MENSILE DI PROGETTAZIONE ELETTRONICA · ATTUALITÀ SCIENTIFICA · NOVITÀ TECNOLOGICHE

Elettronica In

www.elettronicain.it

€ 6,00 - Anno XXVI - n. 243 - MARZO 2020

9 771124 854008 00243

Micro telecamera wireless con ESP32

- I bus per automotive
- I²C Extender
- Circuito giratore
- Domotica per tutti
- Sirena bitonale

Intelligenza artificiale su Raspberry Pi

Bike tracking con SigFox

CONOSCERE E USARE kivy

Poste italiane Spa - Spedizione in abbonamento Postale: D.L. 353/2003 (conv. in L. 27/02/2004 n°46) art. 1 - comma 1 - DCB Milano. Prima immissione: 03/03/2020.

Più sicurezza con le nuove telecamere WiFi

Sicurezza subito, installazione facile e veloce!

cod. FR769

€ 73,00

Telecamera WiFi motorizzata

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

Prezzi IVA inclusa.

Telecamera WiFi da 2Mpx con pannello solare e batteria

cod. FR767

€ 125,00

- Sensore d'immagine FHD 1080P CMOS 1/2.7 PS5230
- Risoluzione Video 1920x1080/15 fps 640x360/30 fps
- Modalità Day/Night automatica
- Illuminatore IR con LED IR da 850 nm e portata di circa 10 metri
- Obiettivo 2G2P HD, F=2,8, con angolo di ripresa di 120°
- Alimentazione con 2 batterie al litio tipo 18650 ricaricate tramite pannellno solare
- integrato
- Formato di compressione video H264
- Protocolli: Network TCP/IP, HTTP, TCP, UDP, SMTP; Wi-Fi 802.11b/g/n
- Registrazione su SD-Card (da 4GB a 64GB)
- Grado di protezione IP67
- S.O. supportati: iOS10.0, Android 5.0 o superiore
- Temperatura di funzionamento da -10°C a +50°C

cod. FR724N

€ 98,00

Telecamera WiFi da 1.3 Mpx ad ottica fissa

- Elemento sensibile 1/3 CMOS
- ISP Hisilicon Hi3518E
- Risoluzione 1.3 Mpx – 1280x1080p
- Frame rate 1280x1080@25fps
- Apertura angolare 110°
- 6 LED IR da 850nm
- Sensore PIR motion detection
- Portata LED IR 6 metri
- Alimentazione con 2 batterie 18650 (incluse)
- Temperatura di lavoro da -20°C a 55°C
- Dimensioni 95x53x80 mm
- Peso 550g

cod. FR761

€ 42,00

Telecamera WiFi a 360° nascosta in una lampadina

- Potenza lampadina 6W
- Sensore di immagine da 2Mpx CMOS
- Visuale di ripresa 360°
- Definizione immagine 1080PH
- Microfono bidirezionale per ascoltare e parlare direttamente attraverso la telecamera
- Modulo Wi-Fi integrato
- Night & Day con tecnologia IR-Cut
- Registrazione su SD-Card (max 128GB)
- Compressione video H.264
- Funzione motion detection
- Push di allarme e invio email
- Peso: 200 g

cod. FR766

€ 42,00

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

- Elemento sensibile CMOS colori 1/3", 2 Mpx
- Risoluzione massima: 1920x1080
- Obiettivo: 360° fisheye
- Illuminatore IR: 3 LED
- Portata: 10 m
- Night&Day automatico
- Motion detection con push di allarme
- Porta Ethernet RJ45: 10/100Mbps
- Protocolli: ONVIF 2.0,TCP/IP, HTTP, TCP, UDP, SMTP, FTP, DHCP, DNS, DDNS, NTP, UPnP, PPPoE, P2P
- Sicurezza Wi-Fi: WPS encryption
- Visione remota tramite Smartphone (Android)
- Uscita video: IP
- APP: IP PRO per Android scaricabile gratuitamente
- Alimentazione 12 Vdc < 1A (con adattatore di rete non incluso)
- Temperatura di lavoro: da -10°C a +50°C
- Dimensione 15,2x16,5x 4 cm
- Peso 250 g

FUTURA ELETTRONICA®

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

Caratteristiche tecniche e vendita on-line su: www.futurashop.it

2020, l'anno dell'auto elettrica per tutti

L'auto elettrica sembra ormai aver raggiunto il traguardo tanto sospirato, ovvero quella della sostanziale parità di costi e prestazioni rispetto alle vetture a combustione interna. Dopo anni di corsa all'elettrico, con vetture dai costi esorbitanti e dalla scarsa autonomia, possiamo finalmente affermare che è arrivata l'ora dell'elettrico per tutti. Negli ultimi mesi sono infatti arrivate sul mercato una serie di utilitarie il cui costo di listino è di poco superiore ai 20 mila euro e la cui autonomia è più che sufficiente per un normale tragitto casa-lavoro o per il classico tour quotidiano di impegni in città. A questa categoria di vetture appartiene, ad esempio, la nuova Seat Mii electric che costa "su strada", completa di accessori, 23.250 Euro e dichiara un'autonomia di 260 km. La Mii ha un motore di 61 kW (83 CV) con coppia massima di 212 Nm che consente di accelerare (... la cosa bella anche in una piccola elettrica) da 0 a 50 km/h in 3,9 secondi e di raggiungere una velocità massima di 130 km/h; la batteria agli ioni di litio da 32,3 kWh si ricarica tranquillamente in una notte mediante l'impianto elettrico di casa. E se il costo può sembrare ancora un po' alto, dobbiamo considerare che con gli incentivi statali e con lo sconto concessionario il prezzo può arrivare a circa 15.000 Euro (o anche meno se abitate in una regione che prevede un ulteriore bonus). Non dobbiamo dimenticare poi che il costo di esercizio di una vettura elettrica è sicuramente inferiore, non si paga il bollo, ecc. Naturalmente la vettura spagnola non è la sola ad offrire queste caratteristiche; molto simili sono i costi e le prestazioni della e-Up, della Skoda Citygo e di altre vetture ancora. Ed è veramente un peccato che progetti come quello della Sion, la prima *Solar Electric Car,* siano ancora in alto mare nonostante la recente campagna di Community Funding abbia avuto grande successo. La possibilità di ricaricare le batterie con i pannelli solari montati sulla vettura (pur con una ricarica minima, pari ad una percorrenza di una trentina di chilometri al giorno), potrebbe convincere molte altre persone a passare all'elettrico, specie quanti (giustamente) obiettano che l'uso di una vettura elettrica, stante l'attuale mix energetico italiano, non fa altro che spostare l'inquinamento atmosferico dalle città dove la vettura viene utilizzata, ai luoghi dove sorgono le centrali che producono elettricità bruciando combustibili fossili. Con la Sion, considerando che un pendolare percorre meno di trenta chilometri al giorno, potremmo muoverci in auto con la certezza assoluta di non inquinare. Un po' come succede in alcune nazioni del nord Europa (Scozia, Danimarca, ecc.) dove l'energia elettrica viene prodotta quasi esclusivamente da fonti rinnovabili. Non a caso in questi paesi la vendita di vetture elettriche ha raggiunto cifre da record. Che dire? Compatibilmente con quelle che sono le capacità produttive dei car maker e la scarsa diffusione dei punti di ricarica, il 2020 sarà sicuramente l'anno dell'auto elettrica o perlomeno l'anno che verrà ricordato per aver dato il via alla mobilità elettrica su larga scala.

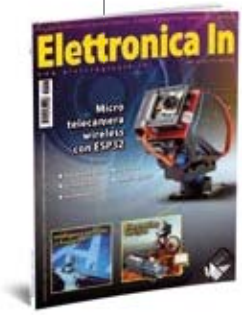
Con lo sconto concessionario e gli incentivi statali il prezzo può arrivare a circa 15 mila Euro o anche meno, se la vostra regione prevede ulteriori bonus.

ON LINE

elettronicain.it

SI RINGRAZIA

Fiera di Novegro
Futura Elettronica
Microchip
RM Elettronica
TME Italia S.r.l.



In copertina:
Sfruttiamo un modulo ESP32 per realizzare una microcamera wireless con tanto di controllo PAN/TILT.

Arsenio Spadoni
arsenio.spadoni@elettronicain.it

Elettronica In

Rivista mensile, anno XXVI n. 243
MARZO 2020

EDITORIALE

Direttore Responsabile
Arsenio Spadoni
arsenio.spadoni@elettronicain.it

Redazione
Stefano Garavaglia, Paolo Gaspari,
Boris Landoni, Davide Scullino,
Gabriele Daghetta, Alessandro Sottocornola
redazione@elettronicain.it

Grafica
Alessia Sfulcini
alessia.sfulcini@elettronicain.it

Ufficio Pubblicità
Monica Premoli (0331-752668)
monica.premoli@elettronicain.it

Ufficio Abbonamenti
Tel. 0331-752668
abbonati@elettronicain.it

Direzione, Redazione, Pubblicità
FUTURA GROUP srl - Divisione Editoriale
via Adige 11 - 21013 Gallarate (VA)
Tel. 0331-752668

Abbonamenti
Annuo 10 numeri Euro 45,00
Estero 10 numeri Euro 45,00 (digitale)
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.elettronicain.it/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

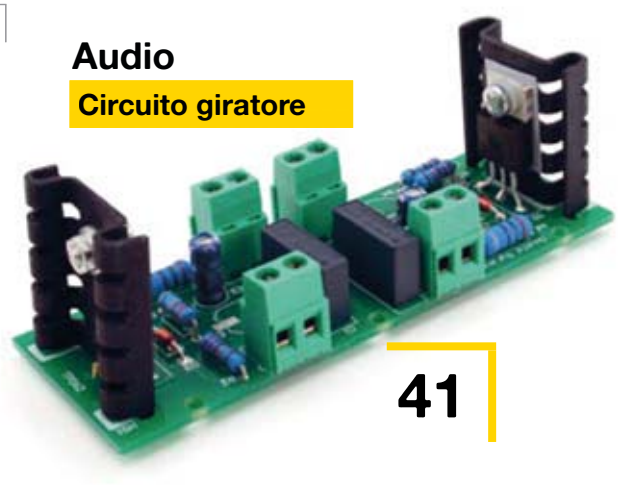
Rivista mensile registrata presso il Tribunale di Milano con il n. 245 il 03/05/1995.
Prezzo di copertina Euro 6,00.
Gli arretrati nei formati cartaceo e digitale (pdf) sono acquistabili sul sito della rivista al prezzo di Euro 6,00.
Poste Italiane Spa - Spedizione in abbonamento Postale - D.L. 353/2003 (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.

Contenuti

01 RUBRICHE

Audio
Circuito giratore



23 ARTICOLI

Progetto di Copertina

Didattica
Domotica per tutti

Didattica
Sirena bitonale

Wireless
Bike Tracking

71

Didattica
I Bus per automotive

83

Didattica
Corso Kivy

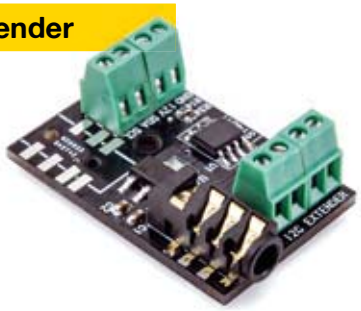
113

Didattica
Intelligenza artificiale su Raspberry Pi

103

Periferiche
I²C Extender

97



DAI FORMA ALLE TUE IDEE!

cod. **FOAMCUTTER**

€ 320,00

PLOTTER TAGLIA POSTIROLO - IN KIT

Realizza, intagliandole a caldo, sagome di ogni genere partendo da lastre di polistirolo, polistirene ed altri materiali espansi a bassa temperatura di fusione. Collegalo subito al PC e utilizzalo per realizzare tutto quello che vuoi: modelli di vario genere, scritte, scatole, sagome per decorazioni, lettere scatolate, loghi, simboli, puzzle 3D e molto altro!

CARATTERISTICHE TECNICHE:

- **Tecnica di taglio:** a filo caldo
- **Materiali lavorabili:** espansi a bassa temperatura di fusione
- **Dimensioni:**
  - **Plotter:** 775x690x220 mm
  - **Area lavorabile:** 500x480 mm
  - **Spessore lavorabile:** 120 mm
- **Risoluzione:** 0,5 mm
- **Tolleranza X Y:** 0,012 mm
- **Velocità di taglio:** 300 mm/minuto (varia in funzione della tipologia di materiale lavorato)
- **Alimentazione:** 12 VDC / 3 A
- **Temperatura filo:** regolabile da software
- **Gestione:** da PC tramite interfaccia USB

Prezzo IVA inclusa.

FUTURA ELETTRONICA®

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

www.futurashop.it

Q&A

Queste pagine sono dedicate alle richieste, ai suggerimenti ed alle segnalazioni dei lettori. Raccomandiamo, per quanto possibile, di proporre argomenti di interesse generale. Contattateci all'indirizzo: **redazione@elettronicain.it**

Accendere le luci con la voce

Ho acquistato recentemente un interruttore WI-Fi che utilizzo con una APP del mio smartphone per accendere e spegnere le luci del soggiorno. Ho visto recentemente dei prodotti simili che però funzionano anche con Amazon Alexa e altri assistenti vocali. Posso utilizzare il mio dispositivo anche con questi prodotti?

Roberto Martelli > Roma

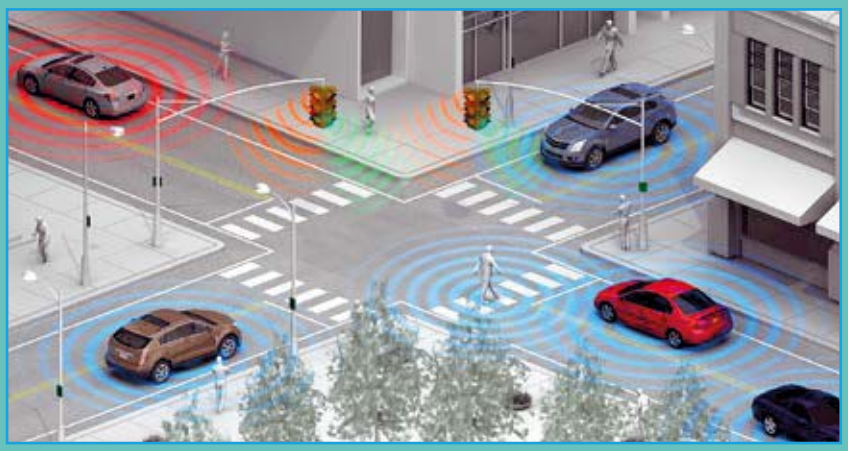
R: Teoricamente si, ammesso che tu disponga degli strumenti e delle capacità necessarie per lavorare con Alexa Skill kit e che conosca le caratteristiche del tuo interruttore Wi-Fi.
Molto più semplice è l'impiego con un nuovo interruttore intelligente già predisposto per operare, oltre che con la tua APP su smartphone, anche con i comandi dei vari assistenti vocali, Amazon Alexa, Google Assistant, ecc.
Questi dispositivi possono essere facilmente abbinabili all'assistente vocale senza dover scrivere una linea di codice.
D'altra parte il costo di questi prodotti è decisamente alla portata di chiunque. Ad esempio, la nuova presa intelligente Wi-Fi FR779 di Futura Elettronica (www.futurashop.it) costa appena 21,90 Euro.
Si tratta, di una presa smart Wi-Fi compatibile con sistemi Amazon Alexa, Google Assistant e IFTTT.
Può essere gestita tramite APP da smartphone, per accendere / spegnere luci ed elettrodomestici, da qualsiasi luogo in cui si dispone di una connessione Internet. Dispone anche di funzione timer per pianificare giorni e orario di accensione e spegnimento dei dispositivi collegati ed è dotata di sistema per il monitoraggio del consumo di energia elettrica, che consente di individuare quali sono le apparecchiature con maggior consumo e di farne un uso più attento e intelligente in modo da tagliare i costi della bolletta elettrica. La presa è in grado di supportare un carico complessivo di 3400W con corrente massima di 16A.
L'abbinamento sia all'APP dello smartphone che ad Alexa è molto semplice e intuitivo.

