperature.h>
#include <dht11.h>
#include <SoftwareSerial.h>

float temp = 0.0;
int u,t;
dht11 DHT;

#define ONE_WIRE_BUS 4 // Data wire (DS18B20) is plugged into port 4 on the Guppy
OneWire oneWire(ONE_WIRE_BUS); // Setup a oneWire instance to communicate with any OneWire devices
DallasTemperature sensors(&oneWire); // Pass our oneWire reference to Dallas Temperature

// You should get Auth Token in the Blynk App.
char auth[] = "RtwN0d0s6x3XbzHlvVoXXjLX-HCEyZyIq"; // Your auth token here !!!!

// Your WiFi credentials. Set password to "" for open networks.
char ssid[] = "Fishino"; // Your WiFi SSID here !!!
char pass[] = "guppy_63"; // Your WiFi password here !!!

BlynkTimer timer;

// This function sends Arduino's up time every second to Virtual Pin. In the app, Widget's reading frequency
// should be set to PUSH. This means that you define how often to send data to Blynk App.

void myTimerEvent() {

    // You can send any value at any time.
    // Please don't send more that 10 values per second.

    Blynk.virtualWrite(V2, temp);
    Blynk.virtualWrite(V1, u);
}

void setup()
{
    // Debug console
    Serial.begin(9600);
    // Start up the library
    sensors.begin();
    Blynk.begin(auth, ssid, pass);
    // Setup a function to be called every second
    timer.setInterval(1000L, myTimerEvent);
}

void loop()
{
    Blynk.run();
    sensors.requestTemperatures(); // Send the command to get temperatures
    temp = sensors.getTempCByIndex(0);
    DHT.read(6);
    u=DHT.humidity;
    //t=DHT.temperature;
    timer.run(); // Initiates BlynkTimer
}

```

razioni: la definizione del tempo, trascorso il quale verranno inviati i dati al server, mediante la funzione **timer.setInterval(tempo, nome procedura)** con tempo che rappresenta l'intervallo prescelto (1000L = 1 secondo) e che si trova nella sezione di setup (**void setup()**) del firmware; e poi il nome procedura (nel nostro listato è **myTimerEvent()**) che rappresenta la procedura incaricata di inviare le variabili prescelte al server.

In questo caso la procedura **myTimerEvent()** invia all'app due variabili V1 e V2 di tipo Virtual Pins e associa a V1 la variabile **u** che contiene la misura dell'umidità e a V2 la variabile **temp** che contiene il valore della temperatura misurata. Per quanto detto, questa procedura invierà una volta al secondo (**timer.setInterval(1000L, myTimerEvent)**) i dati di umidità e temperatura che il nostro Guppy misura mediante i sensori ad esso collegato.

La parte setup del firmware contiene, oltre all'inizializzazione del sensore su bus OneWire, 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 la prima linea di codice `#define BLYNK_PRINT Serial`, normalmente disabilitata per non occupare memoria, potete controllare se la scheda è stata autenticata e quindi è "nel sistema", come si può vedere in **Fig. 6**. Infatti, potete leggere che la scheda Guppy dapprima si è connessa alla rete locale e dopo un po' di tempo comunica il suo IP address al server Blynk che risponde dichiarando che la connessione è avvenuta con successo. Il firmware ora eseguirà il **void loop()** classico, che analizziamo nel dettaglio. La prima funzione da richiamare è quella di avviare Blynk con **Blynk.run()**. Tale funzione serve per elaborare i comandi in arrivo ed eseguire la gestione della connessione Blynk: ovviamente è necessaria! Le righe successive richiedono la misura delle due variabili ai due sensori e memorizzano in **temp** ed **u** le due informazioni che la scheda trasmetterà al server. Infine è richiamata la funzione **timer.run()** che inizializza il timer come spiegato prima. Il firmware lato Guppy finisce qui e, come promesso, è estremamente semplice e facilmente modificabile a vostro piacimento per gestire i vostri smart-sensors.

L'APP PER ANDROID

Ovviamente descrivere l'app sul cellulare è decisamente più difficile, non perché sia complicato ma non dovendo scrivere nessuna linea di codice e

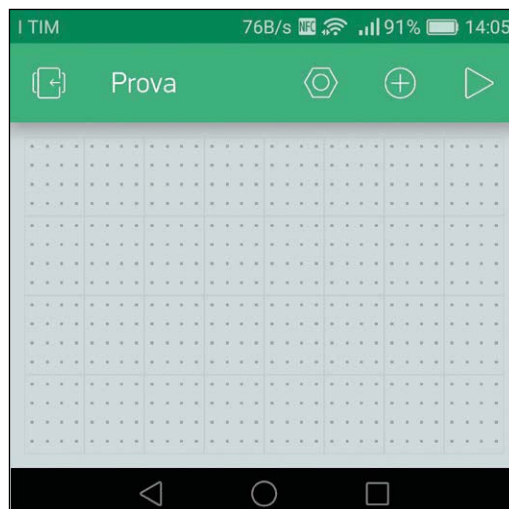
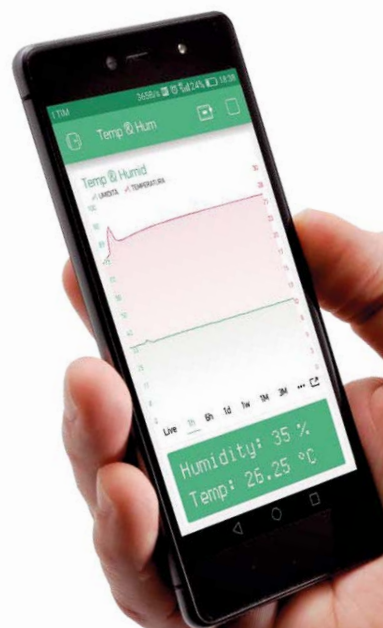


Fig. 7
Creazione di un nuovo progetto Blynk: ambiente di editing per la nuova app da creare.

basandosi solo su alcuni widget da posizionare, non si può seguire una linea logica dettata dalle righe di un software scritto in maniera tradizionale. Iniziamo a dire che ogni progetto di Blynk ha le sue impostazioni di base che si scelgono quando avviamo la costruzione di una nuova app (Create New Project), che possiamo riassumere con:

Project Name, Device, Theme.
La prima rappresenta ovviamente il nome dell'app, il Device permette di scegliere la piattaforma che vogliamo connettere (Arduino Nano) e con quale tipo di connessione (WiFi). Il Theme è semplicemente il colore dello sfondo (scuro o chiaro, diciamo che non è il massimo delle scelte possibili). Date queste semplici informazioni si "crea" il progetto e viene subito inviato alla nostra email il token di autenticazione collegato a questo progetto. A questo punto



Dalla applicazione è possibile monitorare i sensori e visualizzare su grafico i valori rilevati.

Blynk può controllare direttamente i Pin I/O digitali e analogici sull'hardware. Sebbene non sia necessario scrivere codice per queste operazioni, non sempre è sufficiente! Blynk prevede un'ulteriore tipologia di Pin: i Virtual Pins che possono inviare tutti i tipi di dati dal vostro microcontrollore all'App Blynk e viceversa. Tutto ciò che si collega al vostro hardware sarà in grado di parlare con Blynk. Con i Virtual Pins è possibile inviare qualcosa dall'App, elaborarlo sul microcontrollore e quindi inviarlo di nuovo allo smartphone. È possibile attivare funzioni, leggere dispositivi I2C, convertire valori, controllare servo e motori DC, ecc.

I Virtual Pins possono essere utilizzati per interfacciarsi con librerie esterne (Servo, LCD e altri) e implementare funzionalità personalizzate. L'hardware può inviare dati ai widget sul Pin virtuale in questo modo:

```
Blynk.virtualWrite(pin, "abc");
Blynk.virtualWrite(pin, 123);
Blynk.virtualWrite(pin, 12.34);
Blynk.virtualWrite(pin, "hello", 123, 12.34);
```

Viceversa, è possibile inviare qualsiasi dato da Widgets nell'applicazione sullo smartphone al vostro hardware. Tutti i widget controller possono inviare dati ai Pin virtuali sull'hardware. Ad esempio, il codice seguente, su Arduino, mostra come ottenere i valori dal Widget Pulsante nell'App:

```
BLYNK_WRITE(V1) //Button Widget sta scrivendo per il pin V1
{
  int pinData=param.asInt();
}
```

Quando si preme un pulsante, Blynk App invia 1. Al secondo clic invia 0. Il Pin virtuale è quindi un modo per scambiare tutti i dati tra il vostro hardware e la Blynk app e viceversa.

Pensate ai pin virtuali come a dei canali per l'invio di dati. Assicuratevi di differenziare i Pin virtuali dai pin GPIO fisici sul vostro hardware. I Pin virtuali non hanno alcuna rappresentazione fisica. Tutti i valori di Virtual Pin vengono sempre inviati come stringhe e non esistono limiti pratici per i dati che possono essere inviati. Tuttavia, ci sono alcune limitazioni sul lato hardware quando si tratta di numeri. Ad esempio, il numero intero su Arduino è a 16 bit, consentendo l'intervallo da -32768 a 32767. Per interpretare i dati in arrivo come Integer, Float, Doubles e Strings, utilizzare:

```
param.asInt();
param.asFloat();
param.asDouble();
param.asStr();
```

Se il widget che stiamo usando nell'app prevede due dati in uscita (per esempio x e y), allora lato Arduino occorre specificare quale variabile stiamo leggendo, come nell'esempio successivo:

```
BLYNK_WRITE(V1) {
  int x = param[0].asInt();
  int y = param[1].asInt();
  // Do something with x and y
  Serial.print("X = ");
  Serial.print(x);
  Serial.print(" Y = ");
  Serial.println(y);
}
```

È inoltre possibile ottenere i dati RAW dal buffer param:

```
param.getBuffer()
param.getLength()
```

siamo introdotti nella schermata di "costruzione" della nostra app, proposta nella **Fig. 7**. In essa osserviamo l'ambiente di editing dell'app. Sulla riga in alto, oltre al nome che abbiamo dato all'app, ci sono quattro simboli: il primo da sinistra è quello che ci permette o di spostarci tra le varie app che abbiamo creato, o di entrare sulla pagina per creare una nuova app, oppure di uscire dal software. Ovviamente basta scorrere le schermate a destra e a sinistra con il dito per effettuare la scelta. Il secondo simbolo (un esagono con un cerchio inscritto) è il pulsante che ci permette di variare i settings dell'applicazione che stiamo scrivendo. Possiamo cambiare il nome; l'accesso condiviso dell'app oppure no (se condiviso, allora viene generato un link da dare all'amico con cui vogliamo condividere l'app, dietro pagamento di 1000 crediti); possiamo usare un Shortcut se il cellulare lo prevede nel suo

Launcher di applicazioni; i settings dell'hardware a cui è connessa l'app; le informazioni circa il token di autenticazione; il tema dell'app (abbiamo visto, scuro o chiaro); se vogliamo mantenere lo schermo sempre acceso (consente di utilizzare l'applicazione Blynk senza andare alla modalità di sospensione, cosa che di solito tutti i dispositivi mobili fanno); notificare che 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; non mostrare le notifiche quando il sistema è offline: infatti per scopi di debug, ogni volta che l'hardware va offline, il Server Blynk vi avviserà con un popup nell'app di questo evento. Tuttavia, quando il debug non è necessario, queste notifiche sono prive di significato. Quindi questo interruttore consente di disattivare questi popup. Possiamo infine copiare il progetto oppure cancel-

larlo. Il terzo simbolo (un più cerchiato) ci 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.

I WIDGET

I widget sono moduli di interfaccia. Ognuno di essi svolge una specifica funzione di input/output durante la comunicazione con l'hardware.

Ci sono quattro tipi di widget:

- **Controller** - utilizzato per inviare comandi che controllano l'hardware: in questa categoria ci sono bottoni, slider, joystick, cioè tutti quei possibili controlli che permettono di interagire con il nostro hardware remoto;
- **Display** - utilizzati per la visualizzazione dei dati da sensori e altre fonti: tra i Display abbiamo Led, caselle di testo ove appare la variabile da visualizzare, gauge, LCD (il classico LCD a una riga o a due righe), il Terminal (una specie di visualizzatore di messaggi da/per l'hardware remoto), le chart (SuperChart, cioè la possibilità di graficare le variabili in modi diversi), lo streaming di video o la visualizzazione di immagini.
- **Notifiche** - inviare messaggi e notifiche su Twit-

ter, per e-mail, o push al cellulare stesso;

- **Interfaccia** - widget per eseguire determinate funzioni GUI (menù, tab, tabelle,...);

Esistono poi altri widget che non appartengono ad alcuna categoria; per esempio l'Eventor, che noi useremo, che permette di analizzare “eventi” e quindi di prendere delle decisioni. Abbiamo anche un Music Player che permette di aggiungere suoni alle app.

Ogni Widget ha le proprie impostazioni. Alcuni dei Widget (ad esempio Bridge) abilitano solo la funzionalità e non hanno impostazioni. Altre permettono di interagire con l'hardware remoto mediante i pin digitali, analogici e i pin virtuali che abbiamo già visto.

I Pin digitali: rappresentano pin I/O digitali fisici sull'hardware remoto. I pin abilitati per PWM sono contrassegnati con il simbolo ~.

I Pin analogici - rappresentano pin I/O analogici sull'hardware remoto.

I Pin virtuali - non hanno alcuna rappresentazione fisica. Essi sono utilizzati per trasferire tutti i dati tra Blynk App e l'hardware.

Iniziamo dunque a costruire la nostra app per gestire i sensori in remoto sul Guppy.

Il primo widget che aggiungiamo è il SuperChart

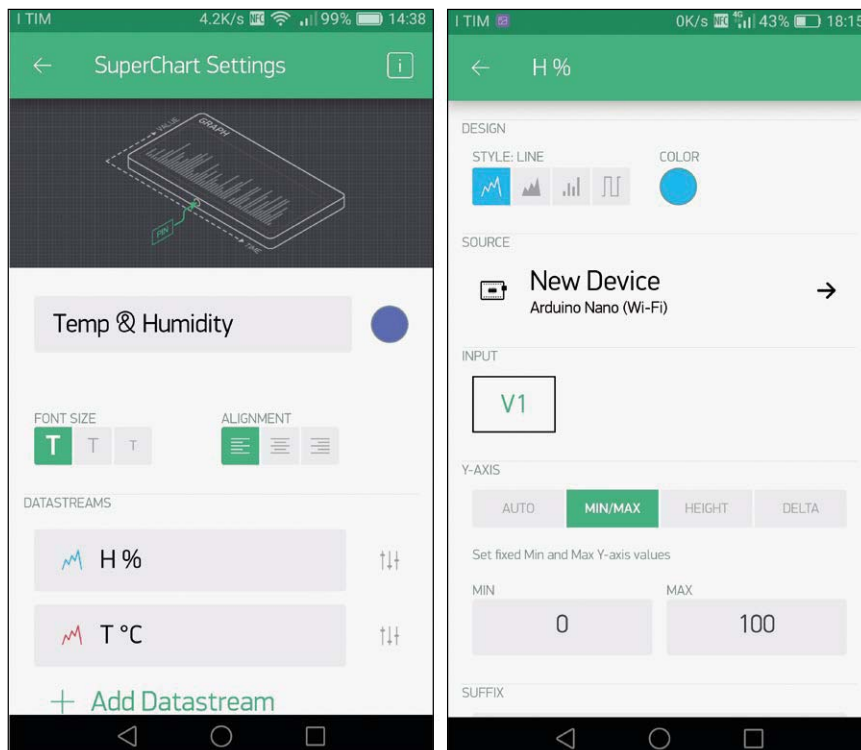


Fig. 8
Settings
del widget
SuperChart.

Fig. 9
Settings del
widget LCD.



che ci permetterà di osservare nel tempo la variazione delle nostre due variabili inviate dal Guppy (temperatura e umidità). Prima di tutto occorre chiarire una interessante caratteristica del sistema Blynk. 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 non eravamo online. Tutto ciò se il nostro Guppy è acceso ed online! Trasciniamo il widget SuperChart sull'area di editing. Qui possiamo ridimensionarlo opportunamente (tenere premuto il dito sul widget per abilitare il ridimensionamento) e, toccando con il dito il widget, possiamo visualizzare i suoi settings, che vediamo in **Fig. 8**. Possiamo dare un titolo al grafico, ma ovviamente la parte più importante è collegare le variabili passate dal Guppy al widget. Questo si fa semplicemente pigiando "Add datastream" che permette di scegliere quale variabile visualizzare (nel caso in figura colleghiamo il Virtual Pin V1 che rappresenta l'umidità) usando Input, e settando il tipo di grafico, il suffisso (in questo caso %) e così via, in maniera molto intuitiva e semplice.

Si possono aggiungere al grafico tutte le variabili che si vogliono! Nel nostro caso dobbiamo solo aggiungere il Virtual Pin V2 che rappresenta la

temperatura (come l'abbiamo definito nel firmware del Guppy). Una volta che abbiamo terminato i settings del widget SuperChart (costo 900 crediti), passiamo ad inserire un LCD a due righe 16 caratteri (400 crediti), che ci permette di visualizzare le due variabili semplicemente. Anche in questo caso trasciniamo il display sull'area di editing dove possiamo posizionarlo e ridimensionarlo a piacimento. Quindi passiamo al setting dei parametri per farlo funzionare (**Fig. 9**). Possiamo aggiungere le due variabili (V1 e V2), indicare il loro valore minimo e massimo, la velocità di aggiornamento (è meglio usare il sistema Push, cioè aggiornare quando è disponibile), e quindi è possibile aggiungere un messaggio esplicativo sul display (ad esempio: "Humidity: /pin0/ %" che permetterà di avere sul display il messaggio "Humidity: 33 %", cioè abbiamo aggiunto un testo ad una misura collegata ad un pin specifico (in questo caso pin0). Infine aggiungiamo due widget interessanti: l'Eventor (500 crediti) e l'Email (100 crediti).

Il widget Eventor (**Fig. 10**) consente di creare semplici regole di comportamento o eventi. Diamo un'occhiata a un tipico caso d'uso: leggere la temperatura dal sensore DS18B20 e inviare una notifica push quando la temperatura è superiore a un certo limite. Eventor è utile quando è necessario modificare le condizioni in tempo reale senza ricaricare il nuovo Sketch sull'hardware. È possibile creare tutti gli eventi che si rendono necessari. Eventor potrebbe anche essere attivato dal lato dell'applicazione: dovete solo assegnare il widget allo stesso pin del vostro Evento all'interno di Eventor.

Consideriamo un evento semplice: se la temperatura > 25 inviare una notifica mediante e-mail (widget Email). Quando la temperatura supera la soglia di 25°C, viene attivata un'azione di notifica. Se la temperatura continua a rimanere al di sopra della soglia di 25, non verranno attivate azioni. Ma se la temperatura scende sotto la soglia e poi passa di nuovo sopra, la notifica verrà inviata di nuovo. Eventor supporta anche gli eventi Timer. Ad esempio, è possibile impostare un pin V1 come ON/HIGH alle 21:00:00 ogni venerdì. Con Eventor Time Event potete assegnare più timer sullo stesso pin, inviare qualsiasi stringa/ numero, selezionare giorni e fuso orario. In questo caso il widget timer si basa sull'ora del server e non sull'ora del telefono. A volte l'ora del telefono potrebbe non corrispondere all'ora del server.

Infine occorre notare che gli eventi vengono attivati una sola volta quando viene soddisfatta

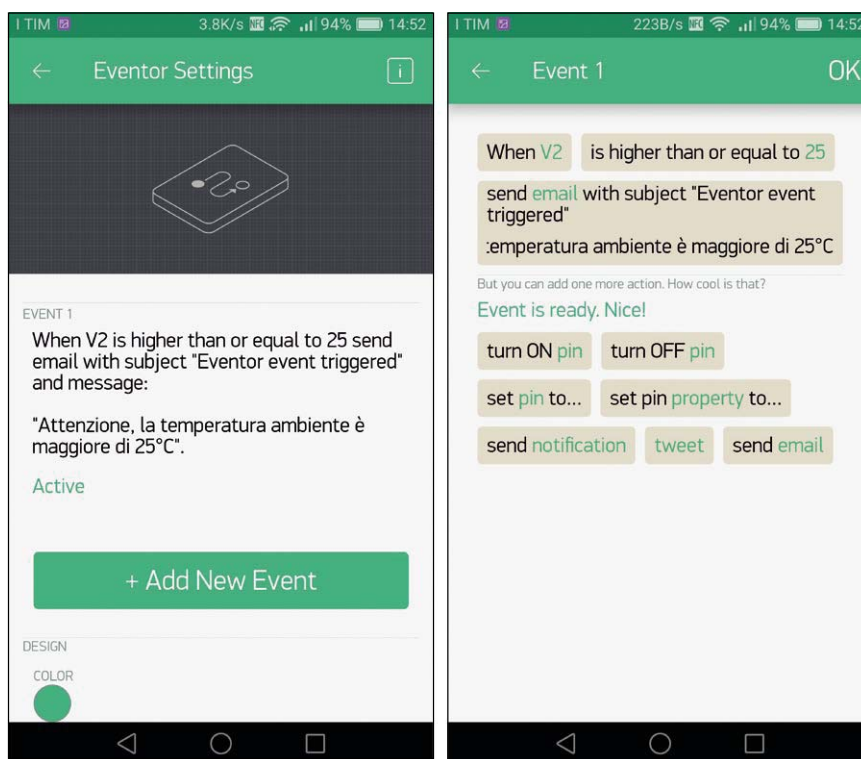


Fig. 10
Settings
del widget
Eventor.

la condizione. Questo significa che il concatenamento degli eventi non è possibile. L'ultimo widget da inserire è **Email**, che prevede solo l'indirizzo di e-mail a cui inviare la notifica generata da Eventor. A questo punto, se tutto è OK, l'app funzionerà senza problemi avviandola con il triangolo in alto a destra. Abbiamo speso 1900 crediti dei 2000 a disposizione gratis, per cui ci rimangono anche 100 crediti per aggiungere qualcosa, e questo lo lasciamo al lettore (per esempio aggiungere un LED da 100 crediti, in maniera tale da collegarlo ad un LED fisico sulla Guppy connesso ad un pin digitale a piacere, il quale si accende mediante Eventor che invia all'hardware una notifica quando la temperatura è minore di una certa soglia).

CONCLUSIONI

Abbiamo realizzato un semplice sistema IoT con pochi componenti, a basso costo, connesso con uno smartphone Android. Ovviamente l'idea proposta oltre ad essere un progetto da realizzare, pronto all'uso, utile per risolvere problemi pratici di vario genere, può essere usato come spunto per una propria idea o come base di studio per apprendere, in dettaglio, le soluzioni e le strategie adottate per la realizzazione di altri progetti. In genere, l'esperienza ricorda che si fa di "necessità virtù": nel momento in cui occorre realizzare una

propria applicazione perché ci rendiamo conto della sua utilità nel nostro quotidiano, la spinta a realizzarla diventa maggiore, e anche le difficoltà sono meno ardue. Solo pochi anni fa sviluppare una app per cellulare connessa ad un hardware remoto (che potrebbe anche trovarsi nell'altra parte del mondo) era impresa ardua e pochi sapevano come muoversi. Con gli strumenti oggi descritti (Blynk) e con l'hardware a nostra disposizione (a costi estremamente ridotti) il tempo di sviluppo può ridursi a poche ore. Chiudiamo ricordando che sul nostro sito è possibile reperire il firmware già compilato e il codice sorgente. □



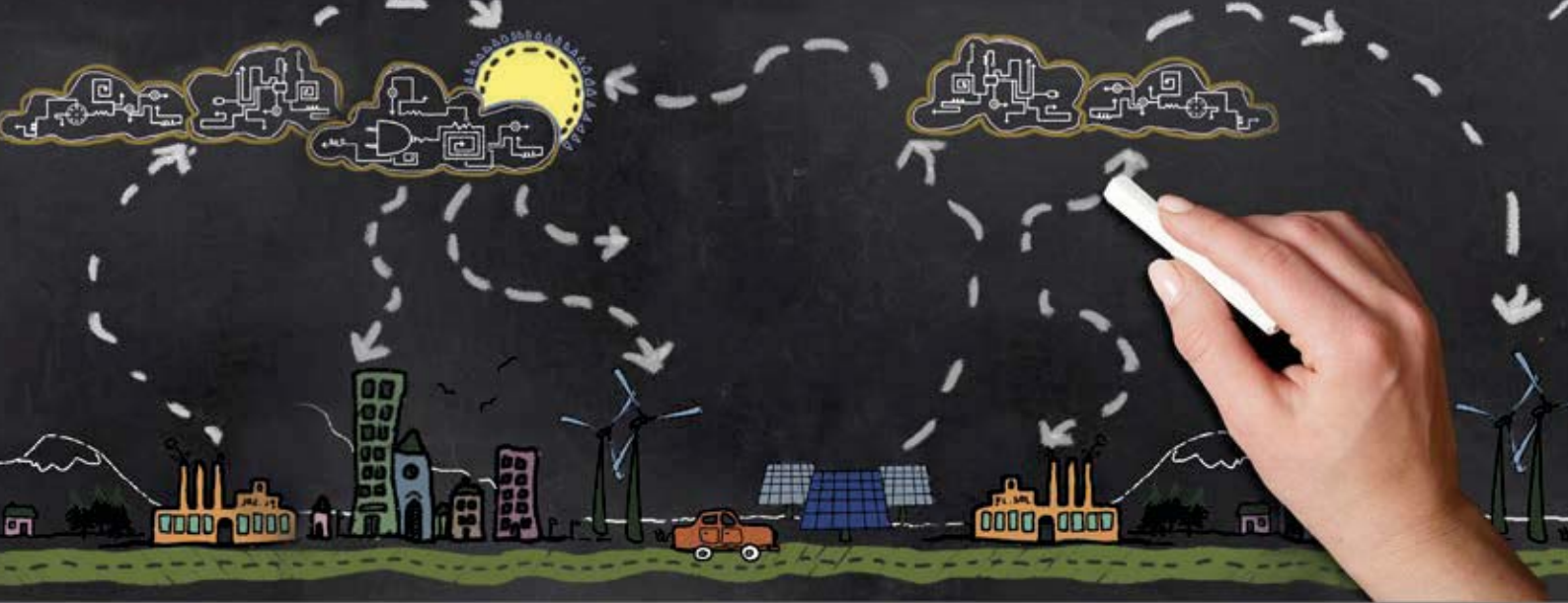
Cosa occorre?

I componenti utilizzati in questo progetto sono disponibili presso Futura Elettronica. La scheda Fishino Guppy (cod. GUPPY) è in vendita a Euro 39,00, il sensore di temperatura (cod. DS18B20) è disponibile a Euro 2,90, il sensore di umidità (cod. DHT11) costa Euro 3,50. 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>**





Reinventa il tuo progetto

Semplifica Con la Piattaforma di Sviluppo Curiosity

La tua Curiosity Nano Development Platform oltre ad essere molto conveniente offre una varietà di kit di valutazione basati su microcontroller (MCU), che ti consentono di esplorare facilmente diverse architetture per il tuo progetto embedded a 32 bit. Potrai operare in un'unica piattaforma di sviluppo o personalizzare in base alle tue esigenze per le singole applicazioni.

I kit di valutazione Curiosity Nano basati su MCU a 32 bit possono essere scalati utilizzando le schede Curiosity Nano Base for Click™ consentendoti così di creare un ambiente di sviluppo in grado di portare il tuo progetto, dall'esplorazione al prototipo funzionante, su un'unica piattaforma. Le funzionalità di programmazione e debug on-board eliminano la necessità di un programmatore esterno, riducendo così i costi di sviluppo.

Caratteristiche salienti

- Possibilità di scalare da MCU 8/16 bit a 32 bit per ottenere prestazioni più elevate
- Punto di ingresso a costi inferiori per la valutazione dei dispositivi
- Supporto completo per programmazione e debug on-board
- L'aggiunta di schede Click espande le funzionalità per una progettazione di sistema più completa

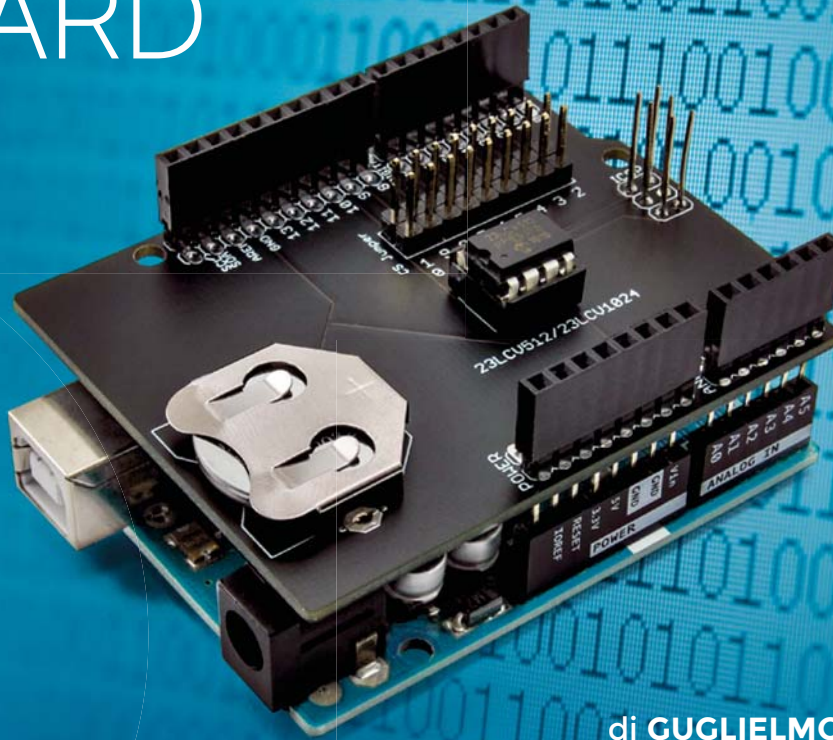
Microchip offre una serie di esempi su MPLAB® Harmony Integrated Software Development Framework, completi di distinta base (BOM), codice utente e note applicative per avviare rapidamente il tuo progetto e far arrivare più velocemente il prodotto sul mercato.



microchip.com/curiositynano

Il nome e logo Microchip, il logo Microchip, MPLAB e PIC sono marchi industriali registrati di Microchip Technology Incorporated negli U.S.A. e altri Stati. Tutti gli altri marchi industriali menzionati appartengono ai rispettivi titolari.
© 2021 Microchip Technology Inc. tutti i diritti riservati.
DS60001678A, MEC2357A-ITA-02-21

nvSRAM BOARD



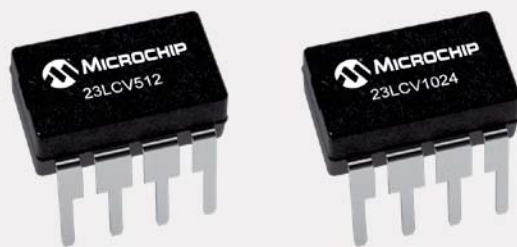
di **GUGLIELMO BRACUGLIA**

Aggiungiamo una RAM “non volatile” dove scrivere e mantenere, grazie a una pila tampone, dati permanenti.

M

olto spesso sul forum di Arduino gli utenti chiedono come salvare, con una certa frequenza, delle informazioni in maniera “non volatile”, in modo che, in caso di mancanza di alimentazione, al ritorno il loro sistema riprenda con le condizioni lasciate al momento dello spegnimento. Tutte le schede Arduino basate su MCU AVR dispongono di una certa quantità di memoria EEPROM e, se la velocità di salvataggio delle informazioni non è eccessiva, in considerazione del fatto che ogni singola cella di detta memoria è garantita per un numero massimo di riscritture pari a 100.000, è ovviamente possibile ed economicamente conveniente utilizzare tale memoria interna alla MCU. Tuttavia 100.000 scritture garantite possono sembrare un numero enorme ed effettivamente più che sufficiente nella maggior parte dei casi, ma il problema è che, se le scritture

Fig. 1
Due memorie
nvSRAM
Microchip.



diventano molto frequenti, come, ad esempio, una scrittura ogni 10 secondi (caso realmente apparso in una domanda sul succitato forum), le 100.000 scritture si esauriscono in soli 10.000 secondi, ovvero, arrotondando per eccesso, in 167 minuti che sono meno di tre ore di funzionamento! Quindi, in tutti quei casi in cui il salvataggio delle informazioni è veramente sporadico, come ad esempio nelle impostazioni delle temperature di un termostato (che normalmente si modificano di rado) l'uso della EEPROM integrata è certamente la soluzione più semplice ed economica, ma, appena la frequenza di salvataggio delle informazioni raggiunge la periodicità dei minuti, allora occorre pensare ad un altro supporto di memorizzazione, veloce, affidabile ed in grado di conservare i dati anche in assenza della tensione di alimentazione. Attualmente il mercato offre diversi tipi di memorie riscrivibili ed in grado di conservare l'informazione anche in caso di mancanza dell'alimentazione principale, ciascuna con i suoi pregi ed i suoi difetti. Qui di seguito ne elenchiamo alcuni tipi evidenziando i pro ed i contro.

EEPROM

Acronimo di "*Electrically Erasable Programmable Read-Only Memory*"; normalmente presenti in dimensioni ridotte in alcune MCU (es. Arduino UNO/Leonardo hanno una EEPROM), sono in grado di conservare le informazioni in esse memorizzate

anche in mancanza di alimentazione. Hanno il difetto di essere piuttosto lente nelle operazioni di scrittura e di avere un ciclo di vita limitato nel numero di scritture possibili per ogni singola cella di memoria (ad esempio, per quelle presenti sulle MCU AVR, il numero di riscritture garantito è di 100.000). La ritenzione dei dati memorizzati è normalmente di qualche decina di anni in assenza di tensione. Ad esempio, per le EEPROM presenti nelle MCU AVR essa è garantita per un massimo di 20 anni. Il costo di un chip da 512 Kbit è circa di un dollaro.

FRAM

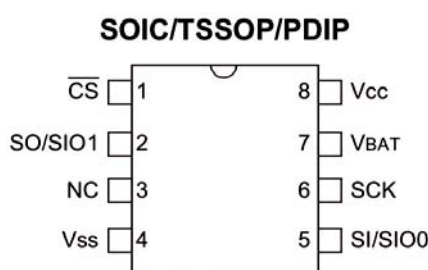
Acronimo di "*Ferroelectric Random Access Memory*" sono memorie simili alle classiche RAM ma, con il vantaggio, di mantenere le informazioni memorizzate anche in mancanza di alimentazione. Rispetto alle EEPROM hanno l'enorme vantaggio di permettere la riscrittura di ogni singola cella un numero pari ad almeno 10^{14} volte o, per alcuni modelli a bassa tensione, anche di più e di essere in grado di conservare le informazioni per un periodo anche superiore ai 150 anni. Anche la velocità di accesso ai dati è circa un fattore 100 volte superiore a quella delle EEPROM e, per tutte queste caratteristiche sono, tecnologicamente, tra le migliori. Purtroppo però, il loro costo è circa un ordine di grandezza superiore a tecnologie meno sofisticate (quali EEPROM o EERAM); difatti, per un chip da 512 kbit, esso è superiore ai 10 dollari.

EERAM

Acronimo di "*Electrically Erasable Random Access Memory*", sono a tutti gli effetti delle SRAM con in parallelo delle celle di memoria simili alle EEPROM. Quando sono alimentate, offrono tutti gli stessi vantaggi di una RAM, quindi numero di scritture per cella praticamente illimitate, velocità di lettura/scrittura, ecc. ecc. Nel momento in cui viene rilevata una mancanza di alimentazione, grazie alla corrente accumulata in un condensatore da collegare tra due suoi piedini, il contenuto della SRAM viene trasferito nella parte simile ad una EEPROM permettendone quindi la conservazione. Al ritorno dell'alimentazione avviene il procedimento opposto: il contenuto salvato nella parte simile ad una EEPROM viene trasferito nella SRAM ripristinando la situazione a prima della mancanza di alimentazione. Chiaramente anche la simil EEPROM presente in tali memorie ha un numero massimo di scritture per cella paragonabile con quello delle vere EEPROM (quindi, garantite le solite 100.000 scritture), ma, contrariamente ad esse in cui ad ogni

Fig. 2
Piedinatura della
NVRAM.

Package Types



variazione del contenuto viene effettuata la scrittura, nelle EERAM la lettura/scrittura viene sempre fatta in RAM e la parte simile alla EEPROM è scritta solo quando manca l'alimentazione. Potremmo quindi dire che il limite, per queste memorie, è il numeri di "spegnimenti/accensioni" che si attesta attorno al valore garantito di 100.000. Il loro costo, a parità di capacità, è circa il doppio di una semplice EEPROM, costo ampiamente ripagato dalla velocità e dalla tecnica utilizzata; esso si attesta, per un chip da 512 Kbit, attorno ai due dollari e mezzo.

nvSRAM

Acronimo di "Non Volatile Static Random Access Memory", sono anche esse a tutti gli effetti delle SRAM, ma sono dotate di due pin per il collegamento di una piccola batteria. Quindi, quando sono alimentate dall'alimentazione principale, sono delle vere SRAM, con tutti i vantaggi che ne conseguono mentre, quando l'alimentazione principale viene a mancare, esse si pongono in stand-by a bassissimo assorbimento (tipicamente attorno ai 4 μ A) e mantengono le informazioni memorizzate grazie ad una batteria di backup. Se si considera l'assorbimento tipico in stand-by, si capisce bene che con una semplice batteria a bottone tipo CR2032, si possono conservare le informazioni, in assenza di alimentazione principale, anche oltre i 4 anni! Il loro costo è inferiore a quello di una EERAM e, per un chip da 512 kbit esso è di circa un dollaro e mezzo, inoltre, vantaggio importante, non essendo presente alcuna parte simile ad una EEPROM, hanno un numero di letture/scritture ed un numero di "accensioni/spegnimenti" praticamente illimitato rendendole, tra quelle esaminate, probabilmente le migliori su cui far ricadere la scelta e sulle quali è difatti ricaduta la nostra per realizzare una semplice shield, compatibile Arduino Uno/Leonardo/Mega, ma facilmente realizzabile anche per altre schede Arduino.

La serie nvSRAM, su cui è ricaduta la nostra scelta, è quella prodotta in due tagli da Microchip, la 23LCV512, che ha una capacità di 64K x 8 bit (pari a 512 Kbit) e la 23LCV1024, che ha il doppio della capacità, ovvero 128K x 8 bit (pari a 1 Mbit). Per facilitarne il montaggio si è scelta la versione con il package DIL, facilmente montabile anche su breadboard o su basette millefori (Fig. 1).

La pin-out di entrambe queste memorie è visibile in Fig. 2. Come si intuisce dal nome dei piedini, queste memorie sono fatte per collegarsi alla MCU attraverso un bus SPI, quindi condividendo, con altre

periferiche SPI, i pin MISO (pin 2 del chip marcato SO), MOSI (pin 5 del chip marcato SI) ed il SCK (pin 6 del chip marcato SCK). Il pin 1 del chip, marcato CS, è il pin che permette di selezionare il chip. Seguendo lo standard SPI Mode 0 (con MSBFIRST), tale pin deve essere tenuto normalmente a livello HIGH e, solo quando si desidera comunicare con il chip, va portato a livello LOW. Il pin 8, VCC, va collegato all'alimentazione principale (che può andare da 2,5 a 5,5V) mentre il pin 4, VSS, va collegato al GND. Infine, il pin 7, VBAT, va collegato ad una batteria esterna (con tensione dai 1,4 a 3,6V) che viene utilizzata per il mantenimento dei dati in assenza di alimentazione principale. La massima velocità di clock, sul pin 6, SCK, è di 20 MHz. Ricordiamo che la frequenza predefinita del clock del bus SPI di Arduino UNO/Leonardo/Mega è di 4 MHz e che la massima è di 8 MHz (con un clock di CPU di 16 MHz, selezionando fosc/2).

Passiamo ora ad esaminare in dettaglio i diagrammi temporali per le fasi di scrittura e lettura di dette memorie. Il diagramma temporale di una scrittura nella memoria (ovvero dati che arrivano dalla MCU attraverso il piedino MOSI e vengono letti dal chip tramite il piedino SI) è quello rappresentato in Fig. 3.

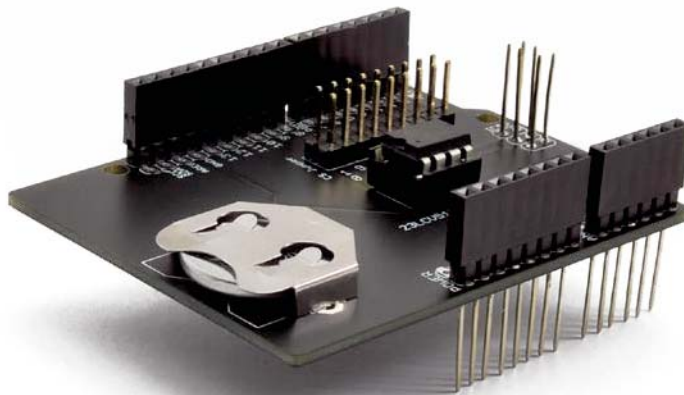
CARATTERISTICHE TECNICHE

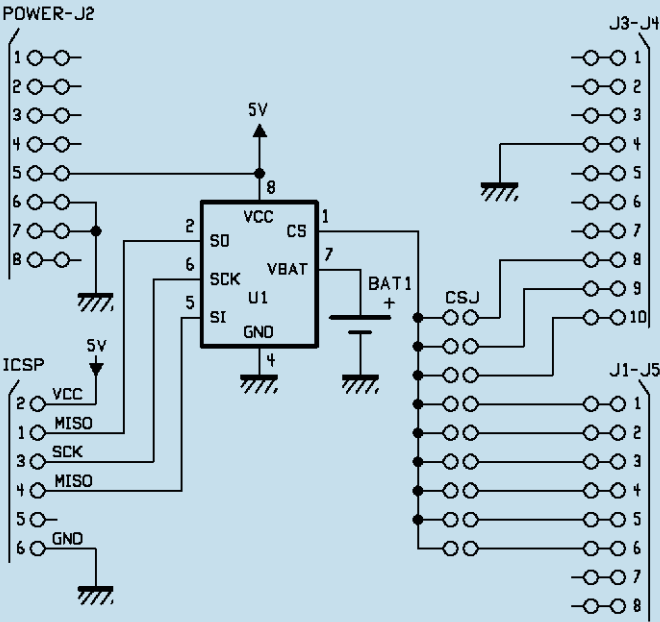
Funzione:
Memoria non volatile
aggiuntiva

Batteria tampone:
Pila 3V a bottone

Capacità:
512 - 1.024 kB

Formato:
Shield Arduino UNO





Invece il diagramma temporale di una lettura dalla memoria (ovvero dati che vanno verso la MCU attraverso il piedino MISO e vengono trasmessi dal chip tramite il piedino S0) è quello rappresentato in **Fig. 4**. La durata dei vari intervalli, identificati dai vari numeri nei diagrammi riportati, è riepilogata nella **Tabella 1**.

Questi chip hanno varie modalità di accesso ai dati. La modalità di accesso “Byte”, permette la lettura di un solo byte per ogni transazione; la modalità di accesso “Page” permette la lettura all’interno di una singola “pagina” (la memoria, al suo interno, è organizzata in pagine da 32 byte); la modalità di accesso “Sequential” permette la lettura di uno o più byte in sequenza, lettura che non ha una lunghezza limite e che quindi permette di leggere/scrivere liberamente blocchi di dimensioni variabili (da 1 ad N byte). Come vedremo, nella libreria che accompagna la scheda che abbiamo sviluppato (ma utilizzabile anche indipendentemente da tale scheda, eventualmente su un circuito realizzato su breadboard o millefori), si è scelto di lavorare in modalità di accesso “Sequential”, che è quella che offre più libertà di manovra. Dedichiamoci ora alla “nVRAM Board” che abbiamo realizzato e visibile in **Fig. 5**. Essa è estremamente semplice ed accoglie la memoria nVRAM (a scelta, in una delle due versioni, 512 kbit o 1.024 kbit avendo queste la stessa piedinatura), una serie di jumper per selezionare quale piedino utilizzare per il segnale di CS (è possibile selezionare dal pin D2 al pin D10 di Arduino), i necessari connettori per il collegamento della scheda con Arduino stesso (si ricorda che, per garantire la compatibilità con differenti schede, i segnali SPI sono prelevati dal connettore ICSP) ed infine l’alloggiamento per

Tabella 1 - Parametri caratteristici del diagramma del ciclo di lettura.

ID.	SYMBOL	TIME	MIN	MAX	UNITS
2	T _{CSS}	CS setup time	25	-	ns
3	T _{CSH}	CS hold time	50	-	ns
4	T _{CSD}	CS disable time	25	-	ns
5	T _{SU}	Data setup time	10	-	ns
6	T _{HD}	Data hold time	10	-	ns
7	T _R	CLK rise time	-	20	ns
8	T _F	CLK fall time	-	20	ns
9	T _{HI}	Clock high time	25	-	ns
10	T _{LO}	Clock low time	25	-	ns
11	T _{CLD}	Clock delay time	25	-	ns
12	T _V	Output valid from clock low	-	25	ns
13	T _{HO}	Output hold time	0	-	ns
14	T _{DIS}	Output disable time	-	20	ns



SERIAL INPUT TIMING

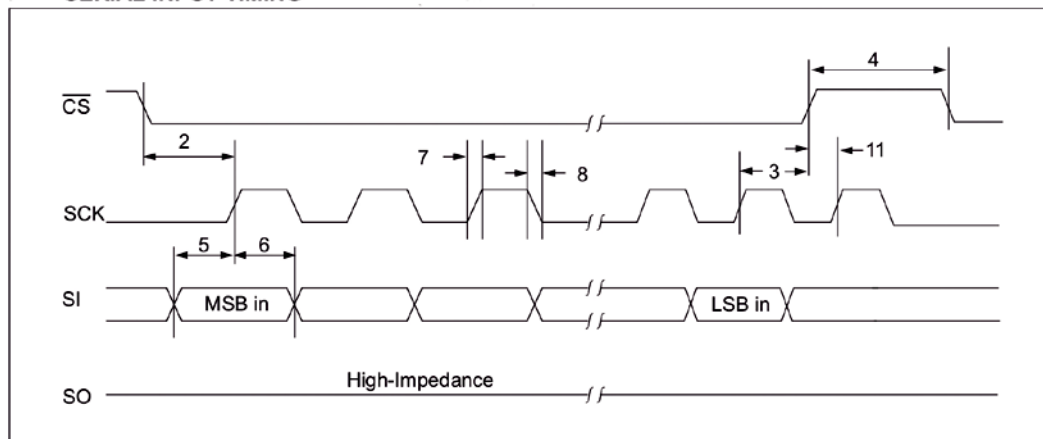


Fig. 3
Diagramma temporale del ciclo di scrittura.

SERIAL OUTPUT TIMING

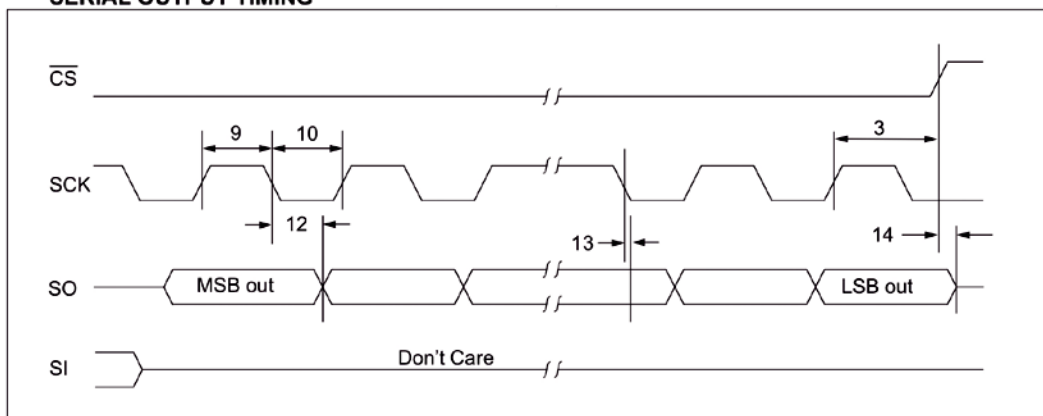


Fig. 4
Diagramma temporale del ciclo di lettura.

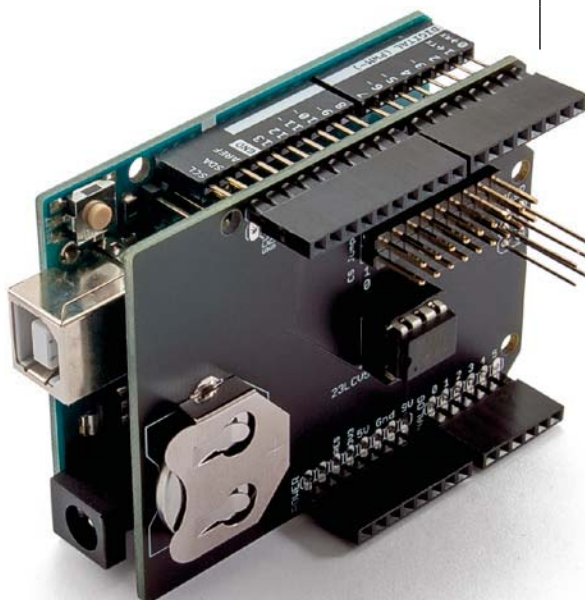
una comune batteria a bottone di tipo CR2032 (si rammenta che la batteria va inserita con il simbolo del + verso l'alto ed il lato con il - verso il circuito stampato). Come già detto, la comunicazione con il bus SPI avviene tramite il connettore ICSP a 6 pin così da permetterne il funzionamento sia su Arduino UNO/Leonardo che su Arduino MEGA.

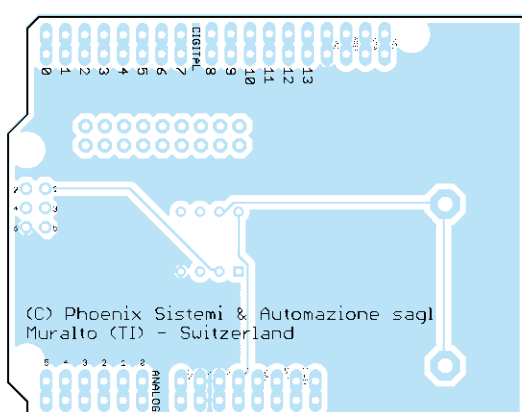
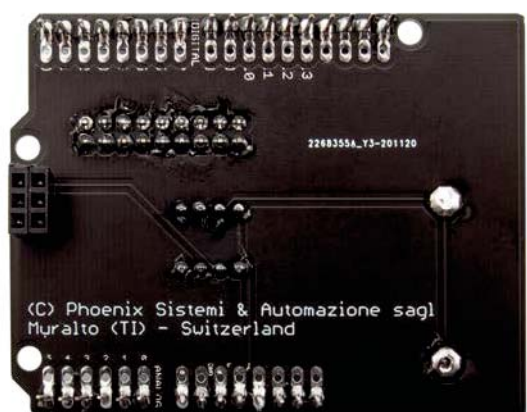
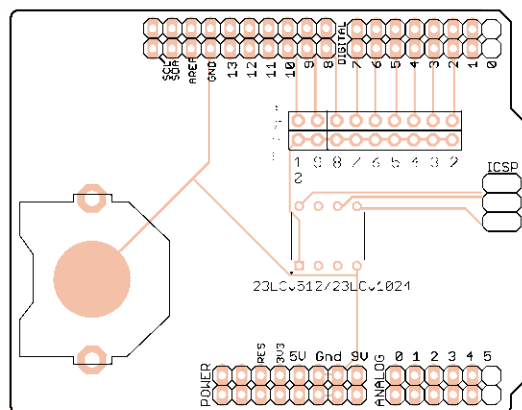
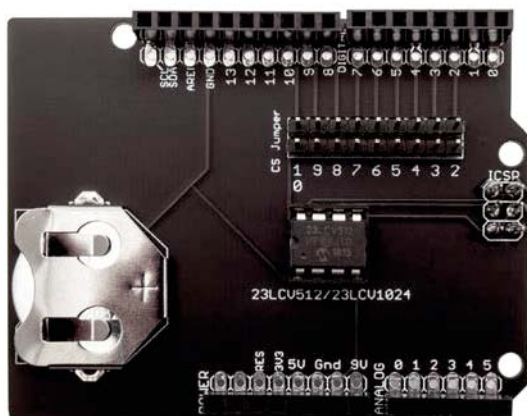
SCHEMA ELETTRICO E REALIZZAZIONE

Altrettanto semplice è lo schema elettrico di detta scheda, che descriviamo qui di seguito; in esso sono evidenti i collegamenti con la batteria di backup, con il connettore ICSP e il jumper per la selezione del pin che controlla il CS.

Per l'utilizzo di tale board, si è scritta una libreria che, per facilitarne l'uso, ricalca, per quanto possibile, le funzioni della libreria EEPROM standard di Arduino. La prima cosa da fare è decidere se si intende utilizzare una nvSRAM da 512 kbit o da 1.024 kb. Per ridurre al minimo l'occupazione della libreria, questa cosa è decisa con l'utilizzo di una *#define* all'interno del sorgente della libreria, quindi,

Fig. 5
Lo shield nvSRAM montato su Arduino UNO.





Elenco Componenti:

U1: 23LCV512-I/P

Varie:

- zoccolo 4x4

- strip maschio 9 poli
- strip maschio-femmine 3x2 (ICSP)
- strip per arduino 6 poli

- strip per arduino 8 poli
- strip per arduino 10 poli
- portabatterie per CR2032
- Circuito stampato S1546 (68x54)

occorrerà aprire il file `src/NVSRAM.h` e scegliere il tipo di memoria che si intende usare.

In pratica, occorrerà andare alle linee numero 36 e 37 del sorgente (linee relative alla versione 1.0) e togliere il commento davanti alla riga che indica la memoria scelta e metterlo davanti a quella non scelta. Ad esempio, per utilizzare le nvSRAM da 512 kbit dovremo avere il codice nel seguente modo:

```
#define NVSRAM_512 // #define NVSRAM_1024
```

Fatto questo, potremo includere nel nostro sorgente `.ino` la libreria con la riga:

```
#include <NVSRAM.h>
```

e quindi istanziare la classe con la riga:

```
NVSRAM myNVSRAM( 10 );
```

dove, tra parentesi metteremo il numero di pin Arduino che intendiamo usare come pin di CS. L'istanza nasconde anche un secondo parametro, di tipo bool e che è di valore "true". Tale parametro indica se la libreria deve o meno inizializzare il bus SPI. Nel caso si volesse fare manualmente l'inizializzazione di detto bus e non automaticamente da parte della libreria, occorrerebbe istanziare la classe fornendo anche il secondo parametro:

```
NVSRAM myNVSRAM( 10, false );
```

▼ Listato 1

```
/*
Sample usage of NVSRAM library with different Arduino data types.
Guglielmo Braguglia - Dec. 2020
Please, verify that you have selected the correct memory chip
you are using, inside the library.
*/
#include <NVSRAM.h>
/*
Leave the following #define uncommented if you want to do a complete
WRITE/READ BACK cycle. Comment if you want only to do a READ BACK cycle
to verify battery data retention
*/
#define WR_RD
/*
Leave the following #define uncommented if you want to use the automatic
default SPI bus initialization. Comment if you want to use a manual SPI
bus initialization with specific parameters
*/
#define SPI_INIT
/*
-----
*/
#ifdef SPI_INIT
NVSRAM myNVSRAM ( 10 );
#else
NVSRAM myNVSRAM ( 10, false );
#endif
void setup ( void ) {
/*
Declares different types of variables/structure to save/restore
Address is the NVSRAM memory address used to store data
b (one byte) start at address 0x0000
i (two bytes) start at address 0x0001
f (four bytes) start at address 0x0003
a (ten bytes) start at address 0x0007
s (17 bytes) start at address 0x0011
*/
byte b = 0;
int i = 0;
float f = 0.0;
byte a[10] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
struct myStruct {
byte by = 0;
int in = 0;
float fl = 0.0;
byte ar[10] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
} s;
unsigned long tStart = 0, tEnd = 0;
unsigned int block_crc = 0;
byte j;
delay ( 500 );
Serial.begin ( 115200 );
#ifndef SPI_INIT
SPI.begin();
SPI.beginTransaction(SPISettings(8000000, MSBFIRST, SPI_MODE0));
myNVSRAM.begin();
#endif
#ifdef WR_RD
Serial.println ( "Demo program WRITE/READ begins ..." );
/*
Erase all the NVSRAM memory
*/
tStart = micros();
myNVSRAM.erase();
tEnd = micros();
Serial.print ( "Memory erased in " );
Serial.print ( tEnd - tStart );
Serial.print ( " µsec; " );
Serial.print ( "Bytes erased: " );
Serial.print ( myNVSRAM.length() );
Serial.print ( ", µsec per byte≈ " );
Serial.println ( ( tEnd - tStart ) / myNVSRAM.length() );
*/
*/

```


Listato 1 - Segue

```
Serial.println( );
#else
Serial.println ( "Demo program READ ONLY begins ..." );
#endif
#ifdef WR_RD
b = 127;
myNVSRAM.write ( 0x0000, b );
#endif
b = myNVSRAM.read ( 0x0000 );
Serial.print ( "Byte b reading with read (expected 127): " );
Serial.println ( b );
b = myNVSRAM[ 0 ];
Serial.print ( "Byte b reading with [] (expected 127): " );
Serial.println ( b );
#ifdef WR_RD
i = -127;
myNVSRAM.put ( 0x0001, i );
#endif
myNVSRAM.get ( 0x0001, i );
Serial.print ( "Integer i reading with get (expected -127): " );
Serial.println ( i );
#ifdef WR_RD
f = 127.12;
myNVSRAM.put ( 0x0003, f );
#endif
myNVSRAM.get ( 0x0003, f );
Serial.print ( "Float f reading with get (expected 127.12): " );
Serial.println ( f );
#ifdef WR_RD
for ( j = 0; j < 10; j++ ) a[j] = j;
myNVSRAM.put ( 0x0007, a );
#endif
myNVSRAM.get ( 0x0007, a );
Serial.print ( "Array a reading with get (expected 0..9): " );
for ( j = 0; j < 10; j++ ) {
Serial.print ( a[j] );
if ( j < 9 ) Serial.print ( ", " );
}
Serial.println ( );
#ifdef WR_RD
s.by = 127;
s.in = -127;
s.fl = 127.12;
for ( j = 0; j < 10; j++ ) s.ar[j] = j;
myNVSRAM.put ( 0x0011, s );
#endif
myNVSRAM.get ( 0x0011, s );
Serial.println ( "Structure s reading with get ..." );
Serial.print ( " - byte by value (expected 127): " );
Serial.println ( s.by );
Serial.print ( " - integer in value (expected -127): " );
Serial.println ( s.in );
Serial.print ( " - float fl value (expected 127.12): " );
Serial.println ( s.fl );
Serial.print ( " - byte array ar value (expected 0..9): " );

for ( j = 0; j < 10; j++ ) {
Serial.print ( s.ar[j] );
if ( j < 9 ) Serial.print ( ", " );
}
Serial.println ( );
Serial.println ( );
block_crc = myNVSRAM.crc ( 0, 34 );
Serial.print ( "CRC16 of first 34 bytes (expected 12711): " );
Serial.println ( block_crc );
Serial.println ( );
Serial.println ( "Test completed." );
#ifndef SPI_INIT
SPI.endTransaction();
#endif
}
void loop ( void ) {
}
```



Fatto questo, tutto è pronto per utilizzare, con una sintassi praticamente uguale a quella della libreria standard EEPROM, la nostra nvSRAM.

Avremo quindi a disposizione i metodi *write()*, *update()*, *read()*, *get()* e *put()* con la stessa sintassi della classe EEPROM (e comunque documentati nel readMe che accompagna la libreria) oltre ad alcuni altri metodi/operatori meno conosciuti e meno utilizzati, alcuni anche loro presenti nella libreria EEPROM ed altri non presenti.

Tra i meno conosciuti troviamo il metodo *length()* che permette di conoscere le dimensioni massime della nostra nvSRAM, che ritorna un "unsigned long" (uint32_t) e la cui sintassi è:

```
myLong = myNVSAM.length( );
```

e l'operatore **[]** che permettere di vedere, in lettura, la nostra nvSRAM come fosse un array, dando direttamente come indice il byte che si intende leggere:

```
myByte = myNVSAM [ myAddress ];
```

Tra i non presenti troviamo il metodo *erase()* che permette, in un solo colpo, di inizializzare tutte le celle della nostra nvSRAM al valore 0x00 e la cui sintassi è:

```
myNVSAM.erase( );
```

Il metodo *begin()*, che va obbligatoriamente utilizzato solo nel caso si scelga l'inizializzazione manuale del bus SPI (ad esempio perché già inizializzato in precedenza o per inizializzarlo con parametri diversi da quello predefiniti):

```
myNVSAM.begin( );
```

ed il metodo *crc()*, che permette di calcolare il crc16 di un blocco di nvSRAM (ad esempio per essere sicuri, alla riaccensione, che non ci siano state modifiche non volute o errori) e che ha la seguente sintassi:

```
myCRC = myNVSAM.crc( myAddress, myLength );
```

Rammentiamo infine che, se per le nvSRAM da 512 kbit, la variabile che viene usata per passare un indirizzo o quella che viene usata per recuperare le dimensioni massime (in bytes) della memoria, può essere dichiarata come "unsigned int" (uint16_t), per le nvSRAM da 1.024kbit tali

variabili devono essere obbligatoriamente degli "unsigned long" (uint32_t) dato che tali valori sono a 24 bit (gli 8 bit più significativi sono garantiti zero).

La libreria può essere scaricata da Internet dal seguente indirizzo:

<https://github.com/gpb01/NVSAM>

Concludiamo vedendo un semplice programma dimostrativo (**Listato 1**) che ci permette di provare tutte le varie funzionalità (programma che si trova nella cartella "examples" della libreria nvSRAM).

La *#define WR_RD* ci permette di verificare il buon funzionamento della nostra scheda. Se tale *#define NON* è commentata, viene eseguito il ciclo completo di pulizia, scrittura e riletture di più variabili nella nvSRAM, se invece tale viene commentata, sia la fase di pulizia che di scrittura viene saltata e viene effettuata solo la fase di lettura delle variabili.

In questo modo si può, una prima volta fare il ciclo completo, poi riprogrammare la scheda con la *#define* commentata, quindi togliere l'alimentazione e, al suo ripristino, verificare che il programma legga correttamente le variabili salvate in nvSRAM. La *#define SPI_INIT* ci permette di provare sia la modalità con inizializzazione automatica del bus SPI con i valori predefiniti che la modalità con inizializzazione manuale e parametri del bus SPI impostati da programma.

Se tale *#define* non è commentata, viene eseguita l'inizializzazione automatica, se invece è commentata, viene eseguita una inizializzazione nel *setup()* del bus SPI con l'utilizzo di parametri personalizzati. □



Cosa occorre?

La memoria da 512Kbit 23LCV512-I/P (cod. 23LCV512IP) è in vendita a Euro 3,90, la memoria da 1024Kbit 23LCV1024-I/P (cod. 23LCV1024IP) costa Euro 4,90. La batteria CR2032 in confezioni da 5 pezzi (cod. CR2032-5) è disponibile a Euro 2,50. Il PCB può essere realizzato utilizzando il nuovo servizio di stampa circuiti stampati (cod. PCBPRODUCTION) direttamente 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>



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

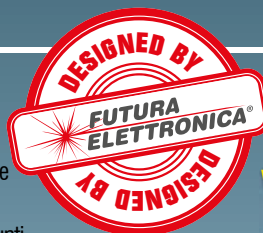


- ✓ Struttura in alluminio per combinare rigidità e leggerezza
- ✓ Assemblaggio semplificato con giunti metallici e parti in ABS stampate a iniezione
- ✓ Piano di stampa con movimentazione X/Y
- ✓ Estrusore movimentato sul solo asse Z
- ✓ Volume di stampa massimo: 20 x 20 x 20 cm
- ✓ Risoluzione: X e 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

€ **499,00**

Cod. **3DRAG/K**

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

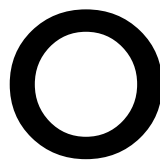


MONITORAGGIO AMBIENTALE con uChip

dell'ING. GIOVANNI CARRERA



Dotato di scheda SD ed RTC, rileva e memorizza temperatura, umidità e pressione barometrica mediante un sensore combinato, mostrando i dati graficamente su display touch.



ggigiorno, anche e soprattutto nel monitoraggio ambientale, ci si orienta verso soluzioni connesse e sempre più integrate nel concetto di IoT, ma non in tutte le applicazioni è possibile o conveniente disporre della necessaria connessione Internet. In casi del genere è possibile salvare i dati

desunti dai sensori e le relative misure su un supporto non volatile come una scheda SD, per poi leggerle ed elaborarle su un Personal Computer. Il progetto qui proposto risponde a tale esigenza applicativa, ma non è semplicemente un *data logger* di dati ambientali, bensì un completo sistema di monitoraggio locale dotato di uscite grafiche in grado di farci capire l'andamento delle variabili osservate, le quali, in questo caso, sono la temperatura, la pressione atmosferica e l'umidità relativa. La rappresentazione grafica è quella tipica dei monitor usati in ambiente medico, con scorrimen-

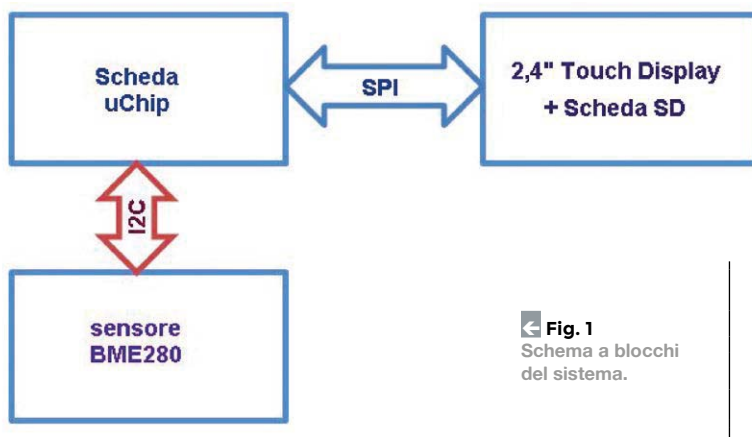


Fig. 1
Schema a blocchi
del sistema.

to da destra a sinistra in modo da visualizzare sempre le ultime misure.
Per il nostro progetto utilizziamo una configurazione hardware descritta dallo schema a blocchi mostrato nella **Fig. 1**. Il cuore del sistema è uChip, un modulo poco più grande di un circuito integrato DIP a 16 pin, dotato di un microcontrollore a 32 bit. Il display TFT, il *touch controller* e la scheda SD sono collegati via bus SPI, mentre il sensore utilizza l'interfaccia I²C-Bus. Per future applicazioni è disponibile anche un'interfaccia seriale, due I/O digitali o tre ingressi analogici o un'uscita analogica. Questo sistema si presta, quindi, per sviluppare numerose altre applicazioni. Sulla scheda SD incorporata nel display, vengono salvati i campionamenti delle tre variabili insieme al *timestamp*, o marca temporale, che permette di elaborare facilmente i dati sul calcolatore. L'hardware del sistema, con modeste modifiche del software, si presta anche al monitoraggio di segnali analogici.

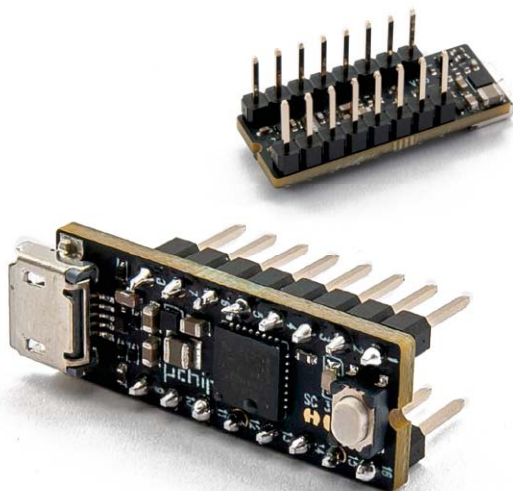


Fig. 2
La scheda uChip.

IL SISTEMA UCHIP

Questa minuscola scheda, visibile in **Fig. 2**, con dimensioni di soli 28,5x10,16 mm, compreso il connettore USB, è prodotta in Italia e commercializzata da Futura Elettronica (futura-shop.it). Essa è basata sulla MCU Cortex M0 SAMD21-E18A, a 32 bit con memoria Flash da 256 kB (248 kB utili, i restanti occupati dal bootloader integrato) e RAM statica da 32 kB ed è compatibile con Arduino Zero. Ha due connettori strip da 8 pin con passo identico a un chip DIL16. In **Fig. 2** sono visibili entrambe le facce della scheda. Uchip può essere alimentata da 3,3V a 5V sul pin 16 (VEXT) o a 5V dalla presa micro USB e ha un sistema di alimentazione molto sofisticato, come si vede nello schema funzionale di **Fig. 3**.

Sono presenti uno switching Boost da 500 mA (Step-Up: converte i 3,3V da VEXT in 5V su USB) e un Buck converter da 1 A (Step-Down: da 5V USB a 3,3V su VEXT), inoltre ha anche un regolatore lineare LDO (*Low Drop-Out*: bassa caduta di tensione ingresso-uscita).

Quest'ultimo regolatore, un NCP114 a 3,3V con tensione di dropout di 135 mV a 300 mA, alimenta la MCU e rende disponibili 3,3V 100 mA sul pin 15 mediante ponticello a saldare. La tensione VEXT non deve superare il valore di 5,5V, valore limite del regolatore LDO.

Si può alimentare la scheda dalla porta USB a 5V, usando un normale alimentatore da cellulare, oppure applicare una tensione da 3,3V a 5V sul pin 16 (VEXT). Nel primo caso, abilitando il regolatore Buck settando VEXT_SELECT (PA15), uChip alimenta altri dispositivi esterni a 3,3V fino a 1A dal pin 16 che diventa un'uscita di alimentazione. Nel secondo caso si può collegare un alimentatore da 3,3V a 5V su VEXT o una batteria al litio di opportuna capacità al pin 16 che ora diventa un ingresso. Un diodo ideale, realizzato con un MOSFET P a bassa resistenza di conduzione, isola l'uscita del Buck. Nel caso si utilizzi la porta USB come host, essa è alimentata dal Boost, attivato automaticamente dal pin ID del connettore OTG. Il Boost è anche attivato o disattivato con il BOOST_ENABLE (PA14).

Nel caso in cui si voglia ridurre il rumore generato dagli alimentatori switching, occorre disabilitare entrambi i convertitori Boost e Buck in questo caso, la MPU non è più alimentata dalla USB ma dal pin 16 (VEXT), tensione che non deve superare i +5,5V. Durante lo sviluppo del software, il Buck è abilitato dalla presenza della tensione USB. I due alimentatori switching sono controllati via software usando

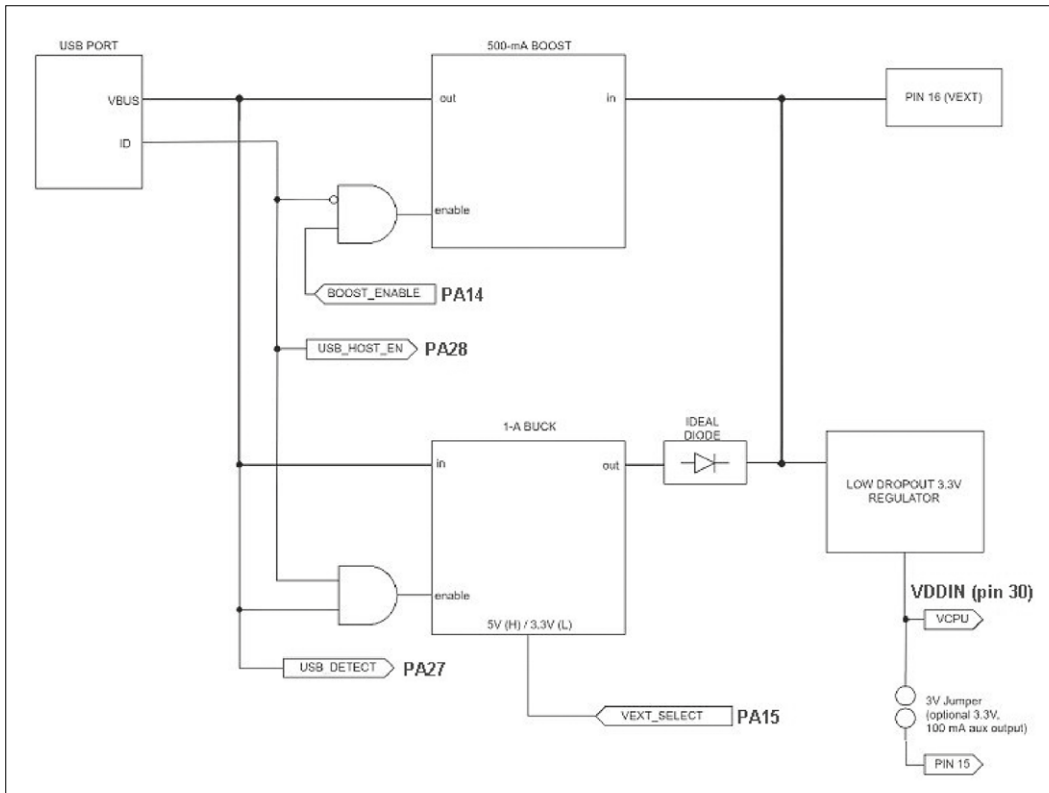


Fig. 3
L'alimentazione
della scheda
uChip.

la specifica libreria o intervenendo sui pin di controllo. In **Fig. 4** è visibile la pin-out della scheda. In **Tabella 1** sono indicati i pin disponibili di uChip e quelli usati dal progetto.

IL DISPLAY

Il display è un TFT touch da 320x240 pixel da 2,4 pollici con 18-bit per 262k colori con codifica

a 6 bit per colore. Il chip di controllo usato per questo schermo è un ILI9341 mentre il *Touch Screen Controller* è un XPT2046. Le immagini di **Fig. 5** mostrano la scheda display usata; come si vede, da un lato ha un connettore da 14 pin per il display e per il *touch controller*, dal lato opposto uno da 4 pin per la SD-Card. In **Fig. 6** è mostrato lo schema di questa scheda, mentre in **Tabella 2**

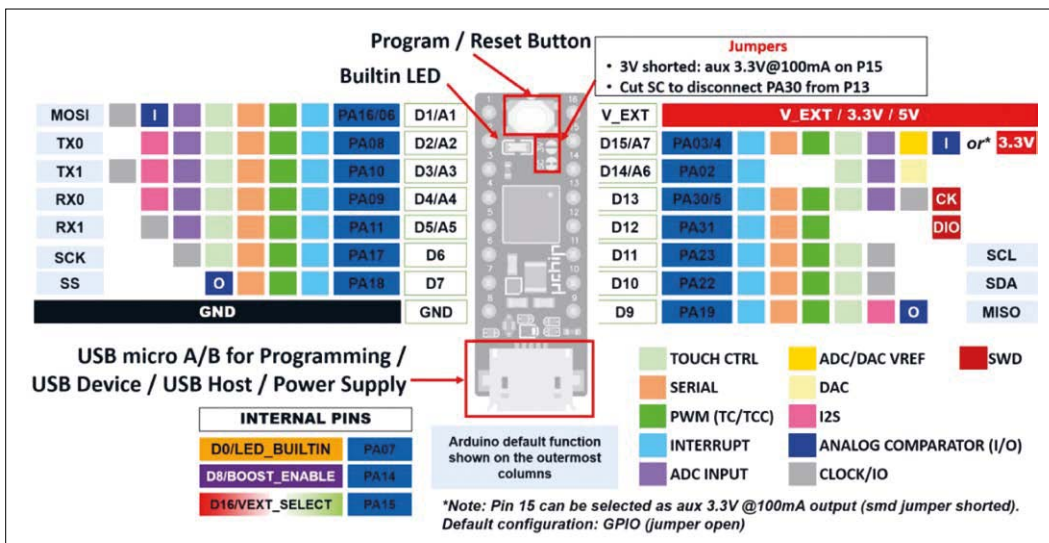


Fig. 4
Pinout di
uChip.

Fig. 5
Il display TFT.

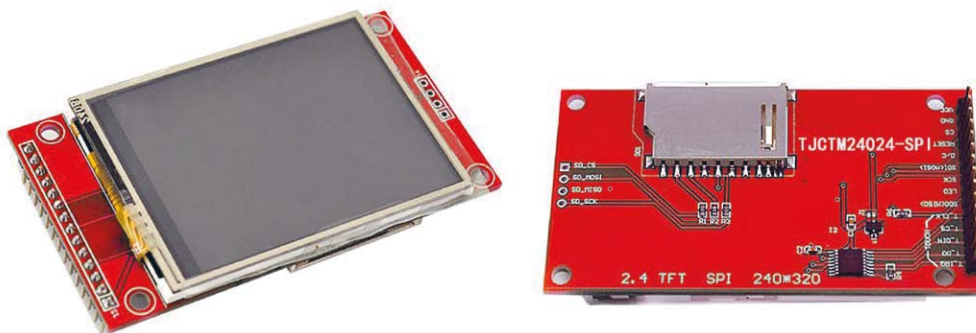


Tabella 1
I pin di uChip usati nel progetto.

PIN	NOME	USO
1	D1/A1	MOSI SD&TFT
2	D2/A2	Touch CS
3	D3/A3	Tx1 Serial1
4	D4/A4	SD_CS
5	D5/A5	Rx1 Serial1
6	D6	SCK SD&TFT
7	D7	TFT_CS
8	GND	GND
9	D9	MISO_SD
10	D10	BME_SDA
11	D11	BME_SCL
12	D12	TFT D/C
13	D13	Touch IRQ
14	D14/A6	Analog I/O
15	D15/A7	TFT backlight
16	VEXT	+3.3Vout

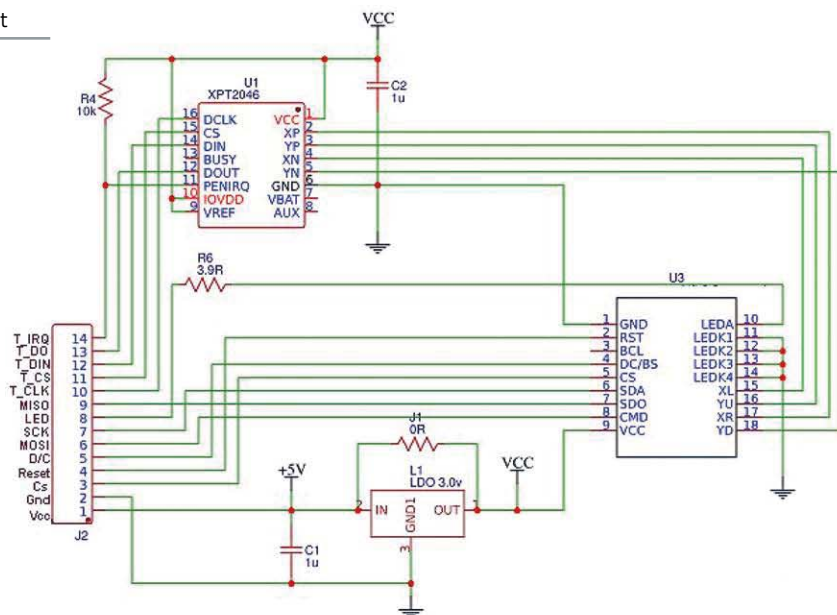
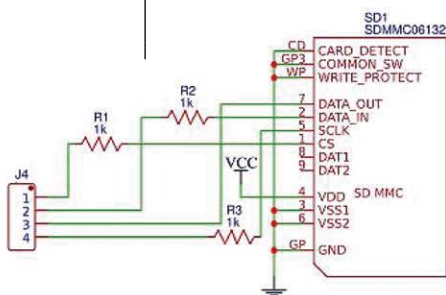
si vedono i collegamenti con uChip realizzati in questo progetto. I segnali della scheda sono TTL a 3,3V e questo non è un problema per uChip che lavora a questa tensione. Il display può essere comunque alimentato a 5V grazie ad un regolatore LDO che fornisce l'esatta tensione per alimentare il TFT e la SD. Per ridurre i pin di I/O e per ottimizzare l'interfaccia, sono stati uniti in un unico bus SPI il controller grafico del display, la SD e il *touch controller*. Per risparmiare pin, si è anche rinunciato al reset pilotato dalla MCU collegandolo a un RC (47kΩ e 100nF) sul pin 4 del TFT.

Per saperne di più sul display touch utilizzato si può consultare il documento "a-Si TFT LCD Single Chip Driver 240RGBx320 Resolution and 262K color Specification" della ILI TECHNOLOGY CORP, reperibile in Rete su vari siti ed eventualmente anche su www.ilitek.com.

IL SENSORE BME280

Il BME280 è un sensore della Bosch combinato

Fig. 6
Schema della scheda display.



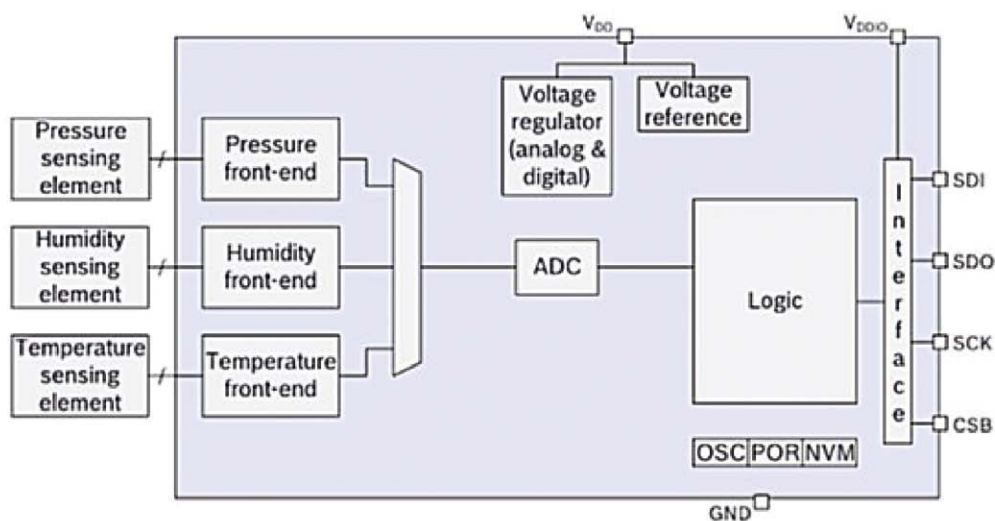


Fig. 7
Schema
funzionale
del sensore
BME280.

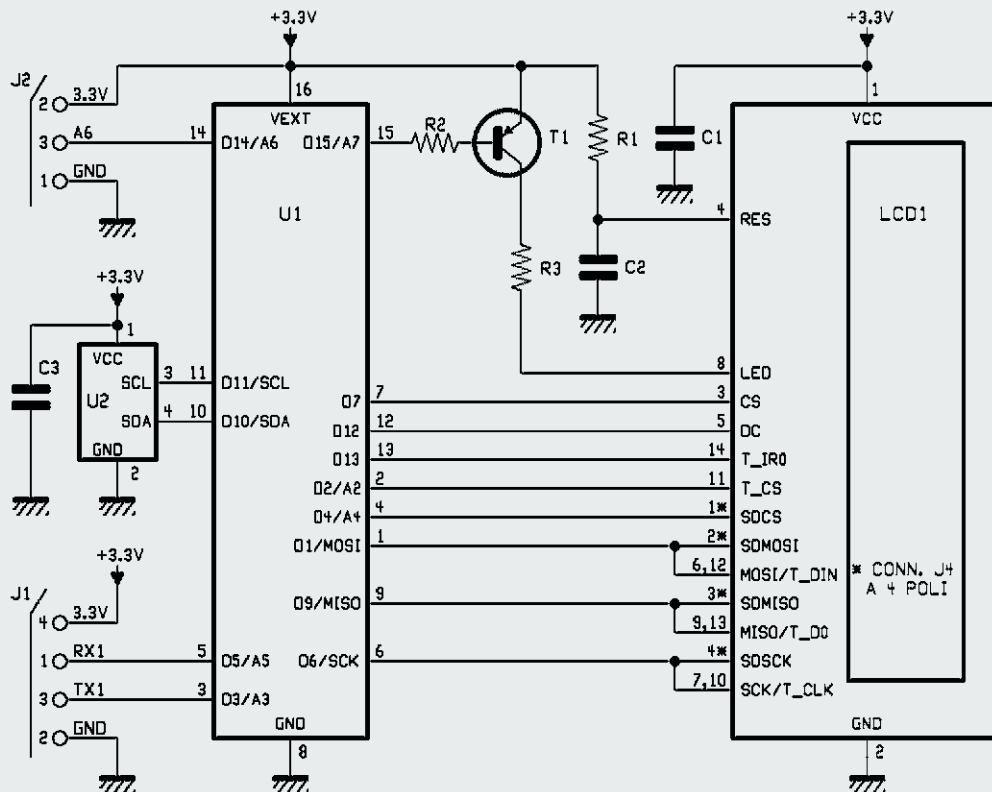
di temperatura, pressione barometrica e umidità relativa della Bosch, incapsulato in un minuscolo case metallico di soli 2,5x2,5x0,93 mm. Inoltre consuma pochissimo, per cui è particolarmente indicato per telefoni cellulari, ricevitori GPS od orologi.

Ha un'accuratezza sulla pressione di $\pm 0,12$ hPa che è equivalente a circa ± 1 m come differenza di altitudine, con una deriva termica di solo

12,6 cm/°C. Il sensore di temperatura ha una risoluzione di 0,01 °C e un'accuratezza di $\pm 0,5$ °C a 25 °C; quanto al sensore di umidità relativa, è dichiarata un'accuratezza di ± 3 %RH a 25 °C. Il sensore ha la stessa piedinatura del popolare BMP280, che è però privo del sensore di umidità. Esso è alimentabile con una tensione continua compresa fra 1,7 e 3,6V; nel nostro caso è alimentato a 3,3V dal regolatore LDO (*low drop-out*)

Tabella 2
Collegamenti del display.

J2 PIN	NOME	FUNZIONE	PIN uChip	NOME
1	Vcc	+3.3 .. +5v	16	
2	Gnd	Gnd	8	
3	Cs	TFT CS	7	D7
4	Reset	TFT Reset	--	
5	D/C	TFT D/C	12	D12
6	MOSI	TFT&Touch&SD MOSI	1	D1/MOSI
7	SCK	TFT&Touch&SD SCK	6	D6/SCK
8	LED	TFT LED	15	D15
9	MISO	TFT&Touch&SD MISO	9	D9/MISO
10	T_CLK	Touch CLK	6	D6/SCK
11	T_CS	Touch CS	2	D2
12	T_DIN	Touch DIN	1	D1/MOSI
13	T_DO	Touch DO	9	D9/MISO
14	T_IRQ	Touch IRQ	13	D13



interno al modulo (alla breakout board sulla quale si trova commercialmente disponibile il sensore). La **Fig. 7** mostra lo schema funzionale del sensore. Tutti i dettagli sul sensore Bosch si trovano nel documento tecnico BME280 "Combined humidity and pressure sensor" reperibile in Rete.

Il componente ha due distinti pin di alimentazione: Vdd, alimentazione principale dei circuiti analogici e digitali e Vddio, alimentazione dell'interfaccia digitale di I/O. Nella maggior parte dei casi esse sono collegate insieme, come nel caso della breakout board da noi utilizzata nel progetto, visibile in **Fig. 8**.

In **Fig. 9** è stato ricavato lo schema di questa scheda. Essa incorpora un regolatore LDO da 3,3V e un adattatore di livello, necessario per collegarlo a un Arduino Uno o altri microcontrollori alimentati a 5V, nel caso di uChip i segnali devono essere a 3,3V e questi componenti sono del tutto inutili. Questa scheda, avendo solo 4 pin, prevede solo l'interfaccia I²C (CSB=1). Il pin SDO permette di cambiare l'indirizzo (LSB) che per default è SDO=0 (ponticello a massa) per cui l'indirizzo è 1110110 (0x76). Se SDO = 1, operazione che richiede il taglio del ponticello e il collegamento a Vdd, l'indirizzo diventa 0x77, operazione non necessaria in questo caso. La libreria Adafruit BME280 con interfaccia I²C software richiede l'assegnazione dei pin usati per il bus con la funzione **Wire.begin (SDA, SCL)**.

SCHEMA ELETTRICO

Bene, ora che sono stati analizzati i singoli moduli componenti il sistema passiamo a vedere lo schema elettrico del complesso, ovvero della scheda da noi prevista per ospitare la scheda display, la breakout board sensore e la board uChip. È stata prevista anche una interfaccia seriale e un I/O analogico sul pin A6.

Fig. 8
La scheda
BME280.



Per spegnere il display è utilizzato un transistor PNP a bassa V_{CE} di saturazione, la scheda ha già un resistore limitatore $R6 = 3,9\Omega$ (vedere **Fig. 6**) per i quattro led di retroilluminazione e, per ridurre ulteriormente i consumi, si è aggiunto un altro resistore in serie. Quindi la retroilluminazione è comandata da un segnale di I/O (ON = LOW). Si può abilitare l'alimentatore Buck usando la funzione della libreria

uChipSetVextValue(VEXT_3V3);

Per questo sistema si consiglia l'alimentazione da connettore micro USB, questo per ridurre il consumo: utilizzando lo switching Buck di uChip si ottiene un minore consumo, ulteriormente ridotto spegnendo la retro-illuminazione del display nel normale funzionamento per accenderla solo quando deve interagire con l'operatore. Sul pin 4 del connettore J1 dell'interfaccia seriale è possibile alimentare il sistema anche con una tensione VEXT da 3,3V a 5V. Per tensioni maggiori di 3,3V il display apparirà più luminoso. Si può usare anche una batteria agli ioni di litio (3,7V tipici). I consumi aumentano sensibilmente in quanto non è usato il regolatore Buck e il display consuma di più, per cui sarebbe da aumentare la resistenza di $R3$. Alimentando da USB,

i consumi medi del sistema sono molto bassi: circa 12 mA senza retro-illuminazione e 38 mA quando si accende, ossia quando si interviene sul touch display per immettere i parametri o leggere le misure. Il display è spento automaticamente a ogni campionamento e dopo 5 secondi di inattività.

Per future applicazioni sono presenti il connettore J1 per utilizzare la Serial1 per connettersi ad altri dispositivi, come un ricevitore GPS o altro; e anche il connettore J2, che permette di usare l'ingresso analogico A6 a 12 bit o l'uscita del DAC a 10 bit. Questi due connettori, con opportuna inizializzazione di I/O e conseguenti modifiche al programma, permettono anche di monitorare tre segnali analogici A6 (J2), A3 e A5 (J1).

REALIZZAZIONE PRATICA

Per realizzare il sistema abbiamo progettato un circuito stampato a doppia faccia (nel nostro prototipo completo di serigrafia) i cui file gerber da cui partire per prepararlo sono scaricabili dal nostro sito (www.elettronica.in.it) insieme ai file del progetto.

Una volta inciso e forato il PCB vi si possono montare i componenti del caso, tutti discreti, seguendo la disposizione proposta nel piano di

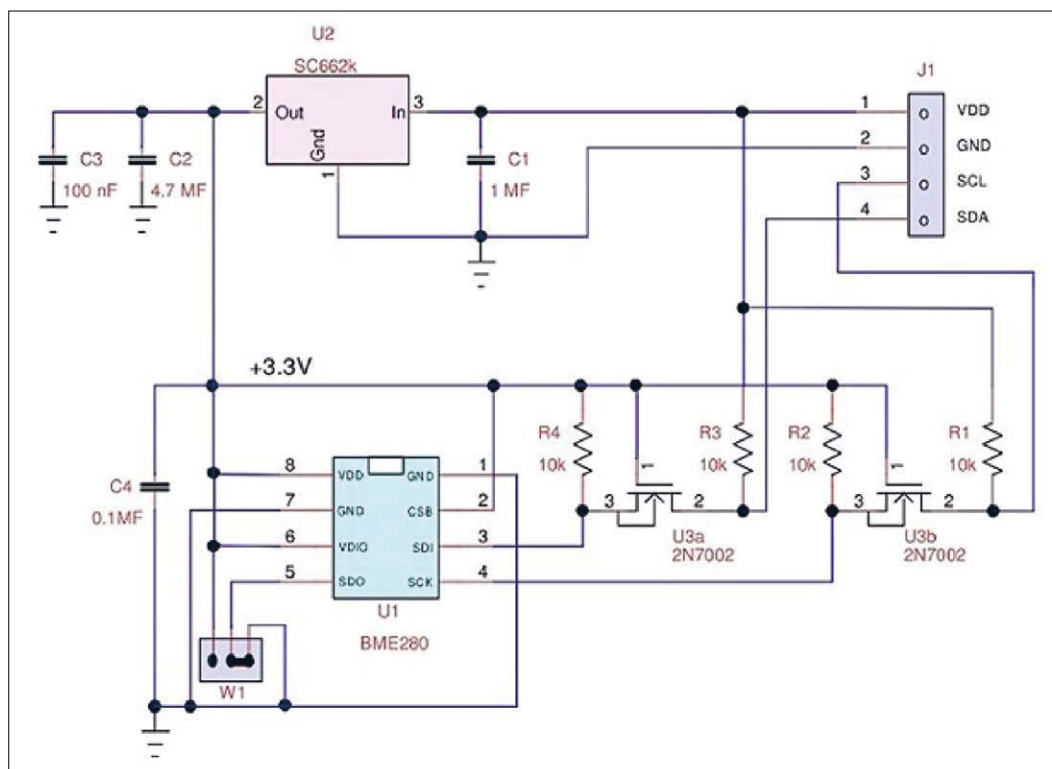


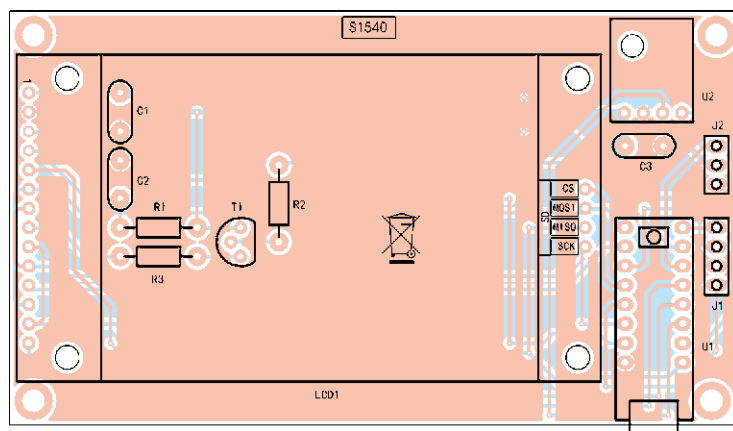
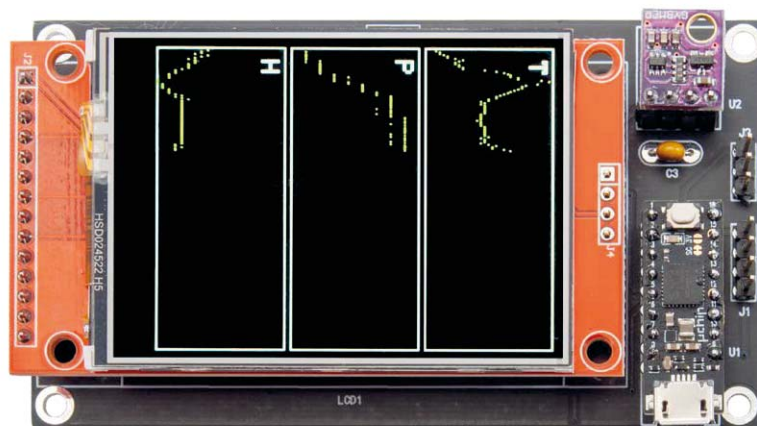
Fig. 9
Schema
della scheda
BME280.

Elenco Componenti:

R1: 47 kohm
 R2: 1 kohm
 R3: 6,8 ohm
 C1,C3: 100 nF
 C2: 100 nF 63V poliestere
 T1: BC547
 U1: uCHIP
 U2: BME280
 LCD1: display touchscreen 240x320 2,4"
 J1: strip maschio 4 poli
 J2: strip maschio 3 poli

Varie:

- strip femmina 4 poli
- strip femmina 8 poli
- strip femmina 14 poli
- Circuito stampato S1540 (96x54)



montaggio visibile in queste pagine; l'ordine di montaggio è il solito, vale a dire prima quelli a più basso profilo come le resistenze e i condensatori, poi il transistor PNP T1, da orientare esattamente come indicato nel piano di montaggio. Infine si inseriscono e si saldano le file di pin-strip necessarie a comporre gli zoccoli per ospitare la scheda display, la breakout board sensore e la scheda uChip, tutte da orientare come mostrato nel piano di montaggio e comunque per ottenere il risultato visibile nella **Fig. 11**. Oltre ai due connettori J1 e J2, non necessari per questa ma per future applicazioni, occorre quindi saldare file di strip femmina da 14 + 4 pin per il display, da 4 pin per il sensore e da 8 + 8 pin per la uChip.

L'OROLOGIO RTC

La MCU SAMD21 ha incorporato un valido RTC (*Real Time Clock*) e uChip usa un quarzo da 32.768 Hz (FK135EIH0.032768) molto preciso con una tolleranza di ± 20 ppm e una stabilità in frequenza di -0.04 ppm/ $\Delta^\circ\text{C}$. Unico inconveniente è che, togliendo l'alimentazione si perde

l'ora impostata, ma questo si può superare con qualche artificio, come si vedrà più avanti.

ALIMENTAZIONE DEL SISTEMA

Per alimentare il tutto ci si può rivolgere a una batteria o ad un alimentatore da rete; il metodo più semplice e capace di fornire autonomia operativa al sistema è l'utilizzo di una batteria al piombo da 12V con un carica cellulare da auto che contiene uno switching step-down da 12/24V a 5V con connettore micro USB per uChip. L'uso di un alimentatore switching, che ha un alto rendimento, permette di ridurre ulteriormente il già basso consumo e aumentare notevolmente l'autonomia.

In alternativa, si potrebbe usare una batteria agli ioni di litio (3,7V) sul pin VEXT, accessibile sui connettori J1 e J2. In questo caso, come si è già detto, i consumi aumentano.

Dove invece è disponibile la tensione di rete, si può ricorrere a un normale alimentatore da 5V per cellulari con connettore micro USB; ciò comporta però l'inconveniente di perdere l'orologio in caso

di black-out anche brevi. Il sistema può essere perfezionato usando due semplici pile da 1,5V collegate con un diodo in serie sulla VEXT. Nel caso di interruzione della rete luce le pile mantengono una tensione di circa 2,4V, bassa per il display ma sufficiente per far funzionare la MCU. Un sistema più complesso di alimentazione potrebbe essere quello di tenere in tampone la batteria al piombo, con un alimentatore stabilizzato da 13,5V. La batteria potrebbe essere anche di modesta capacità (1,2 Ah) garantendo la continuità di funzionamento anche per qualche giorno di black-out.

LE LIBRERIE USATE

Spostiamoci ora sugli aspetti firmware, partendo dalla descrizione delle librerie impiegate e, nello specifico, da quella per il display, il cui controller è un ILI9341; per esso sono disponibili diverse librerie Arduino e noi abbiamo usato la Adafruit_GFX (<https://github.com/adafruit/Adafruit-GFX-Library>), insieme alla Adafruit_ILI9341 (https://github.com/adafruit/Adafruit_ILI9341) scritte dalla brava Limor Fried (Lady Ada). La prima contiene tutte le primitive grafiche e la seconda le specifiche del controller usato dal display. Molto utile anche il manuale della GFX (adafruit-gfx-graphics-library.pdf).

Per il touch controller XPT2046 abbiamo utilizzato la libreria XPT2046_Touchscreen (https://github.com/PaulStoffregen/XPT2046_Touchscreen) di Paul Stoffregen che sfrutta il bus SPI hardware.

Anche per il sensore BME280 sono state utilizzate le librerie Adafruit: Adafruit_Sensor.h (https://github.com/adafruit/Adafruit_Sensor) e Adafruit_BME280.h (https://github.com/adafruit/Adafruit_BME280_Library/blob/master/Adafruit_BME280.h).

Quanto alla uChip, va scaricata la libreria del produttore della scheda (www.itaca-innovation.com), per caricare sulla IDE di Arduino la libreria basta seguire le indicazioni (in inglese) del manuale. Per Windows 10 non occorre scaricare il relativo driver, necessario per le versioni precedenti.

Per le modalità di installazione vedere la guida "uChip quick start manual", Itaca Innovation (l'ultima versione disponibile al momento della stesura di questo articolo è la uChip1.0.11).

FUNZIONAMENTO DEL SISTEMA

Vediamo ora il funzionamento complessivo del

sistema di monitoraggio ambientale, con il quale andremo a registrare temperatura, pressione e umidità relativa sulla scheda SD. L'orologio RTC è molto importante non solo per indicare a ogni misura il tempo esatto, ma anche per scandire il periodo di campionamento.

L'indicazione del tempo: il timestamp

Per avere una corretta collocazione temporale occorre accompagnare gli eventi con una marca che prende il nome di *timestamp*.

Noi umani siamo abituati a indicare il tempo come data e ora, espressi in modo non decimale e anche diversamente espresso in molte zone del mondo.

Questa marcatura del tempo risulta di difficile gestione se dobbiamo valutare differenze di tempo tra un evento e l'altro o creare grafici di serie temporali di campioni.

Nei sistemi operativi Unix e Unix-like il tempo viene rappresentato come secondi trascorsi dalle ore zero del primo gennaio 1970 e prende il nome di *epoca*.

L'epoca Unix è un numero di 32 bit con segno, e il 19 gennaio 2038 si raggiungerà l'*overflow* del contatore con problemi di tantissimi computer che usano un contatore a 32 bit.

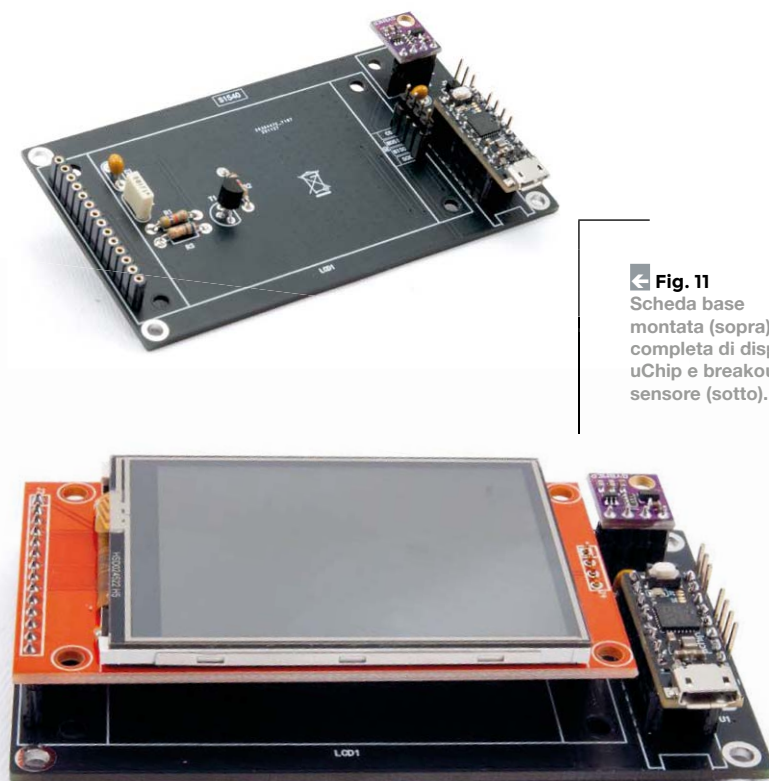


Fig. 11
Scheda base montata (sopra) e completa di display, uChip e breakout sensore (sotto).

**Tabella 3**

Esempio di elaborazione delle misure con foglio elettronico.

Julian	Seconds	T[C]	P [hP]	RH[%]	N. seriale	Giorno	Mese	Anno	Ora	Minuti	Secondi
44115	34800	27.0	933.0	43	44115.402778	11	10	2020	9	40	0
44115	35100	24.6	933.0	49	44115.406250	11	10	2020	9	45	0
44115	35400	23.3	932.9	52	44115.409722	11	10	2020	9	50	0

La libreria RTCLib ha la funzione `getEpoch()` che restituisce l'epoca come *unsigned long*, il cui overflow è invece il 07 Feb 2106 06:28:15.

Anche per una più facile leggibilità, in questo progetto si è preferito indicare il *timestamp* con due numeri interi (long) di massimo cinque cifre. Il primo numero rappresenta i giorni trascorsi dal 1° gennaio 1900, naturalmente esso tiene conto degli anni bisestili, il secondo numero indica i secondi a partire dalla mezzanotte. In questo modo è facile plottare la serie temporale delle misure ambientali.

Questo metodo è usato anche dai fogli elettronici come Excel, che ha come riferimento iniziale il 1° gennaio 1900 e non il 1970 mentre Matlab ha il 1° gennaio dell'anno zero, per passare da un sistema all'altro basta sommare i giorni di differenza.

Con la libreria RTCLib il giorno seriale si calcola facilmente dall'epoca, dividendola per i secondi nel giorno (86400) e prendendo la parte intera. Per avere come riferimento il 1° gennaio 1900 occorre sommare i giorni corrispondenti agli anni dal 1900 al 1970:

```
days1900= long(epoch/86400)+ 25569; //days elapsed
since 1/1/1900
```

Per quanto riguarda i secondi del giorno, basta trasformare ore, minuti e secondi in secondi che rappresentano la frazione del giorno:

```
secondspd= ho*3600 + mi*60 + se; //seconds elapsed
in the day
```

Il programma trasferisce questi due numeri interi (long) sulla SD insieme alla temperatura in [°C], la pressione barometrica in hectopascal [1 hP = 1 millibar] e all'umidità relativa in [%].

Questi cinque numeri sono separati da virgole. Il file ha nome fisso e le misure sono via via appese in coda.

Ecco come si presenta il file ASCII:

```
44115,34800,27.0,933.0,43
```

```
44115,35100,24.6,933.0,49
```

```
44115,35400,23.3,932.9,52
```

```
.....
```

Nel programma di elaborazione su PC si sono espressi i secondi in frazione di giorno = secondi/86400 sommandoli poi ai giorni.

Si sono elaborati i dati in Matlab, ma si possono elaborare anche con un foglio elettronico. In Excel la funzione `DATA.VALORE("11/09/2020")` restituisce il numero seriale 44085, la data è passata come stringa. Dato il numero seriale, la funzione inversa `GIORNO(44085)` restituisce il numero 11, `MESE(44085)` il numero 9 e `ANNO(44085)` il numero 2020. Se si inserisce anche la frazione di giorno, si possono ricavare anche l'ora, i minuti e i secondi. La **Tabella 3** ne è un esempio:

Basta importarli come file testo con la virgola come separatore, quindi calcolare il numero seriale e da questo ricavare giorno, mese, anno, ora, minuti e secondi con le funzioni Excel che hanno lo stesso nome.

GENERAZIONE DEL NOME DEI FILE

Si è utilizzata l'epoca per generare i nomi dei file dati con la risoluzione di soli 10 secondi. Partendo da un riferimento molto vicino, ossia il primo di novembre 2020, con quattro cifre si posso indicare i giorni per altri 27 anni e con altre quattro cifre si possono identificare i 10 secondi del giorno (8640/giorno), supponendo di non creare un file per tempi inferiori ai dieci secondi.

Per cui è generato il nome del file lungo 8 caratteri. Esso viene generato ad ogni inizio acquisizione e stampato sulla finestra "MISURE".

Qui di seguito, il programma di generazione:

```
// days since 01 Nov 2020 00:00:00
ng= int((epoch-1604188800)/86400);
// seconds/10 from midnight
nm= int(secondspd/10);
```



```
// file name
String fn=Conv4digits(ng)+Conv4digits(nm)+
".TXT";
// String to char array conversion
fn.toCharArray(filename,13);

String Conv4digits(int n) {
  if (n < 10) return "000" + String(n);
  else if (n < 100) return "00" + String(n);
  else if (n < 1000) return "0" + String(n);
  else return String(n);
}
```

Il nome dei file generati da Arduino non deve superare gli otto caratteri, come nel vecchio DOS, per cui il programma controlla che il numero di giorni non superi i quattro caratteri, ossia che sia compreso tra 0000 e 9999, con queste istruzioni:

```
// check the lower limit of days
if (ng < 0) ng = 0;
// days must be 4 characters long
if (ng > 9999) ng= 9999;
```

LA GRAFICA DEI DATI AMBIENTALI

Per rappresentare i diagrammi storici delle tre grandezze misurate con il sensore BME280 occorre memorizzare le misure in un buffer di opportune dimensioni.

Il display TFT ha una definizione di 320x240 pixel, che si può organizzare come 240 punti sull'asse X e 320 sull'asse Y.

Volendo usare tutti i 240 pixel per rappresentare le misure nelle ventiquattro ore si deve campionare ogni $1440/240 = 6$ minuti (= 360 s), il che va più che bene per misure ambientali.

Con la MCU SAMD21 di uChip non ci sono i problemi di memoria che ci sarebbero con un Arduino Uno e una matrice di 3x240 interi (1440 byte) non è critica. Si potrebbe usare anche una matrice di float (2880 byte), ma è meglio non perdere troppo tempo nelle conversioni durante la grafica mentre si dispone di tempi elevati tra una misura e l'altra.

Si è scelto di avere un unico buffer GBuff organizzato come array di tre righe per 240 colonne di elementi interi:

```
int GBuff [3][240]; // data buffer for graphics
```

Le tre righe rappresentano la temperatura, la pressione barometrica e l'umidità relativa. Le misure sono in float e, tenendo presente l'ac-

curatezza del sensore e il miglior dettaglio dei grafici, si trasferiscono nel buffer gli interi che rappresentano la temperatura in decimi di grado, la pressione in decimi di hP e l'umidità in per mille trasformate nel vettore GVal[3].

A ogni campionamento si calcolano il massimo e il minimo di ogni variabile GVal[j]:

```
for (int j=0; j <= 2; j++){
  if (GVal[j] > ValMax[j]) ValMax[j] =
GVal[j];
  if (GVal[j] < ValMin[j]) ValMin[j] =
GVal[j];
}
```

Questi valori servono per creare i grafici in auto scala, tracciando le curve tra questi valori si ottiene una rappresentazione più dettagliata. Il massimo e il minimo sono calcolati dall'avvio del sistema. Il trasferimento dei dati nel buffer ha due modi di funzionamento.

All'inizio (first_buffer = true) il registro si riempie a partire da sinistra, dall'indice di colonna BuffInd =0 fino a BuffInd =239, ultimo punto a destra:

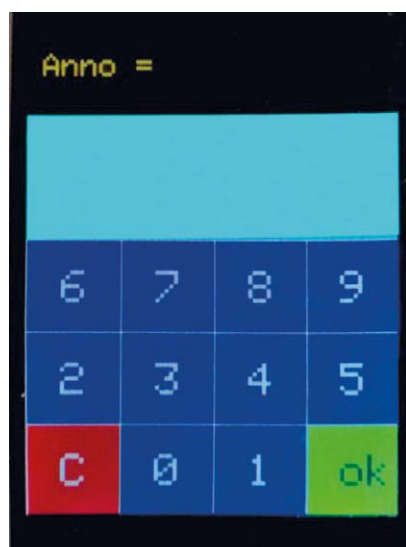
```
if (first_buffer){
  for (int j=0; j <= 2; j++){
    GBuff[j][BuffInd]= GVal[j];
  }
  BuffInd++;
  if (BuffInd == 240) first_buffer = false;
```

Una volta riempito il buffer, cambia il modo di riempimento, tutti gli elementi meno l'ultimo,



Fig. 12
Il menu delle
opzioni.

➔ **Fig. 13**
Tastierino numerico
virtuale del mobile
in legno.



scorrono di una posizione verso sinistra:

```
for (int j=0; j <= 2; j++){
    for (int i=0; i <= 238; i++){
        // shift left previous values
        GBuff[j][i]= GBuff [j][i+1];
    }
}
```

Quindi le misure dell'ultimo campionamento sono trasferite nell'ultima posizione (BuffInd = 239):

```
for (int j=0; j <= 2; j++){
    // update buffer with last values
    GBuff[j][239]= GVal[j];
}
```

La funzione PlotMeasures(int k) graficherà tutte e tre le misure in tre distinti grafici da 240x100. Lo shift dei dati verso sinistra crea uno scorrimento della traccia con un effetto simile a quello dei vecchi registratori a carta e si ottiene un trend storico delle misure nelle ultime 24 ore, assai utile per le previsioni meteo.

IL MENU DELLE OPZIONI

Premendo il display touch, in qualsiasi momento, appare un menu con quattro possibili opzioni, corrispondenti a quattro pulsanti, come si vede in **Fig. 12**.

La prima volta è necessario premere l'ultimo tasto, "SETTAGGI" di colore azzurro, per impo-

stare il periodo di campionamento in secondi, quindi anno (le ultime due cifre), mese, giorno, ora, minuti e secondi. Per impostare questi sette parametri numerici si utilizza la funzione GetNum(String prompt) che attiva la tastiera virtuale numerica, come si vede in **Fig. 13**.

Dopo aver digitato il parametro richiesto, occorre confermare con "ok" o cancellare l'immissione con il tasto "C". Il settaggio termina dopo aver inserito i secondi. Da programma non è possibile l'opzione settaggi se l'acquisizione è in corso, ossia se dal display viene mostrata la scritta "Acquisizione..."

Il secondo tasto, "MISURE", mostra la data e l'ora relativi all'ultimo campionamento, i valori delle misure, il numero dei campioni salvati su SD, i minimi e i massimi delle misure e il nome del file, come si vede in **Fig. 14**.

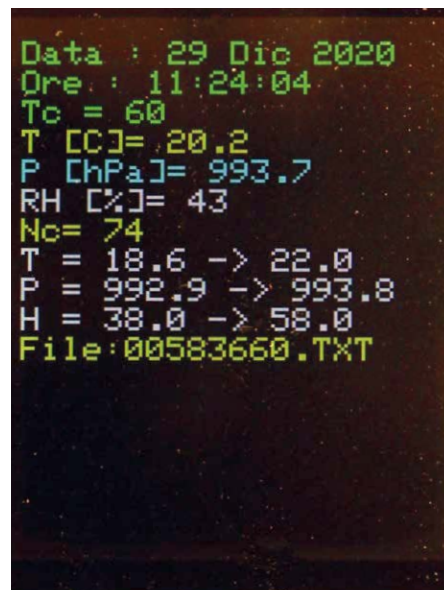
In questo caso sono stati acquisiti su SD Nc=269 campionamenti. Le ultime righe indicano i valori minimi e massimi, che sono anche gli estremi dei grafici.

Nell'ultima versione del programma, nel caso in cui sia stata avviata l'acquisizione, è indicato anche il nome del file su cui si trasferiscono le misure.

Il tasto giallo "GRAFICI" mostrerà i diagrammi temporali delle tre variabili, visibili in **Fig. 15**.

Questo campionamento, come si vede dalla **Fig. 14**, è stato fatto con periodo di 60 secondi, quindi le variabili fisiche sono variate di poco e l'autoscala crea i tipici gradini.

➔ **Fig. 14**
Screenshot delle
misure.



↓ Listato 1

```
% program TPHread.m legge i dati scritti dal sistema UChip
% di misure ambientali con data in giuliano e ora in secondi
% Giovanni Carrera 29 ott 2020
clear
clc
files= dir('*.TXT');
nfile=size(files,1);% numero file trovati

for indf=1:nfile,
    fprintf('%d) %s\n',indf,files(indf).name);
end
while true
    nf = input('numero di file da leggere = ');
    if (nf>=1) && (nf<=nfile), break, end
end
nomefile = files(nf).name;
fid = fopen(nomefile);
if fid==-1
    error([nomefile ': il file non esiste']);
end
DELIMITER = ',';
HEADERLINES = 0;
% Import the file
Data = importdata(nomefile, DELIMITER, HEADERLINES);
Gd = Data(:,1);% vettore giorno in giuliano
t = Data(:,2);% vettore tempo in secondi
julian = Gd+t/86400+693960;% aggiunge la frazione giorno e adatta l'anno
T = Data(:,3);% temperatura in [°C]
P = Data(:,4);% pressione in [hPa]
RH = Data(:,5);% umidità relativa in [%]
Ns = size(Data,1);
clear Data;
disp('Elaborazioni misure ambientali di uChip')
fprintf('Nome file elaborato      = %s\n',nomefile);
fprintf('Data inizio registrazione = %s\n', datestr(julian(1)));
fprintf('Data fine registrazione   = %s\n', datestr(julian(end)));
fprintf('Numero campioni acquisiti  = %d \n', Ns);
Rt = julian(end)-julian(1);
fprintf('Durata registrazione       = %6.2f [minuti]\n', Rt*1440);
tz =julian-julian(1);
figure(1);
subplot(3,1,1), plot(tz,T,'r','LineWidth',2),grid,ylabel('T [C]'),title(['File: ' nomefile])
subplot(3,1,2), plot(tz,P,'g','LineWidth',2),grid,ylabel('P [hPa]')
subplot(3,1,3), plot(tz,RH,'b','LineWidth',2),grid,ylabel('H [%]'),xlabel('days')
```

Il tasto rosa, se non è inserita la scheda SD, rimane senza scritta ed è inutilizzato. Se la SD è presente e non abbiamo ancora avviato l'acquisizione compare la scritta "Acq. Start/Stop". Premendo il tasto avviamo l'acquisizione e il tasto diventa temporaneamente rosso con la scritta "Inizio Acquisizione" per poi tornare rosa con la scritta "Acquisizione...". Quando vogliamo terminare l'acquisizione premiamo questo tasto che diventerà brevemente verde con la scritta "Fine Acquisizione", dopo questo è possibile togliere la SD.

Elaborando le misure con Matlab, si ottengono le seguenti uscite:

Elaborazioni misure ambientali di uChip

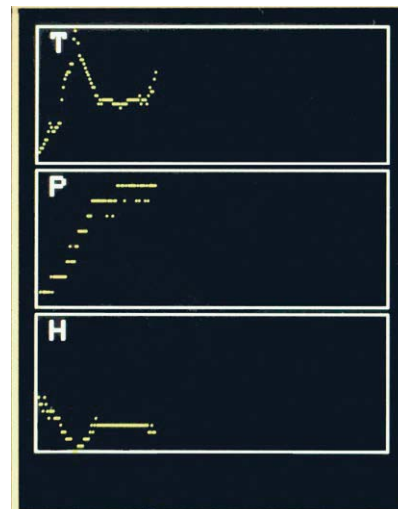
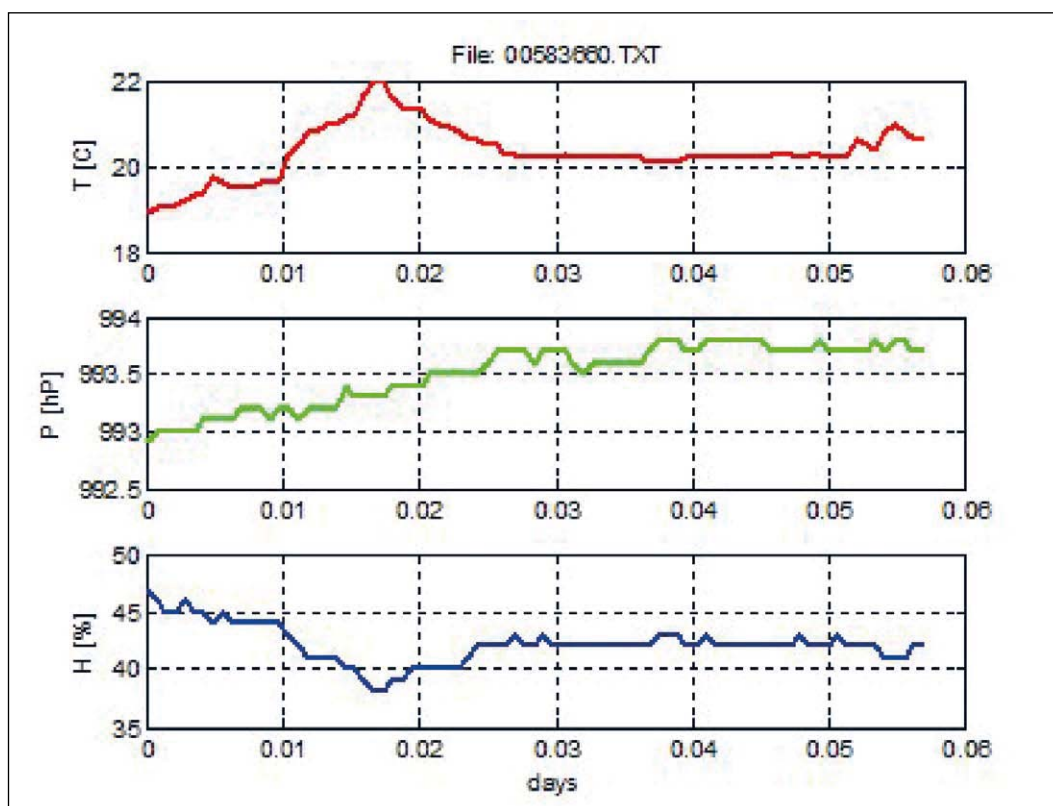


Fig. 15
Screenshot
dei grafici.

➔ **Fig. 16**
Grafici relativi ai
dati salvati sulla SD.



Nome file elaborato = 00583660.TXT
Data inizio registrazione = 29-Dec-2020 10:11:04
Data fine registrazione = 29-Dec-2020 11:33:04
Numero campioni acquisiti = 83
Durata registrazione = 82.00 [minuti]

Il grafico corrispondente è mostrato quello in **Fig. 16**.

Confrontando i grafici di **Fig. 15** con quelli della **Fig. 16** si può notare che sono molto simili, tenendo presente che Matlab unisce i punti con curve interpolate e che le scale orizzontali sono molto diverse.

Il programma scritto in Matlab per elaborare i file è quello nel **Listato 1**.

Chi non ha Matlab, può scaricare gratuitamente dal sito <https://www.gnu.org/software/octave/download> il programma GNU Octave che ora ha anche una buona interfaccia grafica utente. Questo programma funziona anche con Octave senza modifiche.

Il firmware della uChip è troppo voluminoso per entrare in questo articolo, ma lo trovate nella sezione download del nostro sito insieme agli altri file del progetto.

CONCLUSIONI

Il progetto qui proposto è la base per realizzare sistemi di rilevamento di vari parametri, anche non necessariamente ambientali: è sufficiente scegliere i sensori adatti e modificare il firmware. □

Cosa occorre?

Il materiale elettronico utilizzato in questo progetto è disponibile presso Futura Elettronica. La scheda uChip è disponibile con gli strip da saldare (cod. UCHIP) a Euro 19,00 e con gli strip già saldati (cod. UCHIPWSH) a Euro 19,90. Il display touch da 2,4" (cod. LCDTOUCH24SER) è disponibile a Euro 16,90, il sensore di pressione (cod. GY-BME280) è in vendita a Euro 8,50. Il PCB può essere realizzato utilizzando il nuovo servizio di stampa circuiti stampati (cod. PCBPRODUCTION) direttamente 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>



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

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.

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



Alimentale con:

MAX 6
barrette di
LED COB

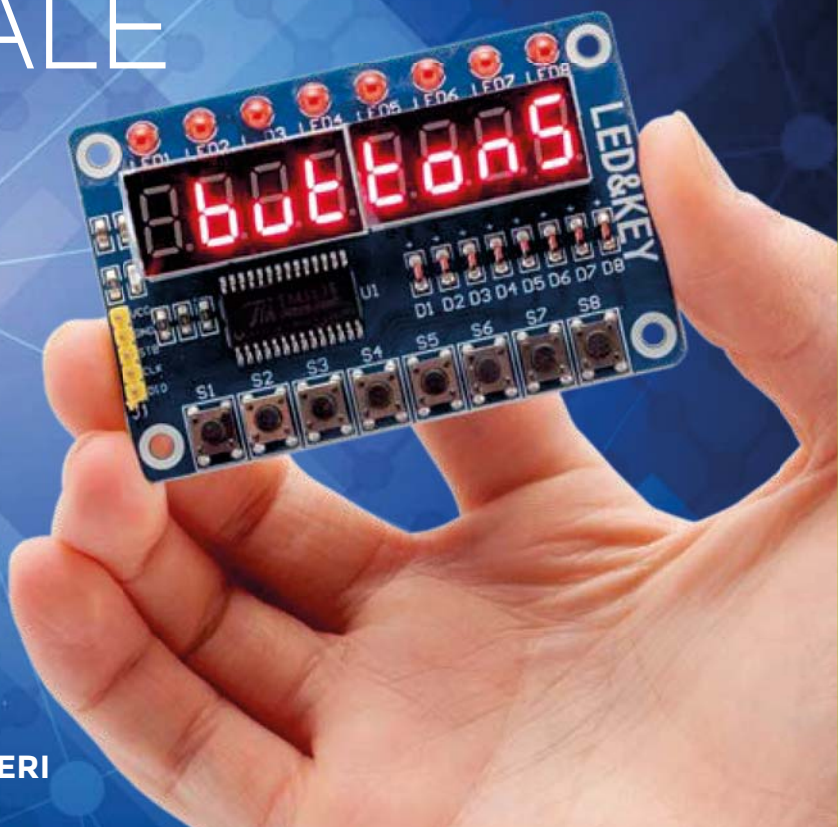
ALIMENTATORE
SWITCHING
12V - 100W -
8,5A

€20,00

Cod. MW10012



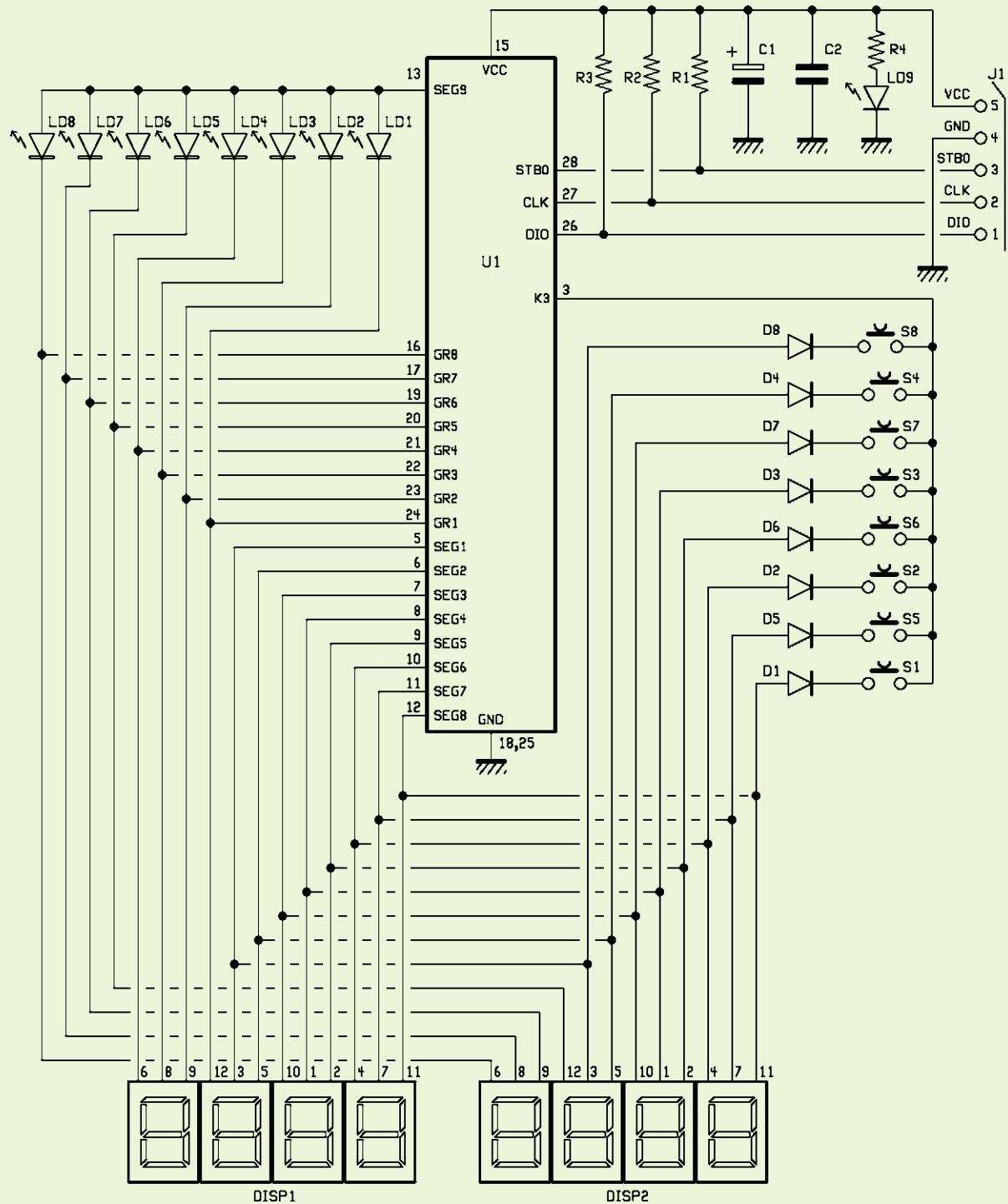
INTERFACCIA UTENTE UNIVERSALE



di **CRISTIANO RUGGERI**

Integra 8 display a 7 segmenti, 8 led e 8 pulsanti, che possono essere gestiti in lettura e scrittura tramite bus bidirezionale.

Uno dei problemi più comuni e più complessi da risolvere quando si lavora con i microprocessori in ambito hobbistico è sempre stato quello dell'interfacciamento con l'utente che, tipicamente, riesce a comprendere solamente messaggi presentati in modo opportuno su display grafici o alfanumerici nonché spie di segnalazione. Analogamente, l'unico mezzo che l'utente ha per passare delle informazioni ad un sistema a microprocessore è spesso riconducibile alla pressione di tasti da una tastiera più o meno complessa, piuttosto che da un touch screen o tramite un mouse e dei menu guidati. Se è vero che oggi i computer e gli smartphone dotati di sistemi operativi moderni riescono ad eseguire comandi impartiti tramite voce, è altresì vero che sistemi piccoli e concettualmente



Elenco Componenti:

R1-R2-R3: 10 Kohm
 R4: 1 Kohm
 C1: 10 uF
 C2: 10 nF
 LD1-LD8: led rosso 3mm
 LD9: led rosso smd
 D1-D2: 1n4148

S1-S8: microswitch da stampato
 DISP1-DISP2: modulo 4 display
 a 7 segmenti rosso
 U1: TM1638

Varie

- Strip maschio 5 poli
 - Circuito stampato S1545 (75x50)

semplici come apparecchi di misura piuttosto che gadget, ancora necessitano di tastiere e display di qualche tipo.

Inoltre la visualizzazione di valori su scale e funzioni (pensate per esempio ad un apparato in grado di leggere contemporaneamente tensione e corrente su due scale di visualizzazione differenti, e di calcolarvi in CosFi nel caso di un'applicazione in alternata) differenti sullo stesso display nel caso di strumenti e apparati da laboratorio, necessita sempre dell'intervento dell'operatore che agisce su tasti e manopole. Da qualsiasi parte la si guardi, sovente è necessario equipaggiare di tasti e display un sistema a microprocessore, poiché in nessun altro modo di facile attuazione si potrebbe implementare la tipica interfaccia utente.

A tal proposito, oggi il mercato offre valide alternative alla realizzazione in totale autonomia di tastiere e sinottici: per esempio i display LCD sono disponibili in tantissime forme e colori, a prezzi ragionevoli e con interfacce di tipo parallelo o seriale. Stesso discorso vale per le tastiere che sono disponibili a pochi tasti così come complete come quelle dei PC.

In questo articolo descriveremo la scheda LED&KEY, venduta dalla Futura Elettronica (con il codice LEDKEY), contenente 8 display a 7 segmenti + punto decimale, 8 LED di segnalazione e 8 pulsanti, che possono essere gestiti piuttosto facilmente attraverso una linea seriale sincrona bidirezionale.

SCHEMA ELETTRICO

Partiamo subito descrivendo lo schema elettrico di tale scheda, basata sul TM1638 LED/key controller prodotto dalla TITAN, azienda cinese che realizza numerosi circuiti integrati per la gestione di tastiere e display.

Il TM1638 viene largamente utilizzato in piccole schede equipaggiate di LED, display e tasti, utili sia in ambito hobbistico che non.

Le caratteristiche principali di questo integrato sono le seguenti:

- gestisce fino a 10 display a 7 segmenti + punto decimale;
- lavora sia a catodo che anodo comune;
- controlla fino a 24 pulsanti;
- dispone di 8 livelli di luminosità;
- lavora con interfaccia sincrona a 3 fili con dati, clock e strobe.

Per ulteriori informazioni potete consultare il datasheet, scaricabile dal sito www.futurashop.it/ *SCHEDA-LED-KEY-ledkey*. Per completezza riportiamo in **Fig. 2** la piedinatura dell'integrato TM1638.

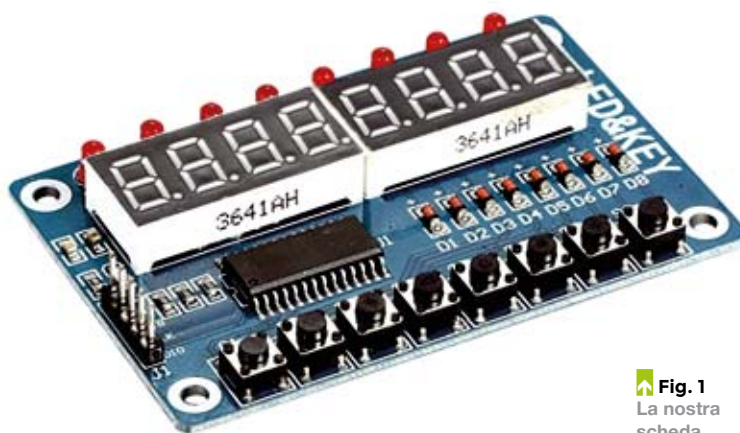


Fig. 1
La nostra
scheda.

Analizziamo uno ad uno i suoi piedini partendo da DIO (piedino 26) che è l'ingresso/uscita dati: in scrittura, il dato è trasferito sul fronte di salita del CLK; in uscita, sul fronte di discesa. In trasmissione, è un open drain e richiede un pull-up.

CLK è l'ingresso di clock e fa capo al pin 27, mentre lo strobe è localizzato sul pin STB (28): inizializza l'interfaccia sul fronte di discesa per ricevere istruzioni; il primo byte che segue tale fronte è considerato un'istruzione.

Quando viene processata un'istruzione gli altri processi vengono bloccati. Con STB a livello alto, il CLK viene ignorato. K1-K3 sono le linee della tastiera (rispettivamente piedini 1, 2, 3) e corrispondono all'ingresso dei pin dati dalla tastiera (hanno resistori di pull-up interni da 10 kohm). SG1-KS1/SG8-KS8 sono le uscite di comando dei segmenti (pin da 5 a 12) ovvero uscita di comando dei segmenti (P

1	K1	STB	28
2	K2	CLK	27
3	K3	DIO	26
4	VDD	GND	25
5	SEG1/KS1	GRID1	24
6	SEG2/KS2	GRID2	23
7	SEG3/KS3	GRID3	22
8	SEG4/KS4	GRID4	21
9	SEG5/KS5	GRID5	20
10	SEG6/KS6	GRID6	19
11	SEG7/KS7	GND	18
12	SEG8/KS8	GRID7	17
13	SEG9	GRID8	16
14	SEG10	VDD	15

Fig. 2
L'integrato
TM1638.

channel open drain) in multiplex e scansione tasti letti attraverso K1+K3.
GRID1-GRID8 (uscite dei digit) ovvero piedini 19-24, permettono l'attivazione in multiplex delle cifre del display ad anodo comune; sono linee open-drain facenti capo a MOSFET a canale N.
Infine Vdd (piedino 15) è l'alimentazione positiva del chip, riferita a massa (ossia a GND, pin 18).

LA PROGRAMMAZIONE

Vista la natura didattica di questo articolo, abbiamo pensato di spiegare come utilizzare la nostra scheda LedKey. In prima istanza siamo stati tentati dall'utilizzare l'immane Arduino, quindi abbiamo iniziato a cercare in rete degli spunti per aiutarci a realizzare delle librerie nostre di facile utilizzo e da condividere con i nostri lettori.
Ma cercando e cercando, abbiamo trovato molte librerie di terze parti che consentono di utilizzare questo dispositivo sia con Arduino che altri sistemi embedded.

Al contrario, non siamo riusciti a trovare alcunché per applicazioni in PICBASIC MELABS, pertanto abbiamo pensato di dare il nostro contributo creando delle librerie di lettura e scrittura della scheda preconfezionate: a voi basterà includere due file specifici in due punti precisi del vostro listato picBasic, e chiamare le varie subroutine nei modi che andremo a spiegarvi in questo articolo teorico.

LE LIBRERIE

In realtà, non dovremmo chiamarle librerie poiché si tratta di due spezzoni di codice vero e proprio scritte in PicBasic Melabs, inseriti in due file distinti che sono:

- LED_KEY_VARS.PBP
- LED_KEY_SUBS.PBP

Il primo contiene tutte le variabili e le costanti necessarie per il funzionamento delle varie subroutine. Le variabili sono molte, tutte di tipo "byte" e "bit", e vengono utilizzate in maniera piuttosto intensiva in tutte le varie subroutine.

Il secondo contiene tutte le subroutine con cui è possibile leggere e scrivere la scheda: le subroutine devono essere invocate dopo aver preparato il pacchetto di dati che esse devono processare qualora si tratti di una scrittura a display; nel caso di una lettura dei pulsanti, invece, è sufficiente invocare la lettura e immediatamente dopo leggere una variabile che contiene il risultato della lettura per poterla poi gestire dal programma principale.

UTILIZZO IN PRATICA

Andiamo dunque a sperimentare un po' con la scheda LED&KEY, premettendo che faremo riferimento alla configurazione hardware proposta nello schema di connessione visibile in queste pagine e chiamato **schema microcontrollore**; come vedete, in questa applicazione immaginiamo di lavorare su un PIC16F88 e pertanto i pin cui faremo riferimento negli esempi saranno quelli di tale microcontrollore. Il connettore LEDKEY serve a connettere la nostra interfaccia utente, PICKIT2 proviene dal programmatore che, oltre a programmare il PIC16F88, fornisce anche la tensione di 5Vcc (opzione questa supportata dal sistema) per alimentare e testare il circuito. Infine DS1302 serve perché tra i nostri esperimenti abbiamo realizzato un orologio digitale.
Bene, ciò detto andiamo subito agli aspetti firmware; la prima cosa da fare è inserire nel vostro listato, all'inizio delle dichiarazioni di variabile, la seguente istruzione:

```
INCLUDE LED_KEY_VARS.PBP
```

La dichiarazione di *include* deve tenere conto del percorso che il compilatore deve fare per cercare il file (path), per cui, per comodità di utilizzo e di ricerca, conviene copiare il file nella stessa cartella dove è contenuto il file del vostro progetto.

In questo modo quando si compilerà il programma, l'ambiente di sviluppo sostituirà in automatico alla riga di INCLUDE tutto il contenuto del file, inserendo di fatto la lista di variabili e costanti esattamente come fareste voi a mano.

Successivamente, occorre inserire nel vostro codice, ad esempio per comodità alla riga precedente la chiusura di programma END, verso il basso, dove sapete che non aggiungerete altro codice, la seguente istruzione (anche qui vale il precedente discorso del path):

```
INCLUDE LED_KEY_SUBS.PBP
```

Trattandosi di subroutine, è comunque conveniente posizionarle virtualmente alla fine del vostro listato, lasciandovi visibilità sulle sub di vostra ideazione. Infine è necessario dire al vostro programma quali sono i tre pin con cui il micro comunicherà con la scheda.

Per fare ciò si devono aggiungere, nella sezione di definizione degli I/O, le tre linee di codice che definiscono i tre pin di comunicazione con la scheda. Nell'esempio sotto riportato abbiamo ipotizzato di utilizzare i primi tre bit della porta C di un PIC, ma



nulla ci vieta di usare altri pin di altre porte, anche mischiati. Infatti nel nostro esercizio abbiamo usato i PB0, PB1 e PB2.

```
SYMBOL STROBE = PORTC.2
SYMBOL SCL = PORTC.1
SYMBOL SDA = PORTC.0
```

A questo punto il vostro programma tipo assomiglierà come struttura a quello visibile nel **Listato 2**. Se non ci sono errori nella vostra parte di codice, lanciando una compilazione il sistema genererà un HEX senza alcuna segnalazione di errore.

LE SUBROUTINE

Vediamo ora le varie subroutine nel dettaglio, tenendo presente che quello che noi volevamo ottenere è qualcosa di facile da comprendere ed utilizzare, senza appesantire troppo il vostro sorgente. L'integrato adottato in questa scheda è molto potente e consente di fare molte cose, come ad esempio l'indirizzamento diretto sia per la scrittura di un LED o display specifico, sia per la lettura di un tasto specifico. Consente anche la gestione in scrittura ad incremento automatico della posizione del carattere, così come la scrittura per singolo bit di ogni elemento luminoso...

Tutto ciò è molto utile e molto interessante, nonché molto bello da avere e usare, se si hanno a disposizione tanti kB di memoria sacrificabili.

Ma noi volevamo che questo oggetto fosse utilizzabile anche con microprocessori molto piccoli, così abbiamo ideato una serie di routine generalizzate, un po' "chiuse" se vogliamo, ma praticamente "plug&play" (usando un termine ormai sdoganato).. Queste routine sono:

- CLEAR_ALL: spegne tutti i display e i led. Non necessita di alcun parametro.
- SET_ALL: accende tutti i display e led. Non necessita di alcun parametro.
- CLEAR_DIGITS: spegne tutti i display. Non necessita di alcun parametro.
- SET_DIGITS: accende tutti i display. Non necessita di alcun parametro.
- CLEAR_LEDS: spegne tutti i LED. Non necessita di alcun parametro.
- SET_LEDS: accende tutti i LED. Non necessita di alcun parametro.
- WRITE_DOTS: scrittura diretta dei LED. Parametro DOTS. Prima di essere chiamata, occorre scrivere nella variabile DOTS i LED da accendere o spegnere. Ad esempio:
-DOTS = %00000000: tutti spenti

-DOTS = %11111111: tutti accesi

-DOTS = %10000000: acceso solo il primo a sinistra.

- PRINT_TEXT: contiene il testo da mandare ai display a 7 segmenti. Necessita dei parametri DGT(0÷7) e DP. Prima di invocarla, occorre impostare le variabili DGT da 0 a 7 (array di 8 byte) e la variabile DP che contiene i punti decimali da visualizzare. DGT(0) è il display a sinistra. DGT(7) quello a destra. Nelle variabili da DGT(0) a DGT(7) si deve inserire il carattere alfanumerico da visualizzare a display mentre nella variabile DP si devono definire in modo binario i punti decimali da accendere.

Nel seguente esempio si vuole scrivere a display la parola "PAUSA", partendo dal primo display a sinistra, e si vuole che ogni carattere abbia acceso anche il suo punto decimale.

DGT(0) = "P"

DGT(1) = "A"

DGT(2) = "U"

DGT(3) = "S"

DGT(4) = "A"

DGT(5) = " "

DGT(6) = " "

DGT(7) = " "

DP = %11111000

GOSUB PRINT_TEXT

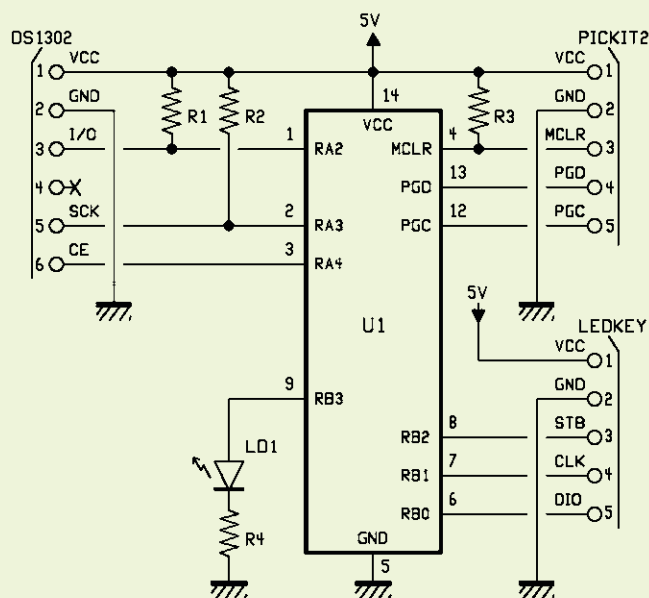
- DISPLAY_BRIGHT: controlla la luminosità del display. Necessita del parametro DISP_BRIGHT, che deve essere posto uguale ad una delle contanti da BRIGHT_0 a BRIGHT_8, che definiscono la luminosità rispettivamente dal spento (BRIGHT_0) al massimo (BRIGHT_8) su 8 livelli.
Ad esempio:

DISP_BRIGHT = BRIGHT_8 ' massima luminosità

GOSUB DISPLAY_BRIGHT ' invoca la sub.

- GET_SWITCH: legge i tasti. La lettura viene restituita sulla variabile SWITCH dove, in binario, MSb = tasto tutto a sinistra
LSb = tasto tutto a destra.
Ad esempio se dopo una GOSUB GET_SWITCH la variabile SWITCH vale %11110000, significa che sono premuti i quattro tasti a sinistra. Analogamente %00000000 direbbe che nessun tasto è premuto come %11111111 direbbe che tutti i tasti sono premuti.
- WRITE_DISPLAY, WRITE_MODE, SEND_CHAR sono routine di supporto a tutte le altre e non andrebbero mai invocate, dato che ci pensano le altre sub a chiamarle al momento opportuno, an-





Elenco Componenti:

R1-R2-R3: 10 Kohm
R4: 560 ohm
U1: PIC16F88
LD1: led rosso 5 mm

che se, volendo approfondire il discorso dell'indirizzamento diretto, i più esperti potrebbero anche utilizzarle per velocizzare le operazioni. È infatti logico che scrivere ogni volta 8 display e 8 led necessita più cicli macchina di quanti ne occorrebbero per puntare ad un indirizzo direttamente e scrivervi dentro un valore preciso.

Vediamo ora le routine nel dettaglio, secondo l'ordine di livello ossia partendo da quelle a più basso livello, più semplici, e salendo via via verso quelle

più complesse, che richiamano a loro volta altre subroutines annidate.

SEND_CHAR

A meno che non sia utilizzata da un utente esperto che ha approfondito il discorso dell'indirizzamento diretto delle memorie, essa non viene mai invocata se non dalle varie sub in automatico ogni volta che il micro deve inviare un'istruzione di qualche tipo alla scheda.

Le istruzioni possono essere di tipo "leggi tastiera" o "scrivi display e LED". È la routine più utilizzata, poiché è asservita a tutte le altre: in pratica invia, attraverso le linee preposte alla comunicazione, il dato di 1 byte contenuto nella variabile DATA_IO, utilizzando il protocollo seriale sincrono. Utilizza la sola funzione picbasic SHIFTOUT e il contenuto di DATA_IO non viene normalmente stabilito dall'utente, se non in alcuni casi, come ad esempio quando si imposta la luminosità del display, ma anche in questo caso in modo indiretto.

SEND_CHAR:

```
shiftout SDA,SCL,4,[data_io]
' LSB first, CLOCK idle HIGH
return
```

GET_SWITCH

Legge i pulsanti e il suo utilizzo è molto semplice: la si invoca ed immediatamente dopo (o quando servirà) si legge il contenuto della variabile SWITCH bit a bit. Il bit 7 corrisponde al primo pulsante a sinistra e il bit 0 all'ultimo (lettura bitwise).

Quando viene chiamata, per prima cosa imposta la variabile DATA_IO a READ_MODE (\$42) quindi RESETTA lo strobe per iniziare la scrittura del comando (chiama la SEND_CHAR già vista).

A questo punto inizia la lettura di quattro byte (memorizzati nelle variabili da SW1 a SW4) tramite la funzione SHIFTIN, operazione terminata dallo strobe SETTATO.

Infine la variabile SWITCH viene scritta bit a bit verificando lo stato dei bit 6 e 2 delle quattro variabili SW1-SW4. Va notato che l'integrato TM1638 è in grado di gestire una matrice di tasti molto estesa, mentre qui sono usati solamente 8 pulsanti, mappati dal costruttore della scheda nei bit 6 e 2 delle quattro variabili, secondo la maschera nel codice nel riquadro.

Il perché di questa scelta, piuttosto che inserirli tutti in una unica variabile, non ci è dato di sapere; comunque la parte difficile l'abbiamo fatta noi, a voi basta leggere il contenuto della singola variabile

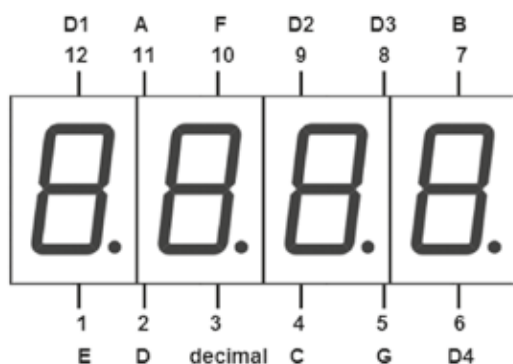


Fig. 3
La pinout
del display.

SWITCH per ricavare lo stato dei vari pulsanti.

```
GET_SWITCH:
    data_io = read_mode
    ' Impostazione per lettura di 4 byte.
    low strobe
    ' Inizio lettura
    gosub SEND_CHAR:
    shiftin sda,scl,4,[sw1,sw2,sw3,sw4]
    ' Legge i 4 byte.
    high strobe
    ' Fine lettura.
    ' MSB primo, legge il dato dopo il clock.
    switch.7 = sw1.6 ' Converte i 4 byte letti
    switch.6 = sw2.6 ' in un singolo byte
    switch.5 = sw3.6 ' contenuto nella var
    switch.4 = sw4.6 ' SWITCH.
    switch.3 = sw1.2 '
    switch.2 = sw2.2 '
    switch.1 = sw3.2 '
    switch.0 = sw4.2 '
    return
```

CLEAR_ALL e SET_ALL

Come intuibile dai nomi, la prima spegne completamente tutti i LED e display, mentre la seconda accende tutto; entrambe le operazioni vengono effettuate con 16 scritture ricorsive di 0 o 255 a tutte le locazioni, partendo da un indirizzo base, dove si trova il primo display a sinistra, e impostando l'incremento automatico dell'indirizzo.

Di fatto, anche se ha due nomi distinti, che altro non sono che due LABEL di programma, la routine è una sola.

L'inizio avviene SETTANDO o RESETTANDO il bit SET_CLEAR il cui valore, 1 o 0, determina l'accensione o lo spegnimento totale. Quindi è chiamato il modo scrittura con l'istruzione GOSUB WRITE_MODE, routine questa che esegue le operazioni seguenti:

- assegna alla variabile DATA_IO il valore AUTO_ADDR (\$40), che dice alla scheda di passare automaticamente al carattere successivo ad ogni nuovo carattere ricevuto;
- abbassa il pin di STROBE, manda il carattere e alza il bit di STROBE per finire la scrittura del carattere.

Ora il programma torna alla routine principale, che mette in DATA_IO l'indirizzo di partenza ossia FIX_ADDR (\$C0) che punta al primo display di sinistra. Viene attivato ancora lo STROBE per avviare una nuova scrittura a display e ancora una volta invocata

la SEND_CHAR già vista in precedenza.

Parte quindi un ciclo di 16 iterazioni in cui a seconda dello stato di SET_CLEAR viene mandato al display o \$00 o \$FF per 16 volte.

Tutti i cicli pari azzerano o accendono un display, mentre tutti i cicli dispari accendono o spengono un led, secondo la mappatura del TM1638.

La routine termina quindi con la chiamata alla sub DISPLAY_BRIGHT, che imposta runtime la luminosità del display. Come già anticipato è possibile impostare 9 valori da spento a massimo nella variabile DISP_BRIGHT semplicemente ponendola uguale a una costante compresa tra BRIGHT_0 (spento) e BRIGHT_8 (massima luminosità) prima della GOSUB DISPLAY_BRIGHT. Volendo essa può essere invocata a piacimento, dopo avere scritto in DISP_BRIGHT il valore di luminosità voluto, ad esempio per enfatizzare l'esito di un'operazione facendo brillare più o meno i caratteri.

```
CLEAR_ALL:
    low SET_CLEAR
    goto UPDATE_ALL

SET_ALL:
    high SET_CLEAR

UPDATE_ALL:
    gosub WRITE_MODE: 'chiama autoincremento
    STROBE = 0 'inizio scrittura
    data_io = FIX_ADDR 'Imposta indirizzo
    gosub SEND_CHAR: ' Scrittura
    for counter = 1 to 16 '
    if SET_CLEAR=0 then '
    data_io = 0 ' spegne tutto
    else '
    data_io = 255 ' accende tutto
    endif '
    gosub SEND_CHAR ' Scrittura
    next '
    Strobe = 1 ' Fine scrittura
    gosub DISPLAY_BRIGHT ' Imposta luminosità.
    return

WRITE_MODE:
    DATA_IO = AUTO_ADDR
    strobe = 0
    gosub send_char:
    strobe = 1
    return

DISPLAY_BRIGHT:
    data_io = disp_bright
    strobe = 0
    gosub SEND_CHAR:
    strobe = 1
    return
```



CLEAR_DIGITS e SET_DIGITS

Anche in questo caso il nome la dice lunga: la prima azzerata tutti i display a 7 segmenti e la seconda li accende tutti. Funzionalmente è simile alla CLEAR_ALL/SET_ALL, ma ci sono delle differenze. Innanzitutto, invece di eseguire 16 scritture in autoincremento, ne esegue solamente 8 nelle posizioni pari da 0 a 14, occupate dai display, utilizzando l'indirizzamento diretto alla locazione di memoria relativa al display. Poi viene coinvolta un'altra subroutine asservita, la WRITE_DISPLAY, il cui compito è quello di mandare prima l'indirizzo dove scrivere e poi il carattere da scrivervi. Infine, il contatore di loop ora è rappresentato dall'indirizzo del display (DISP_ADDR) che ad ogni ripetizione viene incrementato di 2 unità. Si fa infine uso di una nuova variabile, la SEG_VAL, qui scritta direttamente a 0 o 255, che come vedremo contiene la conversione dal carattere alfanumerico al suo equivalente (con la migliore approssimazione possibile) in 7 segmenti.

```
CLEAR_DIGITS:      ' Spegne tutti i display.
                   low SET_CLEAR      '
                   goto UPDATE_DIGITS '

SET_DIGITS:        ' Accende tutti i display
                   high SET_CLEAR     '
                   'GOTO UPDATE_DIGITS '

UPDATE_DIGITS:     '
                   for DISP_ADDR = 0 to 14 step 2
                   ' parte il loop a ind. diretto
                   if SET_CLEAR=0 then '
                   SEG_VAL = 0         ' Spegne i display.
                   ELSE                '
                   seg_val=255         ' Accende i display.
                   endif               '
                   gosub write_display:
                   ' Chiama scrittura indirizzamento diretto.
                   next                '
                   gosub display_bright ' Imposta luminosità.
                   return              '

WRITE_DISPLAY:
                   data_io = FIX_ADDR + DISP_ADDR
                   ' imposta indirizzo.
                   strobe = 0
                   ' strobe 0 per inizio scrittura
                   gosub send_char:    ' manda indirizzo
                   data_io = SEG_VAL  ' imposta il digit
                   gosub send_char:    ' manda il digit
                   STROBE = 1          ' finita scrittura
                   gosub display_bright: ' manda luminosità.
                   return              '
```

CLEAR_LEDS e SET_LEDS

Come nelle precedenti CLEAR_DIGITS e SET_DIGITS è funzionalmente identica come concetto e come variabili di servizio; utilizza sempre l'indirizzamento diretto per puntare alle locazioni di memoria, ma in questo caso a quelle dispari, che contengono l'informazione di led (assegnato al bit 0 della cella di memoria) acceso o spento. Stiamo parlando dei LED posti sopra i display.

```
CLEAR_LEDS:        ' Spegne tutti i led.
                   low SET_CLEAR      '
                   GOTO UPDATE_LEDS  '

SET_LEDS:          ' Accende tutti i led.
                   high SET_CLEAR     '
                   'goto UPDATE_LEDS '

UPDATE_LEDS:       '
                   for DISP_ADDR = 1 to 15 step 2
                   if set_clear=0 then '
                   SEG_VAL = 0         ' Spegne i led.
                   else                '
                   seg_val = 1         ' accende i led
                   endif               '
                   gosub write_display:
                   ' Chiama scrittura indirizzamento diretto.
                   next                '
                   gosub display_bright ' Imposta luminosità
                   return              '
```

WRITE_DOTS

Utilizza l'indirizzamento diretto della scrittura per accendere e spegnere in base alle necessità i led posti sopra i caratteri a 7 segmenti. Prima di chiamarla occorre impostare bit a bit la variabile DOTS, che contiene lo stato di ON/OFF di ogni LED. L'MSb corrisponde al primo led a sinistra mentre l'LSb corrisponde all'ultimo a destra. Ogni bit della variabile DOTS ha un alias che lo contraddistingue: ad esempio DOT7 è il bit 7 di DOTS, che rappresenta il primo LED a sinistra, posto sopra il primo display a sinistra. Analogamente DOT0 è l'ultimo LED a destra. Si inizia impostando la variabile DOTS quindi si invoca la WRITE_DOTS che punta agli indirizzi dispari di memoria in un ciclo di 8 volte, in cui viene incrementato l'indirizzo di scrittura e assegnato ad esso il livello logico del bit. Nel primo ciclo la routine assegna al primo LED a sinistra il contenuto di DOT7, nel secondo ciclo assegna al secondo LED il contenuto di DOT6 e così via fino alla fine.

```
WRITE_DOTS:
                   for DISP_ADDR = 1 to 15 step 2
                   ' led tutti digit A display dispari
```




```

select case disp_addr '
case 1
    seg_val = dot7
case 3
    seg_val = dot6
case 5
    seg_val = dot5
case 7
    seg_val = dot4
case 9
    seg_val = dot3
case 11
    seg_val = dot2
case 13
    seg_val = dot1
case 15
    seg_val = dot0
end select
gosub write_display
Scrive i led.
next
return

```

PRINT_TEXT

La più lunga delle sub, serve per scrivere a display i messaggi richiesti (**Listato 1**). Lavora a indirizzamento diretto, ma dal punto di vista dell'utilizzo questo non traspare. Prima di chiamarla, deve essere

↓ Listato 1

```

' DGT(0) = "C"
' DGT(1) = "I"
' DGT(2) = "A"
' DGT(3) = "0"
' DGT(4) = " "
' DGT(5) = " "
' DGT(6) = " "
' DGT(7) = " "
' DP = %11110000
'gosub PRINT_TEXT:
PRINT_TEXT:
    for CNT = 0 to 7
        digit = DGT(CNT)
        SELECT CASE digit
            case "0"
                seg7(CNT) = %00111111
            case "1"
                seg7(CNT) = %00000110
            "tutti i case per i numeri"
            case "9"
                seg7(CNT) = %01101111
            case " "
                seg7(CNT) = %00000000
            case "a", "A"
                seg7(CNT) = %01110111
            case "b", "B"
                seg7(CNT) = %01111100
            case "c", "C"
                seg7(CNT) = %01011000
            "tutti i case delle lettere"
            case "z", "Z"
                seg7(CNT) = %01011011
            case "-"
                seg7(CNT) = %01000000
            case "_"
                seg7(CNT) = %00001000
            case "."
                seg7(CNT) = %10000000
        END SELECT
    next CNT
    ' aggiunge il punto decimale dove necessario.
    'DP = %11111111 solo per prova.
    if dp.7= 1 then seg7(0) = seg7(0) | dp_mask ' OR bitwise con bit7.
    if dp.6= 1 then seg7(1) = seg7(1) | dp_mask
    if dp.5= 1 then seg7(2) = seg7(2) | dp_mask
    if dp.4= 1 then seg7(3) = seg7(3) | dp_mask
    if dp.3= 1 then seg7(4) = seg7(4) | dp_mask
    if dp.2= 1 then seg7(5) = seg7(5) | dp_mask
    if dp.1= 1 then seg7(6) = seg7(6) | dp_mask
    if dp.0= 1 then seg7(7) = seg7(7) | dp_mask
    for disp_addr = 0 to 14 step 2
        seg_val = seg7(disp_addr / 2)
        gosub write_display: ' chiama scrittura diretta.
    next
    gosub display_bright: ' Imposta luminosità.
    return

```

▼ Listato 2