I sistemi V2X, chi vincerà?

Ho letto che è in atto un braccio di ferro tra due differenti sistemi di comunicazione di sicurezza per autovetture connesse, meglio noti come V2X, ovvero Vehicle to everythings, dove everythings può essere l'infrastruttura stradale, le altre vetture, i pedoni, ecc. Che differenza c'è tra i due sistemi e chi, secondo voi, prevarrà?

Luigi Colombo > Milano

R: Difficile fare pronostici, certo che il primo sistema, quello che si basa su rete Wi-Fi e che risponde allo standard 802.11p è pronto da tempo, anche se a livello di infrastruttura poco o nulla è stato ancora fatto nel frattempo. Questo sistema di basa su tecnologia non licenziata a 5,9 GHz esclusivamente assegnata allo sviluppo dell'auto connessa e ai servizi ITS (Intelligent Transport



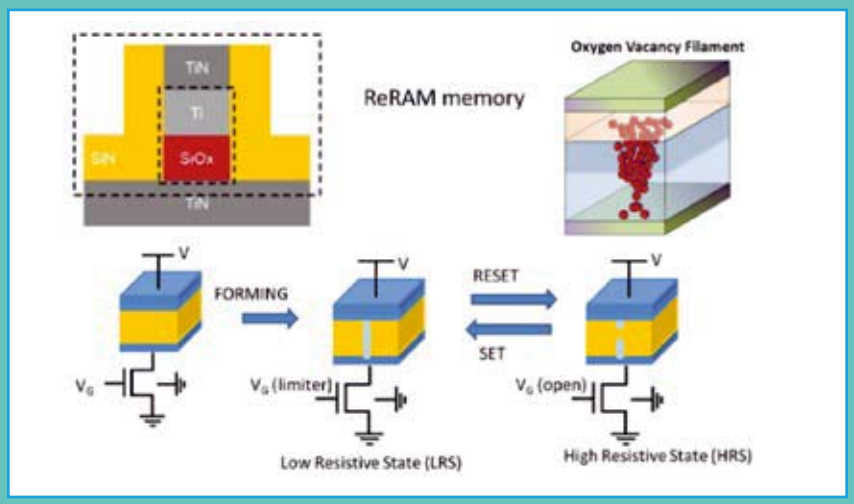
System). Ad oggi l'unica vettura di serie dotata di questo sistema è la nuova Volkswagen Golf che implementa la soluzione di comunicazione salvavita RoadLINK V2X di NXP. Il vantaggio di V2X basato su Wi-Fi è la comunicazione solida, a bassa latenza e in tempo reale, indipendentemente dalle marche di auto: consente la conoscenza e la comunicazione tra auto, infrastrutture stradali come semafori o segnali stradali e altri utenti della strada come ciclisti e pedoni.
È una tecnologia collaborativa che consente di "attingere" ai dati dei sensori circostanti e dalle auto reciprocamente equipaggiate per avvertire di pericoli e prevenire incidenti.
Aiuta i veicoli a "vedere" oltre un miglio più avanti e dietro gli angoli nascosti per fornire un allarme tempestivo di ostacoli, pericoli e condizioni stradali.
Per quanto riguarda la tecnologia C-V2X, la normativa fa riferimento alle Release 3GPP 14, 15 e 16; l'obiettivo del 3GPP è ovviamente anche quello di supportare lo sviluppo di nuove opportunità di business per l'industria delle telecomunicazioni arrivando ad un sistema con performance in grado di poter supportare tutti i livelli di autonomia di guida, dalla guida assistita (L1 dello schema SAE 3) fino a quello più sfidante e complesso della guida autonoma (L5), secondo un approccio a fasi incrementali. Secondo i sostenitori del sistema cellulare, sono quattro le ragioni a sostegno di questa tecnologia.
La prima è quella della disponibilità delle relative infrastrutture sul territorio nazionale (autostrade, strade urbane e rurali) e dei relativi costi di copertura. I sistemi cellulari 4G coprono oggi oltre il 98% della popolazione, mentre i sistemi Wi-Fi coprono solo gli ambienti interni e poche strade urbane: la differenza dei costi di copertura è ovviamente a favore del cellulare.
La seconda riguarda le prestazioni. I sistemi cellulari offrono comunicazioni a tutto campo: V2V, V2I, V2P, V2N, e cioè veicolo verso veicolo, verso infrastruttura, verso pedone o ciclista, verso la rete, e cioè a lunghissima distanza. I sistemi Wi-Fi sono solo V2V e V2I, non consentono l'inclusione dei pedoni, dei ciclisti e non tengono conto di ciò che avviene sul territorio a distanza dalle piccole celle Wi-Fi (ad esempio incidenti, autoambulanze in arrivo, attacchi informatici, ecc.).
La terza ragione fa riferimento alle prestazioni tecnologiche: copertura, throughput, latenza, velocità dei veicoli, affidabilità, scalabilità, ridondanze di rete, capacità di broadcasting. I sistemi cellulari offrono prestazioni superiori per tutti questi aspetti.
La quarta ragione, infine, riguarda la sicurezza, con i servizi cellulari che sono dotati di piattaforme su base nazionale che garantiscono a tutta la rete veicolare (autostradale, urbana e rurale) la supervisione degli accessi dei sensori veicolari e la loro identificazione tramite canali di controllo, additivi rispetto ai canali di trasporto dei dati. In caso di attacchi informatici, i dispositivi di cybersecurity dell'intera rete segnalano alla periferia e bloccano l'accesso dei sensori di cui si sono impadroniti gli attaccanti. Ciò non avviene nel caso della rete Wi-Fi.
In Europa, da tecnico il problema è diventato anche politico, con gli organi comunitari che si sono espressi in maniera contradditoria in passato. Sarà il nuovo Consiglio appena insediato a decidere sul da farsi. Di sicuro la lobby delle telecomunicazioni non mollerà la presa su questa che è anche un'altra importante opportunità di business in ambito LTE/5G.

Memristors: a che punto siamo?

A che punto siamo con la tecnologia delle RAM resistive che tanto interesse avevano suscitato quando vennero annunciate? Esistono già dei prodotti commerciali?

Marino Lucarelli > Firenze

R: Al momento esistono pochissimi prodotti commerciali realizzati con questa tecnologia, anche se, proprio il 2020 dovrebbe rappresentare



↑ La cella ReRAM di Weebit è costituita da due strati di metallo con uno strato di ossido di silicio (SiOx), materiali che vengono utilizzati nelle linee di produzione esistenti dei dispositivi CMOS.

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.elettronicain.it/webforum

l'anno della svolta.
Alla tecnologia ReRAM stanno lavorando molte piccole aziende come Crossbar e Weebit Nano che stanno cercando di superare gli ultimi ostacoli di natura tecnologica e produttiva, prima dell'eventuale commercializzazione.
Traguardo al quale sembra essere arrivata Adesto con la linea di memorie CBRAM, dispositivi in grado di funzionare anche in ambienti difficili.
Attualmente le ReRAM costituiscono una classe di dispositivi ancora in via di sviluppo, ma le loro caratteristiche le rendono in prospettiva interessanti. Infatti l'utilizzo di ossidi binari dei metalli di transizione è compatibile coi processi standard di produzione dei CMOS. Le strutture per la commutazione unipolare non richiedono un transistor di selezione ma basta un diodo con struttura a pila, permettendo quindi uno sfruttamento più razionale della superficie, a vantaggio della densità d'integrazione.
È già stata dimostrata la possibilità di realizzare dispositivi veloci, con tempi di commutazione dell'ordine di 10 ns così come è stata dimostrata la possibilità di realizzare dispositivi multilivello, in grado di memorizzare diversi bit in un'unica cella di memoria.

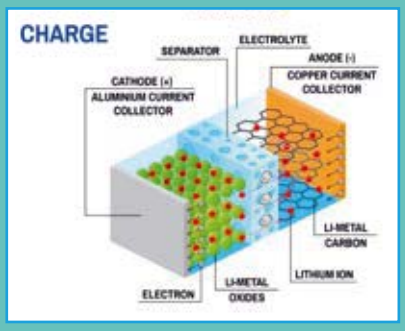
Sulla natura del funzionamento del forming e della commutazione resistiva sono stati formulati diversi modelli, tuttavia la classe dei materiali con proprietà di commutazione resistiva è tanto ampia da presumere che la commutazione sia riconducibile a diversi meccanismi. Inoltre il funzionamento non dipende solo dal materiale utilizzato, ma anche dal processo di deposizione del film e dal tipo di elettrodi utilizzati.
A IEDM 2019, Leti ha presentato un lavoro di ricerca che ha delineato come fabbricare una rete neurale di ispirazione bio completamente integrata, combinando sinapsi basate su ReRAM e neuroni analogici, misurando al contempo una riduzione di 5 volte del consumo di energia rispetto a un chip equivalente mediante codifica formale. L'implementazione della rete neurale è fatta in modo tale che le sinapsi siano posizionate vicino ai neuroni, il che consente l'integrazione diretta della corrente sinaptica.
All'inizio di quest'anno, il produttore di ReRAM Crossbar insieme ad altri, ha dato vita a SCAiLE (SCalable AI for Learning at the Edge), un consorzio di AI dedicato a fornire una piattaforma AI accelerata e a risparmio energetico. Il gruppo si concentra sulla combinazione di ReRAM con hardware di accelerazione avanzato e reti neurali ottimizzate per creare soluzioni pronte ed efficienti dal punto di vista energetico con capacità di apprendimento e riconoscimento degli eventi senza supervisione.
Non dimentichiamo, infatti, che tra le caratteristiche delle ReRAM vi è quella di effettuare operazioni di calcolo direttamente nel chip di memoria. Una caratteristica che ha un potenziale enorme.

Ricarica standard o ultraveloce?

I sistemi di ricarica per auto elettriche cercano di ridurre sempre più il tempo d'attesa con ricariche ultrarapide. Ricariche continue di questo tipo possono, alla lunga, danneggiare o ridurre la vita utile delle batterie dell'auto?

Aldo Barbero > Biella

R: Effettivamente il problema sussiste, anche se, secondo alcune fonti (Tesla), è praticamente insignificante, mentre per altre l'impiego di caricabatterie sempre più veloci può ridurre la vita della batteria in maniera sensibile.
Una ricerca pubblicata recentemente su Energy and Environmental Science, condotta da Daniel Abraham degli Argonne National Laboratory, ha evidenziato come cariche troppo veloci siano piuttosto deleterie per le normali batterie al litio. Viene danneggiato soprattutto l'anodo di grafite della batteria al litio al cui interno si dispongono gli ioni litio dopo aver ceduto un elettrone al catodo. La risposta secondo questo studio è affermativa: maggiore è la velocità di carica, minore sarà la durata della batteria.



Tutto è più facile con i nuovi radiocontrolli a 433 MHz!

Controlla facilmente l'irrigazione, le luci di casa, il cancello elettrico, l'impianto d'allarme e molto altro con i nuovi trasmettitori e ricevitori a 1-2-4 canali operanti sulla banda dei 433 MHz. Forniscono un segnale stabile a lunga distanza e un bassissimo consumo di corrente in modalità stand-by. I ricevitori possono essere alimentati a 220VAC o a 12VDC in base al modello.

RICEVITORE

2 CANALI 220VAC
cod. EMY220RX2 **€ 16,00**

2 CANALI 12VDC
cod. EMY12RX2 **€ 16,00**

4 CANALI 12VDC
cod. EMY12RX4 **€ 19,00**

Istruzioni in italiano!

RICEVITORE

1 CANALE 220VAC
cod. EMY220RX1
€ 13,00

1 CANALE 12VDC
cod. EMY12RX1
€ 13,00

TRASMETTITORE

1 CANALE
cod. EMYTX1
€ 11,00

TRASMETTITORE

2 CANALI
cod. EMYTX2 **€ 12,00**

TRASMETTITORE

4 CANALI
cod. EMYTX4 **€ 13,00**

Ai ricevitori va abbinato esclusivamente il trasmettitore con lo stesso numero di canali.

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

Prezzi IVA inclusa.

Caratteristiche tecniche e vendita on-line su: www.futurashop.it

Tracciare merci con NB-IoT e BT

Nordic Semiconductor equipaggia con le proprie soluzioni NB-IoT il Cloud Tracker della Meshtech; questo apparato può monitorare continuamente parametri ambientali come la temperatura, ma anche se una spedizione è stata fatta cadere, inclinata o piegata, la posizione dei singoli articoli di spedizione (ad esempio carrelli, pallet o scatole) all'interno di una spedizione, l'ordine corrispondente e la posizione geografica dell'intera spedizione in qualsiasi parte del mondo.
Il Cloud Tracker Meshtech è un dispositivo dotato di sensore magnetico a effetto Hall, sensore di temperatura e accelerometro triassiale. È alimentato da sei batterie interne da 3,6 V AA e include tutte le antenne necessarie (LTE e Bluetooth LE) e una scheda SIM preinstallata (eSIM) che può funzionare in tutto il mondo. L'uso di un SoC Nordic nRF52811 non solo fornisce al Cloud Tracker Meshtech la capacità di comunicare con altri dispositivi Bluetooth nelle vicinanze e sensori specializzati, ma anche la piena compatibilità nativa con altre tecnologie wireless a bassissima potenza come Thread e Zigbee, se necessario. Il Nordic Semiconductor nRF9160 SiP (System-in-Package) multimodale ha un modem LTE-M/NB-IoT integrato ed è certificato per applicazioni IoT cellulari globali. È basato su un processore ARM Cortex-M33 a 64 MHz dedicato, 1 MB di memoria Flash e 256 kB di RAM, front-end RF integrato ed incapsulato in un package compatto delle dimensioni di appena 10x16x1 mm. Il modulo incorpora periferiche analogiche e digitali, oltre a implementare la gestione automatizzata di alimentazione e clock, l'ARM TrustZone e l'ARM CryptoCell 310 per la sicurezza a livello applicazione.
www.nordicsemi.com

Condizionamento via IoT

Rittal ha lanciato un adattatore per la sua linea di condizionatori "Blue e" in grado di connetterli, in modo semplice e veloce, ai sistemi intelligenti di Condition Monitoring e di IoT (Internet of Things). L'adattatore consente di mettere in rete la gestione degli impianti di condizionamento. Nello specifico, permette la gestione di dati e il monitoraggio di 10 unità di raffreddamento in configurazione master-slave, la raccolta e la registrazione dei dati relativi ai valori elettrici e termomeccanici, l'analisi di efficienza energetica, la gestione degli allarmi relativi a guasti e al superamento delle soglie impostate.
Alla gestione delle macchine si aggiungono il controllo e il monitoraggio della componentistica elettrica e di comando nelle applicazioni outdoor, quali eventuali impianti fotovoltaici, sistemi di controllo di illuminazione pubblica e sistemi di videosorveglianza.