```
' Definizione dei registri del micro. Dipendono dal micro scelto.
' Questa parte è a discrezione dell'utente.
...
' Queste definizioni sono obbligatorie; i pin sono a titolo di esempio
' e possono essere diversi.
SYMBOL STROBE = PORTA.2
SYMBOL SCL = PORTA.1
SYMBOL SDA = PORTA.0
...
' da qui in poi ci sono le definizioni di pin del programma utente.
...
'-----
' Definizione di tutte le variabili e costanti utente.
INCLUDE LED_KEY_VARS.PBP ' obbligatoria insieme alle variabili
' da qui in poi le variabili utente.
...
'-----
GOSUB SET_ALL          ' accende tutto
pause 1000
GOSUB CLEAR_ALL        ' spegne tutto
pause 1000
GOSUB SET_LEDS         ' accende tutti i led
pause 1000
GOSUB CLEAR_LEDS       ' spegne tutti i led
pause 1000
GOSUB SET_DIGITS       ' accende tutti i digit
pause 1000
GOSUB CLEAR_DIGITS     ' sopegne tutti i digit
pause 1000
MAIN:
  gosub GET_SWITCH      ' legge i pulsanti
  DOTS = SWITCH         ' li trasferisce
  gosub WRITE_DOTS      ' sui led
  if DOT7 = 0 then      ' se pulsante 1 a riposo
    DGT(0) = "-"        ' scrive "-" su display 1
  Else                  ' altrimenti
    DGT(0) = "1"        ' scrive "1"
  Endif
  if DOT6 = 0 then      ' se pulsante 2 a riposo
    DGT(1) = "-"        ' scrive "-" su display 2
  else                  ' altrimenti
    DGT(1) = "2"        ' scrive "2"
  Endif
  if DOT5 = 0 then      ' e così via.
    DGT(2) = "-"
  else
    DGT(2) = "3"
  endif
  if DOT4 = 0 then
    DGT(3) = "-"
  else
    DGT(3) = "4"
  endif
  if DOT3 = 0 then
    DGT(4) = "-"
  else
    DGT(4) = "5"
  endif
  if DOT2 = 0 then
    DGT(5) = "-"
  else
    DGT(5) = "6"
  endif
  if DOT1 = 0 then
    DGT(6) = "-"
  else
    DGT(6) = "7"
  endif
  if DOT0 = 0 then
    DGT(7) = "-"
  else
    DGT(7) = "8"
  endif
  gosub PRINT_TEXT      ' scrive tutti i display
  pause 100             ' pausa.
  GOTO MAIN
'-----
INCLUDE LED_KEY_SUBS.PBP ' include tutte le "librerie"
'-----
END                      ' fine programma
```



messo il testo da scrivere nell'array DGT. DGT(0) è il primo display a sinistra e DGT(7) l'ultimo a destra. Occorre anche impostare bit a bit la variabile DP, in cui il bit 7 rappresenta il punto decimale del primo display a sinistra e il bit 0 il punto del display a destra.

Una volta preparate le 9 variabili (si veda l'esempio nel codice in cui si scrive CIAO seguita da quattro spazi vuoti e ogni lettera ha il punto decimale acceso), viene invocata con l'istruzione GOSUB PRINT_TEXT.

La prima cosa che fa è avviare un ciclo di 8 ripetizioni in cui va a leggere ogni singolo carattere dell'array e lo converte nel corrispettivo in formato 7 segmenti. Per esempio, nel nostro caso il primo carattere a sinistra, DGT(0), è "C" che in 7 segmenti corrisponde ai segmenti d(bit3), e(bit4) e g(bit6) ossia %01011000. La conversione viene effettuata attraverso un select case in cui sono stati inseriti tutti i caratteri alfanumerici di interesse insieme alla relativa conversione. Le conversioni vengono poi memorizzate in un altro array di 8 byte, chiamato SEG7; quindi il carattere contenuto nella variabile DGT(0) avrà il suo corrispondente a 7 segmenti nella variabile SEG7(0) e così via.

Terminata la conversione degli 8 caratteri, occorre assegnare ad essi il punto decimale, se necessario, sapendo che esso si trova sempre nel bit 7. Per fare ciò occorre mettere in OR bit per bit il contenuto di ogni variabile SEG7 con la maschera %10000000, contenuta nella costante DP_MASK, ma solo se il relativo bit nella variabile DP è settato. In altre parole, se il bit 7 di DP è a 1 significa il carattere contenuto in DGT(0) deve avere il punto decimale, quindi il suo equivalente a 7 segmenti, contenuto in SEG7(0), deve essere messo in OR con DP_MASK per settare il bit 7 del punto decimale. Il risultato è che se prima SEG7(0) valeva %01011000 (c.), dopo l'OR varrà %11011000 (c.).

Naturalmente lo stesso discorso vale per ogni carattere. Dopo aver convertito tutti i caratteri nell'equivalente a 7 segmenti e assegnato ad essi il punto decimale se necessario, non rimane che avviare il loop di 8 scritture ad indirizzamento diretto sugli indirizzi pari (caratteri) per mostrare a display il testo richiesto.

CODICE DI ESEMPIO

Nel **Listato 2** vedete un programma generico di test, che a distanza di un secondo accende tutto, spegne tutto, accende e spegne i soli LED, accende e spegne i soli display, avvia un ciclo che legge i pulsanti e ne mostra la condizione sia sui LED che sul display,

scrivendo 1 sul display 1 se il tasto 1 è premuto, 2 per il successivo, 3 per il successivo ancora e così via. La definizione dei registri del micro è legata al micro stesso, pertanto è a vostra discrezione. Come potete notare, è sufficiente includere variabili e costanti in cima al sorgente e le subroutine in coda, in modo da avere visuale libera per quanto riguarda il codice sorgente utente.

CONCLUSIONI

Come premesso all'inizio dell'articolo, in queste pagine volevamo proporvi una soluzione Hardware/Firmware molto semplice da utilizzare, che permettesse di aggiungere ad una qualsiasi piattaforma a microprocessore o controllore PIC, anche di dimensioni contenute, un'interfaccia utente dotata sia di pulsanti che display e led, compatta e semplice da utilizzare sia dal punto di vista HW che FW. Come hardware direi che ci siamo: compatta, rettangolare, necessita solo di 5 fili e 4 fori di fissaggio, più una feritoia per il display su un eventuale pannello. A tal proposito, vi ricordiamo che qualora volesse realizzare un contenitore ad hoc, Futura Elettronica offre un efficiente servizio di taglio laser. Per quanto riguarda il FW, a noi il risultato ottenuto sembra piuttosto buono, seppur certamente migliorabile come dimensioni ed efficienza. A voi il compito di divertirvi utilizzandolo ed eventualmente migliorarlo, nel puro spirito della didattica che portiamo avanti. Continuate a seguirci perché nei prossimi numeri pubblicheremo il progetto di un orologio molto semplice, che abbiamo realizzato per sperimentare tanto la scheda a display quanto una scheda di cui non abbiamo ancora parlato, che monta un real time clock modello DS1302. 



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 utilizzato (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 un PCB del il 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>**



VASTA GAMMA DI STRUMENTI PER IL LABORATORIO

Oscilloscopio 2 Canali 70 MHz

- ✓ Frequenza di campionamento fino a 1 GS/s
- ✓ Funzione FFT integrata
- ✓ Software per analisi in tempo reale con il PC
- ✓ Porta USB Host

COD. DS072

€ 319,00

IL PIÙ VENDUTO



Disponibile anche
nella versione a 200MHz

COD. DS0202

€ 459,00

Oscilloscopio 2 Canali 100 MHz con generatore di forme d'onda

COD. DS0FG102

€ 429,00

- ✓ Frequenza di campionamento fino a 1 GS/s
- ✓ Funzione FFT integrata
- ✓ Software per analisi e controllo in tempo reale con il PC
- ✓ Porta USB Host / USB Device



Disponibile anche
nella versione a 200MHz

COD. DS0FG202

€ 549,00

Oscilloscopio a forma di penna - 1 canale

- ✓ Larghezza di banda 20 MHz
- ✓ Frequenza di campionamento 96 MSa/s
- ✓ Tensione max IN 50V
- ✓ Funzione FFT integrata
- ✓ Software per analisi in tempo reale con PC
- ✓ Alimentazione tramite porta USB
- ✓ Dimensioni 230 x 42 x 25 mm
- ✓ Peso 263 grammi



COD. PS02020

€ 89,00

Oscilloscopio USB per PC 2 Canali 20 MHz



COD. DS0PC22

€ 89,00

- ✓ Frequenza di campionamento 48 MS/s
- ✓ Funzione FFT integrata
- ✓ Interfaccia USB 2.0 per PC (alimentazione da USB)
- ✓ Software e memorizzazione delle forme d'onda

Disponibile anche
nella versione a 100MHz

COD. DS0PC102

€ 290,00

Generatore di funzioni e forme d'onda arbitrarie

USCITA FORME D'ONDA ARBITRARIE

- ✓ Frequenza onda DC~25 MHz
- ✓ 1 canale
- ✓ Frequenza di campionamento 200 MSa/s
- ✓ Risoluzione verticale 12 bit

€ 206,00

COD. FAW1025G

FREQUENZIMETRO

- ✓ Frequenza fino a 50 MHz
- ✓ Impedenza d'ingresso > 100 kohm
- ✓ Ampiezza segnale d'ingresso da 400 mVpp a 18 Vpp



Generatore di funzioni/forme d'onda arbitrarie fino a 10 MHz

€ 309,00

COD. TP0073



- ✓ 1 Canale
- ✓ Frequenza di campionamento 125 MS/s
- ✓ Frequenza max. di uscita 10 MHz
- ✓ Forme d'onda standard: sinusoidale, quadra, rampa, triangolare, impulso, noise, DC, arbitraria
- ✓ Supporta le modulazioni AM, FM, PM, PWM, ASK, FSK, PSK
- ✓ Ampiezza di uscita 1 mVpp ~ 10 Vpp (50 Ω); 2 mVpp ~ 20 Vpp (High Z)

Prezzi IVA inclusa.

FUTURA ELETTRONICA®

Futura Group srl
Via Adige, 11 • 21013 Gallarate (VA)
Tel. 0331/799775

Caratteristiche tecniche
e vendita on-line su:

www.futurashop.it

di DANIEL SCAINI

RICONOSCIMENTO TARGA DA WEB

Usiamo il modulo ESP32-CAM per rilevare e riconoscere la targa di un veicolo ed azionare un attuatore con cui aprire un garage, sollevare una sbarra, attivare un allarme e altro ancora.



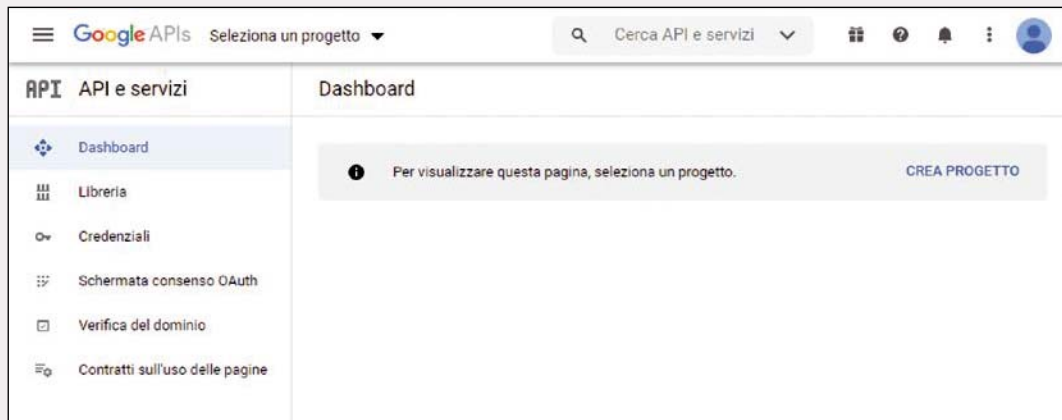
Il tema della domotica è in continua evoluzione: vediamo sempre più frequentemente videocitofoni connessi ai quali è possibile accedere da remoto, ma anche termostati "intelligenti" comandabili anche all'esterno della propria casa e che riescono a capire le nostre esigenze. L'attuale tendenza è infatti quella di riuscire a sfruttare la tecnologia e la rete per poter avere

sempre il controllo della propria abitazione.

Non solo, questa tendenza coinvolge anche diversi luoghi pubblici o lavorativi. Infatti, ultimamente sempre più servizi sono stati automatizzati attraverso questi nuovi strumenti di impronta domotica; basti pensare che in alcuni parcheggi si può pagare attraverso il telepass e quindi senza dover fare un biglietto o pagare presso uno sportello. Andiamo incontro ad un mondo totalmente connesso, in grado di assecondare ogni nostro desiderio; un mondo dove la capacità di analizzare e discriminare immagini e video gioca un ruolo chiave.

Anche noi abbiamo intrapreso questa via: abbiamo presentato, ad esempio, un dispositivo in grado di rilevare gli accessi e fornire una prova fotografica dell'evento tramite un messaggio su Telegram, ma anche un progetto in grado di verificare se la persona inquadrata indossava o meno la mascherina.

Fig. 1
La pagina web di Google API developer.



In questo filone si inserisce l'articolo qui proposto, il quale presenterà uno strumento in grado di azionare un attuatore se verrà riconosciuta e ritenuta valida la targa del veicolo inquadrata con la telecamera.

Le applicazioni sono innumerevoli: dal semplice garage di casa, fino alla gestione di un ingresso aziendale, in modo tale da permettere al dipendente di entrare senza attesa.

Addirittura in una nostra prova, al posto dell'attuatore, è stata inserita la comunicazione diretta con il garage attraverso il protocollo MQTT, particolarmente adatto alla domotica.

Una base che offre diversi spunti per la personalizzazione da parte di chi tra voi se la sentirà.

Per quest'oggetto abbiamo utilizzato niente di più di un ESP32-CAM AI-Thinker, un sensore PIR, due

LED (uno per segnalare l'azione corrente ed uno per simulare l'attuatore) ed alcune chiamate REST verso Google per aiutare il nostro hardware, sicuramente poco propenso a fare l'elaborazione delle immagini, nel suo compito.

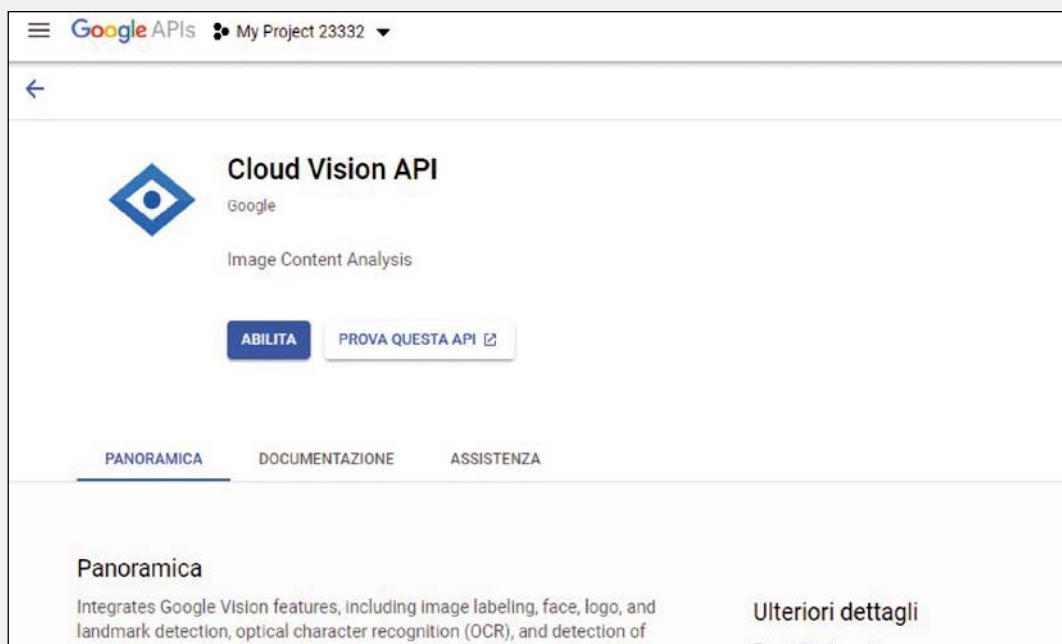
PRINCIPIO DI FUNZIONAMENTO

Come interagiscono fra loro questi dispositivi?

Il sensore di movimento PIR rileva la presenza di un veicolo o di un ostacolo, a quel punto interviene la camera che scatta la foto della targa e la invia ad un nostro server di appoggio.

Il compito di quest'ultimo è di fare, in prima battuta, il forward della richiesta verso l'API di Google. Una volta ricevuta in risposta la targa del veicolo (sotto forma di caratteri alfanumerici), il nostro server controlla sul proprio database o file locale

Fig. 2
La scheda dell'API che ci interessa.



se il veicolo è stato riconosciuto o no.

Una procedura abbastanza semplice che, così costruita, crea meno impatto a livello computazionale per la ESP32-CAM, lasciando tutta la post-elaborazione al server ed a Google.

Questo dettaglio è una grande svolta; infatti sfruttando la rete si va a creare un sistema distribuito, che permette di dividere adeguatamente il carico per l'elaborazione e l'attuazione riuscendo a sfruttare nelle terminazioni oggetti computazionalmente meno performanti.

Ma quali sono i passi da compiere per creare questo sistema distribuito?

Ebbene, prima di tutto è necessario abilitare le API di Google per l'elaborazione della nostra immagine. Successivamente è necessario creare una pagina web sul nostro server di appoggio da poter chiamare con la nostra camera.

Ed infine dovremo caricare il firmware, che vedremo nel dettaglio, sulla nostra ESP32-CAM, così da scattare le foto quando il sensore PIR rileva del movimento.

GOOGLE

Innanzitutto per poter usufruire dei servizi di "big G" è necessario possedere un account Google. Andando sul browser, una volta loggati, inseriamo nella URL questo link:

<https://console.developers.google.com>

allorché ci troveremo davanti una pagina web come quella mostrata in **Fig. 1** che è quella della Google API developer.

Clicchiamo in alto sulla sinistra sulla dicitura "Seleziona un Progetto" e nel popup che comparirà selezioniamo la voce "Nuovo Progetto".

Seguiamo i passaggi indicati, inserendo un nome e cliccando su "Crea", per tornare alla schermata iniziale.

Ora, da qui selezioniamo la voce "Libreria" nel menu che troviamo sulla sinistra: verranno mostrati una serie di servizi sottoforma di rettangoli o tab. A noi interessa "Cloud Vision API"; selezionandolo apriremo la scheda di questa API (**Fig. 2**) nella quale andremo a cliccare sul pulsante "Abilita".

Una volta fatto ciò, abbiamo abilitato l'API corretta per il nostro servizio di riconoscimento oggetti, ora mancano solo alcuni passaggi per poter generare la chiave con la quale potremmo accedere al servizio mediante chiamate REST (REpresentational State Transfer).

Clicchiamo il logo di Google in alto a sinistra per

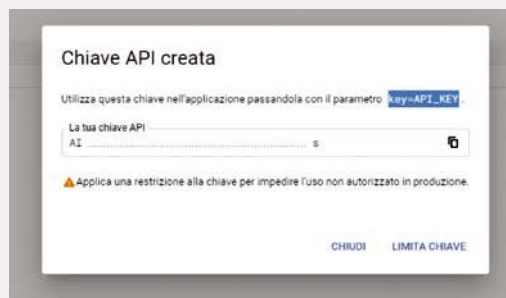


Fig. 3
La chiave API che ci occorre.

tornare alla schermata iniziale e da lì questa volta andremo a premere la voce "Credenziali" sempre all'interno del menu laterale.

Scegliamo la voce "Crea credenziali" e successivamente "Chiave API" ottenendo come risultato qualcosa di simile a quanto mostrato nella **Fig. 3**. Ovviamente per questa chiave possono essere impostati dei limiti di chiamata, ma evitiamo questa gestione e ci limitiamo ad aver ottenuto la nostra chiave.

Infine per poter abilitare totalmente il servizio bisogna inserire un metodo di fatturazione nel caso in cui si sfori il numero massimo di transazioni del piano gratis.

Ci sono da fare alcune precisazioni su questo servizio: infatti Google fissa una quota di 1000 interazioni al mese per questa API, superate le quali si deve sostenere un costo che decresce in base all'uso. A scopo didattico/personale è un limite molto alto, ma se si pensa di adottare questo sistema in un parcheggio o un posto pubblico sarà necessario affrontare una spesa.

Di seguito rechiamoci su "Collega un account di fatturazione" e dopo una prima conferma, seguiamo le istruzioni e completiamo i passaggi.

Per testare la chiave possiamo utilizzare un qualsiasi generatore di chiamate REST.

Un software abbastanza conosciuto è sicuramente Postman, uno strumento utilissimo per testare le API in modo molto semplice e intuitivo senza la necessità di creare a mano le classiche classi di test. Con Postman potremmo effettuare la chiamata e verificare la correttezza delle operazioni svolte senza scrivere alcuna riga di codice.

È sufficiente settare una chiamata come nella

Fig. 4, inserendo i seguenti parametri:

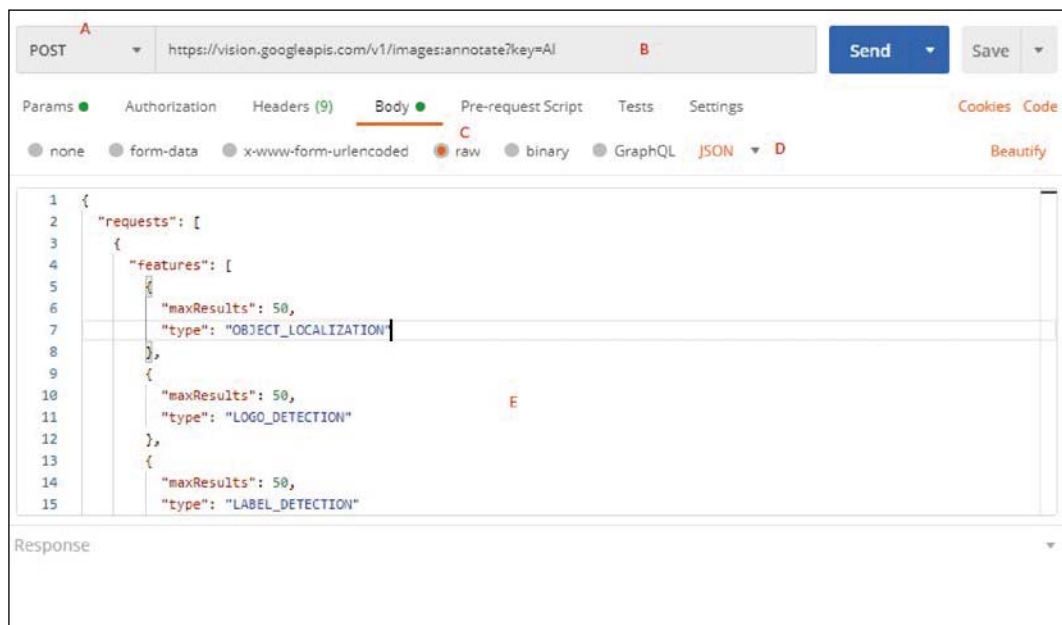
A => POST

B => https://vision.googleapis.com/v1/images:annotate?key=LA_CHIAVE_GENERATA_AL_PASSO_PRECEDENTE

C => raw



Fig. 4
La chiamata REST.



D => JSON

E => il JSON che trovate nel **Listato 1**, sostituendo al tag IMAGE_BASE64 l'immagine che volete far scansionare a Google nel formato Base64.

In risposta, se tutto è stato impostato correttamente, riceveremo un JSON contenente un array di oggetti response contenenti vari dettagli, dal modello della macchina alla targa.

Questa API e questi settaggi possono essere usati anche per determinare quali oggetti sono presenti in una scena o per altri scopi sempre legati alla computer vision.

SERVER

Questa entità avrà il compito di legare i due attori cardini dell'intero sistema, ovvero Google e il nostro modulo camera. Potremmo definirla come un trasportatore di informazioni, ma in realtà la sua funzione non si ferma qui: infatti oltre a questo, elabora la risposta in ritorno da Google, interpretandola, e controllando se la targa è presente tra quelle autorizzate.

Questa sezione dovrà esporre un pagina web verso l'esterno e per far questo esistono due soluzioni. Una prima soluzione, mirata ad un utente molto più esperto, può essere quella di crearsi da solo lo spazio di hosting. In questo caso si potrà usare una semplice Raspberry Pi sulla quale possiamo per semplicità montare una distribuzione server e nella quale installeremo un'architettura LAMP.

Listato 1

```
{
  "requests": [
    {
      "features": [
        {
          "maxResults": 50,
          "type": "OBJECT_LOCALIZATION"
        },
        {
          "maxResults": 50,
          "type": "LOGO_DETECTION"
        },
        {
          "maxResults": 50,
          "type": "LABEL_DETECTION"
        },
        {
          "maxResults": 50,
          "type": "DOCUMENT_TEXT_DETECTION"
        }
      ],
      "image": {
        "content": " IMAGE_BASE64"
      },
      "imageContext": {
        "cropHintsParams": {
          "aspectRatios": [
            0.8,
            1,
            1.2
          ]
        }
      }
    }
  ]
}
```



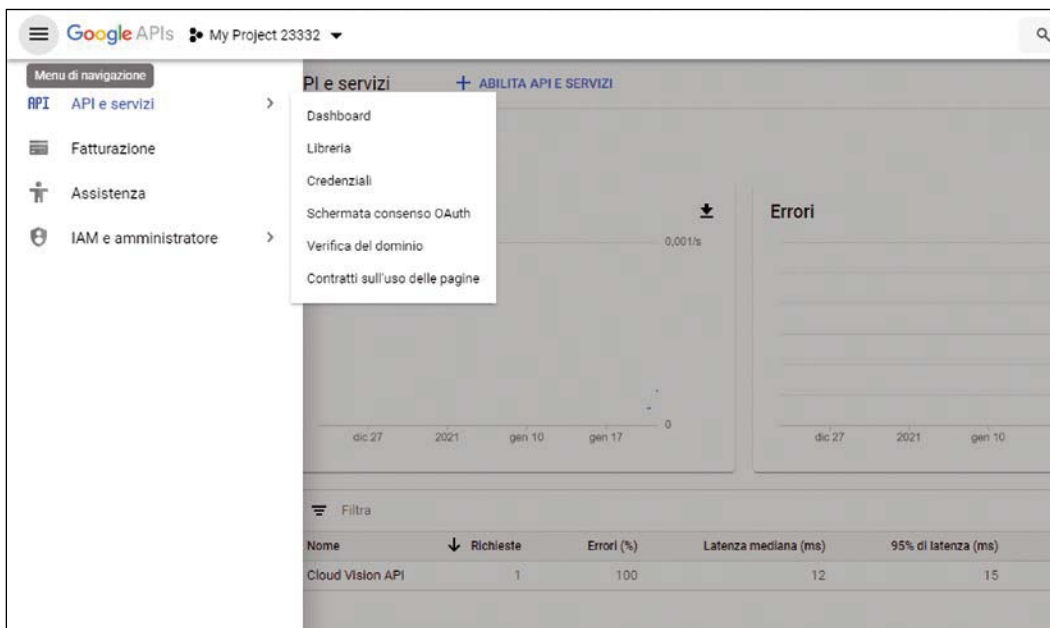


Fig. 5
La gestione
dei file nel
sito di test.

LAMP è l'acronimo delle componenti software che la compongono, ovvero Linux, Apache, MySQL ed infine PHP. Oltre alla precedente configurazione l'utente dovrà munirsi anche di un indirizzo statico al quale associare l'indirizzo locale del suo nuovo RaspServer. Questa associazione solitamente è fornita da un servizio ddns (Dynamic-DNS). Una seconda via è sicuramente quella di acquistare un servizio già pronto all'uso; a tale riguardo evidenziamo come la Mondadori Media (<https://it.altervista.org>) metta a disposizione una piattaforma, a scopo didattico, nella quale è possibile confezionare siti o avere servizi di hosting semplicemente registrandosi gratuitamente. Questa alternativa è molto valida sia per i neofiti, sia per chi, nonostante sia in possesso di ampie competenze, vuole partire subito e evitare ogni problema o "seccatura" di configurazione. Accedendo dunque alla pagina it.altervista.org e cliccando sul tasto "Crea sito" e seguendo le istruzioni è possibile iscriversi al servizio. In particolare, nella seconda schermata consiglio di cliccare su tasto "Vuoi solo il servizio di hosting?", per evitare che venga creata la piattaforma per il sito vero e proprio ma venga lasciato tutto lo spazio libero per poter sviluppare. Una volta scelto il nome e finita la registrazione dovremo solo aspettare la e-mail di conferma, all'interno della quale troveremo Username e Password per un primo accesso. Terminata questa prima parte di configurazione

accediamo al nostro servizio attraverso il login e una volta all'interno clicchiamo su "Gestione file" (**Fig. 5**). Ora possiamo visualizzare, creare ed eliminare i vari file. La parte server espone una pagina web (*getPhoto.php*), costruita per semplicità in php, ed un file di testo. Quest'ultima è una semplificazione dell'apparato per permettere un facile starter anche a chi magari è meno avvezzo a lavorare con entità come database o server. Creiamo il file di testo chiamato *data.txt* nella stessa cartella della pagina, e al suo interno inseriamo le varie targhe:

ED485KC
TR123TP

Vediamo nel dettaglio come si compone questa pagina:

```
<?php
$uploaddir = ".";
$uploadfile = $uploaddir . basename( $_
FILES['photo']['name'] . " " );
if(move_uploaded_file($_FILES['photo']['tmp_name'],
$uploadfile)){
    $data = file_get_contents("." . $_FILES['photo']
['name'] . " " );
```

Nella prima parte andremo a ricevere il file proveniente dalla videocamera e tramite un gioco di set e get, salvare e riprendere il file.

↓ Listato 2

```
$curl = curl_init();

curl_setopt_array($curl, array(
    CURLOPT_URL => 'https://vision.googleapis.com/v1/images:annotate?key=GOOGLE_API_KEY',
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_ENCODING => '',
    CURLOPT_MAXREDIRS => 10,
    CURLOPT_TIMEOUT => 0,
    CURLOPT_FOLLOWLOCATION => true,
    CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
    CURLOPT_CUSTOMREQUEST => 'POST',
    CURLOPT_POSTFIELDS => '{
        "requests": [
            {
                "features": [
                    {
                        "maxResults": 50,
                        "type": "OBJECT_LOCALIZATION"
                    },
                    {
                        "maxResults": 50,
                        "type": "LOGO_DETECTION"
                    },
                    {
                        "maxResults": 50,
                        "type": "LABEL_DETECTION"
                    },
                    {
                        "maxResults": 50,
                        "type": "DOCUMENT_TEXT_DETECTION"
                    }
                ],
                "image": {
                    "content": "'.base64_encode($data).'"
                },
                "imageContext": {
                    "cropHintsParams": {
                        "aspectRatios": [
                            0.8,
                            1,
                            1.2
                        ]
                    }
                }
            }
        ]
    }',
    CURLOPT_HTTPHEADER => array('Content-Type: application/json'),
));

$response = curl_exec($curl);

curl_close($curl);
```

↓ Listato 3

```
$resObj = json_decode($response, true);
if(isset($resObj["responses"]) && count($resObj["responses"])>0 && (isset($resObj["responses"][0]
["textAnnotations"]) && count($resObj["responses"][0]["textAnnotations"])>0 && isset($resObj["responses"]
[0]["textAnnotations"][0]["description"])))
    && (isset($resObj["responses"][0]["labelAnnotations"]) &&
searchCar($resObj["responses"][0]["labelAnnotations"]))) ){
    $tg = searchTG($resObj);
    $res = explode(PHP_EOL, $tg);
    if($res[0]==""){
        $res = $res[1];
    }else{
        $res = $res[0];
    }
    $stringLicSea = str_replace(' ', ' ', trim(preg_replace('/\s\s+/', ' ', $res)));
```



Questo è un sistema di codifica che consente la conversione di dati binari in stringhe di testo ASCII. Inoltre la chiave non è immagazzinata nel client, ma nel server e questo implica una maggior sicurezza e protezione da eventuali attacchi, per esem-

Una volta ricevuta la risposta da Google, andiamo quindi a convertire il testo in un oggetto, perché più facile da maneggiare. Di questo controlliamo che sia realmente una targa ed estrapoliamo il dato pulito, senza spazi o altri caratteri particolari. Riportiamo qui di seguito le due funzioni di ricerca veicolo e ricerca targa.

```
$datab = file_get_contents("./data.txt");
$datab = explode(PHP_EOL, $datab);
foreach($datab as $item){
```

```
function searchTG($resObj){
    $qlt_h = 768;
    $qlt_w = 1024;
    $res = null;
    $rect = new Tmp();
    foreach($resObj["responses"][0]["localizedObjectAnnotations"] as $value){
        if($value["name"]=="License plate"){
            $res = $value;
            break;
        }
    }
}
```


Listato 4 - Segue

```
if($res!=null){
    foreach($resObj["responses"][0]["textAnnotations"] as $valuep){
        if(($valuep['boundingPoly']['normalizedVertices'][0]['x']*$qlt_w)<=$valuep['boundingPoly']['vertices'][0]['x'] &&
            ($valuep['boundingPoly']['normalizedVertices'][0]['y']*$qlt_h)<=$valuep['boundingPoly']['vertices'][0]['y'] &&
            ($valuep['boundingPoly']['normalizedVertices'][2]['x']*$qlt_w)>=$valuep['boundingPoly']['vertices'][2]['x'] &&
            ($valuep['boundingPoly']['normalizedVertices'][2]['y']*$qlt_h)>=$valuep['boundingPoly']['vertices'][2]['y'])){
            $res = $valuep["description"];
            break;
        }
    }
    return $res;
}
return "";
}

function searchCar($arrayLabel){
    $res = 0;
    foreach($arrayLabel as $value){
        if($value["description"]=="Vehicle registration plate"){
            $res++;
            if($res==1){
                break;
            }
        }
    }
    if($res==1){
        return true;
    }
    return false;
}

class Tmp{}
?>
```

```
if($stringLicSea==$item){
    echo "open";
    exit;
}
echo "close";
}else{ echo "close"; }
}else{
    echo "close";
}
```

Ricerchiamo all'interno del file, creato inizialmente, un record simile a quello della targa fotografata. In caso positivo rispondiamo *open*, altrimenti *close*. Chiaramente tutte le label posso essere cambiate ma in accordo con il codice del client.

In alternativa a queste ultime righe, se siamo utenti più esperti e abbiamo già avuto a che fare con strutture dati più complesse, possiamo optare per una ricerca tramite query a database. La piattaforma di Altervista ci può essere d'aiuto anche in questa operazione, offrendo un servizio,

anch'esso completamente gratis, di database remoto con un'interfaccia molto semplice con cui operare una chiamata PhpMyAdmin. Nel dettaglio, in questo articolo vi descriveremo l'implementazione di una connessione ad un database MySQL. Come primo passo riportiamo, nel **Listato 4**, lo script da utilizzare per la creazione della tabella del database e per l'inserimento in essa di alcuni dati di prova.

Nella prima funzione, quella per la ricerca della targa, ci sono due valori: *\$alt_h* e *\$alt_w*. Questi rappresentano la risoluzione della foto, quindi se viene cambiata la parte client che invia la foto, editando la qualità, va modificata anche in queste variabili.

HARDWARE

Come già utilizzato in un precedente articolo la configurazione hardware del client si compone di un modulo ESP32-CAM (**Fig. 6**). Questo hardware è composto da un modulo ESP32, da una telecamera, un alloggiamento per SD-Card e tutto il necessario



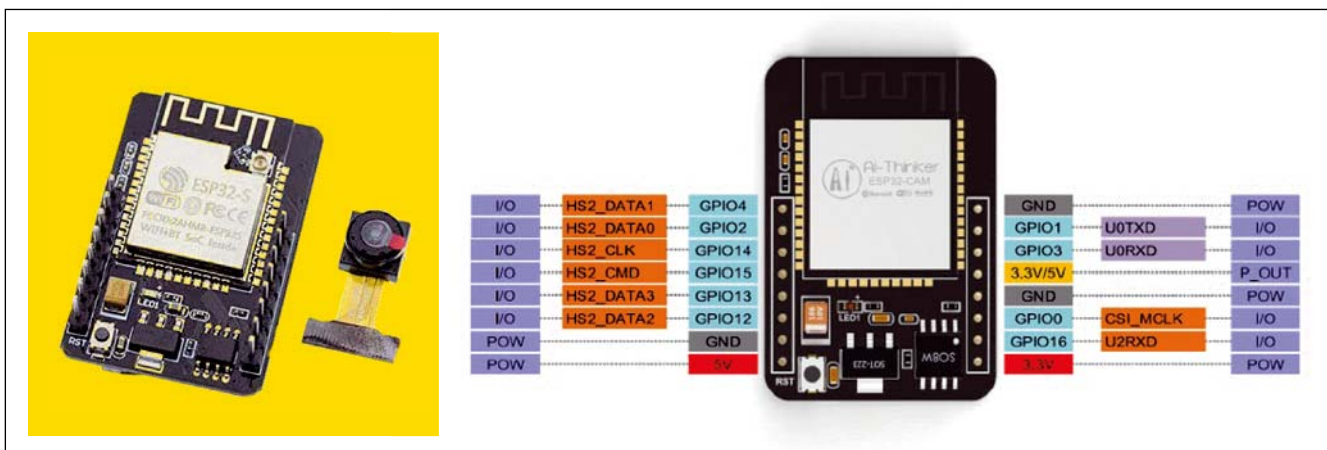


Fig. 6
L'hardware del progetto.

per far funzionare questi componenti. Nel dettaglio possiamo notare che il circuito ESP32CAM si basa sull'ESP32, un microcontrollore programmabile con WiFi e Bluetooth integrati ed un'ulteriore RAM esterna di ben 4MB. Non solo, il nuovo connettore per la fotocamera può ospitare un modulo OV2640 o OV7670: il primo viene fornito con il modulo stesso e presenta una risoluzione di 2 Megapixel. Sulla scheda è presente anche un LED, ad alta luminosità, che può essere impiegato come flash o illuminatore della scena da riprendere. Naturalmente i numerosi GPIO montati sulla scheda ci permettono eventualmente di montarne uno esterno nel caso di scarsa illuminazione. Per piccole applicazioni questo chip si presenta molto robusto e con una grande potenza di elaborazione, basandosi su due core di elaborazione a 32 bit da 120 MHz. Quest'ultima caratteristica gli permette di avere un frame-rate abbastanza alto, ovviamente in funzione del formato e delle dimensioni: indicativamente si riesce ad arrivare ad un throughput fino a 8 JPEG in SVGA (800x600) al secondo. Con questo prototipo scatteremo foto con una risoluzione di 1024x768, ma in caso di problemi di rete o di latenze nella comunicazione la risoluzione può essere tranquillamente scalata. Inoltre, secondo le prove fatte, sarebbe opportuno installare l'antenna sul ESP32 in modo tale da poter sfruttare una maggiore potenza di rete. Ovviamente alla telecamera sarà affiancato un sensore di movimento PIR (**Fig. 7**), per poter rilevare la presenza di un eventuale veicolo. Il comportamento del PIR è molto semplice: nella nostra configurazione parte con un segnale basso (≈ 0) e una volta che rileva l'ostacolo alza il suo segnale (≈ 1). Grazie a questo sensore riusciamo

a minimizzare il numero di foto scattate, come accennato qualche paragrafo indietro e quindi riusciamo ad ottimizzare la concretezza di questo strumento. Vediamo ora come poter collegare i vari attori che compongono il nostro sistema client in modo efficiente. Come prima cosa colleghiamo la ESP32-CAM ad un Arduino UNO che useremo come programmatore. In questo passaggio può essere utilizzato un convertitore USB/TTL al posto dell'Arduino UNO, rendendo il cablaggio, almeno in fase di programmazione estremamente più facile. Come primo passaggio identifichiamo i pin di TX e RX sulla nostra ESP32-CAM, chiamati rispettivamente U0TXD e U0RXD e li andremo a collegare con i pin TX e RX dell'Arduino UNO (o ai pin TX e RX del convertitore USB/TTL). Nel modulo della camera questi pin rappresentano anche i pin di GPIO1 e GPIO3. Collegheremo i pin GND e GPIO0 di quest'ultimo a massa, insieme ai pin GND e RESET di Arduino e i due a 5V fra di loro. Naturalmente per tutti i contatti da porre a massa o al 5V possiamo aiutarci con una breadboard, come



Fig. 7
Il sensore PIR che rileva le auto.

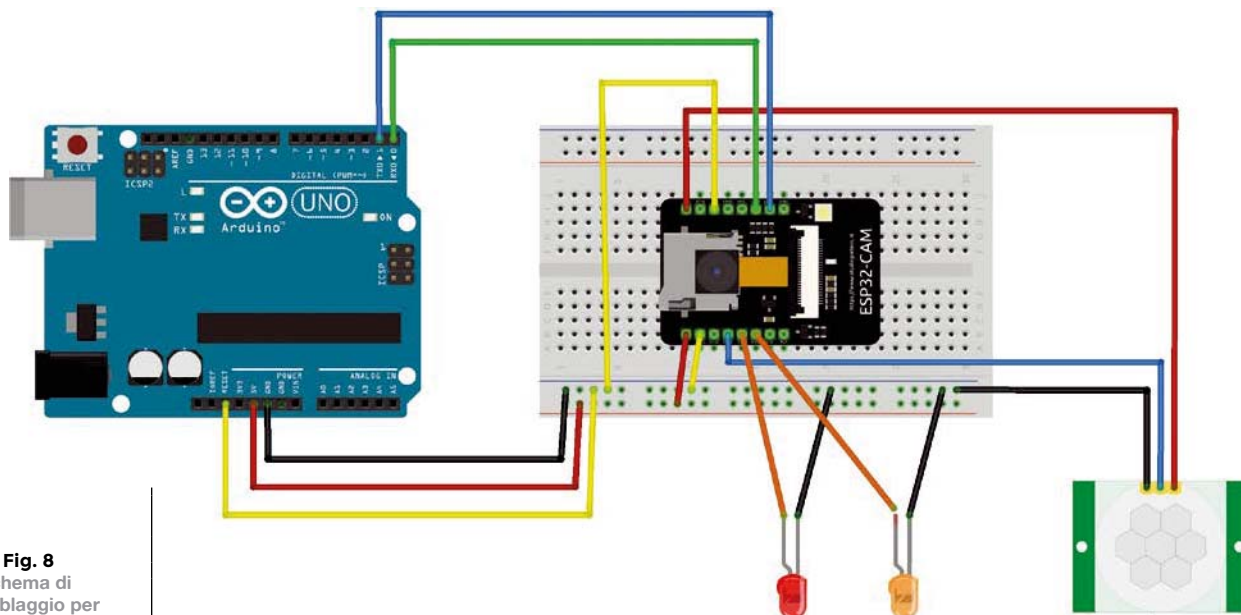


Fig. 8
Schema di cablaggio per la programmazione

pure per le altre connessioni. Riassumendo in forma tabellare quanto espresso precedentemente, si hanno le seguenti corrispondenze tra linee di Arduino e pin della ESP32:

TX Arduino -> U0TXD ESP32
RX Arduino -> U0RXD ESP32
GND Arduino -> GND ESP32
RESET Arduino -> GPIO0 ESP32
RESET Arduino -> GND Arduino ed ESP32

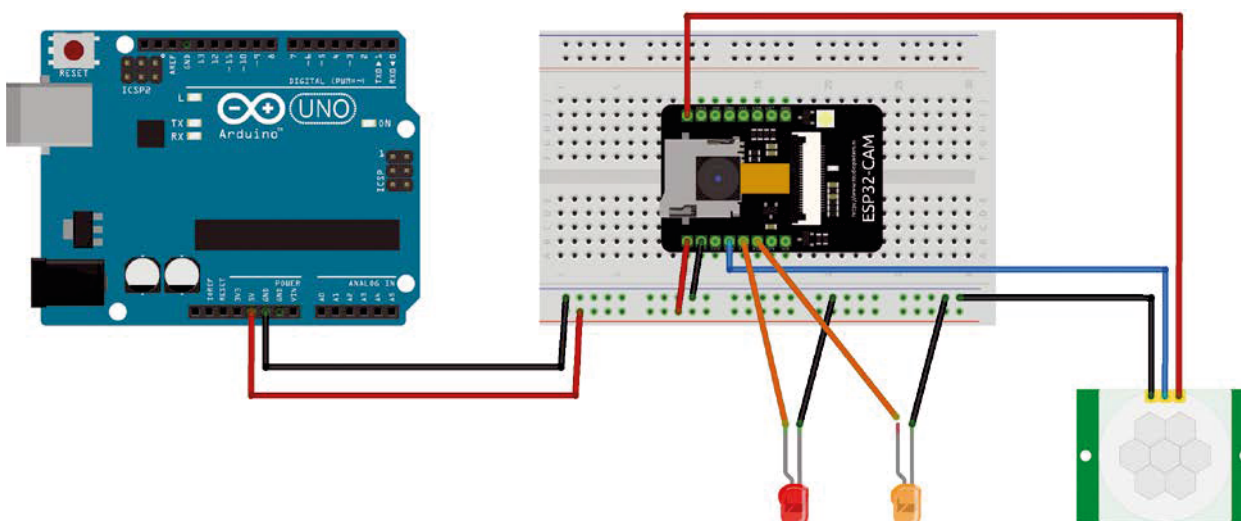
Fig. 9
Schema di cablaggio per l'utilizzo.

che poi significa quanto proposto nella **Fig. 8**. Oltre a queste connessioni, bisogna relizzare quelle dell'alimentazione +5V, ovvero unire tutti i fili positivi del sensore PIR, della ESP32-CAM e

della scheda Arduino UNO. I due LED terminano a massa (GND).

Aggiungiamo ora il collegamento con il PIR: oltre all'alimentazione, quindi 3V3 e GND ai rispettivi VCC e GND del sensore, porteremo il cavo per il segnale di attuazione, partendo dal pin GPIO13. Affianchiamo a quest'ultimo anche un LED, sul pin GPIO15, in modo tale da poter avere un feedback visivo nelle varie fasi di elaborazione.

Esp32->PIR
Esp32->LED
3V3->VCC
GND->GND
GPIO13->OUT



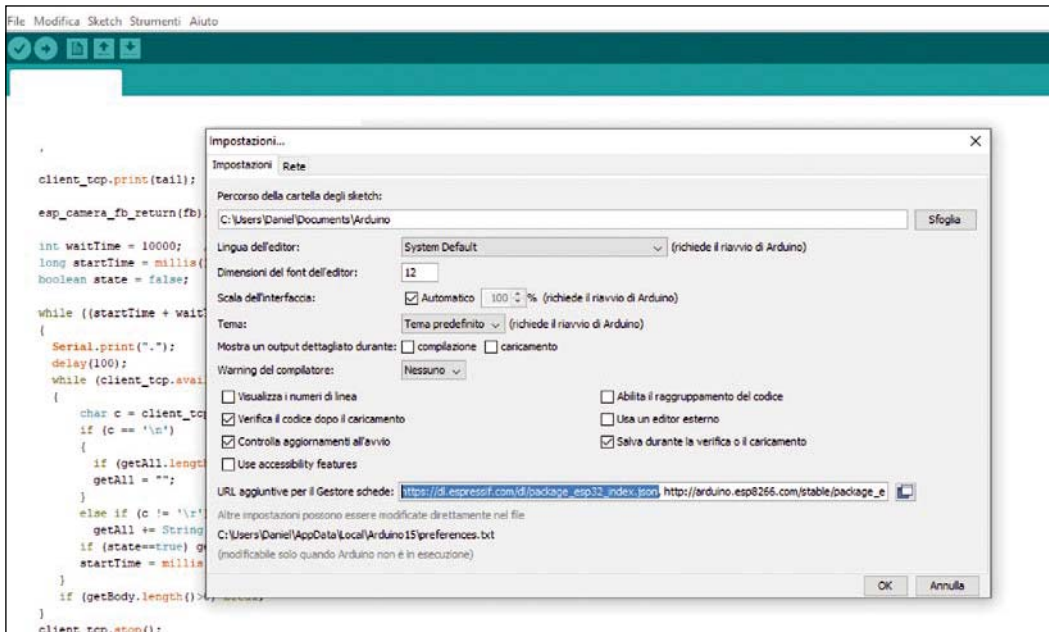


Fig. 10
Aggiunta dei driver per la ESP32-CAM nell'IDE Arduino.

Lo schema di cablaggio del sistema per l'utilizzo a regime è quello proposto nella **Fig. 9**; in pratica corrisponde al circuito finito e programmato.

Una volta che abbiamo programmato il nostro modulo ESP32-CAM, dobbiamo assolutamente rimuovere il collegamento GPIO0→GND.

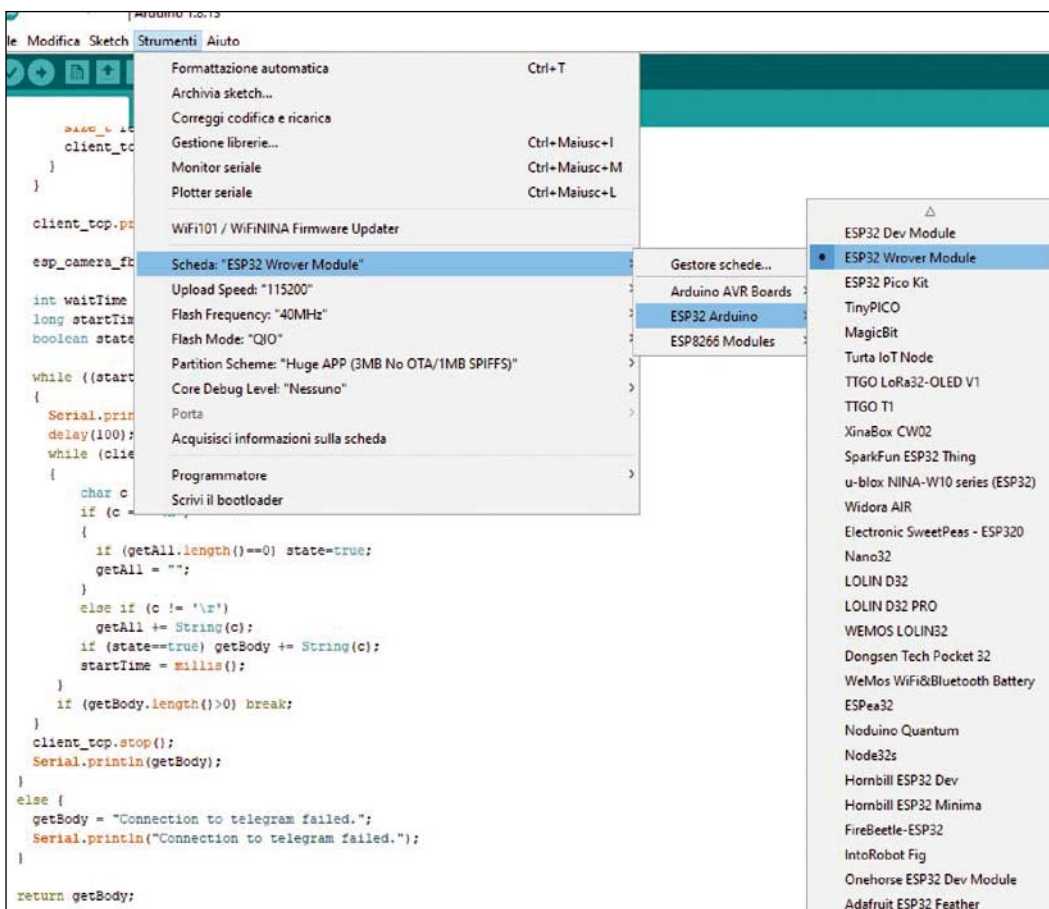


Fig. 11
Scelta dell'hardware nell'IDE Arduino.

Listato 5

```
void setup(){
    WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);

    Serial.begin(115200);
    delay(10);
    pinMode(gpioLED, OUTPUT);
    pinMode(gpioATTUATORE, OUTPUT);
    WiFi.mode(WIFI_STA);
    Serial.println("");
    Serial.print("Connecting to ");
    Serial.println(ssid);
    WiFi.begin(ssid, password);
    long int StartTime=millis();
    while (WiFi.status() != WL_CONNECTED){
        delay(500);
        if ((StartTime+10000) < millis()) break;
    }

    Serial.println("");
    Serial.println("STAIP address: ");
    Serial.println(WiFi.localIP());
    Serial.println("");

    if (WiFi.status() != WL_CONNECTED){
        Serial.println("Reset");

        ledcAttachPin(4, 3);
        ledcSetup(3, 5000, 8);
        ledcWrite(3,10);
        delay(200);
        ledcWrite(3,0);
        delay(200);
        ledcDetachPin(3);
        delay(1000);
        ESP.restart();
    }
    else{
        ledcAttachPin(4, 3);
        ledcSetup(3, 5000, 8);
        for (int i=0;i<5;i++) {
            ledcWrite(3,10);
            delay(200);
            ledcWrite(3,0);
            delay(200);
        }
        ledcDetachPin(3);
    }
}

camera_config_t config;
config.ledc_channel = LEDC_CHANNEL_0;
config.ledc_timer = LEDC_TIMER_0;
config.pin_d0 = Y2_GPIO_NUM;
config.pin_d1 = Y3_GPIO_NUM;
config.pin_d2 = Y4_GPIO_NUM;
config.pin_d3 = Y5_GPIO_NUM;
config.pin_d4 = Y6_GPIO_NUM;
config.pin_d5 = Y7_GPIO_NUM;
config.pin_d6 = Y8_GPIO_NUM;
config.pin_d7 = Y9_GPIO_NUM;
config.pin_xclk = XCLK_GPIO_NUM;
config.pin_pclk = PCLK_GPIO_NUM;
config.pin_vsync = VSYNC_GPIO_NUM;
config.pin_href = HREF_GPIO_NUM;
config.pin_sscb_sda = SIOD_GPIO_NUM;
config.pin_sscb_scl = SIOC_GPIO_NUM;
config.pin_pwdn = PWDN_GPIO_NUM;
config.pin_reset = RESET_GPIO_NUM;
config.xclk_freq_hz = 20000000;
config.pixel_format = PIXFORMAT_JPEG;

if(psrRamFound()){
```

FIRMWARE ESP32-CAM

Prima di procedere alla programmazione dovremmo impostare ArduinoIDE, ovvero il nostro ambiente di sviluppo, per poter scrivere correttamente sul modulo.

Apriamo quindi ArduinoIDE e andiamo nel menu *File->Impostazioni* e aggiungiamo negli URL aggiuntivi per i gestori delle schede questo link https://dl.espressif.com/dl/package_esp32_index.json e facciamo clic su OK (Fig. 10).

Se si dispone già di uno o più URL si usa la virgola per concatenarli, proprio come nell'esempio. Ora scarichiamo correttamente il gestore della scheda: rechiamoci in *Strumenti->Scheda: XXXX->Gestore schede...*, cerchiamo "esp32" e installiamo l'unico risultato della ricerca. (Fig. 7)

Uscendo selezioniamo nuovamente *Strumenti->Scheda: XXXX* e noteremo una nuova voce ESP32 Arduino, la quale al suo interno ha diverse altre scelte. Selezioniamo *ESP32 Wrover Module*, comparirà una nuova riga sempre in quel menu: *Partition scheme*, all'interno del quale andremo a scegliere *Huge APP* (Fig. 11).

Ora abbiamo tutto il necessario per poter caricare il firmware sul dispositivo, colleghiamo quindi l'USB del nostro Arduino UNO e selezioniamo la porta virtuale creata da sistema operativo e clicchiamo *Carica*. Vediamo ora come si compone il codice da uploadare sul nostro device, il quale è suddiviso per le macro aree GLOBALI, FUNZIONE DI SETUP e FUNZIONE DI LOOP.

Nella prima parte andremo a dichiarare tutte quelle variabili globali che useremo all'interno del programma:

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

#include <WiFi.h>
#include <WiFiClientSecure.h>
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"
#include "esp_camera.h"

//CAMERA_MODEL_AI_THINKER
#define PWDN_GPIO_NUM    32
#define RESET_GPIO_NUM  -1
#define XCLK_GPIO_NUM    0
#define SIOD_GPIO_NUM    26
#define SIOC_GPIO_NUM    27
#define Y9_GPIO_NUM      35
#define Y8_GPIO_NUM      34
#define Y7_GPIO_NUM      39
```

↓ Listato 5 - Segue

```
config.frame_size = FRAMESIZE_VGA;
config.jpeg_quality = 10;
config.fb_count = 2;
}else{
    config.frame_size = FRAMESIZE_QQVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
}

// camera init
esp_err_t err = esp_camera_init(&config);
if (err != ESP_OK){
    Serial.printf("Camera init failed with error 0x%x", err);
    delay(1000);
    ESP.restart();
}

sensor_t * s = esp_camera_sensor_get();
s->set_framesize(s, FRAMESIZE_XGA);
}
```

```
#define Y6_GPIO_NUM    36
#define Y5_GPIO_NUM    21
#define Y4_GPIO_NUM    19
#define Y3_GPIO_NUM    18
#define Y2_GPIO_NUM    5
#define VSYNC_GPIO_NUM 25
#define HREF_GPIO_NUM  23
#define PCLK_GPIO_NUM  22
```

```
int gpioPIR = 13;
int gpioLED = 15;
int gpioATTUATORE = 14;
```

Nella seconda parte andremo a inizializzare il modulo camera e ci andremo a connettere all'hotspot WiFi per il quale abbiamo configurato le credenziali nel passo precedente (**Listato 5**).