L'adattatore è una sorta di "retrofit digitale" che adatta le macchine esisenti in modo da renderle integrate nelle applicazioni Internet of Things. L'intero sistema può essere configurato e messo in servizio tramite il web server integrato nell'interfaccia IoT.
www.rittal.com

› L'intelligenza artificiale "in campo"

La Lega Calcio di Serie A ha recentemente avviato una partnership con Math&Sport, startup milanese nata in seno al Politecnico di Milano e gravitante nell'orbita del Polihub, finalizzata a sperimentare l'utilizzo dell'intelligenza artificiale nella stesura della tattica di gioco delle squadre. La collaborazione prevede l'utilizzo di Virtual coach, un tablet che permetterà ai tecnici delle 20 squadre della "massima divisione" di ricevere statistiche in tempo reale e suggerimenti sulla partita in corso, in modo da adattare la propria tattica al mutare dello scenario (per esempio un imprevisto come un infortunio o l'espulsione di un calciatore) seguendo i consigli di un raffinato algoritmo. L'intelligenza artificiale, unita al genio e all'abilità degli allenatori, permetterà di alzare il livello della competizione, rendendo le partite del campionato sempre più avvincenti e spettacolari

Nel corso del girone di ritorno della Serie A TIM 2019/2020, la Lega Serie A doterà tutte le panchine di un tablet con installato il "Football Virtual Coach" sviluppato da Math&Sport, che oltre a fornire dati in real time li analizzerà e sarà in grado di restituire, grazie all'utilizzo dell'Intelligenza Artificiale, indicazioni tecniche precise per gli allenatori e per tutto lo staff. Il Virtual Coach, infatti, sarà in grado di supportare le decisioni

› Lo switch ethernet sale in auto

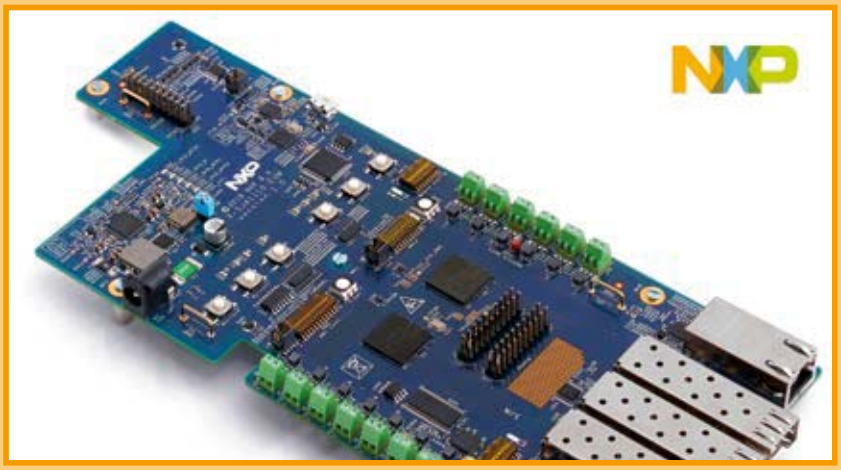
Perfettamente in linea con la recente tendenza a interconnettere le unità elettroniche di bordo delle automobili mediante reti ethernet o da esse derivate come il 100BASE-T1, NXP ha immesso nel mercato uno hub ethernet multigigabit siglato NXP SJA1110: si tratta del primo switch ethernet TSN dedicato all'automotive, disponibile in formato "naked" (ossia la sola scheda elettronica) che offre il supporto alla recente tecnologia 100BASE-T1. Ottimizzato per l'integrazione con il processore di rete per veicoli S32G di NXP, lo switch SJA1110 è anche parte di una soluzione di rete globale che include il circuito integrato di gestione dell'alimentazione VR5510.

La soluzione proposta da NXP permette ai costruttori di automobili di affrontare le sfide imposte dalle nuove tecnologie, prima fra tutte la guida autonoma e l'integrazione con l'infrastruttura di terra nell'ottica del V2V e non solo. In questi ambiti sono necessari nuovi e sofisticati Gateway Service-Oriented e controllori di dominio per ospitare i servizi correlati, come gli aggiornamenti Over-The-Air e le applicazioni basate su grandi moli di dati come la riproduzione di audiovisivi nei sistemi di infotainment e l'acquisizione ed elaborazionbe delle immagini delle telecamere necessarie alla guida autonoma. Le reti di bordo devono essere scalabili e trasferire dati in modo rapido (con minima latenza) e sicuro.

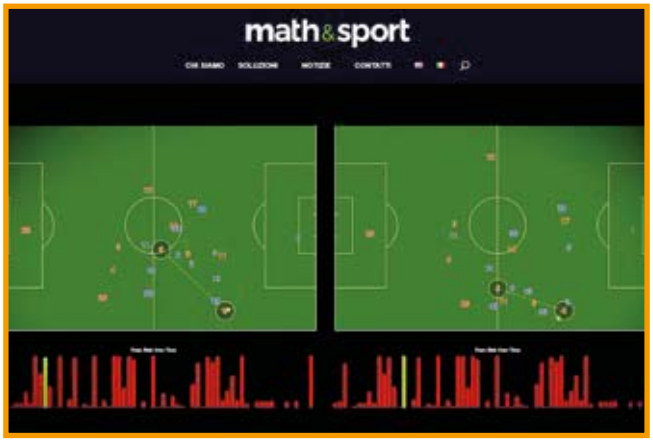
Lo switch Ethernet SJA1110 fornisce scalabilità, sicurezza e prestazioni con hardware adeguato e un numero equilibrato di porte per ECU ad alto e basso numero di porte. L'SJA1110 è allineato ai più recenti standard TSN e offre 100BASE-T1 PHY integrati, funzionalità di sicurezza e protezione assistita da hardware insieme a interfacce multi-gigabit.

Il SJA1110 consente di soddisfare i requisiti ASIL (Automotive Safety Integrity Level) e di elevare di conseguenza la capacità di garantire la sicurezza del veicolo, attraverso, ad esempio, la rilevazione dei guasti nell'ottica della manutenzione predittiva. D'altra parte lo switch beneficia dell'esperienza di NXP in fatto di protezione dalle intrusioni, maturata nei chip per carte bancarie ed e-passport. Il SJA1110 elabora ogni frame ethernet che raggiunge la ECU convalidandolo in base a regole di sicurezza usate anche nei più efficaci firewall.

www.nxp.com



dei tecnici mettendo in evidenza le situazioni critiche e quelle positive della propria squadra e degli avversari. Le informazioni, che arriveranno agli staff tecnici attraverso notifiche push, saranno personalizzabili in base al proprio sistema di gioco e alle strategie da adottare.
Virtual coach, già pronto per funzionare con la tecnologia del 5G, attraverso tecniche di modellazione matematica, algoritmi di machine learning e sistemi di calcolo, riesce a trovare automaticamente, in tempo reale, le relazioni causa-effetto tra le performance di un atleta, un reparto, una squadra, e gli eventi che le hanno generate.
Sarà davvero interessante vedere come, con l'Intelligenza Artificiale, cambierà il modo di giocare!
https://www.mathandsport.com/it/home-2/



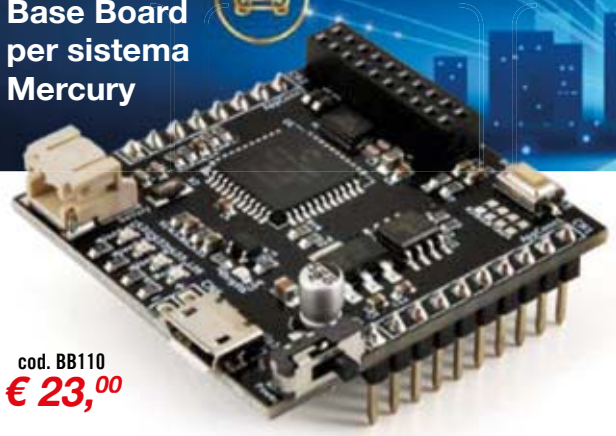
› AI per combattere le microplastiche

Quello delle microplastiche disperse in mare è uno dei problemi che più preoccupano, soprattutto per il rischio di ingerirle mangiando pesce che le ha mangiate e ritenute nel corpo. Per rintracciarle è stato messo a punto un sensore olografico che, abbinato all'intelligenza artificiale, consente di rilevare automaticamente la presenza di microplastiche nell'acqua marina, distinguendole dal microplancton.
Tutto questo è stato realizzato nell'ambito di un lavoro congiunto dell'Istituto di Scienze applicate e sistemi intelligenti del Consiglio nazionale delle ricerche (Cnr-Isasi) e del gruppo di Olografia digitale di Pozzuoli in collaborazione con il gruppo di Intelligenza artificiale di Lecce.

L'inquinamento dei mari dovuto alla plastica è una delle maggiori emergenze ambientali del momento, proprio perché attraverso la fauna marina possono entrare nella catena alimentare e intossicare gli umani. La gravità del problema è accresciuta dalla difficoltà, causata dalle ridotte dimensioni delle particelle inquinanti e dall'eterogeneità dei campioni marini, di identificare le microplastiche attraverso uno screening automatizzato ed accurato sull'acqua di mari e oceani. La combinazione dell'olografia digitale e dell'intelligenza artificiale ha superato tale ostacolo, permettendo di riconoscere decine di migliaia di oggetti appartenenti a diverse classi, con un'accuratezza superiore al 99%: praticamente con la certezza di distinguere le microplastiche dalle alre particelle e dal microplancton. Il nuovo metodo di olografia digitale fornisce un riconoscimento oggettivo di un numero statisticamente rilevante di campioni, fino a centinaia di migliaia di oggetti l'ora, con microscopi realizzabili in configurazioni portatili per analisi in luogo della qualità delle acque.
www.cnr.it

Rendi semplice IoT e connettività con le board MERCURY!

Base Board per sistema Mercury



cod. BB110
€ 23,00

Mercury System è un sistema di sviluppo modulare, hardware/software, specificamente progettato per permettere lo sviluppo di applicazioni IoT e in generale orientate alla connettività. Il sistema è composto dalla Base Board (cod. BB110) che è il "cervello" di tutto il sistema e contiene l'unità logica principale (un microcontrollore) oltre ai vari bus di comunicazione ed interfacce e da un set eterogeneo di schede slave (SB) e schede modem (MB) con le quali interagisce.

LE SCHEDE DEL SISTEMA MERCURY

SCHEDA SLAVE PER SENSORI E ATTUATORI

cod. SB810
€ 13,00

SCHEDA A ULTRASUONI INTERFACCIABILE

cod. SB310
€ 19,00

SCHEDA CON 2 RELÈ

cod. SB110
€ 17,00

SCHEDA MODEM BLUETOOTH

cod. MB310
€ 14,00

SCHEDA DHT22 CON SENSORE DI TEMPERATURA E UMIDITÀ

cod. SB330
€ 20,00

SCHEDA HSD (HIGH SIDE DRIVER) A 4 CANALI

cod. SB140
€ 25,00

SCHEDA A INFRAROSSI 2 CANALI

cod. SB320
€ 12,00

SCHEDA PER STRIP A LED O RING NEOPIXEL

cod. SB120
€ 14,00

SCHEDA EXPANSION DISPLAY 16X2

cod. EB210
€ 23,00

SCHEDA PER CONNESSIONE PLANARE DI 2 SCHEDE

cod. EB110
€ 7,00

SCHEDA PER CONNESSIONE DI 4 SCHEDE

cod. EB111
€ 12,00

SCHEDA CON CONNETTIVITÀ WiFi

cod. MB210
€ 12,00

SCHEDA CON SERVO A 6 CANALI

cod. SB130
€ 16,00

SCHEDA PER GESTIONE ALIMENTAZIONE

cod. PB110
€ 17,00

SCHEDA CON RELÈ PER CARICHI FINO A 10A

cod. SB111
€ 17,00

Prezzi IVA inclusa.

Regolatori µModule DC/DC a quattro uscite da ADI

Analog Devices presenta LTM4668 e LTM4668A, regolatori µModule DC/DC a quattro uscite con corrente di uscita totale fino a 4,8A. I nuovi dispositivi, dotati di PWM controller, FET di potenza, induttori e componenti di supporto, facilitano il processo di progettazione e riducono il consumo energetico e l'ingombro sulla scheda. Sono la soluzione ideale per applicazioni industriali, di telecomunicazioni e networking.

LTM4668 e LTM4668A operano in un intervallo di tensioni d'ingresso comprese tra 2,7V e 17V regolando tensioni di uscita comprese tra 0,6V e 5,5V. I dispositivi supportano il sincronismo in frequenza, il funzionamento PolyPhase, Burst Mode selezionabile, un duty cycle al 100% e bassa corrente di riposo (IQ). L'elevata frequenza di switching e l'architettura in current mode consentono una risposta al transitorio molto rapida alle variazioni di linea e carico, senza pregiudicarne la stabilità.

LTM4668 e LTM4668A sono disponibili in package BGA da 6,25mm x 6,25mm x 2,1mm.

www.analog.com

AS7350, sensore di luce ambiente spettrale ALS da ams

Ams lancia l'AS7350, il sensore di luce ambientale spettrale (ALS) più avanzato per telefoni cellulari di fascia alta. Il nuovo sensore consente immagini di alta qualità anche in situazioni con un contrasto cromatico estremo o in condizioni di sorgente luminosa non ideali / miste, consentendo fotografie di tipo professionale con un dispositivo di consumo.

Per la prima volta, l'AS7350 raggiunge questo obiettivo consentendo l'identificazione della sorgente luminosa attraverso la ricostruzione spettrale per ottenere un preciso bilanciamento automatico del bianco, in qualsiasi condizione di luce.

Identificando accuratamente le condizioni di luce ambientale, l'AS7350 consente una qualità dell'immagine senza pari anche in scene con elevato contrasto cromatico e aiuta i fornitori a continuare il percorso per offrire una qualità di livello professionale nei dispositivi mobili.

www.ams.com

Anritsu presenta la prima famiglia VNA 1-port a 43,5 GHz

Anritsu Corporation presenta il VNA (vector network analyzer) USB ShockLine MS46131A, il primo VNA modulare 1-port disponibile in grado di supportare la misura fino a 43,5 GHz. Con i modelli da 8 GHz e 20 GHz disponibili anch'essi nella serie, l'MS46131A offre vantaggi sia in termini di efficienza che di bassi costi nella misura di antenne e altri dispositivi 1-port per il 5G sub-6 GHz, nonché nelle bande millimetriche (mmWave) dei 28 GHz e 39 GHz.

Molto leggero e compatto, l'MS46131A può essere collegato direttamente al DUT (device under test), eliminando la necessità di cavi di interconnessione. Il risultato è una riduzione dei costi di prova e una migliore stabilità di misura. L'MS46131A è un VNA modulare che può essere configurato per ogni sessione utente, sulla base di ogni porta. Un singolo PC può controllare fino a due strumenti 1-port per un pratico test a doppio sito.

www.anritsu.com

Portenta H7, una board Arduino per il mondo industriale

In occasione del CES2020, Arduino ha annunciato la nuova potente famiglia a basso consumo Arduino Portenta. Progettato per applicazioni industriali impegnative, elaborazione edge per AI e robotica, presenta un nuovo standard per l'interconnessione ad alta densità in grado di supportare periferiche avanzate. Il primo membro della famiglia è il modulo Arduino Portenta H7: utilizza due core Cortex-M7 e Cortex-M4 che funzionano rispettivamente a 480 MHz e 240 MHz e componenti con temperatura di funzionamento di grado industriale (da -40° a + 85° C). Il modulo Portenta H7 è in grado di eseguire codice Arduino, Python e Javascript, rendendolo accessibile a un pubblico ancora più ampio di sviluppatori.