Come si può notare, nel caso in cui non riesca a

↓ Listato 6

```
String callServerCheck(){
    const char* myDomain = "SERVER_SITE ";
    String getAll="", getBody = "";

    camera_fb_t * fb = NULL;
    fb = esp_camera_fb_get();
    if(!fb){
        Serial.println("Camera capture failed");
        delay(1000);
        ESP.restart();
        return "Camera capture failed";
    }

    WiFiClientSecure client_tcp;
    if (client_tcp.connect(myDomain, 443)){
        Serial.println("Connected to " + String(myDomain));

        String head = "--India\r\nContent-Disposition: form-data; name=\"chat_id\"; \r\n\r\n chat-id-123456789\r\n--India\r\nContent-Disposition: form-data; name=\"photo\"; filename=\"esp32-cam.jpg\"\r\n\r\nContent-Type: image/jpeg\r\n\r\n";
        String tail = "\r\n--India--\r\n";

        uint16_t imageLen = fb->len;
        uint16_t extraLen = head.length() + tail.length();
        uint16_t totalLen = imageLen + extraLen;

        client_tcp.println("POST /getPhoto.php HTTP/1.1");
        client_tcp.println("Host: " + String(myDomain));
        client_tcp.println("Content-Length: " + String(totalLen));
        client_tcp.println("Content-Type: multipart/form-data; boundary=India");
        client_tcp.println();
        client_tcp.print(head);

        uint8_t *fbBuf = fb->buf;
        size_t fbLen = fb->len;

        for (size_t n=0;n<fbLen;n=n+1024){
            if (n+1024<fbLen){
                client_tcp.write(fbBuf, 1024);
                fbBuf += 1024;
            }
        }
    }
}
```

Listato 6 - Segue

```
        else if (fbLen%1024>0){
            size_t remainder = fbLen%1024;
            client_tcp.write(fbBuf, remainder);
        }
    }

    client_tcp.print(tail);
    esp_camera_fb_return(fb);

    int waitTime = 10000; // timeout 10 seconds
    long startTime = millis();
    boolean state = false;

    while ((startTime + waitTime) > millis()){
        Serial.print(".");
        delay(100);
        while (client_tcp.available())
        {
            char c = client_tcp.read();
            if (c == '\n')
            {
                if (getAll.length()==0) state=true;
                getAll = "";
            }
            else if (c != '\r')
                getAll += String(c);
            if (state==true) getBody += String(c);
            startTime = millis();
        }
        if (getBody.length()>0) break;
    }
    client_tcp.stop();
    //Serial.println(getBody);
}
else{
    getBody = "Connection to server failed.";
    Serial.println("Connection to server failed.");
}

return getBody;
}

void loop(){
    pinMode(gpioPIR, INPUT_PULLUP);
    int v = digitalRead(gpioPIR);
    if (v==1){
        digitalWrite(gpioLED, HIGH);
        delay(1000);
        digitalWrite(gpioLED, LOW);
        delay(1000);
        String vari = callServerCheck();
        if(vari.indexOf("open") > 0){
            digitalWrite(gpioATTUATORE, HIGH);
            delay(3000);
            digitalWrite(gpioATTUATORE, LOW);
            digitalWrite(gpioLED, HIGH);
            delay(3000);
            digitalWrite(gpioLED, LOW);
        }
        delay(5000);
    }
}
```

connettersi al WiFi o non riesca ad inizializzare la camera il dispositivo si riavvierà: per questo è importante accertarsi di aver cablato tutto in maniera corretta e inserito le credenziali giuste, in modo da

evitare il verificarsi di un bootloop.

Inoltre come accennato precedentemente, la risoluzione dell'immagine può essere settata dal codice alla riga:

```
config.jpeg_quality = 10;
```

Qui si può spaziare da 0 a 63, rammentando che i numeri bassi indicano maggiore risoluzione (e di conseguenza migliore qualità delle immagini) e viceversa.

Infine vi è la porzione responsabile del funzionamento del dispositivo a regime, il cui codice è quello riportato nel **Listato 6**.

In questa porzione troviamo due funzioni: la prima, `callServerCheck`, è responsabile dell'acquisizione dell'immagine e dell'invio di questa verso il server; la seconda parte non fa altro che eseguire ciclicamente un'azione di controllo del sensore radard a infrarossi passivi. In caso di ostacolo oppure di oggetto in movimento che viene rilevato, passa subito il controllo alla prima.

POSSIBILI EVOLUZIONI

Nell'esempio in questione andiamo a pilotare un pin esterno (nel dettaglio GPIO14), alzando il suo stato logico per una decina di secondi, ma nulla ci vieta di aggiungere maggiore complessità, come per esempio andare a pilotare l'apertura di un cancello motorizzato, della serranda o basculante di un garage o attivare un relé intelligente che supporti il protocollo MQTT.

Cerchiamo di comprendere meglio cosa significa e quali possibilità offre questo protocollo: alcuni dispositivi sono comparabili a dei "mini server" che espongono un'interfaccia per poterli usare da remoto senza dover essere fisicamente presenti. Questi solitamente comunicano attraverso un protocollo MQTT, che si basa su almeno due attori: un broker, responsabile dello smistamento dei vari messaggi ed uno o più publisher o client, i quali si litimitano ad inviare o ricevere messaggi.

In breve accade che il publisher si collega al broker e può cominciare da subito ad inviare messaggi con un determinato TOPIC verso quest'ultimo, il quale rilancia il messaggio verso gli altri client collegati a quel determinato argomento.

MQTT è quindi un protocollo altamente ottimizzato ed usato da molti dispositivi per IoT.

Per quanto riguarda il codice, la modifica è abbastanza semplice perché è sufficiente aggiungere le seguenti righe nelle sezioni già descritte del codice in esecuzione sul dispositivo.





Il sistema può essere disposto di fronte alla sbarra di accesso ad un parcheggio.

Nelle variabili GLOBALI:

```
#include <PubSubClient.h>
WiFiClient espClient;
PubSubClient mqttClient(espClient);
```

Nella funzione di SETUP:

```
char* server = "SERVER_SITE_MQTT";
mqttClient.setServer(server, MQTT_PORT);
boolean rc = mqttClient.connect("myClientID");
```

o in caso di connessione protetta:

```
boolean rc = mqttClient.connect("myClientID", "myUsername", "myPassword");
```

Nella funzione di LOOP:

```
if(vari.indexOf("open") > 0){
    mqttClient.publish("TOPIC", "COMMAND");
}
```

Questa è comunque una delle tante modifiche possibili al codice; nulla vieta di realizzarne altre finalizzate a specifiche applicazioni.

CONCLUSIONI

Bene, con questo abbiamo terminato la descrizione del nostro sistema di riconoscimento di targhe che sfrutta l'elaborazione in cloud di Google per fare il "grosso" del lavoro e ottenere le lettere e i numeri identificati nel fotogramma passato dalla

ESP32-CAM tramite Internet.

Ricordiamo che il progetto è un'applicazione demo attraverso la quale vi abbiamo spiegato come, sfruttando servizi in Rete e su Cloud di grandi provider, è possibile svolgere operazioni complesse e che se svolte localmente avrebbero richiesto grandi risorse di calcolo, utilizzando hardware semplici capaci, però, di interagire con Internet per inviare richieste ed elaborare le risposte.

La nostra applicazione può essere personalizzata e adattata in base alle proprie esigenze, magari cambiando il sensore e/o l'attuatore per utilizzare come evento trigger una condizione differente dall'avvicinamento di un'automobile e per attivare dispositivi di vario genere, come anche passare il segnale di riconoscimento a una successiva elaborazione.



Cosa occorre?

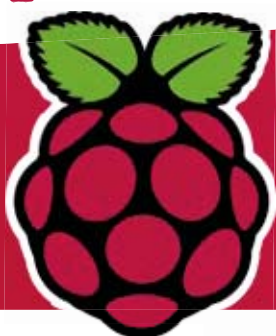
I componenti utilizzati in questo progetto sono disponibili presso Futura Elettronica. La board con ESP32 e camera (cod. ESP32-CAM) è in vendita a Euro 16,50, il modulo PIR (cod. PIRMOD) è disponibile a Euro 5,00. I prezzi si intendono IVA compresa.

Il materiale va richiesto a:

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

Raspberry Pi 4

**UTILIZZALA
SUBITO CON
I NUOVI KIT**



Starter kit completo con Raspberry Pi 4 da 2 GB

Prezzi IVA inclusa.



€ 89,00

Cod. RPI4FULLG2

- » Board Raspberry Pi 4 modello B con 2GB di RAM.
- » Alimentatore 5 VDC / 3 A uscita type C e interruttore.
- » Contenitore plastico per Raspberry Pi 4 tipo B, colore bianco e rosso.
- » Cavo da HDMI (tipo A) a micro HDMI (tipo D), lunghezza 1,5 metri.
- » Micro SD card da 32GB con sistema operativo NOOBS precaricato.



Disponibile anche con Raspberry Pi 4 modello B da 4GB di RAM **COD. RPI4FULLG4 € 105,00**
e con Raspberry Pi 4 modello B da 8GB di RAM **COD. RPI4FULLG8 € 129,00**

Starter kit con Raspberry Pi 4 da 2 GB



€ 75,00

Cod. RPI4STARTERG2

- » Board Raspberry Pi 4 modello B con 2GB di RAM.
- » Alimentatore 5 VDC / 3 A uscita type C e interruttore.
- » Contenitore plastico per Raspberry Pi 4 tipo B, colore bianco e rosso.



Disponibile anche con Raspberry Pi 4 modello B da 4GB di RAM **COD. RPI4STARTERG4 € 89,00**
e con Raspberry Pi 4 modello B da 8GB di RAM **COD. RPI4STARTERG8 € 115,00**



L'immediato futuro vede ogni genere di dispositivo connesso e se questo può essere fonte di una gran quantità di dati fruibili dalla Rete, l'esistenza di una gran quantità di nodi periferici rappresenta un'infinità di accessi disponibili per portare attacchi informatici. Analizziamo il problema e valutiamo le possibili soluzioni.

La diffusione dell'Internet delle cose rappresenta un'opportunità unica di sviluppo e crescita sotto molti aspetti, che toccano l'evoluzione della diagnostica, l'analisi dei comportamenti, lo studio delle abitudini ai fini del marketing, il miglioramento delle condizioni di vita in generale, il monitoraggio e l'analisi di processi e di apparati tecnologici distribuiti, ma tanto altro ancora; ma l'IoT passa attraverso la diffusione massiccia di sensori in ogni ambito, perché servono a raccogliere quei dati (anche noti come Big Data) che permettono poi l'analisi finalizzata all'applicazione nei succitati settori (e non solo) e tale diffusione implica altrettante potenziali "porte aperte" o backdoor, come si usa dire nel gergo della Sicurezza IT, che gli hacker possono sfruttare per portare dei cyber attacchi ai server che presiedono, tipicamente in Cloud, alle reti IoT. Infatti una miriade di sensori connessi significa anche una capillare e sconfinatamente ampia rete informatica, attaccabile con una certa facilità (perché la cosiddetta "superficie attaccabile" è vastissima) se non vengono adottate strategie ed architetture a prova di penetrazione e non vengono corrette le vulnerabilità. In questo articolo proveremo a fare il punto sull'architettura e la tecnologia dell'IoT, focalizzando il discorso sulle problematiche di sicurezza che si portano dietro.

CHE COS'È L'IoT

Internet delle cose (Internet of Things) significa dotare dispositivi e soprattutto oggetti di uso comune ed anche indossabili, di quell'elettronica che permette loro di rilevare (mediante sensori e dispositivi di digitalizzazione) dati dall'ambiente circostante o dal corpo o veicolo cui vengono applicati e trasmetterli tramite Internet a server di raccolta e di elaborazione, tipicamente in Cloud Computing. Ma significa anche connettere a server remoti terminali, erogatori, interfacce utente, macchinari e apparati di ogni genere attraverso Internet a una rete di computer che provvede alla raccolta dei dati e talvolta ad impartire comandi. La connessione alla Rete è, per la natura dei dispositivi, quasi totalmente wireless, il che implica la possibilità di penetrarla, da parte dei cyber-attaccanti, senza dovervi entrare in contatto, ma sfruttando connessioni radio come il WiFi.

PERCHÉ L'IoT È A RISCHIO SICUREZZA

Si tratta di un pericolo concreto perché un ipotetico attaccante potrebbe introdursi nella comunicazione con una certa facilità, sia perché non sempre il progettista dei sistemi ha l'accortezza di adottare algoritmi di protezione e cifratura sicuri, sia perché una connessione radio è per sua natura più vulnerabile, dato che non serve andare localmente ad attaccare un cavo per cercare di entrarvi. Peraltro esistono sul web parecchi tool nati ufficialmente per condurre dei test di penetrabilità (i cosiddetti Penetration Test) al fine di identificare le falle di sicurezza e le vulnerabilità per risolverle, ma troppo spesso impiegati dagli hacker per entrare in un sistema informatico e da lì scolarlo fino ad arrivare a conoscerne il contenuto, violarlo, sottrarlo o renderlo inutilizzabile per poi chiedere un riscatto (è il caso dei ransomware e dei cryptolocker) in denaro per sbloccare i dati. Tra questi tool c'è KaliLinux, una distribuzione Linux nata per simulare cyber attacchi a vari livelli, che raggruppa tutta una serie di tool altrimenti rintracciabili in Rete distintamente, da configurare e via di seguito. La principale problematica insita nell'IoT è quindi rendere sicura la comunicazione ed evitare intrusioni, identificare e chiudere le backdoor; l'importanza si comprende considerando che un sistema IoT parte da una serie di dispositivi periferici e da un eventuale edge computer (un elaboratore posto sul luogo dove si trovano sensori e apparati IoT egde) che ne pre-elabora i dati per compattarli e inviare sulla linea dati il minimo payload possibile al fine di limitare l'occupazione di banda (e di risorse sulla piattafor-

ma Cloud cui si appoggia il sistema) per poi arrivare a dei server spesso virtualizzati in ambiente Cloud. Se un attaccante riesce a penetrare la connessione, magari sottraendo l'identità a un dispositivo che ha determinati privilegi di accesso, può portare un attacco informatico sia di tipo verticale, sia di tipo orizzontale: per verticale si intende che dalla linea Internet che accede al server Cloud e superando l'eventuale firewall perimetrale può penetrare sempre più in profondità fino al server principale; per orizzontale significa che una volta entrato in una "macchina" può propagare l'attacco a quelle adiacenti, collegate sulla stessa VLAN dell'ambiente Cloud, del cluster o del rack.

Quindi, una volta fatta l'enumerazione delle risorse (ciò dopo aver esplorato il sistema ed aver catalogato server, storage, interfacce di rete) presenti nel sistema può concentrarsi sulle più interessanti, muovendosi per portare attacchi mirati, per esempio attraverso una privilege escalation che consiste nel cercare di ottenere i privilegi di accesso come quelli di utenti esistenti di cui si ruba l'identità o, peggio ancora, dell'amministratore, che ha, per così dire, potere di vita o di morte sull'infrastruttura attaccata. Tra gli attacchi più pericolosi si annoverano anche gli Exploit, per i quali esistono tool specifici (sempre in rete) come Metasploit. Quindi entrando da un dispositivo IoT poco protetto, paradossalmente si può portare attacchi ai server con cui si interfaccia; considerato poi, che nel Cloud si ha a che fare con Macchine Virtuali (VM o Virtual Machine che dir si voglia) ovvero server simulati con sistemi operativi virtualizzati su un sistema operativo ospite e attraverso un hypervisor (per esempio VMware) o ambienti di virtualizzazione come VirtualBox, si pone il problema dell'isolamento delle risorse, ovvero di come il Cloud Provider ha studiato l'isolamento tra macchine virtuali, le quali non sono computer fisici ma software in esecuzione su una stessa macchina. Spesso non si tratta neppure di VM, ma di applicazioni Cloud eseguite su container, quindi unità ancor più astratte e che di fatto sono isolate a livello software ma non nella realtà fisica e delle risorse utilizzate; tanto per dare l'idea, due macchine virtuali possono risiedere su uno stesso disco rigido ed essere separate dall'hypervisor, il quale potrebbe non riuscire a isolare un attacco informatico ricevuto da una delle VM.

Quando si parla di IoT non si intende, però, solamente il rapporto fra end-device e server Cloud ma anche di problematiche originate dalla proliferazione di dispositivi domestici connessi e integrati nelle



funzioni domotiche, nella videosorveglianza wireless ecc. e comunque affacciati al web, magari per attingere a funzioni gestite da remoto da parte dei fornitori dei prodotti. In questo caso le vulnerabilità critiche creano un rischio elevato, perché non solo è possibile violare i dispositivi e la loro connessione wireless, ma anche accedervi da Internet inserendosi nella connessione con il fornitore dei servizi; si possono quindi carpire immagini e suoni degli ambienti, codici di blocco delle porte e dei sistemi d'allarme ecc. I dispositivi domestici intelligenti non protetti, o nodi, sono un bersaglio interessante per gli hacker, poiché si intromettono nelle reti domestiche, rubano e utilizzano in modo improprio i dati personali.

Tra gli attacchi che possono essere portati ci sono anche i DDoS, dove l'hacker sfrutta host casuali per colpire un obiettivo specifico. Più specifico dell'ambiente IoT è l'attacco Mirai, che mira a prendere il controllo dei dispositivi connessi al fine di distribuirli in attacchi denial of service (DDoS) non autorizzati e distribuiti in modo massiccio.

PIATTAFORME IoT E SICUREZZA

Per proteggersi dagli attacchi, i grandi provider di soluzioni per IoT hanno implementato accorgimenti in grado di garantire la necessaria sicurezza; questo significa che per poter scambiare i dati con essi, i dispositivi che ci si vogliono interfacciare devono rispondere a un protocollo di sicurezza e autenticazione.

Tra questi provider si trovano Amazon, Microsoft, Google, che esigono un'autenticazione anche sofisticata da parte dei dispositivi IoT che vogliono accedere ai sistemi. Per soddisfare tale requisito, i produttori di semiconduttori e moduli per IoT implementano nei propri prodotti i protocolli di autenticazione richiesti dalla connessione alle suddette piattaforme, allo scopo di rendere sicuri i dispositivi IoT edge.

Sicurezza a livello di applicazione

A questo livello, in cui vengono forniti il backend e le applicazioni utente, si implementano controlli di accesso tramite autorizzazioni, ruoli, password complesse, crittografia in transito, registrazione e monitoraggio.

La crittografia è a livello Transport Layer Security (TLS, **Fig. 1**) che garantisce che il traffico tra dispositivo e server sia indecifrabile da qualsiasi potenziale intercettatore. Il TLS viene utilizzato comunemente per accedere ai siti web e prevede un'autenticazione sicura, durante la

sessione, dell'host (non del computer o cliente che vi si connette) attraverso uno username e una password. Siccome tali credenziali possono essere sottratte da un hacker, per elevare il livello di sicurezza si ricorre ai certificati; il certificato utilizza la crittografia asimmetrica, che implica una separazione dei ruoli: la parte che rilascia il certificato (CA=autorità di certificazione) garantisce il collegamento tra il dispositivo fisico e la chiave pubblica, la qual da sola non è sufficiente. Inoltre, il verificatore non ottiene mai nulla di valore (come la password) per essere in grado di autenticare il dispositivo. Questo offre un livello di sicurezza molto più elevato, ma sfortunatamente eleva la complessità dei sistemi.

La CA dispone di un certificato radice e di una chiave privata che devono essere custoditi con cura; la chiave può essere utilizzata per generare una CA intermedia con lo scopo di firmare altre chiavi, ad esempio per i dispositivi IoT.

I certificati per autenticare i dispositivi sono formati dalle coppie di chiavi privata e pubblica e sono scritti in ogni dispositivo in fabbrica (in una ROM) ovvero creati e salvati in un'area di memoria Flash protetta.

Affinché i dispositivi possano autenticarsi al Cloud IoT, le loro chiavi pubbliche devono essere caricate nel Cloud stesso. Per eseguire la mutua autenticazione, il dispositivo dovrà memorizzare la propria chiave privata, chiave pubblica, uno stack TLS con mutua autenticazione, un certificato con la chiave pubblica dell'endpoint cui connettersi nel Cloud. Per elevare il livello di sicurezza è conveniente che i dispositivi periodicamente cambino le chiavi.

Fig. 1
Autenticazione
tramite TLS.

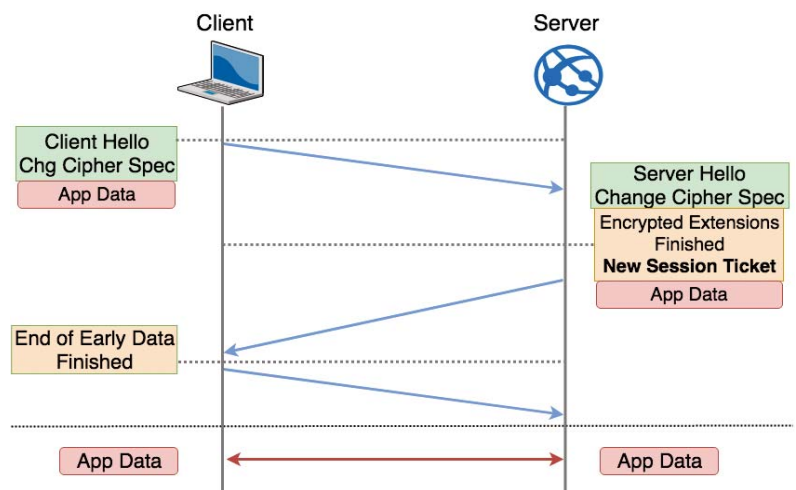




Fig. 2
Un gateway smart home dotato di chip di sicurezza permette di gestire dispositivi come applicazioni domotiche e telecamere di videosorveglianza.

Sicurezza a livello di hardware

Archiviare in modo sicuro le chiavi private nei dispositivi IoT può risultare difficile; per elevare la sicurezza bisogna passare all'implementazione a livello hardware. Per questo occorre integrare nei dispositivi un chip in grado di memorizzare in modo sicuro le chiavi stesse: si tratta di un processore di sicurezza dotato di funzionalità anti-manomissione che bloccano tutti i tentativi di hackerare il dispositivo per carpirne le chiavi.

In generale sempre più architetture delegano a un chip specifico la gestione della sicurezza delle connessioni.

Il chip di sicurezza può essere integrato in un gateway tramite cui un sistema domotico o di videosorveglianza (**Fig. 2**) sfruttando la connessione WiFi, si collegano a Internet; in questo modo è possibile bloccare tentativi di accesso fraudolento dalla Rete e alla rete, portato ad esempio da un hacker penetrato attraverso la connessione WiFi in dispositivi come termostati e telecamere di sorveglianza. Complice, anche, il fatto che i sistemi come le telecamere di sicurezza funzionano per periodi di tempo più lunghi ed è complicato eseguire aggiornamenti periodici.

Componenti del genere ne esistono diversi, sovente certificati dai Cloud IoT provider per la connessione dei dispositivi. In questo contesto va precisato che esistono processori multicore che integrano crittografia e memoria ad accesso sicuro in un SoC (System-on-Chip); si tratta di microcontrollori abilitati alla crittografia.

In alternativa è possibile affiancare al processore di una piattaforma esistente un coprocessore di sicurezza, che poi è il chip di sicurezza ed elementi

crittografici, che operano attraverso l'integrazione di crittografia, autenticazione e archiviazione sicura dei dati (**Fig. 3**).

Un coprocessore di sicurezza è un microcontrollore a basso costo che consente anche a nodi IoT piccoli ed economici di implementare potenti metodi crittografici per l'autenticazione reciproca e la derivazione della chiave di sessione; solleva il microprocessore o microcontrollore principale dal lavoro di elaborazione crittografica, che a sua volta porta a un minore consumo energetico e un tempo di autenticazione più veloce. I coprocessori di sicurezza contengono chiavi e certificati di sicurezza. Altro esempio è nella famiglia Microchip SAML10/11 di MCU a 32 bit low-power, capaci di offrire sicurezza a livello di chip e basati sulla tecnologia Arm TrustZone. Le MCU integrano un'ampia varietà di periferiche, le funzionalità di sicurezza, clock a 32 MHz, fino a 64 kB di Flash e 16 kB di SRAM.

Il micro SAM L11-KPH aggiunge una radice della chiave di attendibilità fornita in fabbrica per assegnare alla MCU un'identità sicura che può essere utilizzata per lo sviluppo di applicazioni IoT sicure. SAM L11 e SAML11-KPH sono supportati da un ecosistema di sicurezza completo che include la soluzione Kinibi-M della Trustonic e l'architettura Secure Deploy della Secure Thingz.

Gli MCU SAM L11 sono PSA Certified Livello 1, che garantisce l'implementazione delle migliori pratiche di sicurezza integrate per i prodotti IoT, tra cui una protezione anti-manomissione a 256 byte su RAM, True Random Number Generator (TRNG), standard AES di codifica avanzata, Secure Hash Algorithm (SHA) e storage sicuro delle chiavi. Altra soluzione di Microchip è il CEC1702: si tratta di un microcontrollore basato su architettura ARM Cortex-M4 con integrata una soluzione completa abilitata per la crittografia hardware, che consente l'avvio sicuro del firmware di sistema. Questo microcontrollore a 32 bit a basso consumo ma potente e programmabile, protegge i dati attraverso la crittografia a chiave pubblica e la convalida, garantendo che il firmware sia stato firmato digitalmente e non sia stato alterato.

La sua suite di crittografia hardware riduce i tempi di elaborazione di ordini di grandezza rispetto alle soluzioni software e, ad esempio, fornisce un miglioramento delle prestazioni da 20x a 50x per l'accelerazione PKE, nonché un miglioramento 100x per la crittografia/decrittografia.

La famiglia CEC1702 comprende il dispositivo CEC1702Q-B2-I / SX (libero) e il CEC1702Q-S1-I/X

che va utilizzato con il firmware Soteria personalizzato, disponibile in base a un contratto di licenza firmato (SLA). Il processore conta su 480 kB di SRAM per codice e dati, su una robusta suite di crittografia hardware e 2,5 kB bit di memoria OTP programmabile dall'utente.

L'avvio protetto fornisce una radice di attendibilità basata sull'hardware. Il CEC1702 può sostituire o integrare il microcontrollore esistente nel device IoT perché è un micro completo di funzionalità di autenticazione e crittografia semplici e intuitive per le applicazioni connesse, che possono sia essere fornite nella modalità coprocessore di sicurezza, sia integrate nell'applicazione.

AMAZON AWS IoT

AWS IoT è una piattaforma Cloud gestita da Amazon Web Services che consente ai dispositivi connessi di interagire facilmente e in sicurezza con le applicazioni Cloud e altri dispositivi. Affinché i dispositivi possano interagire con AWS IoT tramite il protocollo MQTT (Message Queue Telemetry Transport) che è quello tipico dell'IoT, devono prima autenticarsi utilizzando l'autenticazione reciproca TLS e quindi scambiandosi le chiavi (handshake). Per archiviare queste chiavi in modo sicuro sui dispositivi IoT da connettere ad AWS IoT si può contare su chip di sicurezza come l'ATECC508A della Microchip, che dispone di un meccanismo di archiviazione delle chiavi sicuro: una volta memorizzata una chiave nell'ATECC508A, il suo contenuto non può essere letto. Il chip realizza questo con funzioni di accelerazione crittografica basate su hardware che gli consentono di eseguire operazioni crittografiche molto rapidamente e a basso consumo. Microchip può precaricare i certificati sull'elemento di sicurezza durante la produzione, prima della consegna. La combinazione di questa funzionalità con il supporto di AWS IoT per autorità di certificazione personalizzate e la registrazione just-in-time può semplificare il provisioning e la sicurezza dei dispositivi. Per facilitare utilizzo e integrazione dell'ATECC508A nei dispositivi che si connetteranno ad AWS IoT, Microchip rende disponibile un kit di valutazione chiamato Zero Touch Secure Provisioning Kit, che include un microcontrollore SAM G55 Cortex-M4, il chip di sicurezza ATECC508A e un modulo 802.11 b/g/ ad alta efficienza energetica ATWINC1500 e viene fornito con istruzioni per iniziare a lavorare con AWS IoT. I settori attenti all'affidabilità, tra cui l'automazione domestica e degli edifici, stanno adottando sempre più soluzioni hardware incentrate su processore

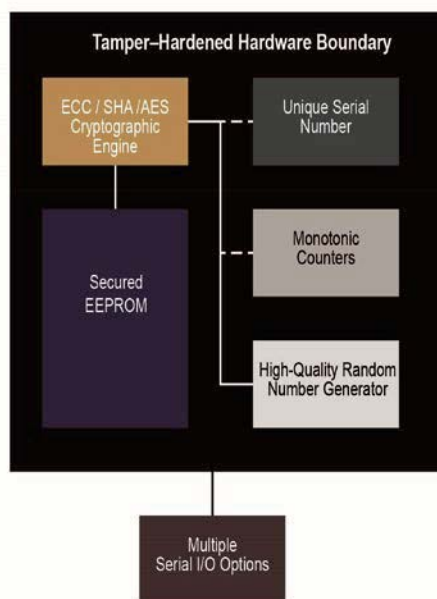


Fig. 3
Schema a blocchi di un coprocessore di sicurezza: si distinguono i blocchi encryption, authentication e data-storage (Microchip).

di sicurezza, che dev'essere integrato nelle prime fasi del ciclo di progettazione, quindi la scelta della piattaforma hardware cui affidarsi è fondamentale ed è la prima da fare.

Proteggere i dispositivi IoT è importante, ma ancora più importante è la protezione dell'elemento di raccolta dei dati di questi dispositivi per esempio un server web o un servizio Cloud, come Amazon Web Services (AWS). In altre parole, l'implementazione della sicurezza è fondamentale a più livelli in tutto il progetto, dal sensore al nodo della Smart Home, al collegamento del servizio Cloud.

Fig. 4
Il chip SAML 1011 Microchip implementa la sicurezza delle connessioni.



Fig. 5
L'integrato
Mediatek per
la cifratura e
l'autenticazione
di dispositivi IoT
al Cloud Azure
Sphere.



Questo ci porta a un aspetto critico della sicurezza negli ambienti di automazione domestica, ossia l'autenticazione del dispositivo al Cloud. Come accennato, un tipo di autenticazione da dispositivo a Cloud è il TLS, tuttavia se è implementato a livello software, non avere una memoria sicura per chiavi e dati sensibili consente agli hacker di sfruttare bug del software e penetrare nella memoria del microcontrollore o microprocessore in cui risiedono dati sensibili. D'altra parte, i co-processori di sicurezza che memorizzano chiavi private, certificati e dati sensibili in hardware sicuro facilitano un rafforzamento dei ben noti stack di implementazione TLS, come Secure Socket Layer (SSL) sotto forma di OpenSSL e wolfSSL. La protezione avanzata consente agli sviluppatori di eliminare le vulnerabilità del software creando livelli di sicurezza hardware aggiuntivi. Un coprocessore di sicurezza offre il modo più semplice per implementare uno stack TLS sicuro. Oltre a consentire l'hardening TLS, un coprocessore di sicurezza consente a un nodo IoT l'autenticazione a un servizio Cloud senza un ritardo percepibile dall'utente.

MICROSOFT AZURE SPHERE

Azure Sphere è una piattaforma applicativa Cloud protetta con funzionalità di comunicazione e di protezione integrate per i dispositivi connessi a Internet; implementa un servizio di sicurezza che stabilisce una connessione sicura tra un dispositivo fisico (un microcontrollore o un microprocessore) e il servizio cloud Azure di Microsoft. In pratica l'hardware IoT garantisce il secure boot (avvio sicuro) delle applicazioni e delle connessioni con il Cloud. Il secure boot è qualcosa che impedisce l'avvio di un sistema operativo se questo è stato alterato, impedendo, per esempio, che un nodo IoT corrotto da un attacco informatico possa instaurare una connessione con il Cloud.

In questo modo l'attacco malevolo ai dati diventa pressoché impossibile.

Come è per AWS IoT, la connessione tra un dispositivo Azure Sphere e il servizio di sicurezza del cloud Azure si basa su un sistema di autenticazione dell'identità del dispositivo, garantendo così l'integrità e l'attendibilità del software di sistema. Il servizio fornisce anche il canale protetto utilizzato da Microsoft per scaricare e installare automaticamente gli aggiornamenti del sistema operativo Azure Sphere e gli aggiornamenti delle applicazioni dell'utente sui dispositivi distribuiti. Il cuore della tecnologia risiede in un'unità microcontroller (MCU) protetta e connessa, in cui risiedono un sistema operativo personalizzato basato su Linux e un servizio di sicurezza basato sul Cloud Azure. Il primo MCU certificato Azure Sphere (e per il momento l'unico) è prodotto da Mediatek (<https://www.mediatek.com>). Include il chip Mediatek 3620, siglato MT3620 (**Fig. 5**). Tale componente, che poi è l'hardware di riferimento del corso Azure Sphere che presentiamo a partire da questo fascicolo di Elettronica In, integra:

- processore ARM Cortex-A7 (500 MHz);
- 2 sottosistemi I/O ARM Cortex-M4F (200 MHz);
- 5 interfacce UART;
- 1 interfaccia I²C (Inter-integrated Circuit) con clock del bus a 100 KHz, 400 KHz e 1 MHz;
- 1 interfaccia SPI (Serial Peripheral Interface) con clock fino a 40 MHz;
- 2 interfacce I²S (Inter-integrated System);
- 8 ingressi ADC;
- 12 contatori PWM;
- 72 pin GPIO;
- connettività WiFi.

Il chip componente MT3620 contiene anche il sottosistema di sicurezza Microsoft Pluton con un core Arm Cortex-M4F dedicato, che gestisce l'avvio e il funzionamento sicuro del sistema; il servizio di sicurezza Azure Sphere permette quindi ai microcontrollori dei device IoT una connessione sicura e garantisce che ogni dispositivo venga avviato solo con una versione autorizzata di applicazioni approvate. Inoltre, fornisce un canale protetto tramite cui Microsoft può scaricare e installare automaticamente gli aggiornamenti del sistema operativo per i dispositivi distribuiti sul campo, al fine di limitare i problemi di sicurezza.

Il chip Mediatek MT3620 si rivolge a un'ampia gamma di applicazioni IoT, tra cui smart home, IoT commerciale e industriale e molti altri ambiti, grazie al suo ampio sottosistema di periferiche I/O

che consente flessibilità e libertà di progettazione dei dispositivi.

Il processore per applicazioni ad alte prestazioni dell'MT3620 è un processore applicativo Arm Cortex-A7 che funziona fino a 500 MHz di clock e include cache L1 di grandi dimensioni e cache L2 e SRAM integrata per un funzionamento altamente efficiente su un'ampia gamma di applicazioni. Due sottosistemi I/O Arm Cortex-M4F per uso generico in esecuzione fino a 200 MHz supportano i requisiti delle periferiche on-chip.

Questi due sottosistemi I/O Cortex-M4F sono destinati principalmente a supportare l'elaborazione I/O in tempo reale, ma possono anche essere utilizzati per il calcolo e il controllo generale. I core Cortex-M4F possono eseguire qualsiasi sistema operativo fornito dall'utente finale o eseguire una "app bare metal" senza sistema operativo. La memoria Flash per supportare Cortex-A7 ed entrambi i processori Cortex-M4F è integrata nell'MT3620.

Il sottosistema di sicurezza Microsoft Pluton e il sottosistema WiFi dedicato sono al di fuori di questi tre core accessibili all'utente finale; il sottosistema radio WiFi 802.11a/b/g/n dual-band 1x1 è controllato da un core RISC Andes N9 a 32 bit dedicato; tale sottosistema contiene il transceiver radio in banda base e il MAC progettati per consentire applicazioni ad alto rendimento con grande efficienza energetica. Il funzionamento delle funzionalità di sicurezza MT3620 e della rete WiFi è isolato ed eseguito indipendentemente dalle applicazioni dell'utente finale. Solo le funzionalità hardware supportate da Azure Sphere sono direttamente accessibili agli utenti finali. Di conseguenza, le funzionalità di sicurezza e quelle del WiFi sono accessibili solo tramite API Azure Sphere definite e sono resistenti agli errori di programmazione nelle applicazioni degli utenti finali, indipendentemente dal fatto che queste applicazioni vengano eseguite sui core Cortex-A7 o

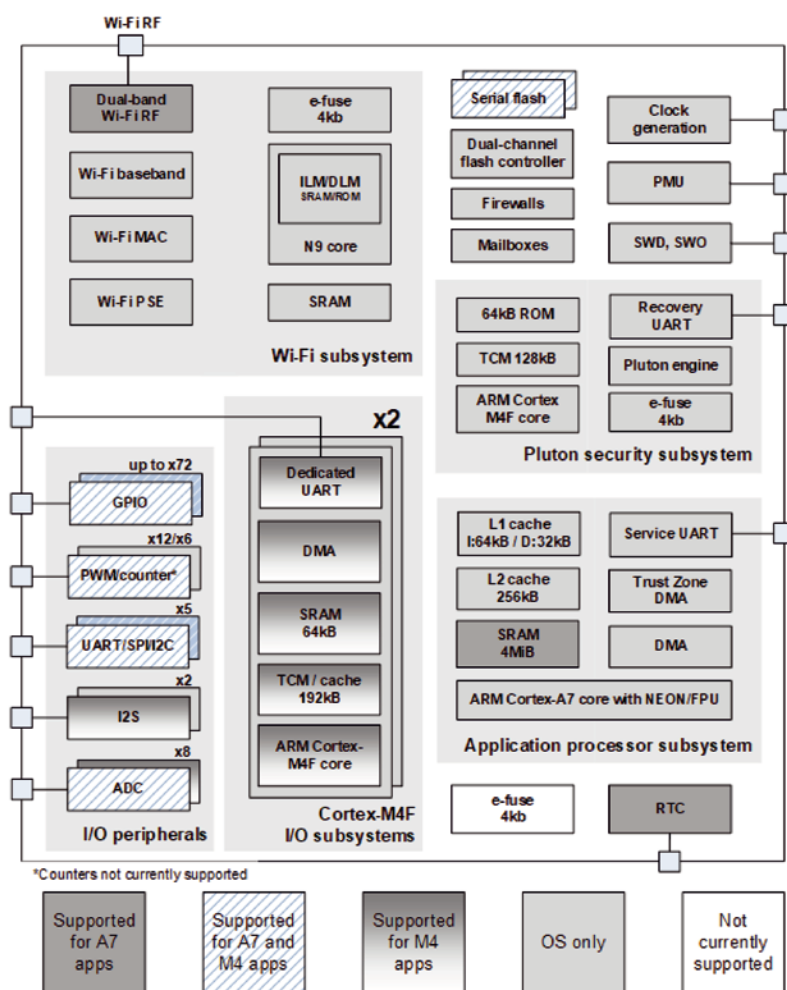


Fig. 6
Schema
a blocchi
del SoC
MediateK

Cortex-M4F accessibili dall'utente.

Microsoft fornisce un potente ambiente di sviluppo basato su Visual Studio che sfrutta il compilatore gcc, consentendo lo sviluppo delle applicazioni dei clienti in C.

Microsoft Azure Sphere MT3620 fornisce un livello di sicurezza senza precedenti per i dispositivi collegati. La soluzione sicura fornisce autenticazione e attestazione del dispositivo, supporta aggiornamenti software over-the-air remoti per mantenere la sicurezza di fronte agli attacchi in evoluzione e automatizza anche la registrazione e la segnalazione degli errori.

MediaTek NeuroPilot-Micro fa parte di NeuroPilot ed è progettato specificamente per applicazioni di apprendimento automatico miniaturizzate. Sfrutta gli strumenti di ottimizzazione NeuroPilot e fornisce framework di reti neurali popolari per l'inferenza. NeuroPilot-Micro SDK supporta TensorFlow Lite per microcontrollori (TFLM) e API di ottimizzazione per l'inferenza e l'ottimizzazione del modello. La **Fig. 5** propone lo schema a blocchi dell'MT3620. La maggior parte dei pin viene multiplexata tra I/O di utilizzo generico (GPIO) e altre funzioni: oltre ai pin GPIO elencati, i GPIO12-23 sono disponibili rispettivamente nei pin MT3620 27-38.

Alcuni pin presentano capacità di multiplexing tra GPIO, PWM (Pulse Width Modification, modifica della larghezza degli impulsi) e contatori hardware. Le funzioni GPIO attualmente supportate sono l'impostazione dell'output alto/basso e la lettura dell'input. Sono supportate anche le modalità di guida aperte di svuotamento/open source e il controllo del livello di attendibilità dell'unità. Gli interrupt esterni sono supportati nel solo core Arm Cortex M4 core, ma non nel core A7.