I tempi di distribuzione vengono ulteriormente accelerati grazie all'uso di Altium Designer e Altium 365 Cloud Platform per la progettazione hardware. Le PMI e i professionisti del design che utilizzano Altium Designer possono ora sfruttare una gamma di risorse di progettazione di riferimento Arduino, da simboli e impronte di componenti convalidati a modelli ed esempi schematici e di layout, rendendo più veloce e più semplice che mai la creazione di progetti hardware personalizzati che integrano l'hardware modulare Arduino.

www.arduino.cc

IC Real-Time Clock per automotive con bassa corrente di standby

Diodes Incorporated ha annunciato che le versioni compatibili con il settore automobilistico delle sue popolari soluzioni RTC (orologio in tempo reale) a bassa potenza, PT7C4363BQ e PT7C4563BQ (con timer regolabile), sono ora disponibili. La loro ampia gamma di temperature li rende adatti per applicazioni automobilistiche, inclusi sistemi di infotainment, display per cruscotti e box telematici (T-Box).
PT7C4363BQ e PT7C4563BQ sono qualificati per il grado 1 AEC-Q100, che copre l'intervallo di temperatura da -40 °C a + 125 °C. Sono in grado di supportare PPAP e sono fabbricati in strutture certificate IATF16949. Un'ampia gamma di tensioni di funzionamento da 1,3 V a 5,5 V consente flessibilità nel design, mentre una corrente di backup di appena 400 nA a 3,0 V allunga la vita operativa.
www.diodes.com

OPTIREG, alimentatori con tecnologia flip-chip

Infineon Technologies ha avviato un processo di produzione dedicato ai package flip-chip, pienamente allineato con le elevate esigenze di qualità del mercato automobilistico. Il primo di questi prodotti è il regolatore di tensione lineare OPTIREG TLS715B0NAV50.
Con la tecnologia flip-chip, i circuiti integrati sono installati capovolti nel package. Con la parte riscaldata dell'IC rivolta verso la parte inferiore del package e più vicina al PCB, l'induttanza termica può migliorare di un fattore compreso tra 2 e 3. La maggiore densità di potenza consente un ingombro notevolmente inferiore rispetto alle tecnologie convenzionali.
L'impronta del nuovo regolatore di tensione lineare di Infineon (package TSNP-7-8, 2,0 mm x 2,0 mm) è inferiore di oltre il 60%

rispetto al prodotto di riferimento (package TSON-10, 3,3 mm x 3,3 mm) mentre la resistenza termica rimane la stessa. Ciò rende il nuovo dispositivo particolarmente adatto in applicazione con spazi limitati, come radar e telecamere. OPTIREG TLS715B0NAV50 fornisce 5 V con una corrente di uscita massima di 150 mA.
La tecnologia flip-chip viene utilizzata nei mercati consumer e industriale per diversi anni.
www.infineon.com

MAX16923: riduce la complessità dei display automotive

I progettisti di sistemi elettronici automotive, possono ora aumentare il numero di display per veicolo riducendo al contempo la complessità del design utilizzando l'IC di alimentazione per display a 4 uscite con timer watchdog MAX16923 di Maxim Integrated. Sostituendo quattro o cinque circuiti integrati discreti con un'unica soluzione di gestione dell'alimentazione, il MAX16923 riduce significativamente le dimensioni della soluzione e rende più facile per i progettisti automobilistici aumentare il numero di display da due a cinque per veicolo, e oltre. Il suo elevato livello di integrazione può ridurre una soluzione di alimentazione per autoveicoli da quattro o cinque circuiti integrati fino ad un singolo chip.
www.maximintegrated.com

MAC-SA5X, il nuovo orologio atomico al rubidio di Microchip

Per soddisfare la domanda di un orologio atomico di dimensioni ridotte, Microchip Technology ha annunciato un nuovo modello con le prestazioni più elevate del settore per dimensioni e potenza. Il nuovo dispositivo offre anche una gamma termica più ampia, miglioramenti delle prestazioni critiche e altri vantaggi rispetto alla tecnologia precedentemente disponibile.
L'orologio atomico in rubidio miniaturizzato MAC- SA5X di nuova generazione di Microchip produce un riferimento di tempo e frequenza stabile che mantiene un alto grado di sincronizzazione con un clock di riferimento, come un segnale derivato dal GNSS. La sua combinazione di basso tasso di deriva mensile, stabilità a breve termine e stabilità durante le variazioni di temperatura consente al dispositivo di mantenere precisi requisiti di frequenza e temporizzazione durante le interruzioni del GNSS.
www.microchip.com

Da NXP il processore i.MX 8M Plus con unità di elaborazione neurale

NXP Semiconductors ha ampliato il suo portafoglio Edge-Verse leader del settore con il processore applicativo i.MX 8M Plus – la prima famiglia i.MX a integrare un'unità di elaborazione neurale dedicata (NPU) per l'inferenza avanzata dell'apprendimento automatico a livello industriale e IoT (Internet-of-Things).
I.MX 8M Plus combina un NPU ad alte prestazioni che offre 2.3 TOP (Tera operazioni al secondo) con un sottosistema Arm-Cortex-A53 Quad-core che funziona fino a 2 GHz, un sottosistema indipendente in tempo reale con un Cortex-M7 a 800 MHz, un DSP audio a 800 MHz ad alte prestazioni per l'elaborazione vocale e del linguaggio naturale, i processori di segnale immagine (ISP) a doppia fotocamera e una GPU 3D per un attrattivo rendering grafico. Con la combinazione di core Cortex-A53 ad alte prestazioni e NPU, i dispositivi periferici saranno in grado di prendere decisioni intelligenti a livello locale imparando e inferendo input con un intervento umano scarso o nullo.La gamma di applicazioni rese possibili con l'i.MX 8M Plus comprende riconoscimento di oggetti e persone per la sicurezza pubblica, la visione industriale delle macchine, la robotica, il riconoscimento dei gesti, ecc.

www.nxp.com

Da ST il primo SoC wireless al mondo che integra un ricetrasmettitore LoRa

STMicroelectronics ha presentato il dispositivo STM32WLE5, il primo SoC LoRa al mondo per la connessione wireless a lunga distanza di dispositivi Internet of Things (IoT). Il SoC STM32WLE5 consente agli sviluppatori di creare dispositivi come sensori ambientali remoti, misuratori, tracker e controller di processo che aiutano a gestire in modo efficiente energia e risorse. Il SoC combina le elevate prestazioni dei microcontrollori STM32 a bassissima potenza con una radio conforme LoRa in un unico dispositivo semplice da usare. Con più brevetti ST in corso di registrazione, in particolare per l'architettura di gestione dell'alimentazione radio, STM32WLE5 garantirà prestazioni uniche. Il software LoRaWAN di ST per le comunicazioni di rete wireless ha superato tutte le certificazioni regionali per l'uso in tutto il mondo. STM32WLE5 è disponibile in un package UFBGA73 da 5 mm x 5 mm. È completamente integrato nell'ecosistema STM32, incluso il supporto software STM32Cube, nonché uno stack LoRaWAN certificato per tutte le regioni e disponibile in formato codice sorgente.

www.st.com.

Semtech rilascia il nuovo dispositivo LoRa Smart Home per applicazioni IoT

Semtech ha annunciato il lancio di LoRa Smart Home, un dispositivo progettato per applicazioni smart home, community e consumer basate su LPWAN. Il transceiver offre un basso consumo, e un ampio raggio di copertura per dispositivi IoT posti in interni o aree non immediatamente prossime, e che si collegano a sensori o attuatori per casi d'uso in ambito sicurezza, ambientale e convenienza.
Il nuovo transceiver è pensato per sensori alimentati a batteria, con funzionamento pluriennale. Vanta una corrente in modalità Sleep di soli 600 nA e 4,6 mA in ricezione attiva. Con supporto per la modulazione LoRa nei casi d'uso LAN a basso consumo e per la modulazione (G)FSK per casi d'uso preesistenti, questo dispositivo è compatibile con le reti esistenti basate su LoRaWAN e supporta protocolli proprietari. La copertura in frequenza continua da 150 MHz a 960 MHz consente il supporto di tutte le principali bande ISM sub-GHz in tutto il mondo.

www.semtech.com

Fotoaccoppiatore TLP2363 con uscita logica da 10Mbps

Toshiba ha lanciato un nuovo fotoaccoppiatore con uscita logica da 10Mbps ad alta velocità per ricevitori PLC con ingresso digitale a 24V e altri apparecchi di misura e di controllo. Il nuovo TLP2363 richiede una corrente di alimentazione di appena 4mA ed è in grado di operare in un ampio intervallo di temperature di esercizio (da -40°C a 105°C). La corrente di ingresso di soglia (IFHL) è specificata con un valore minimo di 0,3mA e massimo di 2,4mA, e garantisce così la conformità allo standard IEC 61131-2 Tipo 1 per gli ingressi digitali. Una caratteristica unica per questo tipo di dispositivi è che tutto ciò che è necessario per completare il progetto è un ponte di diodi e un resistore adatto per il controllo in corrente. Il breve ritardo di propagazione, pari ad appena 80ns, è ideale per impieghi in sistemi ad alta velocità.
Il fotoaccoppiatore offre un chiaro livello 'alto' o 'basso' sull'uscita, sopprimendo il rumore di interferenza, anche con un segnale di ingresso ad escursione lenta di 60s o meno, con fronti di salita o di discesa lenti. Di conseguenza, non è necessario alcun circuito di generazione di forme d'onda, e ciò riduce il numero di componenti con un risparmio di area su PCB e di costi. Nonostante sia alloggiato in un minuscolo e sottile package SO6 a 5 pin con un'altezza massima di 2,3 mm, il dispositivo offre una tensione di isolamento minima di 3.750Vrms.

toshiba.semicon-storage.com

ELECTRONIC DAY

MERCATO DEL CONSUMER ELECTRONICS

4-5 APRILE 2020

ORARI:
sab. 9.30 - 18.00
dom. 9.30 - 17.00

IN CONCOMITANZA CON
HOBBY MODEL EXPO
SPRING EDITION

PARCO ESPOSIZIONI NOVEGRO
MILANO / LINATE AEROPORTO ✈

con il patrocinio di Città di Segrate

www.parcoesposizioninovegro.it

› **02-03 APRILE 2020**

DRONITALY FORUM 2020

MILANO

Dronitaly Forum 2020 in programma il 2 e 3 Aprile presso il Politecnico di Milano, Campus Bovisa, è l'evento che chiama a raccolta la community italiana del comparto *unmanned* per affrontare e stimolare il dibattito attorno ai temi più attuali e urgenti del settore. Nell'attesa di un quadro normativo uniforme a livello europeo che tracci con evidenza le linee entro cui potranno muoversi gli utilizzatori professionali di SAPR, il Forum propone un programma di convegni e presentazioni altamente qualificato e di respiro internazionale, in cui saranno anche gli utilizzatori finali (aziende e PA) a raccontare la propria esperienza diretta fornendo una prospettiva che parte da esempi concreti, per arrivare all'analisi degli scenari di mercato più recenti e all'individuazione di nuovi orizzonti di crescita. Negli anni Mirumir, società che organizza l'evento, ha sempre lavorato insieme alle Università per definire i percorsi tematici più rispondenti alle esigenze della filiera e fornire contributi per orientare l'attività di informazione, formazione e promozione di Dronitaly a favore dello sviluppo del comparto. Il Comitato Scientifico nasce – su iniziativa di Mirumir – per proseguire nell'attività di ricerca e promozione del mercato *unmanned*, in continua evoluzione. I membri del Comitato Scientifico sono: Marco Lovera (Politecnico di Milano), Francesco Guerra (IUAV Venezia), Marcello Chiaberge e Marco Piras (Politecnico di Torino). Il 2020 sarà l'anno dell'entrata in vigore del regolamento EASA e dell'auspicata armonizzazione normativa e operativa del comparto *unmanned* a livello internazionale. Nell'attesa di un quadro uniforme che tracci con evidenza le linee entro cui potranno muoversi gli utilizzatori professionali di SAPR, il Forum propone una serie di incontri altamente qualificati e di respiro internazionale in grado di fornire nuovi orizzonti di crescita, muovendo da un'analisi degli scenari di mercato più recenti.

Scenari normativi, sviluppo del business, tecnologie abilitanti per utilizzi professionali, uso dei mezzi a pilotaggio remoto in scenari industriali e ambientali, cybersecurity e data protection sono alcuni dei grandi temi su cui grandi aziende, enti pubblici, centri di ricerca e investimento, associazioni e start-up saranno chiamati a contribuire offrendo spunti di riflessione e presentando esperienze reali.

Appuntamento il 2-3 Aprile 2020 presso l'Aula Magna Carassa e Dadda, Edificio BL28, Politecnico di Milano, Campus Bovisa, Via Raffaele Lambruschini 4, Milano.

www.mirumir.it

› **01-02 APRILE 2020**

IOTHINGSMILAN

MILANO

IOTHINGS è l'evento italiano più importante nell'ambito delle tecnologie IoT con due appuntamenti annuali: in primavera a Milano e in autunno a Roma. IOTHINGS Milan è parte di INNOVABILITY HUB.

Dal 1 al 3 aprile 2020 si svolgeranno in contemporanea al MIND – Milano Innovation District le nuove edizioni di IOTHINGS, ITALIA5G, EMBEDDEDIOT e IOENERGY. Tre le novità per il 2020: CODER 4.0, FUTURE BUILDING TECH e EDGE FORUM.

Sette eventi riuniti per creare INNOVABILITY HUB: un esclusivo punto di aggiornamento sulle tecnologie più "disruptive" e per favorire la trasformazione digitale delle aziende.

MILAN IOTHINGS 1-2 APRILE 2020

INNOVABILITY HUB si svolgerà nel nuovo MIND – MILANO INNOVATION DISTRICT, realizzato nell'area che ha ospitato Expo 2015 e che si sta trasformando in un distretto della scienza, del sapere e dell'innovazione di eccellenza. MIND diverrà un nuovo quartiere di Milano che includerà il nuovo Ospedale IRCCS Galeazzi, il Campus delle facoltà scientifiche dell'Università Statale di Milano e Human Technopole, un istituto di ricerca internazionale per le eccellenze nei campi Life Sciences, Healthcare, Biotech, Pharma, Agrifood, Nutrition e Data Science. L'evento si articola in una serie di conferenze e seminari e in un'area espositiva dove i visitatori potranno toccare con mano e vedere in funzione le soluzioni offerte dai principali player del settore.

IOTHINGSMILAN si terrà i giorni 1 e 2 Aprile presso Studio 2 – MIND, Milano Innovation District.

www.iothingsmilan.com

MIR 2020, ANCORA PIÙ TECH

L'unico evento italiano dedicato alle tecnologie audio, video, luci per concerti, spettacoli, eventi e location che permette di provare dal vivo la tecnologia delle aziende leader si svolgerà dall'8 al 10 marzo 2020 a Rimini.
Palcoscenici per sperimentare, installazioni all'avanguardia, esperienze raccontate dai brand più innovativi, masterclass con i grandi nomi internazionali dello show business e con i creativi più visionari.
Tre giorni, mille opportunità: innovazioni tecnologiche, eventi, ispirazione e training, l'unica fiera di riferimento per l'intero settore dello Showbiz, che ha saputo costruire un format originale ed esclusivo per raccontare ai professionisti di questo mondo quali sono le migliori tecnologie e i campi di applicazione in tema Light, Sound, Visual, Integrated System & Broadcast.
Una delle novità di questa nuova edizione è lo spostamento delle date. Infatti, la manifestazione di Italian Exhibition Group apre le porte in anticipo rispetto agli anni precedenti e si svolgerà dall'8 al 10 marzo 2020 alla Fiera di Rimini. "*Dopo 4 anni, consideriamo chiusa la prima fase di startup di sviluppo del MIR*", sottolinea **Marco Borroni**, Group Brand Manager di IEG, "*la manifestazione è nata e cresciuta nel mese di maggio, rispetto alla prima edizione abbiamo avuto un'evoluzione incredibile. Abbiamo superato il 60%, sia dei mq occupati, che di aziende partecipanti*". Dunque MIR si è definitivamente affermata. Non è più la "nuova" fiera, ma "la" fiera TECH di riferimento. Un progetto che, secondo gli organizzatori, nei prossimi 3-4 anni ha già in serbo di sviluppare diversi settori collaterali orientandosi anche al mondo della fotografia, dell'home entertainment e degli eventi, impiantistica e tecnologie per lo sport con progetti più verticali.
Anche per questa edizione non mancherà LIVE YOU PLAY, uno dei contenuti più innovativi, da sempre presente al MIR. Continua ad essere l'evento LIVE più atteso dai professionisti che hanno modo di entrare in contatto con le più importanti aziende del settore tech. Grazie a una formula peculiare, si genera un vero e proprio coinvolgimento emozionale dei visitatori che possono vedere e provare le tecnologie audio, luci, video e le strutture installate su un palco, con gli artisti che si esibiscono dal vivo. Quest'anno verranno dedicati 4 padiglioni da 6000 mq l'uno, con vari palchi riservati al teatro e alla musica e un nuovo padiglione interamente destinato alle migliori tecnologie broadcasting.
MIR 2020 vanterà la partecipazione straordinaria di Marco Balich, Chairman di Balich Worldwide Shows, Direttore e Executive Producer di Cerimonie Olimpiche e dei più importanti live show a livello mondiale, che per l'occasione salirà sul palco dell'evento IEG per presentare Future is Illuminated, talk show unico nel suo genere sul mondo delle luci, capace di far rivivere al pubblico presente le emozioni che hanno segnato la storia dei grandi eventi.
Al MIR l'emozione passa anche attraverso la formazione, un punto di forza indiscusso dell'intera manifestazione. Il calendario dei corsi, workshop e seminari sarà come sempre molto ricco.

www.musicinsiderimini.it

› 26-28 MARZO 2020
ENERGYMED 2020
NAPOLI

Dopo il successo delle precedenti edizioni, EnergyMed, "*la Mostra Convegno sulle Fonti Rinnovabili e l'Efficienza Energetica*", giunge alla sua XIII edizione. Le fonti rinnovabili e l'efficienza energetica sono, infatti, sempre più al centro dei piani di azione per la sostenibilità ambientale per cui EnergyMed diventa il contesto ideale per confrontarsi sullo stato dell'arte di settori innovativi legati al solare, all'eolico, alle caldaie ad alta efficienza e a biomasse, al recupero di materia ed energia dai rifiuti, ai veicoli a basso impatto ambientale e ai servizi; attenta all'evoluzione del mercato, la manifestazione vede la consolidata presenza delle quattro sezioni dedicate all'efficienza energetica "EnerEfficiency", al riciclo "Recycle", alla mobilità sostenibile "Mobility" e all'automazione "Automation". Molte le novità di quest'anno, con un'esposizione che supera gli 8.000 metri quadrati, con un vasto programma congressuale e numerosi eventi speciali; infatti, i migliori esperti del settore e i rappresentanti delle principali imprese partecipano ai workshop tematici, agli incontri business to business e alle tavole rotonde. Gli organizzatori intendono, infatti, creare le condizioni ideali per l'aggiornamento professionale e per gli scambi commerciali tra aziende nazionali e straniere, favorendo un processo di internazionalizzazione sempre più proficuo in tale comparto produttivo.
La scelta della Campania e di Napoli rappresenta poi un elemento strategico conferendo alla manifestazione una connotazione "mediterranea" e un ruolo di cerniera tra il resto dell'Europa e i paesi dalle economie emergenti che si affacciano sul "Mare Nostrum".

www.energymed.it

› **26-28 MARZO 2020**

MECSPE 2020

PARMA

Dal 26 al 28 marzo andrà in scena a Parma la diciannovesima edizione della fiera MECSPE, con un nuovo punto di riferimento, uno spazio progettato oltre ogni confine, per valorizzare i giovani e lo sviluppo delle loro capacità, attraverso un'esperienza ancora più dinamica, immersiva e coinvolgente.

Come una fabbrica può attrarre le nuove generazioni?

La difficoltà di reperimento di giovani da inserire negli impianti produttivi italiani rappresenta un tema sempre più cruciale per le PMI del nostro Paese. La percentuale, infatti, degli imprenditori che riscontrano problematiche di questo tipo a gennaio 2020 passa dal 31% al 33% e il mismatch domanda-offerta riguarda da una parte una serie di profili di laureati (difficili da reperire il 39,3% dei candidati in possesso di un titolo universitario), ma anche profili tipici della formazione professionale (35,1% la difficoltà segnalata). Dati del bollettino mensile del Sistema informativo Excelsior, realizzato da Unioncamere e Anpal, confermati dalle ultime analisi condotte dall'Osservatorio MECSPE, secondo cui gli spazi della fabbrica rappresentano, nell'immaginario collettivo giovanile, un luogo contraddistinto da una profonda staticità e ripetitività lavorativa. Un pensiero comune che non tiene in considerazione la forte trasformazione digitale avvenuta negli ultimi anni e l'impiego degli strumenti di Industry 4.0 volti al posizionamento, mediante il dialogo con la tecnologia, della persona al centro della fabbrica.

Per smentire il falso stereotipo e avvicinare i giovani alla fabbrica, MECSPE, la fiera di riferimento per l'industria manifatturiera promossa da Senaf, che si terrà a Parma dal 26 al 28 marzo 2020 prima del suo spostamento a Bologna nel 2021, mette in mostra un'importante novità, che supera ogni confine tecnologico: una "Fabbrica senza limiti", cuore pulsante di 2.000 mq nel Pad. 4 ideato in collaborazione con il CFI – Cluster Fabbrica Intelligente, per incentivare lo scambio e l'interazione di conoscenze fra aziende e giovani, e fornire una reale visione della conformazione degli stabilimenti 4.0.

L'impegno di MECSPE non finisce qui. Infatti, mediante il premio "*Innovazione robotica e automazione*", rivolto ad aziende, startup, università e centri di ricerca che si sono distinti per la creazione di applicazioni caratterizzate da un alto tasso di innovazione e d'impatto, si darà un riconoscimento concreto per la migliore applicazione innovativa nel ramo della robotica e dell'automazione. Altro importante momento celebrativo previsto, il "Premio di Laurea TMP", dell'associazione italiana dei Tecnici delle Materie Plastiche che, in collaborazione con la rivista Plastix, sarà assegnato alle migliori tesi di laurea di giovani studenti, che avranno esaminato e analizzato il settore della plastica dal punto di vista della sostenibilità, del riciclo e della progettazione. Infine, per premiare le aziende espositrici che si sono distinte nell'ambito delle soluzioni 4.0 adottate con successo, la rivista Industrie 4.0 riproporrà il premio "*Award 4.0*".

I numeri di MECSPE PARMA 2019

L'edizione 2019 è stata caratterizzata da numeri in crescita in tutti gli ambiti: 135.000 mq di superficie espositiva, 56.498 presenze professionali, 2.306 aziende presenti, 2.000 mq di Tunnel dell'Innovazione in collaborazione con il Cluster Fabbrica Intelligente, 67 iniziative speciali e convegni. A partire dal 2021, MECSPE proseguirà il proprio sviluppo internazionale trasferendosi nel quartiere fieristico di BolognaFiere.

www.mecspe.com/it

MOSTRE MERCATO

BASTIA UMBRA (PG)
EXPO ELETTRONICA BASTIA UMBRA
Umbria Fiere - Bastia Umbra (PG)
Organizzazione: Blu Nautilus Srl
Telefono: 0541439573
www.expoelettronica.it
info@expoelettronica.it
04 e 05 Aprile 2020

NOVEGRO (MI)
ELECTRONIC DAY + HOBBY MODEL EXPO
Parco Esposizioni di Novegro (MI)
Organizzazione: Comis Srl
Telefono: 027562711
www.parcoesposizioninovegro.it
radiant@parcoesposizioninovegro.it
04 e 05 Aprile 2020

PESCARA
FIERA MERCATO DELL'ELETTRONICA DI PESCARA
Centro Fiere Ibisco - Via Lungofino 187 Città Sant'Angelo - PE
Organizzazione: ARI PESCARA
Telefono: 0854215840
www.aripescara.org
arifiera@aripescara.org
18 e 19 Aprile 2020

PORDENONE
RADIOAMATORE HI-FI CAR
Quartiere Espositivo - Pordenone
Organizzazione: Pordenone Fiere Spa
Telefono: 0434232111
www.fierapordenone.it
info@fierapordenone.it
25 e 26 Aprile 2020

L'elenco aggiornato di tutte le Mostre Mercato del 2020 è disponibile sul sito *www.elettronicain.it* sul quale è anche possibile scrivere un commento sulle fiere visitate.

L'elenco completo dei più importanti eventi nazionali e internazionali di elettronica, sicurezza e fonti rinnovabili è disponibile sul nostro sito: http://www.elettronicain.it/myMain/Fiere/p_fiere.asp

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.elettronicain.it.*

→ e-mail
scrivendo a
abbonati@elettronicain.it

→ telefono
telefonando a
+39-0331-752668

ABBONAMENTO ANNUALE PER RIVISTA **CARTACEA**

10
NUMERI
A SOLI
€45
ANZICHÉ € 60,00

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

La tua casa sotto una nuova luce!

Rinnova e migliora l'illuminazione della tua abitazione.

Set 3 lampade LED bianco naturale 220 VAC - attacco E27 - 11W

cod. HT6011
€ 6,90

Confezione contenente 3 lampade LED con emissione luminosa bianco naturale, angolo di emissione >270°, temperatura colore 4000 K, attacco E27, basso consumo energetico e accensione istantanea. Alimentazione: 175-250 VAC, dimensioni: Ø60x109 mm.

Set 3 lampade LED bianco naturale 220 VAC - attacco E14 - 5W

cod. HT4505
€ 4,90

Confezione contenente 3 lampade LED con emissione luminosa bianco naturale, angolo di emissione >270°, temperatura colore 4000 K, attacco E14, basso consumo energetico e accensione istantanea. Alimentazione: 175-250 VAC, dimensioni: Ø45x81 mm.

Set 3 lampade LED candela bianco naturale 220 VAC attacco E14 - 5W

cod. HT3705
€ 4,90

Confezione contenente 3 lampade LED a candela con emissione luminosa bianco naturale, angolo di emissione >270°, temperatura colore 4000 K, attacco E14, basso consumo energetico e accensione istantanea. Alimentazione: 175-250 VAC, dimensioni: Ø45x81 mm.

Lampada tubolare con LED bianco a luce neutra - 230VAC - E27

cod. APT37N
€ 4,90

Lampada tubolare a LED con luce neutra con potenza di 9W, temperatura di colore 4000K, flusso luminoso 800lm.

Lampada LED con crepuscolare attacco E27 - 10W

SENSORE CREPUSCOLARE INTERNO

800 LUMEN

Lampada a basso consumo energetico. Accensione istantanea, alimentazione: 230 Vac, luce bianca neutra, dimensioni: Ø60x115 mm, attacco E27.

cod. LAMPWSENSOR
€ 7,50

Lampada a LED solare con batteria interna, sensore crepuscolare e di movimento

Lampada a LED 4W da esterno con grado di protezione IP44 Non necessità di collegamenti alla rete elettrica per funzionare in quanto dotato di batteria interna e pannello solare per la ricarica. Dispone di sensore crepuscolare e sensore PIR di movimento. Portata del sensore di movimento (PIR) 3-5 metri. Dimensioni 135x100x50 mm, peso 180 grammi.

cod. SOL01
€ 14,90

Faro a LED da esterno (IP65) con sensore a microonde e telecomando

Composto da 30 LED SMD ad alta luminosità con luce bianca naturale. Dispone di sensore di movimento a microonde e di telecomando per le impostazioni e il controllo a distanza. Può essere acceso/spento manualmente tramite il tasto ON/OFF, può essere impostato per accendersi automaticamente al passaggio di qualcuno. Flusso luminoso: 1400 lumen, temperatura colore 4000 K, angolo di emissione 120°, alimentazione 230 VAC.

cod. LEDA7002NW-BM
€ 29,90

Faro a LED da esterno (IP65) Bianco neutro 50W - 4500 Lumen

Faretto da esterno leggero e elegante, rifinito con vernice epossidica di colore nero. Lampada led a luce bianco neutro (4000K) con una potenza totale di 50 watt e un flusso luminoso di 4500 LUMEN ad elevata efficienza energetica. Accensione istantanea e senza sfarfallamento, resistente all'acqua e alle alte temperature. Completo di staffa di montaggio regolabile.

4500 LUMEN

cod. FB50N
€ 34,00

DOMOTICA PER TUTTI

Concludiamo la nostra rassegna sulla piattaforma per lo sviluppo di applicazioni domotiche CM3-Home, con un approfondimento sulle "rule". Settima e ultima puntata.

di **GUIDO OTTAVIANI**

I dati, grezzi o aggregati, possono essere semplicemente visualizzati su un *HABpanel* o venire inquadrati in una o più regole (*rule*) allo scopo di creare automatismi per comandare un attuatore e, in generale, ottenere un'azione come accendere una luce, alzare una tapparella, avvolgere una veranda, comandare il riscaldamento, eccetera. Nella scorsa puntata di questa serie dedicata alla scheda CM3-Home abbiamo proposto alcuni esempi molto semplici di rule che, allo scatenarsi di un certo evento, eseguono un piccolo script per effettuare, ad esempio, delle conversioni da un'unità ad un'altra, inviare comandi complessi ad attuatori locali o remoti oppure aggiornare delle informazioni sul display. La maggior parte delle volte l'evento coincide con il variare di uno o più *item* ma può essere anche generato a tempo o su variabili ambientali del sistema operativo.

Il linguaggio di programmazione è derivato da Java, quindi si possono seguire le linee guida di questo linguaggio, sempre tenendo conto della gestione ad eventi dei diversi task concorrenti.
Il punto comune tra tutte le componenti di OpenHAB sono, come sarà ormai chiaro, gli *item*. Possiamo considerarli come le variabili globali del nostro ambiente di sviluppo. Il punto sul quale confluiscono tutte le operazioni che abbiamo eseguito in precedenza configurando *binding, thing, channel* e che riportano in un linguaggio unico dati eterogenei di sensori ed attuatori basati su tecnologie anche diversissime tra loro. Gli *item* sono anche il punto di collegamento tra la parte server (le *rule* eseguite su OpenHAB nella CM3-Home) e i client (le GUI che girano sull'app mobile oppure gli *HABpanel* basati su *widget* standard o personalizzati). In quest'ultimo articolo della serie descriveremo applicazioni che esplicano al meglio le potenzialità di OpenHAB: integrare tra loro le tecnologie più disparate per ottenere un cruscotto di controllo molto potente ma facile da usare. Oltre ad utilizzare diverse componenti hardware, disegneremo delle GUI basate sullo stato dell'arte del software di presentazione, come HTML, CSS, javascript, AngularJS, Ajax, eccetera.

SVILUPPO AVANZATO DI *RULE*

Negli articoli precedenti abbiamo visto come leggere informazioni da sensori collegati direttamente o via rete alla CM3-Home, oppure da servizi esterni in Cloud, ed anche come presentarli in un HABpanel in maniera aggregata e trasparente all'utente. Ora spiegheremo come, utilizzando gli stessi dati ma elaborandoli tramite *rule* avanzate, possiamo ottenere informazioni sull'ambiente.

Fig. 1 Esempio di HABpanel meteo avanzato.

METEO

Le previsioni meteorologiche hanno raggiunto dei livelli di precisione veramente alti, grazie alla raccolta di enormi quantità di informazioni da sensori sparsi in tutto il mondo (e fuori da esso) e all'elaborazione con i computer più potenti tramite algoritmi predittivi sempre più elaborati. In Rete ci sono diversi servizi che espongono delle API per utilizzare i risultati di questi calcoli anche nei nostri dispositivi; però quelle così ottenute rimangono sempre previsioni statistiche a medio e breve termine su un territorio esteso. Dal momento che abbiamo a disposizione dei sensori locali che analizzano l'ambiente che ci circonda, possiamo elaborare anche noi delle previsioni molto localizzate e a brevissimo temine che, anche se con algoritmi meno sofisticati, potrebbero esserci più utili per sapere se dobbiamo prendere l'ombrello, se è inutile innaffiare il giardino o se possiamo organizzarci per il pic-nic fuori porta o, magari, per comandare in autonomia tapparelle e verande in caso di pioggia imminente se siamo fuori casa. Potrebbero essere anche più utili delle previsioni nazionali o regionali che debbono essere, per forza di cose, più generiche.
Un algoritmo molto diffuso da anni, utilizzato anche su stazioni meteo piccole e medie, e che ha bisogno solo di poche informazioni, è la versione digitale del metodo Zambretti sperimentato da più di cento anni nella sua versione originaria. Vediamo come implementarlo in OpenHAB con i dati che già ci fornisce la nostra CM3-Home.
In uno dei primi articoli abbiamo visto come rappresentare informazioni meteorologiche su un HABpanel utilizzando un *widget* personalizzato scritto in HTML con gli stili definiti in un CSS esterno. I dati raccolti da sensori collegati alla CM3-Home o prelevati da siti di previsioni meteo on-line sono semplicemente riportati nella pagina HTML o associati a icone in modo automatico.
Ora vogliamo realizzare un pannello più elaborato (**Fig. 1**) nel quale alcuni dati sono semplicemente prelevati dai sensori, mentre altri sono elaborati secondo algoritmi molto usati in meteorologia, come il metodo Zambretti e la tabella Beaufort. Analizziamo le diverse tecnologie integrate in questo widget: temperatura, umidità, pressione barometrica e concentrazione di CO_2 sono rilevati tramite i sensori Netatmo, Sonoff e 1-wire che abbiamo già descritto in precedenza. La variazione di pressione è calcolata dai dati storici memorizzati in InfluxDB, anche questo già spiegato. Le previsioni per i giorni successivi sono ottenute da servizi