Se l'applicazione usa un controller PWM, tutti i pin associati a tale controller vengono allocati per l'uso come output PWM e nessuno di essi può essere usato come GPIO. L'hardware PWM può essere configurato per un clock 32 kHz, 2 MHz (XTAL/13) o 26 MHz (XTAL). Nel core di alto livello (A7), il driver Linux userà sempre il clock a 2 MHz; ciò comporta limitazioni del ciclo di servizio e del periodo nelle applicazioni di alto livello.

GOOGLE IOT

Anche questa piattaforma (quella di "Big G") implementa dei meccanismi di sicurezza che prevedono sia lo scambio di chiavi e la memorizzazione sicura in un co-processore dotato di funzionalità anti-manomissione, sia l'utilizzo di un JWT per l'autenticazione.

Anche in questo caso sono supportati coprocessori crittografici Microchip come l'ATECC608A, che non solo permette la memorizzazione sicura delle chiavi private, ma convaliderà anche il firmware e offrirà un processo di avvio più sicuro (secure boot) per il dispositivo.

L'ATECC608A offre molte funzionalità: per esempio la chiave privata viene generata dallo stesso elemento protetto, non da un'autorità certificatrice (CA); il chip utilizza un generatore di numeri casuali per creare la chiave, rendendo praticamente impossibile la decifrazione. La chiave privata non lascia mai il chip e utilizzando la chiave privata, il chip potrà generare una chiave pubblica che potrà essere firmata dalla CA scelta dalla piattaforma e dal cliente.

Microchip esegue questa firma elettronica in una struttura sicura dedicata collocata negli Stati Uniti, dove un impianto isolato e protetto memorizza le chiavi CA intermedie del cliente (produttore di dispositivi IoT) in un server altamente sicuro collegato alla linea di produzione.

Una volta che i dispositivi protetti hanno generato ciascuno le proprie coppie di chiavi, le chiavi pubbliche corrispondenti vengono inviate all'account Google Cloud del cliente e archiviate in modo sicuro nel gestore dispositivi Cloud IoT Core. Poiché Cloud IoT Core può archiviare fino a 3 chiavi pubbliche per dispositivo, la rotazione delle chiavi può essere eseguita con sicurezza senza problemi. Tutto ciò che il cliente deve fare è fornire una CA intermedia per un determinato lotto di dispositivi alla Microchip, che restituirà un reel di chip securizzati, che il produttore dei dispositivi IoT monterà poi sulle proprie schede.

Autenticazione mediante JWT

Il TLS è una soluzione adatta a proteggere la comunicazione tra dispositivi (PC, sistemi remoti) e il Cloud, ma lo stack di autenticazione non è l'ideale per l'IoT, in quanto lo stack richiesto per l'autenticazione reciproca è di grandi dimensioni e deve avere l'informazione su dove sono archiviate le chiavi, ovvero deve sapere quale elemento sicuro viene utilizzato e come comunicare con esso. Uno stack OpenSSL presumerà che le chiavi siano archiviate in un file system e debbano essere modificate per accedere all'elemento sicuro, ma ciò implica di eseguire sviluppo e test ad ogni aggiornamento dello stack. Con l'arrivo della TLS 1.3 è probabile che questo lavoro dovrà essere eseguito più volte, il che impone ai produttori di device IoT costi non indifferenti.



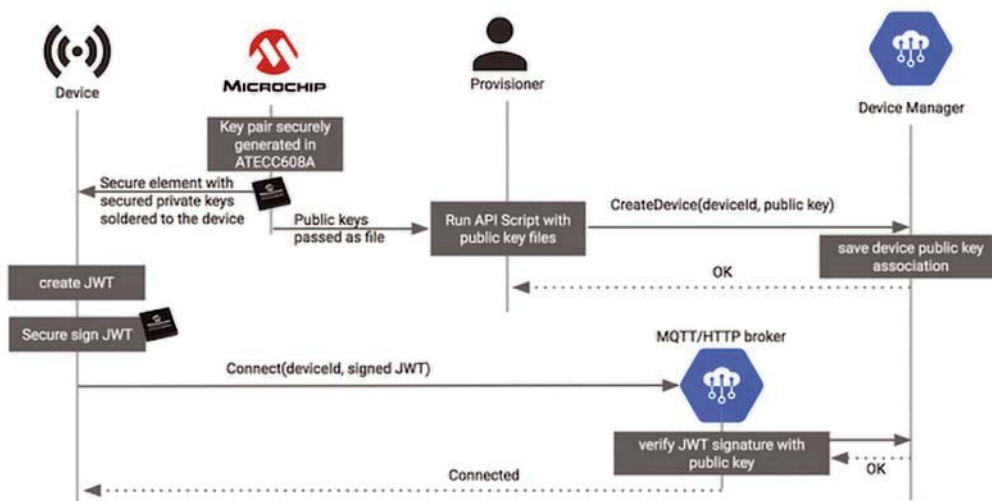


Fig. 7
Processo di sicurezza fra ATECC608 e IoT Device Manager Cloud tramite JWT.

In alternativa è possibile utilizzare uno stack TLS già compatibile con l'elemento sicuro, come WolfSSL, ma tale soluzione comporta un costo di licenza che si aggiunge al costo del dispositivo. Come alternativa a ciò, il servizio Google Cloud IoT permette di utilizzare un JWT, ossia un JSON Web Token, che è una soluzione molto comune per autenticare il dispositivo invece di fare affidamento sull'autenticazione reciproca implementata da uno stack TLS.

Il dispositivo stabilirà una connessione sicura all'endpoint Cloud globale per il Cloud IoT Core (*mqtt.googleapis.com*) utilizzando lo schema TLS, ma invece di attivare l'autenticazione reciproca genererà un JWT molto semplice, lo firmerà con la sua chiave privata e lo passerà al Cloud come password.

Il coprocessore Microchip ATECC608 offre una semplice interfaccia per firmare il dispositivo attraverso il JWT in modo sicuro e senza mai esporre la chiave privata. Infatti il JWT viene ricevuto da Google Cloud IoT, la chiave pubblica del dispositivo viene recuperata e utilizzata per verificare la firma JWT; se valida, l'autenticazione reciproca è effettivamente stabilita (**Fig. 7**).

La convalida JWT può essere impostata dal cliente, ma la sua durata non supera mai le 24 ore, il che la rende molto effimera e difficilmente intercettabile dagli hacker.

L'utilizzo del JSON Web Token porta con sé numerosi vantaggi:

- non esiste alcuna dipendenza dallo stack TLS utilizzato per eseguire l'autenticazione del dispositivo, il che semplifica l'aggiornamento allo stack TLS alla 1.3, quando necessario, senza complicare e rendere onerose le operazioni;
- i dispositivi non hanno bisogno di memorizzare la loro chiave pubblica e il certificato, il che consente di lasciare libera una parte significativa della memoria sul dispositivo;
- il dispositivo non necessita di ospitare uno stack TLS completo, cosa che, anche questa, libera spazio di memoria per l'applicazione;
- i requisiti di memoria sono ben al di sotto dei 50 kB, il che apre la porta all'utilizzo di un MCU (unità microcontrollore) molto più piccolo.

Gli ultimi due punti evidenziano come l'adozione del JSON Web Token rimuova l'intera complessità della gestione dei certificati, consentendo agli sviluppatori di concentrarsi sullo sviluppo del prodotto, sia sotto l'aspetto delle risorse disponibili, sia per quel che riguarda lo sviluppo delle applicazioni IoT.

CONCLUSIONI

Bene, possiamo ritenere di aver terminato; vi abbiamo introdotto il concetto di IoT e quello attualissimo e imperativo della sicurezza dell'Internet of Things, la cui diffusione e la vulnerabilità che ne può derivare ha imposto l'adozione di protocolli e piattaforme di sicurezza in grado di colloquiare con i device periferici.

Abbiamo inoltre spiegato come il concetto di sicurezza e di autenticazione delle connessioni si sia spostato dall'applicazione e dal software, direttamente all'hardware, con la creazione e l'implementazione delle funzioni di sicurezza all'interno di chip specifici, inviolabili e capaci di garantire lo scambio di chiavi di crittografia private in tutta sicurezza, alcuni dei quali progettati secondo le specifiche dei principali provider. ■

STOP AL VIRUS

Protezioni anti-contagio
per la prevenzione e
sanificazione di ambienti.



PREVENZIONE

SEGNALATORE ACUSTICO E OTTICO PER DISTANZA DI SICUREZZA IN KIT

Manteniamo il "distanziamento sociale" con un segnalatore acustico e ottico che ci avverte quando ci stiamo avvicinando troppo al bancone di un negozio, di un chiosco o a uno sportello. Distance Alert è dotato di in un visualizzatore in plexiglass che si illumina di rosso se la distanza è inferiore a un metro e di verde se è quella corretta da mantenere.



cod. DISTALERT
€ 89,00



SATURIMETRO PORTATILE DA DITO

Pulsossimetro a clip leggero con dimensioni contenute. Misura e monitora la saturazione di ossigeno nel sangue (SpO₂) e la frequenza cardiaca (pulsazioni). Dispone di display a colori TFT su cui vengono visualizzati la grafica del battito cardiaco, la saturazione di ossigeno nel sangue e la segnalazione di batteria scarica. Alimentazione tramite due batterie ministilo AAA da 1,5V (acquistabili separatamente); laccetto da polso in dotazione; segnalazione eventuale batteria scarica.

cod. FR804
€ 29,00

TERMOMETRO INFRAROSSI PER TEMPERATURA CORPOREA (DA +32°C A +45°C)

Termometro a infrarossi specifico per la misura a distanza della temperatura corporea. Utilizzo semplice e igienico, consente la misura senza contatto rapida e accurata. Posizionando il termometro tra 5 e 10 cm in meno di un secondo si ottiene, sull'ampio display retroilluminato, la misura della temperatura con l'eventuale lampeggio in caso di superamento della soglia di 37,2°C. Possibilità di visualizzazione in gradi centigradi o in gradi Fahrenheit; spegnimento automatico. Alimentazione tramite due batterie ministilo AAA da 1,5V (incluse).



cod. TP216
€ 59,00

SANIFICAZIONE

SANIFICATORE UVC PER OGGETTI

Sanificatore automatico con LED UVC, particolarmente indicato in tutte le situazioni dove si rende necessario igienizzare, sanificare e comunque disinfettare oggetti potenzialmente depositari di particelle infettive (virus, batteri, ecc.) come mascherine chirurgiche, utensili, occhiali, chiavi, ecc.

È costituito da un contenitore in plexiglass con un coperchio incernierato all'interno del quale si trova una strip a LED UV-C con quattro LED SMD UV-C tipo 3535 da 5 mW ciascuno con lunghezza d'onda della radiazione emessa nella banda dei 265 e i 278 nm, gamma ideale per sterilizzare e sanificare superfici e oggetti. Il set contiene tutte le parti in plexiglass, le parti stampate in 3D, l'elettronica e l'alimentatore.



cod. UVCSAN
€ 80,00

STERILAIR - PURIFICATORE ARIA A RAGGI ULTRAVIOLETTI



cod. FR782
€ 390,00

Purificatore d'aria, dotato di lampada LED (di elevata qualità) a raggi ultravioletti e contenitore in acciaio inossidabile, è in grado di ridurre al minimo la presenza di virus, batteri e muffe presenti nell'aria in ambienti fino a 150 m³. Può essere impiegato anche in presenza di persone e può restare acceso 24 ore al giorno. È adatto per essere utilizzato in abitazioni, ambulatori, camere di hotel, uffici, spogliatoi, negozi, cucine, ecc. Alimentazione 220 VAC; potenza assorbita 13 Watt; lunghezza d'onda lampada LED 254-280 nm; dimensioni 360x115x113 mm; peso circa 4 kg.

SANYO³ - GENERATORE PROFESSIONALE DI OZONO

cod. FR788

€ 680,00

Macchina professionale per l'igienizzazione e la disinfezione di ambienti attraverso l'utilizzo di Ozono. Sanyo³ è progettata e prodotta interamente in Italia, con l'utilizzo di prodotti di elevata qualità e tecnologia innovativa. Offre differenti tipi di programmazione e tramite i timer di accensione automatica consente di igienizzazione del locale ad orari desiderati nell'ambito domestico o di chiusura nell'ambito industriale. Produce 5000 mg/h di ozono che permettono di sanificare in 23 minuti un ambiente di circa 48m³.





Conosciamo la tecnologia di Microsoft per la creazione di applicazioni sicure, basate sull'interazione di hardware protetto e Cloud. Prima puntata: prepariamo la nostra scheda IoT.

CONOSCERE

E USARE

Azure Sphere

di PIER CALDERAN

1

Oggi si parla sempre più spesso di sicurezza informatica o cyber security, ovvero di protezione dei dati dai cosiddetti cyberattack perpetrati da hacker che hanno lo scopo di violare le reti aziendali o istituzionali. Il risultato è spesso uno spaventoso danno economico, sociale o anche peggio. Senza entrare nel dettaglio della questione, diciamo che ormai diventa più che mai necessario pensarci. Non è infrequente sentire notizie di attacchi di spoofing, di denial of service, furto di informazioni e così via. Fortunatamente esistono soluzioni per proteggere dati e applicazioni, soprattutto quelle connesse. Una nota azienda americana, produttrice di distributori automatici di gel sanificante per ospedali, si è attrezzata con un sistema di protezione Azure Sphere per difendersi da possibili attacchi mirati al disordine sanitario e sociale. Anche se è ancora

poco conosciuta, la tecnologia Azure Sphere si sta diffondendo sempre più in tutto il mondo della cyber security. Vediamo di cosa si tratta.

COS'È AZURE SPHERE

Azure Sphere nasce nel 2018 da un'idea di Microsoft sulla sicurezza informatica, ma la tecnologia e i chip certificati che la supportano sono stati ufficializzati nel febbraio 2019. Azure Sphere è una "piattaforma applicativa protetta di alto livello" con funzionalità di comunicazione e di protezione integrate per i dispositivi connessi a Internet. Riportiamo lo slogan di Microsoft dalla sua home page dedicata: *"Ogni anno vengono connessi miliardi di nuovi dispositivi, quindi è più importante che mai proteggere i tuoi. Contribuisci alla protezione di dati, privacy, sicurezza fisica e infrastruttura con Azure Sphere. È il risultato di decenni di esperienza"*

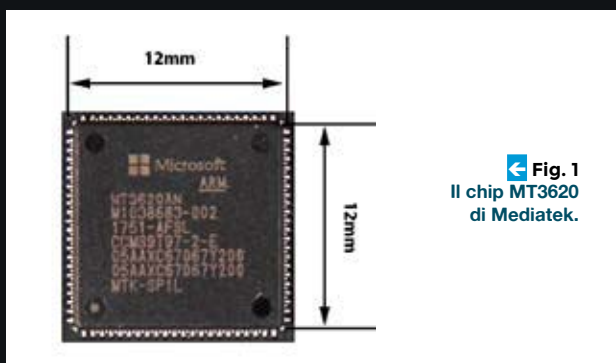


Fig. 1
Il chip MT3620
di Mediatek.

Microsoft in termini di hardware, software e cloud per fornire una soluzione di sicurezza chiavi in mano per i dispositivi IoT". Stiamo parlando di un sistema basato su un servizio di sicurezza che stabilisce una connessione sicura tra un dispositivo fisico (MCU) e il servizio cloud Azure di Microsoft. In pratica è l'hardware che garantisce il cosiddetto avvio sicuro delle applicazioni e delle connessioni con il cloud. Si tratta di un secure boot, ovvero di un avvio sicuro dal dispositivo hardware in poi. In questo modo l'attacco malevolo ai dati diventa pressoché impossibile. La connessione tra un dispositivo Azure Sphere (un chip protetto) e il servizio di sicurezza del cloud Azure si basa su un sistema di autenticazione dell'identità del dispositivo, garantendo così l'integrità e l'attendibilità del software di sistema. Il servizio fornisce anche il canale protetto utilizzato da Microsoft per scaricare e installare automaticamente gli aggiornamenti del sistema operativo Azure Sphere e gli aggiornamenti delle applicazioni dell'utente sui dispositivi distribuiti.

L'HARDWARE AZURE SPHERE

Il cuore della tecnologia risiede in un'unità microcontroller (MCU) protetta e connessa, in cui risiedono un sistema

operativo personalizzato basato su Linux e un servizio di sicurezza basato sul cloud Azure.

Il primo MCU certificato Azure Sphere (e per il momento l'unico) è prodotto da Mediatek (<https://www.mediatek.com>). Include il chip MediaTek 3620, siglato come MT3620.

Caratteristiche principali:

- 1 processore ARM Cortex-A7 (500 MHz);
- 2 sottosistemi I/O ARM Cortex-M4F (200 MHz);
- 5 interfacce UART;
- 1 interfaccia I²C;
- 1 interfaccia SPI;
- 2 interfacce I²S;
- 8 ingressi ADC;
- 12 contatori PWM;
- 72 pin GPIO;
- connettività WiFi.

Il chip MT3620 contiene anche il sottosistema di sicurezza Microsoft Pluton con un core Arm Cortex-M4F dedicato, che gestisce l'avvio e il funzionamento sicuro del sistema. Mediante il servizio di sicurezza Azure Sphere, gli MCU si connettono in modo sicuro al cloud e al Web. Il servizio assicura che il dispositivo venga avviato solo con una versione autorizzata di applicazioni approvate. Inoltre, fornisce un canale protetto tramite cui Microsoft può scaricare e installare automaticamente gli aggiornamenti del sistema operativo per i dispositivi distribuiti sul campo, al fine di limitare i problemi di sicurezza.

Il chip Mediatek MT3620 è incapsulato in un package QFN DR delle dimensioni di appena 12x12 mm (Fig. 1).

I DISPOSITIVI AZURE SPHERE

Attualmente esistono tre schede Azure Sphere che montano il chip MT3620:

- Seed MT3620 Development Kit;



Fig. 2
La scheda Seed
MT3620.

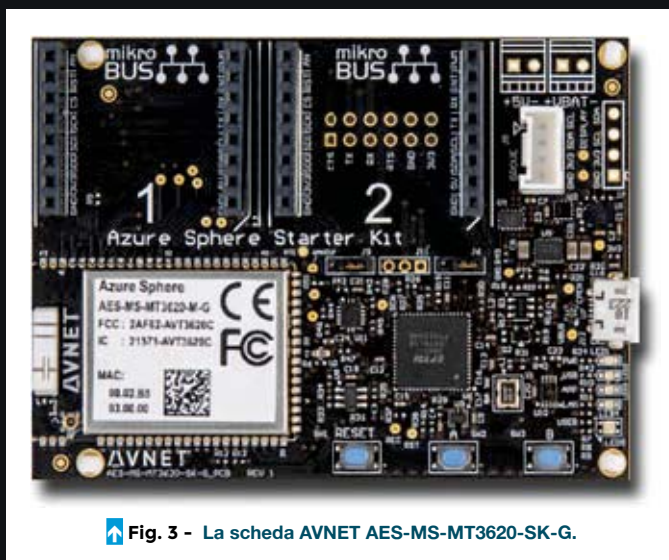


Fig. 3 - La scheda AVNET AES-MS-MT3620-SK-G.

- Avnet Azure Sphere MT3620 Starter Kit;
- Futura Elettronica Azure Sphere MT3620 Demo Board.

La prima scheda Azure Sphere ad apparire sul mercato nel gennaio 2019 è stata la scheda Seeed MT3620 (Fig. 2). A seguire, la AVNET AES-MS-MT3620-SK-G (Fig. 3), a novembre 2019. Ultima in ordine di apparizione, ma non meno importante, la scheda di Futura Elettronica Azure Sphere MT3620 Demo Board (Fig. 4) disponibile nello shop online della Futura Elettronica (www.futurashop.it) con il codice FEAZURESPH.

In questo corso faremo riferimento solo alla scheda Azure Sphere MT3620 Demo Board di Futura Elettronica. Il motivo è che nasce dall'esigenza di fornire un valido supporto di sviluppo integrato con Azure Sphere su un hardware facile da usare e aperto a qualsiasi implementazione esterna. Le limitazioni hardware e la scarsa dotazione delle schede Seeed e Avnet, pur essendo utili per il loro scopo, ci hanno spinto a produrre questa Demo Board volgendo l'occhio sia ai maker evoluti che ai progettisti seri, senza costringere la scelta obbligate dell'hardware da collegare.

La scheda è stata pensata per usufruire fin da subito di periferiche on-board, in modo da avvicinarsi in maniera semplice allo sviluppo di applicazioni Azure Sphere, ma offre numerosi possibilità di collegamenti con hardware esterno.

FUTURA ELETTRONICA AZURE SPHERE DEMOBOARD

La scheda Azure Sphere MT3620 Demo Board (da ora in poi solo Futura MT3620) nasce dalla collaborazione dell'azienda Futura Group e chi scrive. La scheda monta un modulo breakout Avnet Azure Sphere con chip MT3620, e dispone di tutto quel che serve per la sperimentazione e la messa in opera sul campo. La scheda Futura MT3620 può ospitare i seguenti moduli e sensori:



Fig. 4 - La scheda Futura Azure Sphere MT3620 Demo Board.

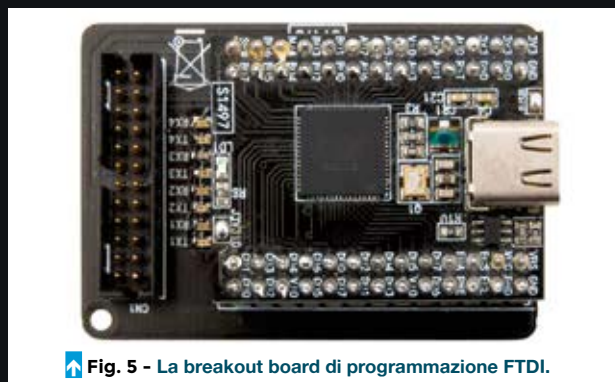


Fig. 5 - La breakout board di programmazione FTDI.

- 1x accelerometro/giroscopio MPU6050 (I²C);
 - 1x modulo RFID MFRC522 (SPI);
 - 1x modulo qualità dell'aria CCS811 (I²C);
 - 1x modulo RTC DS3231RTC (I²C);
 - 1x sensore temperatura/umidità DHT22 (single-wire).
- Inoltre prevede:
- 2x header utente;
 - 3x pulsanti utente;
 - 3x LED utente;
 - 1x alimentazione esterna USB-C e terminale 5VDC;
 - 1x breakout board Avnet Azure Sphere MT3620;
 - 1x slot di programmazione.

La scheda Futura MT3620 è stata progettata per avere a bordo tutto il necessario per lo sviluppo immediato di applicazioni hardware/software; per contenere le dimensioni è stato deciso di non mettere a bordo la parte di connessione USB per la programmazione.

Per questo motivo, la scheda è dotata di un connettore per la programmazione, a cui, nelle fasi appunto di programmazione da computer, va connesso un apposito modulo composto da un PCB adattatore su cui si applica la breakout board FTDI.

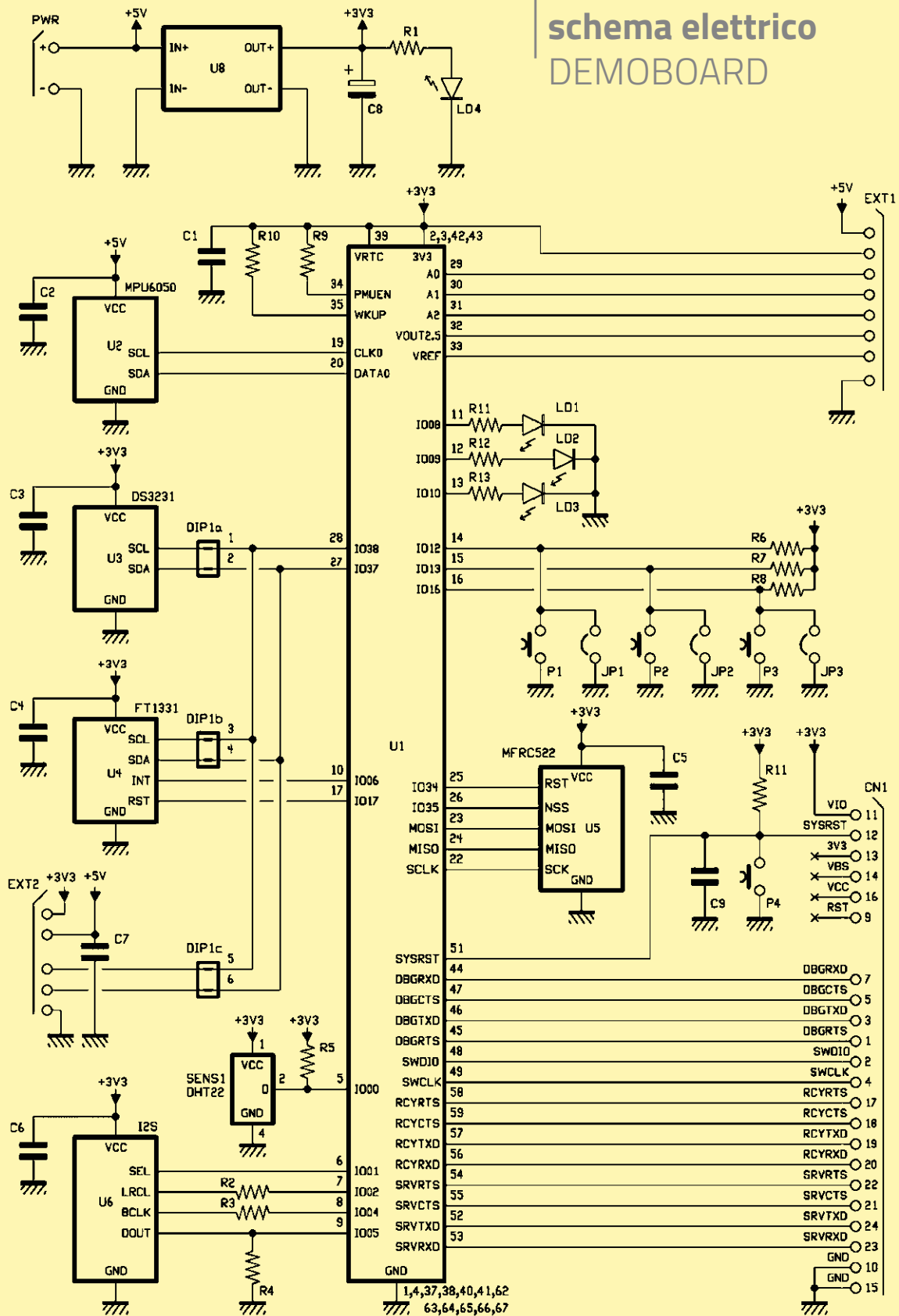
Quindi per programmare la demoboard bisogna collegare con un flat cable il connettore CN1 (vedere schema elettrico) della demoboard con il corrispondente della scheda adattatrice di programmazione sulla quale è stato inserito il modulo basato su FT4232 (Fig. 5); quest'ultimo andrà connesso alla porta USB del computer utilizzato per la programmazione, utilizzando un cavetto USB-C/USB-A.

SCHEMA ELETTRICO DEMOBOARD

Per darvi un'idea di come è composta la scheda di sviluppo, ne abbiamo riportato lo schema elettrico in queste pagine; in esso vedete il componente chiave che è il modulo siglato U1 (si tratta dell'AES-MS-MT3620-M-G) di cui sono indicati sia i pin fisici che i pin GPIO corrispondenti.

I DIP switch possono indirizzare le porte GPIO per diversi utilizzi, rendendo più versatile l'uso della scheda, come spiegheremo più avanti. Per quanto riguarda i componenti a bordo della scheda, abbiamo SENS1 che è il

schema elettrico DEMOBOARD



senore di temperatura e umidità DHT22. U2 è il modulo accelerometro + giroscopio 3 assi, U3 è l'RTC basato sul chip Maxim/Dallas DS3231, mentre U4 è il modulo rilevatore della qualità dell'aria con CCS811. C'è poi il lettore NFC/RFID (U5), il modulo con microfono GY-SPH0645 (U6) con connessione I²S (attualmente non ancora gestita dal modulo MT3620) e il DC/DC converter 3V3/3A (U7) che è un integrato step-down MP1584 il cui compito è ricavare la tensione di alimentazione per i componenti della board, partendo dalla tensione applicata alla morsettiera PWR. Nello schema troviamo anche una serie di pulsanti e LED. Il connettore CN1 è dedicato alla programmazione esterna tramite la scheda di interfaccia USB di seguito descritta.

INTERFACCIA DI PROGRAMMAZIONE

Per collegare la nostra demoboard al computer in modo da programmarla è stata prevista una scheda d'interfaccia da connettere alla demoboard mediante il connettore flat a passo 2x2,54 mm siglato CN1, presente su entrambe le schede. L'interfaccia è basata sugli integrati U1 (74LCX126MTC) e U2 (74HC595PW) e ospita la breakout board basata sul chip FT4232 della FTDI (cod. FT1497K): questo si monta sugli strip siglati CN2 e CN3 nello schema elettrico; la breakout funge quindi da elemento convertitore dei segnali TTL della demoboard in USB e viceversa, quindi contiene il connettore USB-C con il quale si va a interfacciare il sistema al Personal Computer.

Il compito dell'interfacciamento tra seriale TTL (TXD, RXD) e USB è svolto di fatto dalla breakout board; i due integrati di bordo sono riservati a futuri sviluppi. In particolare il 74LCX126MTC è un quadruplo buffer funzionante con livelli logici di 3-3,3V ma caratterizzato da ingressi e uscite 5V-tolerant, mentre il 74HC595 è un 8-bit serial-in/serial-out parallel-out shift register con uno storage register e uscite 3-state (1, 0, open).

Il chip FTDI 4232 permette, una volta caricati i driver, di installare quattro porte COM, ma nel nostro caso è stato programmato (ricordiamo che può essere impostato da software secondo la configurazione desiderata e utilizzando specifici driver) per Azure Sphere in modo da comunicare con tre porte COM.

La breakout board serve anche ad alimentare la scheda durante la fase di programmazione. Al termine della programmazione e del debug, si può tranquillamente sconnettere l'interfaccia con la breakout board e alimentare la scheda direttamente con la presa USB incorporata.

OPERAZIONI PRELIMINARI SULLA SCHEDA

Come tutte le schede Azure Sphere, anche la nostra va preparata seguendo le procedure che spiegheremo dettagliatamente fra poco.

I passi da seguire per preparare la scheda all'uso con il sistema operativo Azure Sphere sono i seguenti:

- claim del dispositivo;
- connessione WiFi;
- creazione di un tenant;
- preparazione del dispositivo allo sviluppo;
- verifica dei certificati nel gruppo di registrazioni.

Configurazione del dispositivo Azure Sphere

Per lo sviluppo di applicazioni per Azure Sphere sono necessari i requisiti seguenti:

- un dispositivo Azure Sphere (nel nostro caso la scheda Futura MT3620);
- un PC che esegue Windows 10 oppure un computer Linux che esegue Ubuntu 20.04 LTS o 18.04 LTS;
- una porta USB libera e, se si sta eseguendo una macchina virtuale, il supporto per il pass-through USB.

A seconda della piattaforma del sistema operativo, Azure Sphere supporta i seguenti ambienti di sviluppo.

- Visual Studio per Windows. Dopo l'installazione di Azure Sphere SDK, è possibile installare l'estensione Azure Sphere per Visual Studio. L'estensione consente di sviluppare ed eseguire il debug di applicazioni Azure Sphere usando l'IDE di Visual Studio.
- Visual Studio Code su Windows o Linux. Dopo l'installazione di Azure Sphere SDK, è possibile installare l'estensione Azure Sphere per Visual Studio Code. L'estensione consente di sviluppare ed eseguire il debug di applicazioni Azure Sphere in Windows o Linux usando l'IDE di Visual Studio Code.
- Interfaccia dalla riga di comando (CLI) su Windows o Linux. Il pacchetto Azure Sphere SDK supporta autonomamente lo sviluppo e il debug di applicazioni Azure Sphere usando l'interfaccia della riga di comando dal command prompt o dal terminale.

Quando la scheda Futura MT3620 viene collegata al Personal Computer, il dispositivo installa tre porte COM.

La prima volta che il dispositivo viene collegato, i driver dovrebbero venire scaricati dal sito della FTDI e installati automaticamente; se i driver non vengono installati automaticamente bisogna fare clic con il pulsante destro del mouse sul nome del dispositivo in Gestione dispositivi e selezionare **Aggiorna driver**. In alternativa, è possibile scaricare il driver da Future Technology Devices International FTDI:

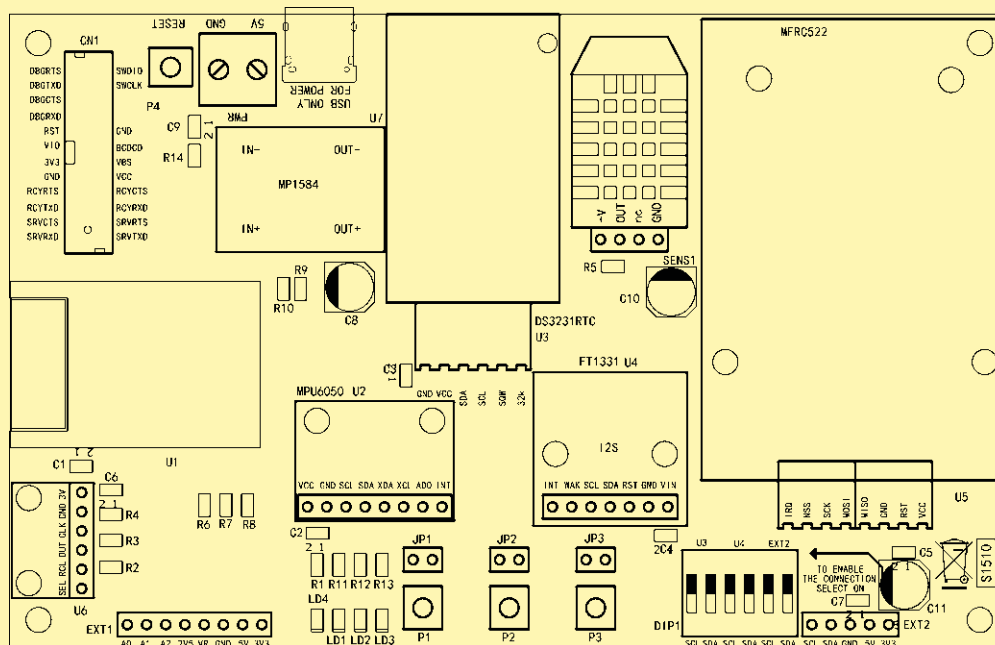
<https://www.ftdichip.com/Drivers/VCP.htm>.

Scegliere il driver adatto alla versione di Windows (32 o 64 bit). Per verificare la corretta installazione, aprire *"Gestione dispositivi"* e cercare tre porte COM. I numeri delle porte COM potrebbero essere diversi da quelli presenti nella **Fig. 6**.

INSTALLARE L'SDK DI AZURE SPHERE SU WINDOWS

Per motivi pratici, in questo corso faremo riferimento solo al sistema operativo Windows 10, anche se è possibile ese-

piano di montaggio DEMOBOARD



Elenco Componenti:

R5, R6, R7, R8, R9, R10, R14 10 kohm (0805)
 R2, R3: 47 ohm (0805)
 R4: 100 kohm (0805)
 R1, R11, R12, R13: 330 ohm (0805)
 C1, C2, C3, C4, C5, C6, C7, C9: 100 nF ceramico (0805)
 C8: 100 µF 16 VL elettrolitico (ø 6mm)
 LD1: LED rosso
 LD2: LED giallo
 LD3: LED verde
 LD4: LED verde
 P1, P2, P3, P4: Microswitch
 SENS1: DHT22
 U1: AES-MS-MT3620-M-G
 U2: Modulo accelerometro giroscopio 3 assi
 U3: Modulo RTC DS3231
 U4: Modulo qualità aria CCS811
 U5: Modulo lettore NFC/RFID
 U6: Modulo microfono GY-SPH0645 I2S

U7: Modulo Step-Down MP1584 3V3/3A
 CN1: Connettore IDC 2x12 vie passo 2mm
 PWR: Morsetto 2 vie passo 5.08 mm
 USB1: Connettore USB TypeC da CS
 DIP1: Dip switch 6 vie

Varie

- Strip femmina 2 vie
- Strip femmina 6 vie
- Strip femmina 7 vie
- Strip femmina 8 vie
- Strip femmina 6 vie 90°
- Strip femmina 8 vie 90°
- Strip maschio 8 vie
- Strip maschio 5 vie
- Strip maschio 2 vie
- Circuito stampato S1510 (133x84 mm)

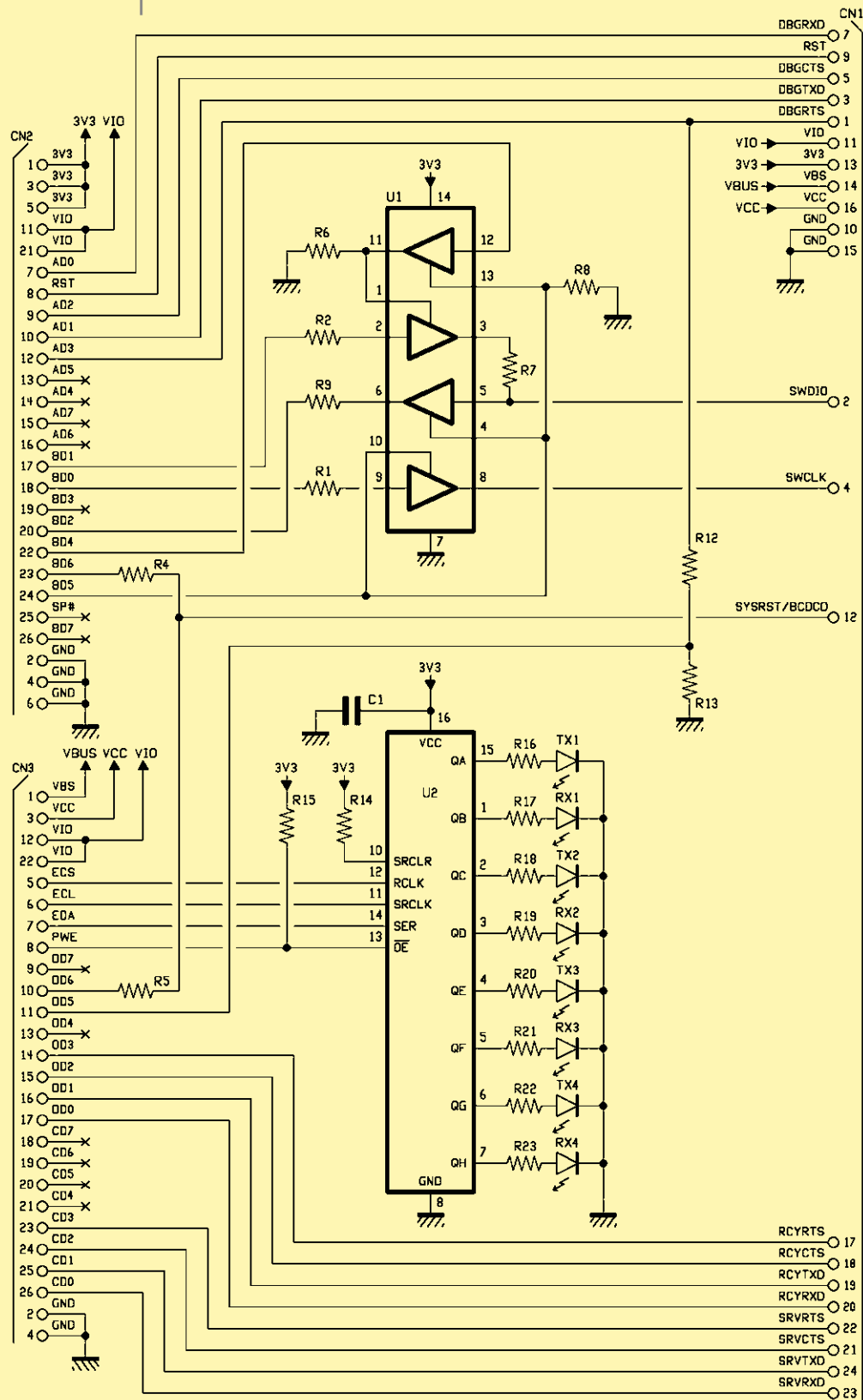
guire tutto su Linux. Scaricare l'SDK dal link seguente:

<https://aka.ms/AzureSphereSDKDownload/Windows>

Eseguire come amministratore il file EXE scaricato per

installare l'SDK. Accettare le condizioni di licenza e quindi fare clic su Avanti. A questo punto si deve fare clic sul pulsante Installa per avviare l'installazione dell'SDK (Fig. 8). Al termine dell'installazione, se richiesto dal sistema operativo bisogna riavviare il PC.

schema elettrico ADATTATORE USB



piano di montaggio ADATTATORE USB

Elenco Componenti:

R1, R2, R3, R7, R9, R10, R11: 33 ohm (0603)
R4, R5, R6, R8: 4,7 kohm (0603)
R12, R13: 2,2 kohm (0603)
R14, R15: 10 kohm (0603)
R16, R17, R18, R19, R20, R21, R22, R23: 330 ohm (0603)
C1: 100 nF ceramico (0603)
RX1, RX2, RX3, RX4: LED verde (0603)
TX1, TX2, TX3, TX4: LED rosso (0603)
U1: 74LCX126MTC
U2: 74HC595PW,118

Varie

- Connettore IDC 2x12 vie passo 2mm

- Strip femmina 2x13 vie passo 2.54mm (2 pz.)
- Circuito stampato S1498 (52x36 mm)

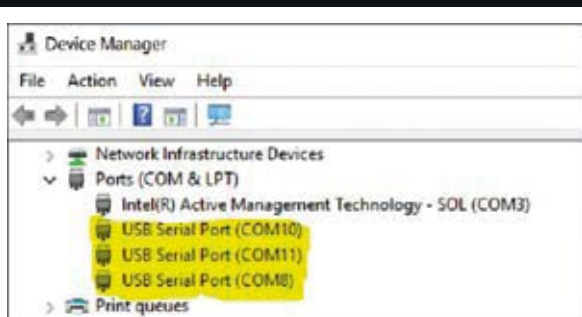
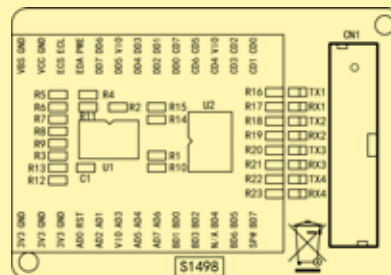


Fig. 6 - Le tre porte COM installate dal driver FTDI.