Listato 1

```
rule “wind”
when
  Item Wind_Speed received update or
  Item Wind_Direction received update or
  Item Wind_Direction_deg received update

then
    val beaufortText = newArrayList
    (
      ‘Calma’,
      ‘Bava di vento’,
      ‘Brezza leggera’,
      ‘Brezza tesa’,
      ‘Vento moderato’,
      ‘Vento teso’,
      ‘Vento fresco’,
      ‘Vento forte’,
      ‘Burrasca’,
      ‘Burrasca forte’,
      ‘Tempesta’,
      ‘Tempesta violenta’,
      ‘Uragano’
    )
  Wind.postUpdate(Wind_Speed.state.toString + “ km/h  “ + Wind_Direction.state.toString)
  var int beaufortValue = Math.round(Math.pow((Wind_Speed.state as DecimalType).floatValue /
4,0.707106781)).intValue
  Wind_Beaufort.postUpdate(beaufortText.get(beaufortValue).toString)
end
```

meteo on-line. La direzione e la velocità del vento possono essere misurate tramite, ad esempio, un anemometro collegato alla centralina Netatmo. La relativa *rule* è descritta nel **Listato 1**, dove i dati di velocità e direzione sono raggruppati in una stringa per poterli rappresentare su un'unica riga della GUI classica (**Fig. 2**). Viene poi calcolato l'indice Beaufort. Usando questo indice nella tabella *beaufortText* si può ottenere la descrizione in chiaro, secondo Beaufort, dell'intensità del vento. In **Fig. 1** si può vedere una situazione di calma di vento con una direzione di provenienza da nord-est. Per calcolare le previsioni con il metodo Zambretti occorre analizzare l'andamento della pressione, in mbar/h, nelle ultime tre ore, riferito alla pressione atmosferica a livello del mare. La centralina Netatmo fornisce il valore di pressione sia assoluta (misurata dal barometro) sia relativa; questa viene calcolata dalla centralina conoscendo l'altezza sul livello del mare della postazione, altezza rilevata a sua volta in fase di installazione tramite il GPS del telefono sul quale gira l'app di configurazione. Se si dispone solo della pressione assoluta è facile calcolare quella relativa conoscendo l'altezza s.l.m. ed applicando delle semplici formule reperibili in Rete. Secondo il NOAA (National Oceanic and Atmospheric Administration) la tendenza della pressione atmosferica è uguale alla differenza tra la pressione atmosferica attuale e quella di tre ore prima, divisa per 3 per ottenere l'andamento orario.
La *rule* è avviata a tempo ogni 10 minuti, lo stesso periodo con il quale sono aggiornati i dati della centralina meteo. I dati storici sono ripresi da InfluxDB, mediando due misure distanti 12 minuti tra loro per essere sicuri di catturare due diversi eventi. La pressione massima e quella minima sono calcolate ad ogni misura e conservate nel DB per avere una storia del luogo e tarare meglio nel tempo l'algoritmo Zambretti (**Listato 2**).
L'andamento della pressione calcolato si usa per assegnare il nome dell'icona che indicherà l'andamento nel *widget* e il flag che sarà usato successivamente per applicare il corretto algoritmo. Il valore dell'andamento di pressione minimo che deve essere usato per considerare la pressione stabile o meno non è molto chiaro, ci sono diverse interpretazioni. In questo esempio è stato usato il valore applicato nel codice reperibile all'indirizzo https://github.com/jim-easterbrook/pywws e sembra dare risultati affidabili.

Fig. 2 Esempio di rappresentazione dei parametri del vento su una GUI classica.

Listato 2

```
rule “Pressure Trend”
when
    Time   cron “0 0/10 * * * ?”
then
   var Number pressure0avg = ((Netatmo_Indoor_Pressure.historicState(now.minusMinutes(0)).state as
DecimalType)+(Netatmo_Indoor_Pressure.historicState(now.minusMinutes(-12)).state as DecimalType))/2
   var Number currentPress = Netatmo_Indoor_Pressure.historicState(now.minusMinutes(0)).state as DecimalType
   var Number maxPress     = MaxPressure.historicState(now.minusMinutes(0)).state as DecimalType
   var Number minPress     = MinPressure.historicState(now.minusMinutes(0)).state as DecimalType

    if (currentPress > maxPress)
    {//store the MAX
        MaxPressure.postUpdate(currentPress)
    }

    if (currentPress < minPress)
    {
        //store the min
        MinPressure.postUpdate(currentPress)
    }

    var Number pressure180avg = ((Netatmo_Indoor_Pressure.historicState(now.minusMinutes(174)).state as
DecimalType)+(Netatmo_Indoor_Pressure.historicState(now.minusMinutes(186)).state as DecimalType))/2

    var Number pressureTrend  = (pressure0avg - pressure180avg) / 3.0
    PressureTrendValue.postUpdate(pressureTrend)

    if (pressureTrend > 1.2)
    {
        PressureTrendStatus.postUpdate(“QuickRise”)
        PressureTrendFlag.postUpdate(1)
    }
    else if (pressureTrend > 0.1)
    {
        PressureTrendStatus.postUpdate(“SlowRise”)
        PressureTrendFlag.postUpdate(1)
    }
    else if (pressureTrend < -1.2)
    {
        PressureTrendStatus.postUpdate(“QuickFall”)
        PressureTrendFlag.postUpdate(-1)
    }
    else if (pressureTrend < -0.1)
    {
        PressureTrendStatus.postUpdate(“SlowFall”)
        PressureTrendFlag.postUpdate(-1)
    }
    else
    {
        PressureTrendStatus.postUpdate(“Steady”)
        PressureTrendFlag.postUpdate(0)
    }
end
```

Abbiamo tutti i dati che ci servono, quindi adesso vediamo come usarli per ottenere gli ultimi due parametri da presentare nel widget: la descrizione delle previsioni e l'icona corrispondente. Per prima cosa definiamo le tabelle usate per le diverse conversioni (**Listato 3**); l'operazione è effettuata ogni 10 minuti e non ci sono altri trigger basati su item. Per definire degli array in OpenHAB occorre usare gli ArrayList, i quali non sono array tradizionali ma, piuttosto, una classe contenitore di Java (trovate i dettagli nei JavaDocs). Vedremo negli esempi come devono essere usati (**Listato 4**).
L'equivalente informatico del regolo Zambretti funziona aggiungendo (algebricamente) dei parametri di correzione alla pressione barometrica relativa misurata, per simulare lo spostamento dei cursori del regolo. La tabella *windCorr* serve per la correzione relativa alla direzione del vento usando la convenzione a 16 punti della rosa dei venti; dividendo l'angolo giro per 22°30' si arriva a stimare

i cosiddetti mezzi venti, che sono gli stessi forniti dai servizi metereologici on-line.
Il regolo originale è sviluppato per funzionare esclusivamente nell'emisfero nord, ma è sufficiente girare di 180° la rosa dei venti nel caso le previsioni siano effettuate nell'emisfero australe.
Il numero intero calcolato è usato come indice per recuperare, dalla tabella, il valore di correzione adatto. Attenzione che la variabile sia effettivamente di tipo **int**, altrimenti l'ArrayList non può es-

Listato 3

```
rule "Zambretti"
when
    Time   cron "0 0/10 * * * ?"
then
    val zambrettiText = newArrayList
    (
        //Zambretti forecast text
        'Bel tempo stabile',
        'Bel tempo',
        'In miglioramento',
        'Bello, in peggioramento',
        'Bello, possibilità di rovesci',
        'Abbastanza bello, in miglioramento',
        'Abbastanza bello, possibili rovesci mattutini',
        'Abbastanza bello, leggeri rovesci',
        'Rovesci, in miglioramento',
        'Variabile',
        'Abbastanza bello, rovesci probabili',
        'Piuttosto instabile, migliora sul tardi',
        'Instabile, probabile miglioramento',
        'Piovoso, schiarite a intervalli',
        'Piovoso, in peggioramento',
        'Variabile, qualche rovescio',
        'Alternanza di nuvole e brevi schiarite',
        'Alternanza di nuvole e rovesci a fine giornata',
        'Alternanza di nuvole e qualche rovescio',
        'Veramente molto instabile',
        'Occasionale pioggia, peggioramento',
        'Pioggia a tratti, molto instabile',
        'Frequenti rovesci',
        'Pioggia, molto instabile',
        'Temporalesco, in miglioramento',
        'Temporalesco, molta pioggia'
    )

    val windCorr = newArrayList
    (
       //pressure value correction with the wind direction
        5.2,  4.2,  3.2,  1.05, -1.1, -3.15, -5.2, -8.35, -11.5, -9.4, -7.3, -5.25, -3.2, -1.15,  0.9,  3.05
    )

    val lutRising = newArrayList
    (
        //Zambretti result in case of rising pressure
        //'A', 'B', 'B', 'C', 'F', 'G', 'I', 'J', 'L', 'M', 'M', 'Q', 'T', 'Y'
           0 ,  1,   1,   2,   5,   6,   8,   9,   11,  12,  12,  16,  19,  24
    )

    val lutFalling = newArrayList
    (
        //Zambretti result in case of falling pressure
        //'B', 'D', 'H', 'O', 'R', 'U', 'V', 'X', 'X', 'Z'
           1 ,  3,   7,   14,  17,  20,  21,  23,  23,  25
    )

    val lutSteady = newArrayList
    (
        //Zambretti result in case of steady pressure
        //'A', 'B', 'B', 'B', 'E', 'K', 'N', 'N', 'P', 'P', 'S', 'W', 'W', 'X', 'X', 'X', 'Z'
           0,   1,   1,   1,   4,   10,  13,  13,  15,  15,  18,  22,  22,  23,  23,  23,  25
    )
```

sere usato. Un'ulteriore correzione, anche questa dipendente dall'emisfero, deve essere effettuata in funzione della stagione.
Il valore di pressione relativa attuale è normalizzato in una scala unica in funzione dei limiti estremi di pressione del luogo di misura. Per ora utilizziamo gli stessi del regolo originale, poi col tempo potremo correggerli analizzando la serie storica dei valori minimo e massimo calcolati in precedenza.
Possiamo ora applicare le tre diverse formule in funzione dell'andamento della pressione (**Listato 5**). L'istruzione ***logInfo*** è usata solo a scopo di debug durante lo sviluppo del codice, presenta nel log viewer di OpenHAB le variabili calcolate per verificarne la correttezza. L'indice ***F*** permette di ricercare l'indice ***Z*** nelle tre tabelle relative ai tre andamenti della pressione. È l'equivalente dell'assegnazione tramite le tre finestrelle del regolo.
L'indice ***Z*** è l'equivalente della lettera trovata con il regolo e permette di recuperare la stringa di descrizione delle previsioni dalla tabella ***zambrettiText***.
Come visto in precedenza, il codice HTML di presentazione è contenuto all'interno del *widget* stesso (**Fig. 3**). La stringa è presentata nel *widget* in HTML utilizzando *l'item* ***ForecastText***:

```
<span id="weather-forecast">{{'%s' | sprintf:itemValue('ForecastText')}}</span>
```

Per rendere ancora più accattivante la presentazione dei dati nell'HABpanel, associamo anche un'icona specifica alla previsione.
In Rete si possono scaricare liberamente delle icone come quelle visibili in **Fig.3**, che provengono dal sito Internet *http://meteoduquebec.com/icones_grzanka.html*.
Nel *widget* sono selezionate tramite *l'item* chiamato ForecastIdx:

```
<td><img id="weather-icon" src="../static/weather-data/images/{{IconSet}}/{{itemValue('ForecastIdx')}}.png" align="left"/></td>
```

Tutto il codice e le icone usati per questo esempio sono reperibili su https://guiott.com/?attachment_id=3167. Le icone devono essere copiate in *html/weather-data/images/colorful*. Gli stili CSS in *html/weather-data/layouts*.
Le *rule* e gli *item* nelle directory corrispondenti ag-

Listato 4

```
var north = true
    var int windIdx=0

    if (north == true)
    {
        windIdx=(Math.floor((Wind_Direction_deg.state as DecimalType).floatValue / 22.5)).intValue
    }
    else
    {
        windIdx=((Math.floor((Wind_Direction_deg.state as DecimalType).floatValue / 22.5) + 8) % 16).intValue
    }

    //1 - pressure value correction by the wind direction, meaning the value got from the windCorr array
    var Number plusWind=0

    if ((Wind_Speed.state as DecimalType) >= 0.3)
    {
        plusWind=windCorr.get(windIdx)
    }

    //2 - the formula depends on the emisphere and on the season
    var summer = (north == ((now.getMonthOfYear >=4) && (now.getMonthOfYear <=9)))

    //3 - the main parameters of the formula are the current barometric pressure...
    val Number baro_lower=950.0
    val Number baro_upper=1050.0
    //normalize pressure
    var Number pressure=0
    pressure=950.0+((1050.0-950.0)*((Netatmo_Indoor_Pressure.state as DecimalType)-baro_lower)/(baro_upper-baro_lower))+plusWind

    //4 - ...and the trend of the pressure in the past 3 hours
    var Number trend=(PressureTrendFlag.state as DecimalType).floatValue
```

giustando le voci secondo i propri sensori specifici. Il *custom widget* si aggiunge all'HABpanel importando il file JSON corrispondente, che contiene anche il codice HTML.

REALIZZARE UN DISPLAY AVANZATO

Vediamo ora come rendere il display interattivo, con una configurazione dinamica delle funzioni necessarie; andremo a interagire con l'HABpanel in modo flessibile e più completo del semplice ON/OFF o regolazione di uno slider.

CRONOTERMOSTATO MULTIPLO

I valori rilevati in vari locali da sensori locali e remoti, possono essere utilizzati, oltre che per la visualizzazione, per il controllo dell'impianto di riscaldamento/raffrescamento. Usiamo, per comodità, l'acronimo inglese HVAC (Heating, Ventilation, Air Conditioning = Riscaldamento, Ventilazione, Condizionamento dell'aria) che è diventato di uso comune in tutte le lingue.
Lo scopo è realizzare un cronotermostato configurabile dall'utente direttamente dal pannello di controllo, per la regolazione automatica della temperatura, in modo molto flessibile e dinamico. Siccome il codice richiesto è molto complesso, non siamo partiti da zero, ma abbiano seguito l'evoluzione dello sviluppo di Geo Magadan, uno dei tanti partecipanti al forum di OpenHAB, che ha gettato le basi per un prodotto molto interessante, anche se non finalizzato. Il software originale si può trovare su Github (*https://github.com/nepotu/ohscheduler*) mentre in *https://guiott.com/?attachment_id=3166* trovate quello da noi modificato per la nostra applicazione. Le modifiche sono estetiche (per renderlo omogeno con gli altri HABpanel già configurati per il display da 7" CM3-Panel), linguistiche (è tutto tradotto in italiano) e funzionali (sono stati inseriti i controlli con isteresi degli attuatori e la visualizzazione diretta dello stato). Studieremo le tecniche usate per ottenere il risultato desiderato analizzando gli elementi principali del codice "server" (le *rule*) e "client" (il *widget* per l'HABpanel).

Lato server

La configurazione dei parametri di un sistema che controlla un numero variabile di stanze, con temperature anche diverse, in più fasce orarie, non è facile: usare un numero fisso di variabili porterebbe ad un software difficilmente gestibile; raggruppando le informazioni in formato JSON, si può avere una configurazione dinamica che si adatta

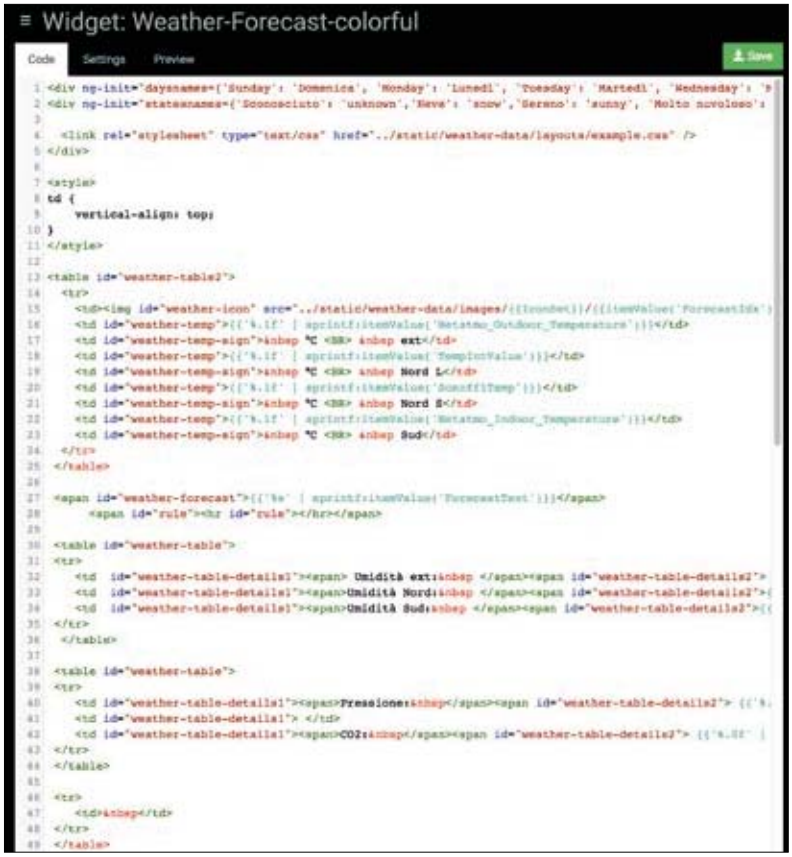


Fig. 3
Codice HTML del widget.

facilmente alle diverse esigenze. L'esempio nel **Listato 6** riporta la configurazione di quattro stanze per una programmazione su più orari e giorni della settimana (in **Fig. 5** vedete una sola stanza).
Nella *rule* ***OHSchedulerHVAC.rules*** (**Listato 7**) un parser analizza, ogni minuto o a seguito di ogni

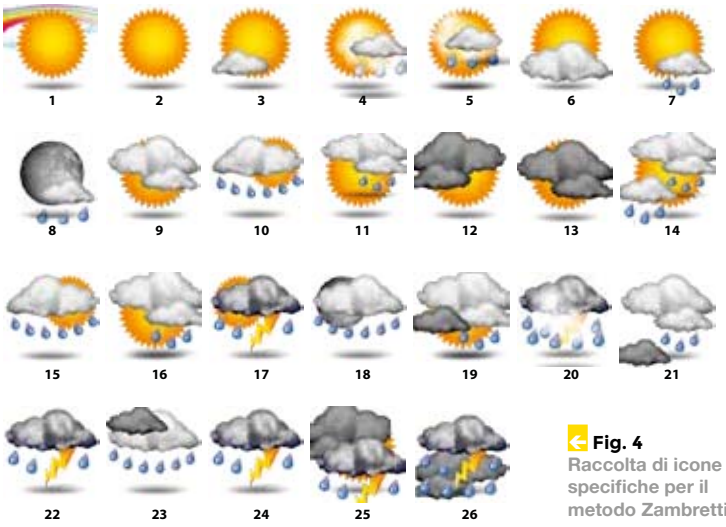


Fig. 4
Raccolta di icone specifiche per il metodo Zambretti.

cambiamento, l'item stringa in formato JSON ed estrae i dati relativi alla variazione successiva:

- **Target_Item_x** – l'item che riceverà la nuova temperatura di riferimento
- **D1-D7** – giorno della settimana (da Lunedì a Domenica)
- **Thhmm** – ora del giorno
- **value** – temperatura di riferimento (set point).

A questo punto imposta un timer con il prossimo cambiamento e non controlla più la stringa JSON fino a quel momento, a meno che non avvenga prima una modifica della programmazione o della modalità di funzionamento.
Al momento impostato nella programmazione, si riattiva e riempie una coda con i nuovi valori da impostare per ogni stanza.
Al variare dell'*item* ***HVAC_Queue*** parte la *rule* dello scheduler che aggiorna il nuovo valore di temperatura di riferimento stanza per stanza (**Listato 8**).
Quando il set point o la temperatura attuale di un membro qualsiasi dei gruppi *grpTemp* o *grpTargetTemp* cambiano, si attiva la regola di controllo degli attuatori. Non sono state specificate, in questa *rule*, le azioni da intraprendere perché possono variare da impianto a impianto.
Nel caso di un impianto di riscaldamento centralizzato potrebbero inviare, ad esempio, il nuovo valore di temperatura stanza per stanza alle manopole termostatiche dei radiatori collegate wireless in

Listato 5

```
var Number F = 0
var Number Z = 0

if (trend == 1)
{// rising pressure
    if (summer) {pressure+=3.2}

    F=0.1740 * (1031.40 - pressure)

    if (F < 0.0) {F = 0}
    else if (F > lutRising.size) {F = lutRising.size-1}
    else {F=(Math.floor(F)).intValue}

    Z=lutRising.get(F)
}
else if (trend == -1)
{// falling pressure
    if (summer) { pressure-=3.2}

    F=0.1553 * (1029.95 - pressure)

    if (F < 0.0) {F = 0}
    else if (F > lutFalling.size) {F = lutFalling.size-1}
    else {F=(Math.floor(F)).intValue}

    Z=lutFalling.get(F)
}
else
{// steady pressure
    F=0.2314 * (1030.81 - pressure)

    if (F < 0.0) {F = 0}
    else if (F > lutSteady.size) {F = lutSteady.size-1}
    else {F=(Math.floor(F)).intValue}

    Z=lutSteady.get(F)
}

ForecastIdx.postUpdate(Z+1)
ForecastText.postUpdate(zambrettiText.get(Z).toString)

logInfo('weather.rules',"Emisfero: "+north+" Wind IDX: "+windIdx+" Pressure: "+Netatmo_Indoor_Pressure.state as DecimalType+" Pressure Trend FLAG: "+PressureTrendFlag.state+" plus wind "+plusWind+" "+now.getMonthOfYear+" summer: "+summer+" F: "+F+" Z: "+Z+" "+ForecastText.state)
end
```

L'ALGORITMO ZAMBRETTI

La capacità di prevedere il tempo locale di pescatori e agricoltori è rinomata e agli inizi del secolo passato qualcuno ha pensato di "normalizzare" queste capacità concentrandole su un regolo circolare. È sufficiente conoscere la pressione barometrica del luogo relativa al livello del mare, la direzione del vento e la tendenza della pressione nelle ultime tre ore.
L'utilizzo è molto semplice: si posiziona il cerchio mobile più esterno (Fig. A) sul segno corrispondente al vento dominante, poi si gira il disco interno finché la freccia corrisponde con la pressione attuale e a questo punto basta leggere la lettera che appare sulla finestrella corrispondente alla tendenza della pressione, con una ulteriore piccola correzione se la misura è effettuata in estate (S) o in inverno (W). Voltando il regolo (Fig. B) si legge la previsione corrispondente alla lettera:

A	Bel tempo stabile
B	Bel tempo
C	In miglioramento
D	Bello, in peggioramento
E	Bello, possibilità di rovesci
F	Abbastanza bello, in miglioramento
G	Abbastanza bello, possibili rovesci mattutini
H	Abbastanza bello, leggeri rovesci
I	Rovesci, in miglioramento
J	Variabile
K	Abbastanza bello, rovesci probabili
L	Piuttosto instabile, migliora sul tardi
M	Instabile, probabile miglioramento
N	Piovoso, schiarite a intervalli
O	Piovoso, in peggioramento
P	Variabile, qualche rovescio
Q	Alternanza di nuvole e brevi schiarite
R	Alternanza di nuvole e rovesci a fine giornata
S	Alternanza di nuvole e qualche rovescio
T	Veramente molto instabile
U	Occasionale pioggia, peggioramento
V	Pioggia a tratti, molto instabile
W	Frequenti rovesci
X	Pioggia, molto instabile
Y	Temporalesco, in miglioramento
Z	Temporalesco, molta pioggia

Pur essendo un metodo empirico, riesce ad avere una precisione superiore al 90% nelle previsioni locali, in particolare se le misure sono eseguite intorno alle 9 del mattino per la giornata. Diversi anni fa qualcuno ha pensato di trovare un algoritmo informatico che riuscisse ad avere le stesse funzionalità: riportando su una curva i risultati del regolo per ogni tendenza della pressione (in aumento, in diminuzione, stabile) si ottengono tre serie che possono essere approssimate con ottima precisione da tre diverse equazioni. Il software originale, in javascript, è stato lo spunto dal quale sono derivati molti altri in tutti i linguaggi di programmazione, compresi dei porting su microcontrollori a 8 bit. La versione digitale corregge anche i limiti del regolo originale che, essendo stato progettato in Inghilterra, non tiene conto delle differenze tra emisfero boreale e australe. Questo stesso algoritmo è tuttora usato, a dimostrazione della sua efficacia, in molti dispositivi commerciali e professionali.

Fig. A - regolo Zambretti, facciata anteriore.

Fig. B - regolo Zambretti, facciata posteriore.

Z-wave o WiFi.
Invece nel caso di un di impianto termoautonomo, potrebbe attivare direttamente l'accensione della caldaia o l'apertura dell'elettrovalvola di controllo del flusso d'acqua. Nel caso di impianto a pompa di calore potrebbero attivare il condizionatore

SCALA BEAUFORT

La Scala di Beaufort è una misura empirica della forza del vento misurata in 13 gradi o numeri (da 0 a 12) indicati col simbolo **Bft**.
Anche se la velocità del vento può essere misurata con buona precisione mediante un anemometro, che esprime un valore in nodi o in chilometri all'ora, è possibile stimare il grado sulla scala Beaufort con la sola osservazione degli effetti del vento sull'ambiente, riportati nella tabella in basso al riquadro.
Il merito di avere perfezionato, nel 1805, una scala contenente dei criteri relativamente precisi per quantificare il vento in mare e permettere in tal modo la diffusione di informazioni affidabili e universalmente comprese sulle condizioni di navigazione si deve all'ammiraglio britannico Francis Beaufort (1774 ÷ 1857). Questo sistema di valutazione ha validità internazionale dal 1° gennaio 1949.
In tabella è riportata la corrispondenza della velocità del vento misurata con un anemometro con il corrispettivo indice Bft.
Riportando su un grafico la corrispondenza tra velocità in km/h e grado Bft (Fig. C) possiamo vedere come la curva è approssimata con ottima precisione dall'equazione:

$$B = \left(\frac{V}{4}\right)^{\frac{1}{\sqrt{2}}}$$

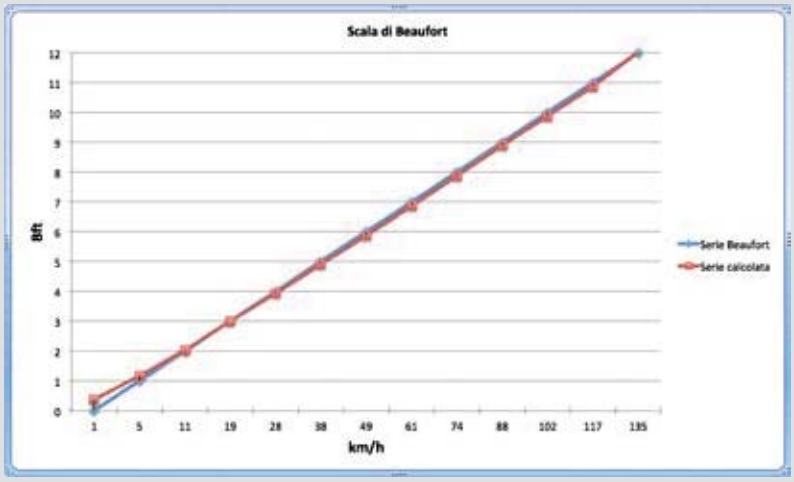


Fig. C - Corrispondenza tra velocità del vento in km/h e Bft.

dove V è la velocità del vento in km/h e B è l'indice Beaufort corrispondente. Usando questo valore come indice in una tabella, è possibile associare a programma la velocità del vento misurata tramite un anemometro o prelevata da un servizio metereologico on-line, con la descrizione della scala Beaufort. La visualizzazione della stringa corrispondente nell'HABpanel fornisce sicuramente un'informazione più immediata rispetto alla velocità o all'indice Bft per chi non sia marinaio o meteorologo.

FORZA BEAUFORT	VELOCITA' IN NODI	VELOCITA' IN KM/H	DESCRIZIONE DEL VENTO	ALTEZZA DELLE ONDE IN METRI	ALTRI EFFETTI AMBIENTALI
0	<1	<1	Calma	0	Il fumo sale verticalmente
1	1-3	1-5	Bava di vento	0.1	Il fumo devìa leggermente
2	4-6	6-11	Brezza leggera	0.2	Si muovono le foglie
3	7-10	12-19	Brezza tesa	0.6	Si agitano foglie e piccoli rami
4	11-16	20-28	Vento moderato	1	La polvere si solleva
5	17-21	29-38	Vento teso	2	Anche gli arbusti oscillano
6	22-27	39-49	Vento fresco	3	Si agitano i grandi rami, i fili sibilano
7	28-33	50-61	Vento forte	4	Si muovono interi alberi, difficile camminare controvento
8	34-40	62-74	Burrasca	5.5	Non si riesce a camminare controvento. Si spezzano i rami
9	41-47	75-88	Burrasca forte	7	Camini e tegole vengono divelti
10	48-55	89-102	Tempesta	9	Alberi sradicati, ingenti danni alle abitazioni
11	56-63	103-117	Tempesta violenta	11.5	Devastazioni gravi
12	64 e oltre	118 e oltre	Uragano	14	Edifici e manufatti distrutti

Listato 6


```
{“HVAC_Bagno_G”:{“D1”:{“T0700”:23,”T1200”:”21.0”},”D2”:{“T0700”:23,”T1200”:”21.0”},”D3”:{“T0700”:23,”T1200”
:”21.0”},”D4”:{“T0700”:23,”T1200”:”21.0”},”D5”:{“T0700”:23,”T1200”:”21.0”},”D6”:{“T0800”:23,”T1300”:”21.0”
},”D7”:{“T0800”:23,”T1300”:”21.0”}},”HVAC_Camera_Letto”:{“D1”:{“T0800”:”22.0”,”T2200”:”20.0”},”D2”:{“T0800
”:”22.0”,”T2200”:”20.0”},”D3”:{“T0800”:”22.0”,”T2200”:”20.0”},”D4”:{“T0800”:”22.0”,”T2200”:”20.0”},”D5”:{“
T0800”:”22.0”,”T2200”:”20.0”},”D6”:{“T0900”:”22.0”,”T2300”:”20.0”},”D7”:{“T0900”:”22.0”,”T2300”:”20.0”}},”
HVAC_Soggiorno”:{“D1”:{“T0800”:”22.0”,”T2200”:”20.0”},”D2”:{“T0800”:”22.0”,”T2200”:”20.0”},”D3”:{“T0800”:”
22.0”,”T2200”:”20.0”},”D4”:{“T0800”:”22.0”,”T2200”:”20.0”},”D5”:{“T0800”:”22.0”,”T2200”:”20.0”},”D6”:{“T09
00”:”22.0”,”T2300”:”20.0”},”D7”:{“T0900”:”22.0”,”T2300”:”20.0”}},”HVAC_Studio”:{“D1”:{“T0800”:”22.0”,”T220
0”:”20.0”},”D2”:{“T0800”:”22.0”,”T2200”:”20.0”},”D3”:{“T0800”:”22.0”,”T2200”:”20.0”},”D4”:{“T0800”:”22.0”,
”T2200”:”20.0”},”D5”:{“T0800”:”22.0”,”T2200”:”20.0”},”D6”:{“T0900”:”22.0”,”T2300”:”20.0”},”D7”:{“T0900”:”22
.0”,”T2300”:”20.0”}}}
```


tramite ripetitore a infrarossi, come abbiamo visto in articolo precedenti.
In ogni caso l'accensione è regolata con isteresi, ossia c'è una soglia (nell'esempio +/- 0,5°C) all'interno della quale non c'è cambiamento; questo serve per evitare accensioni e spegnimenti troppo frequenti dell'impianto che, oltre ad essere inutili, potrebbero danneggiarlo (**Listato 9**). Da notare che all'interno del codice analizzato non c'è mai traccia diretta del nome degli *item* interessati, perché questi sono tutti gestiti in modo dinamico; vediamo come. Il software è configurato in modo da non dover cambiare codice per aggiungere o modificare gli *item*. È possibile aggiungere o togliere stanze semplicemente configurando il *widget*. I nomi degli *item* sono presi in modo dinamico da questo e dalla definizione nel file ***OHScheduler.item*** che, ovviamente, devono coincidere (**Listato 10**).
In questo caso gli *item* sono definiti all'interno della variabile "items" con la struttura riportata nel **Listato 11**.
Per ogni stanza c'è un gruppo di quattro *item*: set point, valore attuale di temperatura, descrizione, stato dell'attuatore. Raggruppando le categorie in gruppi omogenei:

```
Group     grpTemp
Group     grpTargetTemp
Group     grpTargetSwitch
```

è possibile rendere il software autonomamente configurabile semplicemente con qualche operazione sulle stringhe, come ad esempio quanto riportato nel **Listato 12**. Come vedremo in seguito, la visualizzazione sull'HABpanel è gestita dinamicamente con un metodo simile.