Installare Visual Studio e l'estensione Azure Sphere

Per lo sviluppo di applicazioni Azure Sphere, è possibile usare Visual Studio Enterprise, Professional o Community 2019, versione 16.4 o successiva. In questo corso si farà riferimento solo a Visual Studio Community 2019. Se non è presente nel vostro PC, installare Visual Studio da questo link:

<https://docs.microsoft.com/visualstudio/get-started/visual-studio-ide?view=vs-2019>

Per installare l'estensione di Visual Studio per Azure Sphere bisogna scaricarla dal sito Visual Studio Marketplace a questo link (Fig. 7):

<https://marketplace.visualstudio.com/items?itemName=AzureSphereTeam.AzureSphereSDKforVisualStudio2019>

Come vedremo più avanti in questo corso, per l'uso con Visual Studio Code o l'interfaccia della riga di comando di Windows (CLI) sarà necessario installare manualmente CMake e Ninja.

TENANT E CLAIM DEL DISPOSITIVO

Su ogni dispositivo Azure Sphere deve essere effettuato il cosiddetto "claim", termine che potrebbe essere tradotto come richiesta, rivendicazione, ma che continueremo a chiamare claim. Prima del claim, bisogna creare un "tenant" (che vorrebbe dire "affittuario"). Richiedere un dispositivo con il claim significa associare il suo ID univoco e non modificabile, al proprio tenant Azure Sphere. Azure Sphere Security Ser-

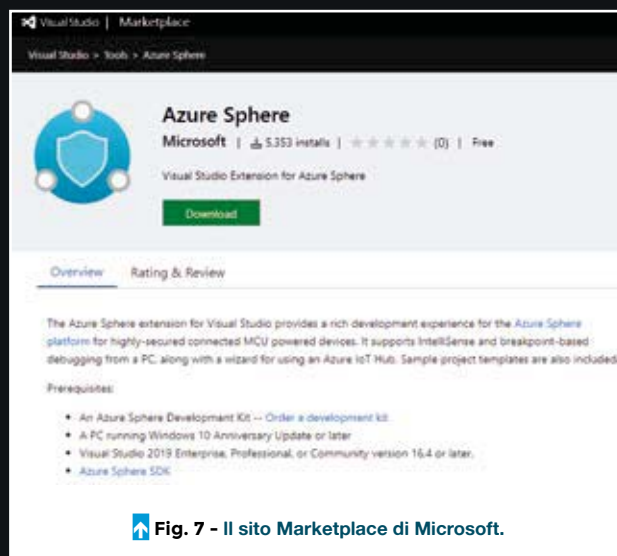


Fig. 7 - Il sito Marketplace di Microsoft.

vice utilizza l'ID dispositivo per identificare e autenticare il dispositivo. Quando si richiede un dispositivo e lo si associa a un tenant NON sarà più possibile modificare l'associazione con quel dispositivo. Si potranno comunque aggiungere utenti e ruoli con autorizzazioni di diverso tipo.

ACCEDERE AL DISPOSITIVO AZURE SPHERE

Dopo aver connesso il dispositivo tramite USB e verificato che siano state create le tre porte COM per la programmazione e il debug, bisogna aprire come amministratore il prompt dei comandi per gli sviluppatori Azure Sphere, che si trova nel menu Start sotto "Azure Sphere Developer Command Prompt". Una volta aperto il prompt dei comandi, bisogna accedere al dispositivo Azure Sphere usando un account Microsoft Azure con il comando seguente (Fig. 8):

```
azsphere login
```

Si fa notare che se non è mai stato effettuato l'accesso a Azure Sphere prima, si può aggiungere il parametro `--newuser` insieme all'indirizzo di posta elettronica dell'account Microsoft:

```
azsphere login --newuser <email-address>
```

Dopo aver scelto l'account ed effettuato il login (Fig. 9) si potrà vedere l'account registrato e l'indirizzo email. Si vedrà anche il tenant selezionato con il relativo ID creato per quel tenant.

SCEGLIERE O CREARE UN TENANT

Dopo aver eseguito il login, è necessario scegliere un tenant Azure Sphere. Se si ha accesso a un solo tenant, questo viene selezionato per impostazione predefinita ed è possibile proseguire con il passaggio successivo. Se si ha accesso a più tenant, è necessario selezionarne uno.

Un tenant di Azure Sphere "isola" i dispositivi di Azure Sphere e fornisce un modo per gestirli.

Se il proprietario del dispositivo ha già un tenant di Azure Sphere, si consiglia di valutare se richiedere l'accesso al tenant invece di crearne uno aggiuntivo.

Dopo aver creato un tenant, non è possibile rinominarlo, eliminarlo o spostarlo in un'altra sottoscrizione di Azure. Ogni ID del dispositivo può essere usato per creare un solo tenant Azure Sphere. All'account che crea il tenant viene assegnato automaticamente il ruolo di amministratore per il tenant. Se la schermata di accesso ad Azure Sphere che appare dopo aver impartito il comando `azsphere login`, indica che non si ha accesso a un tenant di Azure Sphere, è possibile crearne uno in questo modo:

- collegare un dispositivo Azure Sphere al computer tramite USB;
- immettere il comando `azsphere tenant create --name <my-tenant>`;

- sostituire `<my-tenant>` con un nome riconoscibile tipo `ilmiotenant`. Se il nome include degli spazi, bisogna racchiuderlo tra virgolette "il mio tenant";
- i nomi devono essere costituiti solo da caratteri alfanumerici senza simboli.

Dopo la creazione del tenant, verrà richiesto di effettuare nuovamente l'accesso. Accedere con l'account che si utilizzerà per gestire i dispositivi di Azure Sphere.

Il comando `azsphere tenant create` crea un tenant, ma non esegue il claim del dispositivo. È possibile richiedere (claim) il dispositivo dopo aver creato il tenant. Quando si crea un tenant, il servizio di sicurezza di Azure Sphere registra l'ID del dispositivo collegato. Ogni ID dispositivo può essere usato per creare un solo tenant Azure Sphere.

Se si ha già accesso a un tenant e si è assolutamente certi di volerne creare un altro, usare il comando seguente:

```
azsphere tenant create --force --name <my-tenant>
```

EFFETTUARE IL CLAIM DEL DISPOSITIVO

Come già detto, la richiesta del dispositivo si chiama claim e il comando è il seguente (Fig. 10):

```
azsphere device claim
```

Quando il dispositivo viene richiesto all'interno di un tenant, non è più possibile spostarlo in un tenant diverso.

Alla fine del processo, verrà visualizzato un output simile al seguente:

Claiming device.

Successfully claimed device ID '<device ID>' into tenant '<name>' with ID 'd343c263-4aa3-4558-adbb-d3fc34631800'.

Command completed successfully in 00:00:05.5459143.

Attenzione: prima del claim apparirà un avvertimento che tradotto suona così: "Siete sicuri di volere richiedere il

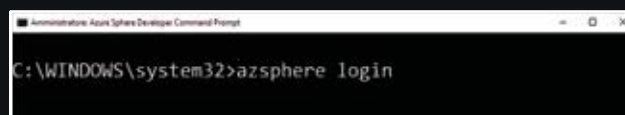


Fig. 8 - Azure Sphere Developer Command Prompt con il comando `azsphere login`.

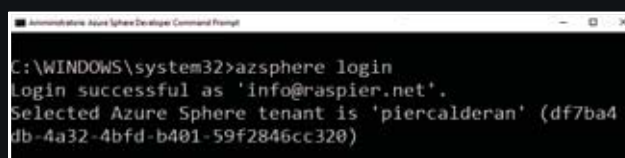


Fig. 9 - Accesso all'account Azure.


```
Administrator: Azure Sphere Developer Command Prompt
C:\WINDOWS\system32>azsphere device claim
warn: You are claiming a device to the following tenant
ID:
warn: df7ba4db-4a32-4bfd-b401-59f2846cc320 ()
warn: Do you want to claim the device ID E59B4F23EC0CF956ED312946E45E1B8741EEE384AF5E22838BF49FACF3C5AB94F4F88064435A897F42A0AAF455D533C1ADCD5BEC6E521C9C8F02DCCD365D1EBD to this tenant?
warn: You cannot change the Azure Sphere tenant this device is claimed to once this action has completed.
Enter 'yes' to continue. Enter anything else to exit.
> yes
Claiming device.
Successfully claimed device ID 'E59B4F23EC0CF956ED312946E45E1B8741EEE384AF5E22838BF49FACF3C5AB94F4F88064435A897F42A0AAF455D533C1ADCD5BEC6E521C9C8F02DCCD365D1EBD' into tenant 'piercalderan' with ID 'df7ba4db-4a32-4bfd-b401-59f2846cc320'.
```

Fig. 10 - Il prompt con il comando `azsphere device claim`.

dispositivo all'interno di uno specifico tenant? Una volta completata questa azione non è possibile modificare il tenant di Azure Sphere richiesto dal dispositivo. Digitare "yes" per continuare o qualsiasi altra cosa per uscire".

CONFIGURARE IL WIFI SUL DISPOSITIVO AZURE SPHERE

Il dispositivo Azure Sphere deve essere connesso a Internet, sia per le operazioni di accesso ai servizi sia per ricevere gli aggiornamenti; vediamo quindi come fare, partendo dal primo step che consiste nel verificare lo stato delle connessioni con il comando seguente:

`azsphere device wifi show-status`

Apparirà lo stato della connessione che indicherà lo stato di nessuna connessione rilevata. Se si conosce il nome della rete (SSID) e la password, basta aggiungere la rete WiFi nel dispositivo usando il comando seguente:

`azsphere device wifi add --ssid <yourSSID> --psk <yourNetworkKey>`

Sostituire <yourSSID> con il nome della rete e <yourNetworkKey> con la password di rete. Si fa notare che i nomi SSID distinguono tra maiuscole e minuscole.

È possibile usare il comando `azsphere device wifi scan` per ottenere l'elenco delle reti WiFi disponibili. Se l'SSID o la chiave di rete includono spazi, racchiudere l'SSID o la chiave tra virgolette.

Dopo aver aggiunto la rete, usare ancora il comando `azsphere device wifi show-status` per controllare lo stato della connessione (Fig. 11). Quando è connesso, il dispositivo Azure Sphere verifica a ogni avvio la presenza di aggiornamenti del sistema operativo Azure Sphere e dell'applicazione. Questo si verifica durante la connessione iniziale a Internet e, successivamente,

ogni 24 ore. Se sono disponibili aggiornamenti, il download e l'installazione possono richiedere fino a 15-20 minuti e potrebbero causare il riavvio del dispositivo.

PREPARARE IL DISPOSITIVO ALLO SVILUPPO E AL DEBUG

Prima di poter compilare e distribuire un'applicazione sul dispositivo Azure Sphere, ovvero nella nostra scheda Futura MT3620, è necessario abilitare lo sviluppo e il debug.

Per impostazione predefinita, i dispositivi Azure Sphere sono "bloccati", ovvero non consentono il caricamento da un computer di applicazioni in fase di sviluppo, oltre a non permettere il debug delle applicazioni. Preparando il dispositivo per il sideload, questa restrizione viene rimossa. Il comando per abilitare allo sviluppo del dispositivo in modo che accetti le applicazioni per il debug, ovvero possa caricare il server di debug nel dispositivo è il seguente:

`azsphere device enable-development`. L'output dovrebbe essere simile alla Fig. 12. Come si può vedere, il comando effettua il download della configurazione e abilita il dispositivo allo sviluppo, quindi distribuisce l'immagine `gdbserver`, `imagepackage` nel dispositivo. A questo punto, il dispositivo è pronto ed è possibile compilare e distribuire un'applicazione nella scheda Azure Sphere.

ESEMPI DELLA SCHEDA FUTURA MT3620

Per prima cosa bisogna clonare il pacchetto di esempi dal GitHub seguente (installare Git, se non è presente nel sistema):

`git clone https://github.com/piercalderan/futura_MT3620.git`

Dopo aver clonato il repository, verrà creata una cartella "futura_MT3620" nella directory di destinazione (Fig. 13). In questo corso verranno spiegati gli esempi contenuti nella cartella "Futura_samples":

- Futura_MT3620_HelloWorld (lampeggio di tre LED);
- Futura_MT3620_ADC_IoT_Central (potenziometro analogico);
- Futura_MT3620_DHT22_IoT_Central (sensore temperatura/umidità);
- Futura_MT3620_GPIO_IoT_Central (LED e pulsanti);
- Futura_MT3620_MPU6050_IoT_Central (accelerometro/giroscopio);

```
Administrator: Azure Sphere Developer Command Prompt
C:\WINDOWS\system32>azsphere device wifi show-status
SSID           : TP-Link_Pier
Configuration   : enabled
Connection state : connected
Security state  : psk
Frequency       : 2427
Mode            : station
Key management  : WPA2-PSK
WPA State       : COMPLETED
IP Address      : 192.168.1.100
MAC Address     : 00:02:b5:03:56:1e
```

Fig. 11 - Lo stato della connessione Wi-Fi.



- Futura_MT3620_RFID_IoT_Central (lettore card RFID);
- Futura_MT3620_RX_UART_IoT_Central (porta seriale);
- Futura_MT3620_inter-core_IoT_Central (applicazione HApp + RTApp).

COMPILARE ED ESEGUIRE L'APPLICAZIONE HELLOWORLD

A questo punto, se tutto è installato e configurato correttamente, si potrà compilare ed eseguire nella scheda il più semplice degli esempi: HelloWorld.

Questa semplice applicazione farà lampeggiare i tre LED incorporati nella scheda a intervalli di un secondo.

Ecco la procedura da seguire passo passo.

1. Avviare Visual Studio senza codice.
2. Dal menu File impartire il comando Apri> CMake... (Fig. 14).
3. Aprire la cartella Futura_samples.
4. Aprire la cartella Futura_MT3620_HelloWorld.
5. Aprire il file CMakeLists.txt che si trova al suo interno.
6. Al termine del caricamento in Visual Studio si vedrà qualcosa di simile alla Fig. 15.
7. Dalla barra del menu, selezionare la voce GDB Debugger (HL Core) per impostare correttamente l'elemento di avvio (Fig. 16).
8. Premere F5 o fare clic su GDB Debugger (HL Core) per eseguire il debug del progetto: Visual Studio compilerà il progetto prima dell'avvio.

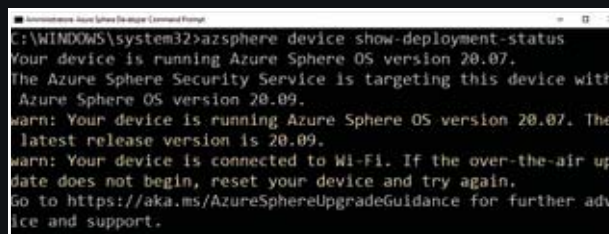


Fig. 12 - Il prompt con il comando `azsphere device enable-development`.



Fig. 13 - La cartella degli esempi Futura MT3620.

9. Attendere qualche secondo mentre Visual Studio compila l'applicazione: verrà creato un pacchetto immagine e inizierà la distribuzione nel dispositivo.

Apparirà un messaggio "Copying application to the device". Quando l'applicazione verrà eseguita nel dispositivo si potranno

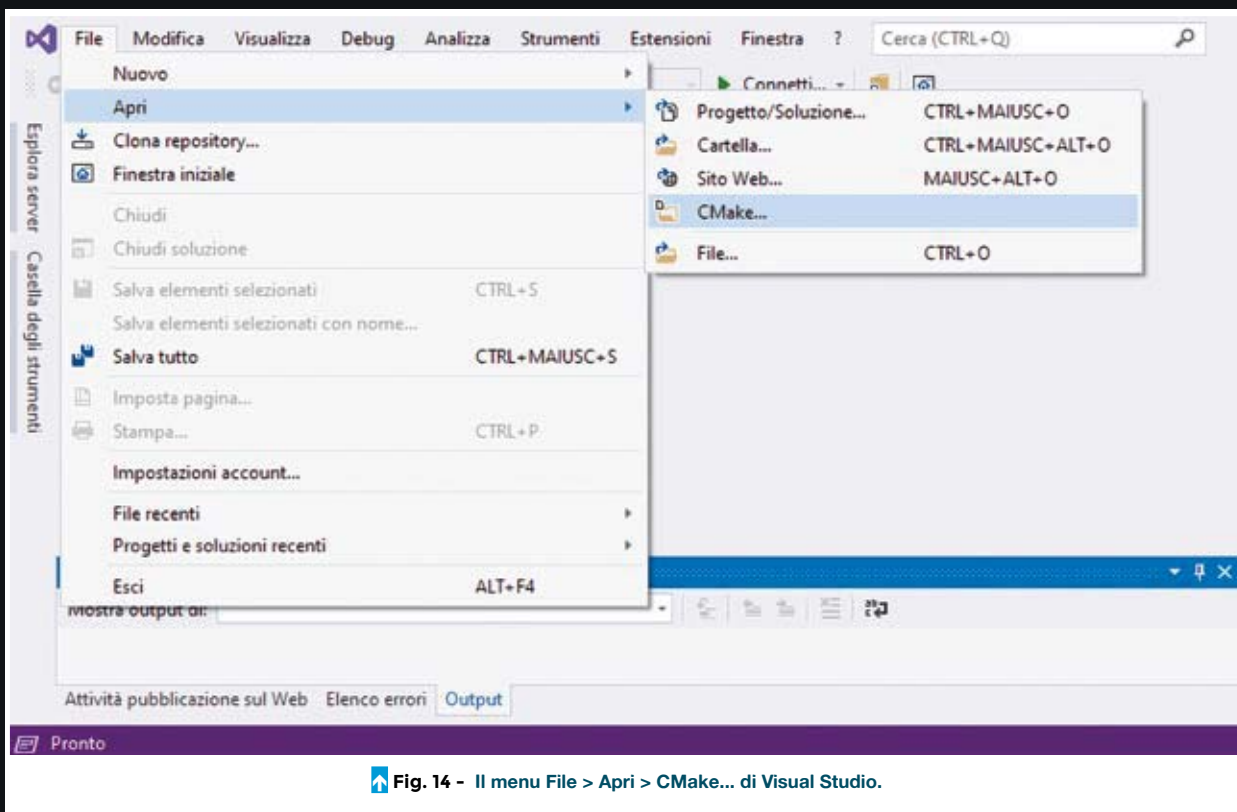


Fig. 14 - Il menu File > Apri > CMake... di Visual Studio.

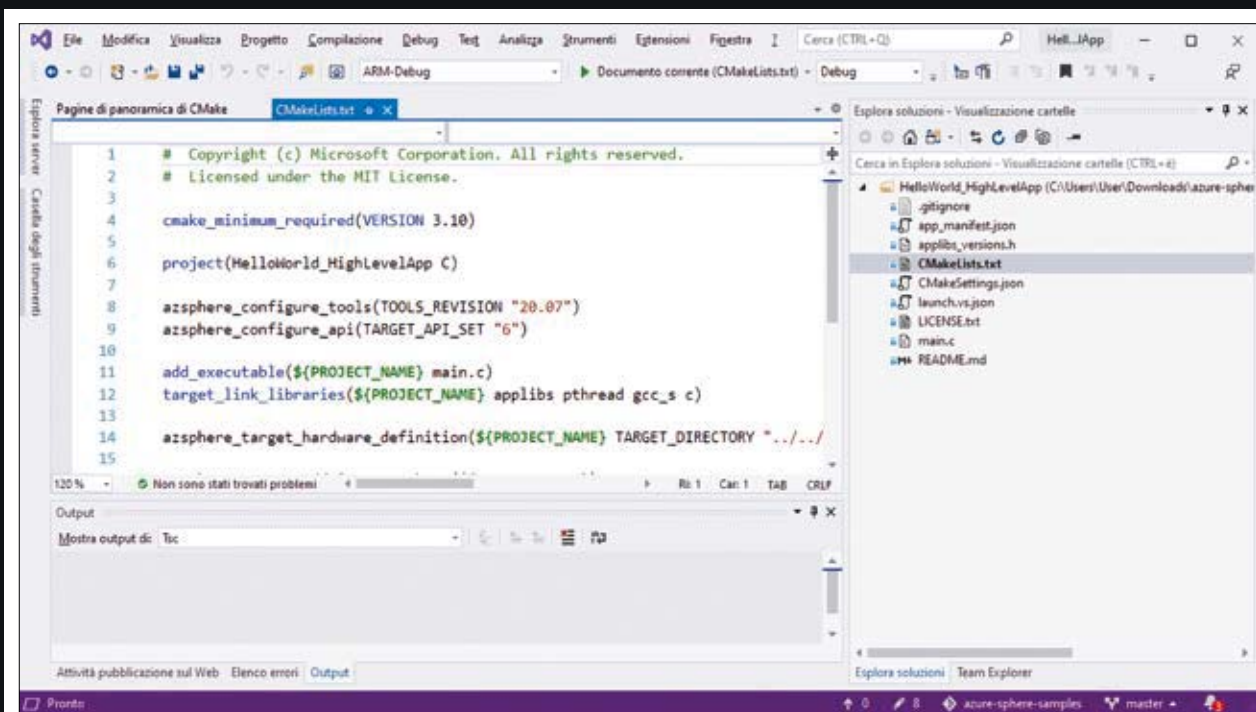


Fig. 15 - L'esempio HelloWorld caricato.



Fig. 16 - L'opzione GDB Debugger (HL Core) selezionata.

no vedere le informazioni di debug nella finestra di "Debug" oppure nella finestra di Output selezionando "Device Output". A questo punto, i tre LED della scheda lampeggeranno. Nella finestra di Output si possono vedere le informazioni del server di debug GDB, con l'indirizzo IP e la porta assegnati:

*Remote debugging from host 192.168.35.1, port 61056
Starting CMake Hello World application...*

CONCLUSIONI

Bene, con questo concludiamo questo primo appuntamento con la tecnologia Azure Sphere per l'IoT. Con la speranza di essere stati sufficientemente chiari, vi diamo appuntamento alla prossima puntata, nella quale approfondiremo gli esempi per la nostra scheda e altri aspetti di Azure Sphere. □



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>

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

Sostituire il telecomando TV? Nulla di più facile!

Il tuo telecomando TV è rotto o non funziona più bene?
È arrivato il momento di sostituirlo con i telecomandi universali a infrarossi compatibili con le TV delle maggiori marche prodotte dal 2000 ad oggi. Dispongono di tutte le funzioni del telecomando originale. Subito utilizzabili!



PRONTI ALL'USO



SAMSUNG

Cod. **TVIR-SAMSUNG**



SONY

Cod. **TVIR-SONY**



PHILIPS

Cod. **TVIR-PHILIPS**



LG

Cod. **TVIR-LG**



Panasonic

Cod. **TVIR-PANASONIC**

€ 5,90
OGNI PEZZO

DA PROGRAMMARE



€ 9,00
cod. **TVFREEDOM2**

2 in 1

DISPONIBILE ANCHE 1 in 1
cod. **TVFREEDOM1**

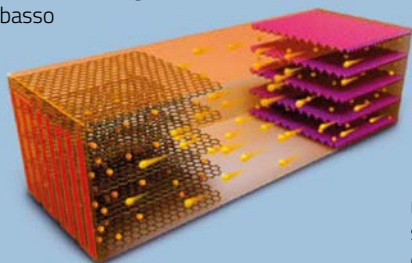
€ 8,50

Telecomando universale, programmabile da PC, per controllare qualunque dispositivo di qualsiasi marca e modello. Include tutte le funzioni dell'originale, configurazione facile e veloce. Mai obsoleto grazie ad aggiornamenti gratuiti. Oltre 260.000 modelli in continuo aggiornamento, può comandare fino a due dispositivi audio/video: TV, TDT, SAT, DVD, BLURAY, proiettori, soundbar, VCR, media center e impianti Hi-Fi delle migliori marche. Richiede il download su PC-Windows di software gratuito per la selezione e la programmazione del telecomando.



10 minuti per 400 km di autonomia

I ricercatori dell'Università di Stato della Pennsylvania (U.S.A.) hanno sviluppato una batteria per auto elettriche capace di accumulare in soli 10 minuti energia sufficiente a percorrere 400 km: un record che fa ben sperare nella possibilità di rifornire le auto elettriche in tempi paragonabili a quelli delle vetture a motore termico. A rendere interessante la tecnologia ci sono aspetti come la possibilità di realizzare accumulatori ad alta densità d'energia e dal basso



costo. La lunga durata e la ricarica rapida sono dovute alla capacità della batteria di auto-riscaldarsi rapidamente fino a 60° C e di raffreddarsi quando sono sconnesse. La ricarica rapida consente di ridurre le dimensioni della batteria, che dovrebbe garantire complessivamente nell'arco della sua vita una durata di più di tre milioni di chilometri. La batteria sviluppata dal team della Penn State University è del tipo litio-ferro-fosfato (LiFePO₄) più economica rispetto a quelle attualmente in uso. Questo segna un altro importante passo nel percorso di transizione alla mobilità elettrica, ad oggi ostacolata dagli elevati costi, dal tempo di ricarica e dalla ridotta autonomia. Queste caratteristiche sembrano poter essere risolte dalle nuove batterie. Per elevare la densità energetica le batterie dei veicoli

elettrici utilizzano catodi di litio-ferro-fosfato (LFP) con ossidi a strati ternari sempre più ricchi di nichel e meno di cobalto. Impiegano inoltre la grafite in aree a bassa superficie, prolungando così la durata a circa due milioni di miglia.

Le batterie agli ioni di litio si degradano quando vengono caricate rapidamente a temperature ambiente inferiori a 50 gradi Fahrenheit perché gli ioni di litio invece di inserirsi negli anodi di carbonio si depositano in creste sulla superficie dell'anodo, formando un rivestimento di litio che riduce la capacità delle celle, ma può anche causare picchi elettrici e condizioni non sicure della batteria.

Le batterie riscaldate al di sopra della soglia di placcatura al litio, sia per riscaldamento esterno che interno, non soffrono di questo problema.

news.psu.edu

Satelliti in orbita con un razzo lanciato da un 747

Virgin Orbit, la compagnia spaziale fondata da Richard Branson (proprietario anche di Virgin Galactic) è finalmente riuscita a mandare in orbita un razzo lanciato nello spazio dall'ala di un Boeing 747 della sua compagnia aerea Virgin Atlantic appositamente modificato e chiamato Cosmic Girl, aprendo la strada a un nuovo metodo di lancio di satelliti a basso costo. Dopo un test preliminare a maggio 2020, stavolta Cosmic Girl è decollato dal Mojave Air and Space Port nel deserto a nord di Los Angeles, ha raggiunto la quota di lancio e qui ha lasciato andare il razzo agganciato all'apposita struttura posta sotto l'ala sinistra dell'aereo; mentre il 747 "scivolava d'ala" e poi virava per allontanarsi dalla zona, il razzo si accendeva in pochi istanti e prendeva la traiettoria che l'avrebbe portato in orbita, a 500 km di quota, raggiunta la quale ha aperto la sommità rilasciando il suo carico composto da satelliti.

Virgin Orbit nasce come alternativa alla tecnica di lancio tradizionale con razzi che partono da terra e si rivolge

a operatori di piccoli satelliti di peso compreso tra i 300 e i 500 chili, un mercato in piena espansione perché lanciare un razzo da un aereo è più semplice che farlo con un lancio verticale da una piattaforma di lancio tradizionale; infatti il Boeing 747 si porta ad alta quota (il lancio è avvenuto a 35.000 piedi) e lì rilascia il razzo, il quale si può dire che parte con 11.000 metri di "vantaggio".

Questo genere di lanci è più economico e non richiede altra infrastruttura che una pista di decollo e atterraggio per il velivolo "lanciatore", che in questo caso è il 747.

Il razzo utilizzato da Virgin Orbit è lungo 21 metri, chiamato LauncherOne e fino al rilascio è agganciato alla parte inferiore di un'ala del Boeing 747 mediante un apparato appositamente installato, poco impattante sul piano dell'aerodinamica, collocato tra i due motori turboventola dell'aereo.

www.virginorbit.com



Un pieno d'idrogeno eolico per il trasporto su strada

Prende il via in Danimarca il progetto H2RES, finalizzato a produrre idrogeno puro da fonte eolica per l'utilizzo come combustibile per l'autotrazione.

A renderlo noto è Ørsted, la società che assieme ad altri sei partner sta portando avanti l'iniziativa. Il progetto dimostrativo di produzione dell'idrogeno da fonte rinnovabile H2RES avrà come sede Avedøre Holme a Copenhagen, dove un'installazione eolica off-shore (ossia in mare) sarà sfruttata per generare elettricità, parte immessa nella rete elettrica locale, parte impiegata per produrre fino a 1.000 kg di idrogeno rinnovabile al giorno, che verrà utilizzato per alimentare il trasporto stradale.

Il progetto dovrebbe entrare in funzione alla fine del 2021.

L'idrogeno per l'autotrazione dev'essere puro, perché così lo richiedono le fuel-cell dedicate all'automobile; il problema è che produrlo costa molto e se ottenuto da elettrolisi dell'acqua, l'elettricità richiesta supera quella prodotta dalle pile a combustibile dell'auto.

Poter contare su una fonte energetica praticamente gratuita come l'eolico rende l'idrogeno conveniente ed ecosostenibile, facendolo reconsiderare come vettore energetico per l'autotrazione.

Ørsted ha collaborato negli ultimi 18 mesi con diversi consorzi in sette progetti di idrogeno rinnovabile in Danimarca, Germania, Paesi Bassi e Regno Unito.

Le risorse eoliche offshore globali praticamente illimitate sono ideali per alimentare l'elettrolisi dell'idrogeno rinnovabile.

Il progetto H2RES studierà come combinare al meglio un elettrolizzatore con l'alimentazione incostante dell'eolico offshore, utilizzando le due turbine eoliche da 3,6 MW di Ørsted ad Avedøre Holme.

orsted.com/en

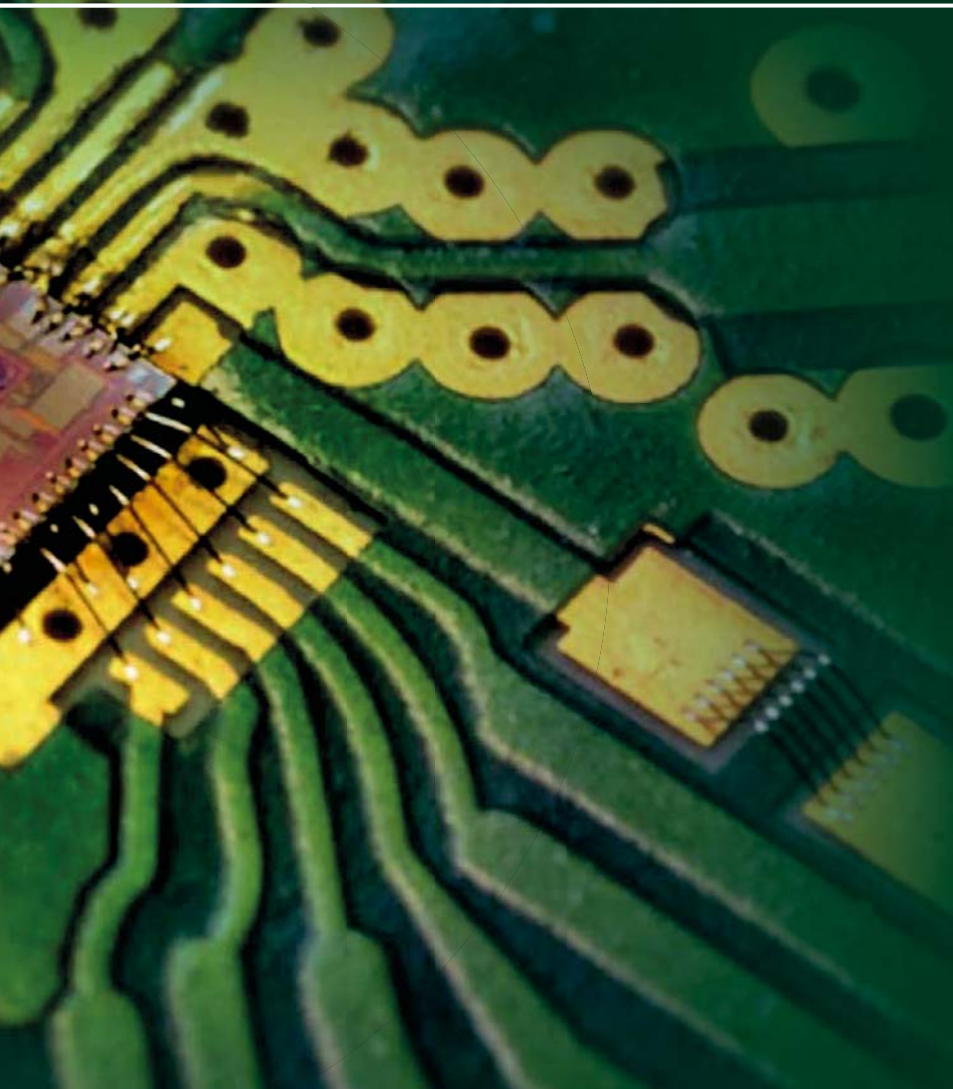


Il freddo rende i computer quantistici più potenti e compatti

Per la maggior parte dei computer quantistici, il calore è considerato un nemico perché causa errori nei qubit falsando le operazioni in esecuzione; per questo i computer quantistici devono essere mantenuti molto freddi, durante il loro funzionamento: idealmente appena un po' sopra lo zero assoluto (-273,15 °C). Tuttavia tale condizione non è facile da attuare perché per far funzionare un computer è necessaria un'interfaccia con il mondo non quantistico, quindi una serie di componenti elettronici tradizionali, i quali scaldano in una certa misura e non funzionano in modo ottimale in ambiente molto freddo.

Per realizzare computer quantistici migliori e più efficienti, i ricercatori stanno cercando di portare la maggior parte possibile di tali componenti elettronici nel frigorifero a diluizione che ospita i qubit stessi. Tra i modi per far funzionare i circuiti a temperature

criogeniche spicca ad esempio quello proposto da Google, che ha sviluppato un circuito integrato criogenico per il controllo dei qubit, collegandolo con altri dispositivi elettronici. Il dispositivo di Google funziona a 4 kelvin all'interno del frigorifero, solo leggermente più caldo dei qubit che si trovano a circa 50 centimetri di distanza. Ciò potrebbe ridurre drasticamente l'ingombro di quelli che ora sono rack da elettronica delle dimensioni di una stanza. Il controllo efficiente dei computer quantistici è fondamentale poiché raggiungono 100 qubit o più. Gli amplificatori criogenici a basso rumore facilitano la lettura dei qubit. Da soli, quei qubit emettono deboli segnali RF. Inserire l'amplificatore a basso rumore (LNA), può amplificare quei segnali e facilitare la lettura dei qubit. Non sono solo i computer quantistici a beneficiare degli LNA criogenici: li utilizzano anche i radiotelescopi e le reti

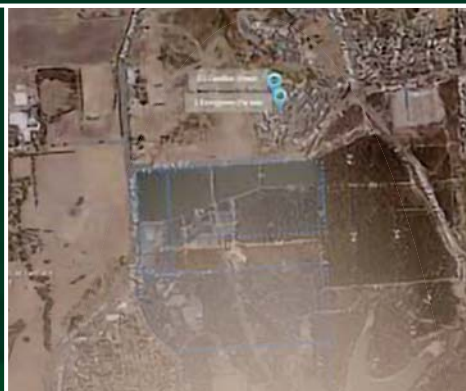


di comunicazione nello Spazio.

I ricercatori della Chalmers University of Technology di Gothenburg, in Svezia, sono tra coloro che cercano di produrre crio-LNA. Il loro circuito utilizza transistor ad alta mobilità elettronica (HEMT), particolarmente utili per la commutazione e l'amplificazione rapida della corrente. I ricercatori utilizzano transistor realizzati con fosforo di indio (InP), un materiale familiare per gli LNA, sebbene l'arseniuro di gallio sia più comune in commercio. Tali transistor in fosforo di indio trovano applicazione in sistemi operanti in condizioni di congelamento profondo, perché il materiale di cui sono fatti lavora meglio, sotto l'aspetto della velocità di conduzione degli elettroni, a basse temperature piuttosto che a temperatura ambiente. I ricercatori hanno sperimentato a lungo l'utilizzo degli HEMT in fosforo di indio negli LNA, ma puntano a spingere i loro

circuiti a funzionare a temperature più basse e a utilizzare meno energia. I dispositivi funzionano a partire da 4 kelvin, una temperatura che li rende idonei a operare nel frigorifero a diluizione di un computer quantistico. Altro elemento importante di un computer quantistico è il cablaggio: numerosi fili collegano i qubit alla loro elettronica di controllo, leggendone gli stati e restituendo i segnali agli ingressi. Alcuni di questi cavi possono essere eliminati utilizzando un multiplexer RF (RF MUX) ossia un circuito in grado di controllare i segnali da e verso più qubit; una ricerca recente ha sviluppato multiplexer del genere in grado di unire i qubit nel frigorifero del computer quantistico. A differenza di molti circuiti criogenici, che funzionano a 4 kelvin, il nuovo RF MUX può funzionare fino a temperature di alcuni millikelvin.

eds.ieee.org



In Australia, il più grande fotovoltaico mobile

SA Water, utility australiana, realizzerà il più grande parco fotovoltaico mobile al mondo, nella Riserva Happy Valley di Adelaide. Il sistema sarà costituito da 30.000 pannelli solari, posizionati in prossimità del bacino idrico della riserva. Il primo modulo è già stato installato e l'impianto, a regime, dovrebbe sviluppare energia per oltre 17 GWh l'anno.

La particolarità di questa installazione è nella tipologia: l'impianto sarà infatti composto da pannelli solari dell'australiana 5B, i quali sono blocchi modulari "plug and play" prefabbricati e precablati, che vengono depositi sul sito e quindi interconnessi. Tale configurazione "mobile" richiede una preparazione minima del sito e lascia la libertà di spostare i blocchi in altri luoghi. Ecco perché l'impianto della Happy Valley sarà il più grande parco fotovoltaico mobile.

La riserva è uno degli oltre 30 siti che SA Water ha selezionato per l'installazione di pannelli solari nell'Australia meridionale e per la sua realizzazione l'utility australiana ha studiato misure di riduzione dell'impatto visivo e ambientale, quali la minimizzazione dell'area occupata e la piantagione di vegetazione nativa per creare strutture protettive e schermi naturali.

Il parco fotovoltaico alimenterà l'impianto di trattamento delle acque della Happy Valley. L'array della riserva fa parte di un progetto più ampio dedicato alle rinnovabili con cui SA Water mira all'installazione di oltre 500.000 pannelli fotovoltaici, per un totale di 242 GWh di produzione di elettricità "verde" all'anno.

L'iniziativa si affianca ad altre in programma in siti, per un totale di oltre 160.000 pannelli fotovoltaici.

www.sawater.com.au

Accelerare le particelle con la stampa 3D

Di recente gli ingegneri del CERN hanno raggiunto un raffreddamento senza precedenti, pari a -40°C , all'interno dello spazio estremamente limitato della striscia di fotorealizzazione del Large Hadron Collider, l'acceleratore di particelle nato per sperimentare l'origine della materia. Tale temperatura è necessaria per rilevare efficacemente le particelle create all'interno dell'acceleratore, ma per ottenerla servivano raffreddatori

troppo sottili per essere ottenuti con le tecniche tradizionali. Qui è entrata in gioco la stampa 3D additiva Direct Metal Printing (DMP) della di 3D Systems, che ha permesso di raffreddare le strisce di fotorealizzazione, lunghe circa 140 metri e larghe meno di due millimetri, mediante barre refrigeranti in titanio



stampate in 3D, alte 263 mm.

I requisiti del sistema di raffreddamento imponevano una minima quantità di materiale interposto fra le strisce rivelatrici e il liquido di raffreddamento e la superficie da raffreddare, tanto minima da non poter essere ottenuta con le tradizionali tecniche di lavorazione del titanio. Con la stampa 3D è stato invece possibile raggiungere lo scopo, stampando uno spessore di parete di 0,25 mm. La tecnica

utilizzata è stata quella di deposizione e sinterizzazione mediante laser della polvere di titanio, che ha portato alla realizzazione del materiale LaserForm TiGr23, una lega di titanio ad alta resistenza.

it.3dsystems.com

Ridurre la CO₂? Basta aspirarla

Il problema dei gas serra e in particolare dell'anidride carbonica è uno dei più grandi del nostro tempo e non basta puntare a ridurne la produzione, dato che più di tanto non è fattibile.

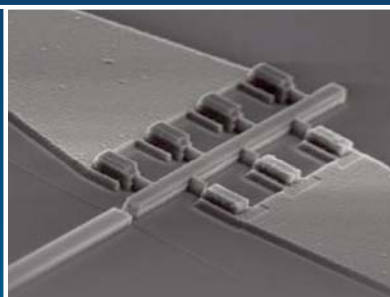
Allora scendono in campo tecnologie per assorbirla dall'aria, come quella della canadese Carbon Engineering, che mediante ventilatori giganti sparsi estrarrà l'anidride carbonica dall'aria e la pomperà in profondità nel sottosuolo. Una volta completato, il progetto potrebbe catturare 1 milione di tonnellate di anidride carbonica all'anno, facendo il lavoro di pulizia dell'aria di circa 40 milioni di alberi. Siccome non ci si può limitare a catturare e imprigionare la CO₂, gli impianti puntano a riciclarla. L'impianto che Carbon Engineering realizzerà in Texas utilizzerà banchi di ventilatori, ciascuno di circa 8,5 metri

di diametro, per aspirare l'aria in una struttura chiamata contattore, dove si trova una rete di plastica rivestita con una soluzione di idrossido di potassio, che si lega all'anidride carbonica.

Una serie di processi chimici concentra e comprime la CO₂, poi riscaldata a 900°C per riciclarla. Il gas può anche essere combinato con l'idrogeno, come avviene nell'impianto pilota nella Columbia Britannica, per produrre petrolio greggio sintetico.

L'anidride carbonica non convertita può essere iniettata nei pozzi petroliferi più vecchi, sequestrandola nel sottosuolo e spingendo fuori, nel contempo, il petrolio rimanente. Impianti del genere possono aspirare dall'aria milioni di tonnellate equivalenti al giorno.

carbonengineering.com



La guida d'onda plasmonica collega i processori con la luce

I collegamenti in fibra ottica sono il metodo principale per trasferire dati tra cluster di computer nei data center; per raggiungere lo scopo ci vorrebbero ricetrasmittitori a plasmoni, che peraltro potrebbero essere costruiti sopra il processore. I plasmoni sono onde di densità elettronica generate quando la luce colpisce l'interfaccia tra un metallo e un isolante; possono viaggiare lungo una guida d'onda ma si esauriscono rapidamente perché il metallo assorbe la luce. I ricercatori dell'Università di Toronto hanno inventato una struttura capace di ridurre la perdita, chiamata guida d'onda plasmonica ibrida accoppiata (CPHW): si tratta di una pila composta da silicio, ossido di stagno e indio conduttore, biossido di silicio, alluminio e altro silicio. Questa combinazione forma due tipi di giunzioni a semiconduttore, con l'alluminio che contiene il plasmon in comune tra i due.

All'interno del metallo, il plasmon nella giunzione superiore interferisce con quello nella giunzione inferiore in modo da ridurre la perdita. Utilizzando il CPHW come base sono stati realizzati un modulatore che trasforma i bit elettronici in bit fotonici, e un fotorelevatore, che fa il contrario.

Il sistema CPHW consuma meno di 1/3 dell'energia per bit trasmesso rispetto a quelli fotonici e potrebbe trasmettere a velocità di 150 gigabit senza bisogno di correzione degli errori.

opto.utoronto.ca



Sfoggia on-line i nostri cataloghi

**FUTURA
ELETTRONICA®**



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.