Lato client

La visualizzazione nell'HABpanel è realizzata sfruttando molto bene le potenzialità dell'HTML5. Un tutorial su come sviluppare *widget* personalizzati in OpenHAB è reperibile in *https://community.openhab.org/t/habpanel-development-advanced-features-start-here/30755* e potete fare riferimento ad esso per comprendere alcune scelte progettuali.
Il web server di OpenHAB è dotato dello stato dell'arte della tecnologia di sviluppo per tale ambiente. Si possono usare le potenzialità del toolkit Bootstrap, con il suo sistema di griglia e le classi già pronte per tabelle, pulsanti, tab, drop menu eccetera, senza dover disegnare stili CSS complessi. Si possono anche utilizzare le direttive AngularJS per una completa interattività, come ad esempio le *ng-knob* usate come strumenti e le librerie *VisJS* per il *timeline* con il quale interagire sulla programmazione oraria/giornaliera. A questi si aggiungono le funzioni specifiche dell'HABpanel che permettono un'interazione completa con gli *item*. Il codice HTML del *widget* carica i file javascript e gli stili CSS. Il file ***ohscheduler.tpl.html*** è il *template* html per il *widget* usato dallo scheduler: utilizzando le diretti-

Fig. 5
Esempio di programmazione delle temperature per una stanza.

ve Bootstrap e AngularJS, costruisce un HABpanel dinamico e fluido, che si adatta al contenitore. Nel codice javascript contenuto in **ohscheduler.js** ci sono tutti i metodi per disegnare strumenti e pulsanti ed interagire con essi, oltre alle utility usate dal programma. Dal momento che non tutto è parametrizzato esternamente, alcuni parametri possono essere modificati nel codice per personalizzare il proprio HABpanel, come, ad esempio, nella funzione che inizializza le diverse viste:

```
// OHS Config - store scheduler and view parameters
    function OHSConfig() {
// Init configuration object tabIdx: 0 - Status
```

Listato 7

```
//HVAC Scheduler - Target Temperature Management Rules
import java.util.concurrent.locks.ReentrantLock

//Global variables
var ReentrantLock tempCtlLock = new ReentrantLock()
var ReentrantLock tempQueueLock = new ReentrantLock()
var Timer tempCtlTimer = null

var hysteresis = 0.5

// HVAC Scheduler - Target Temperature Management
rule "HVAC Scheduler - Target Temperature Management"
when
    Time   cron "0 0/1 * * * ?"           or
    Item   HVAC_Mode received update      or
    Item   HVAC_Mode received command     or
    Item   HVAC_Mode changed              or
    Item   HVAC_Schedule received update  or
    Item   HVAC_Schedule received command or
    Item   HVAC_Schedule  changed

then
    if (tempCtlLock.tryLock()) {
        try {
            //tempCtlLock.lock()
            if (HVAC_Mode.state == 'AUTO' || HVAC_Mode.state == 'FREDDO' || HVAC_Mode.state == 'CALDO') {
                val DateTime timeNow = now()
                val Number trimToMin = 60000
                val String runningMode = (HVAC_Mode.state as StringType).toString
                var String triggeredBy = null
                try {
                    triggeredBy = triggeringItem.name
                } catch(Exception e) {
                    triggeredBy = receivedCommand
                }
                // Cancel the timer if the command was received on HVAC_Mode or HVAC_Schedule change
                if (tempCtlTimer !== null && triggeredBy !== null) {
                    tempCtlTimer.cancel()
                    tempCtlTimer = null
                }
                if (tempCtlTimer === null || tempCtlTimer.hasTerminated) {
                    var String currentTime = null
                    var String temperatureQueue = null
                    var DateTime upToTime = null
                    if (timeNow.getHourOfDay() < 10) {
                        currentTime = 'T0' + timeNow.getHourOfDay() + timeNow.getMinuteOfHour
                    } else {
                        currentTime = 'T' + timeNow.getHourOfDay()+ timeNow.getMinuteOfHour
                    }
                    if (HVAC_Schedule.state != NULL) {
                        grpTargetTemp.members.forEach [ targetItem | {
                                var String jsonHVACSchedule = (HVAC_Schedule.state as StringType).toString
                                var String jsonItemSchedule = null
                                var String jsonItemDaySchedule = null
                                // Parse the item schedule
//..... JSON parsing code follows
```

Listato 8

```
// HVAC Scheduler - A queue item, used to set the target temperature for multiple items at the same time
rule "HVAC Scheduler - Target Temperature Queue"
when
    Item HVAC_Queue received update  or
    Item HVAC_Queue received command or
    Item HVAC_Queue changed
then
    if (tempQueueLock.tryLock()) {
        try {
            if (HVAC_Queue.state != NULL && HVAC_Queue.state != '') {
                grpTargetTemp.members.forEach [ targetItem | {
                        var Number j = -1
                        var String[] itemsTempArray = null
                        itemsTempArray = (HVAC_Queue.state as StringType).toString.split(';')
                        while ((j=j+1) < itemsTempArray.length) {
                            if (itemsTempArray.get(j.intValue()).split('=').get(0) == targetItem.name) {
                                logInfo('temperature-control.rules', 'Target temperature for item ' + targetItem.name
                                    + ' changed from ' + targetItem.state
                                    + ' to ' + DecimalType.valueOf(itemsTempArray.get(j.intValue()).split('=').get(1)))
                                targetItem.postUpdate(DecimalType.valueOf(itemsTempArray.get(j.intValue()).split('=').get(1)))
                            }
                        }
                    }
                ]
                // Cleanup the queue
                HVAC_Queue.postUpdate('')
                logInfo('temperature-control.rules', 'Target Temperature Queue cleaned up')
            }
        } catch(Exception e) {
                logInfo('temperature-control.rules', 'Target Temperature Queue control error: ' + e.toString)
        } finally {
                tempQueueLock.unlock()
        }
    }
end
```

```
// 1 - Schedule,2 - AddEdit
        var ohsConf = { ........
```

Oppure la funzione che configura gli oggetti legati alla temperatura:

```
// OHS Temperature
    function OHSTemperature() {
        // Init temperature object
        var ohsTemp = {
        .....
```

O per tradurre i termini in italiano:

```
// OHS Week Days
    function OHSDays() {
        var ohsDays = [{
        ........
```

La funzione OHSKnobs configura gli strumenti di misura, i *gauge*, secondo le direttive ng-knob di AngularJS. Anche questa può essere modificata secondo le proprie esigenze, aggiungendo indicazioni, cambiando colori o tipologia di strumento:

```
// OHS Knobs
    function OHSKnobs() {
        var ohsKnobs = {};
        ........
```

In modo simile si può modificare la visualizzazione della *timeline*

```
// OHS Timeline
    function OHSTimeline() {
        var ohsTimeline = {};
        .......
```

Le altre funzioni interagiscono direttamente con il *widget* in modo bidirezionale, aggiornando la visualizzazione in funzione dello stato degli *item* (il canale di collegamento tra l'ambiente client e quello server), oppure modificando i parametri di intervento delle *rule*. L'interazione avviene in modo dinamico e il numero di strumenti per la visualizzazione della temperatura cambia in funzione dei

Listato 9

```
// Thermostat Action. Drive the HVAC control items according to the current temperature and set point
//        < (-1)      = (0)       > (1)      Setpoint (switchItem)
//AUTO: ON  heater     OFF      ON  cooler
//COLD: OFF cooler     OFF      ON  cooler      (OFF heater)
//HEAT: ON  heater     OFF      OFF heater      (OFF cooler)
rule "Thermostat Action"
when
    Member of  grpTemp changed or
    Member of  grpTargetTemp changed
then
    grpTargetTemp.members.forEach [ targetItem |
    {
       var switchItem = grpTargetSwitch.members.findFirst[ t | t.name == targetItem.name + "_Switch" ] as NumberItem
       var tempItem   = grpTemp.members.findFirst[ t | t.name == targetItem.name + "_Temp" ] as NumberItem
       var Number setPoint = (targetItem.state as DecimalType).floatValue
       var Number setPointUP = setPoint + hysteresis;
       var Number setPointDOWN = setPoint - hysteresis;

       if(switchItem.state == 99)
       {
       }
       else
       {//the measure is not too old to be considered valid
           if((tempItem.state as DecimalType).floatValue >=setPointUP)
           {//TOO HOT
               switchItem.postUpdate(1)
               if (HVAC_Mode.state == 'AUTO' ||  HVAC_Mode.state == 'FREDDO')
               {
                   logInfo('temperature-control.rules', 'Switch OFF HEATER')
                   logInfo('temperature-control.rules', 'Switch ON  COOLER')
                   //PUT code to act on your system here
               }
               else if (HVAC_Mode.state == HVAC_Mode.state == 'CALDO')
               {
                   logInfo('temperature-control.rules', 'Switch OFF HEATER')
                   logInfo('temperature-control.rules', 'Switch OFF COOLER')
                   //PUT code to act on your system here
               }
           }
           else if((tempItem.state as DecimalType).floatValue <=setPointDOWN)
           {//TOO COLD
               switchItem.postUpdate(-1)
               if (HVAC_Mode.state == 'AUTO' ||  HVAC_Mode.state == 'CALDO')
               {
                   logInfo('temperature-control.rules', 'Switch ON  HEATER')
                   logInfo('temperature-control.rules', 'Switch OFF COOLER')
                   //PUT code to act on your system here
               }
               else if (HVAC_Mode.state == HVAC_Mode.state == 'FREDDO')
               {
                   logInfo('temperature-control.rules', 'Switch OFF HEATER')
                   logInfo('temperature-control.rules', 'Switch OFF COOLER')
                   //PUT code to act on your system here
               }
           }
           else
           {//COMFORT
               switchItem.postUpdate(0)
               logInfo('temperature-control.rules', 'Switch OFF HEATER')
               logInfo('temperature-control.rules', 'Switch OFF COOLER')
               //PUT code for your system here
           }
       }
    }
    ]
end
```

Listato 10

```
// itemtype itemname        “labeltext [stateformat]”     <iconname>    (group1,group2,...)     [“tag1”, “tag2”, ...] {bindingconfig}
// ------------------------------------------------------------------------------------------------------------------------------------------- //

// -- Thermostat Items -- //
Group       grpTemp
Group       grpTargetTemp
Group       grpTargetSwitch

String   HVAC_Mode     “HVAC Mode [%s]”    <switch>              [“HVACMode”]
String   HVAC_Schedule “HVAC Schedule [%s]” <calendar>           [“HVACSchedule”]
String   HVAC_Queue    “HVAC Queue [%s]”                         [“HVACQueue”]
DateTime HVAC_Next_Change_Time “HVAC Next Change Time [%1$ta %1$tR]” <calendar> [“HVACNextChange”]

Number   HVAC_Soggiorno_Temp “Temp Soggiorno[%.1f °C]”  <temperature> (grpTemp)
          { mqtt=”<[broker1:tele/TH10_sensor/SENSOR:state:JSONPATH($.SI7021.Temperature)]” }
Number   HVAC_Soggiorno      “Temp Soggiorno Set [%.1f °C]” <temperature> (grpTargetTemp)
          [“LRTargetTemp”]
Number   HVAC_Soggiorno_Switch “Temp Soggiorno Status [%d]”  <switch>   (grpTargetSwitch)
          [“LRswitch”]

Number   HVAC_Bagno_G_Temp  “Temp Bagno giorno[%.1f °C]”    <temperature> (grpTemp)
          { mqtt=”<[broker1:tele/TH10_sensor2/SENSOR:state:JSONPATH($.SI7021.Temperature)]” }
Number   HVAC_Bagno_G  “Temp Bagno Giorno Set [%.1f °C]” <temperature> (grpTargetTemp)
          [“BGTargetTemp”]
Number   HVAC_Bagno_G_Switch “Temp Bagno Giorno Status [%d]” <switch> (grpTargetSwitch)
          [“BGswitch”]

Number  HVAC_Camera_Letto_Temp  “Temp Camera Letto [%.1f °C]”  <temperature> (grpTemp)
          { channel = “netatmo:NAMain:b108a673:70ee502af646:Temperature” }
Number  HVAC_Camera_Letto  “Temp Camera Letto Set[%.1f °C]” <temperature> (grpTargetTemp)
          [“MBTargetTemp”]
Number  HVAC_Camera_Letto_Switch “Temp Camera Letto Status[%d]”  <switch> (grpTargetSwitch)
          [“MBswitch”]

Number  HVAC_Studio_Temp  “Temperatura Studio [%.1f °C]”   <temperature> (grpTemp)
Number  HVAC_Studio “Temp Studio Set[%.1f °C]”             <temperature> (grpTargetTemp)
          [“KBTargetTemp”]
Number  HVAC_Studio_Switch “Temp Studio Status[%d]”        <switch>  (grpTargetSwitch)
          [“KBswitch”]
```

parametri impostati sul *widget*, cioè del numero di *item* definiti.

INSTALLAZIONE

In questo caso il *widget* contiene solamente il codice HTML che richiama i file necessari.

```
<div oc-lazy-load=”{name: ‘OHSchedulerCtl’, files:
['/static/habpanel/scheduler/vis.min.js?v=4.21.0','/
static/habpanel/scheduler/angular-vis.js?v=4.16.0','/
static/habpanel/scheduler/ohscheduler.js?v=1.0-
RC0','/static/habpanel/scheduler/ohscheduler.
css?v=1.0-RC0'], serie: true, cache: false}”>
    <div ng-controller=”OHSchedMainCtl as ohs”>
        <widget-scheduler conf=”{{config}}”></wid-
get-scheduler>
```

Listato 11

```
// Widget Items: [{“setItem”:”HVAC_Soggiorno”,”readItem”:”HVAC_Soggiorno_Temp”,”label”:”Soggiorn
o”,”switchItem”: “HVAC_Soggiorno_Switch”},{“setItem”:”HVAC_Camera_Letto”,”readItem”:”HVAC_Came-
ra_Letto_Temp”,”label”:”Camera Letto”,”switchItem”: “HVAC_Camera_Letto_Switch”},{“setItem”:”HVAC_
Studio”,”readItem”:”HVAC_Studio_Temp”,”label”:”Studio”,”switchItem”: “HVAC_Studio_
Switch”},{“setItem”:”HVAC_Bagno_G”,”readItem”:”HVAC_Bagno_G_Temp”,”label”:”Bagno Giorno”,”switchItem”:
“HVAC_Bagno_G_Switch”}]
```

Fig. 6 Esempio di configurazione del *widget*.

```
    </div>
</div>
```

La pagina HTML principale, il codice javascript e il file con gli stili devono essere messi in *html/habpanel/scheduler*. Il file *OHScheduler.widget.json* deve essere importato nell'HABpanel per inserire il *custom widget*. Nel file *OHScheduler.items* c'è un esempio di come devono essere definiti gli *item*, i quali devono essere configurati secondo i propri sensori. Ci sono due file di *rule* diversi per la funzione HVAC e quella ON/OFF; in questo esempio analizzeremo solo la funzione HVAC.
L'intera configurazione dello scheduler può essere memorizzata in InfluxDB per averla sempre disponibile anche in caso di riavvio della CM3-Home. È sufficiente aggiungere gli *item* necessari al file *persistence/ influxdb.persist*:

```
HVAC_Mode, HVAC_Schedule, OnOff_Mode, OnOff_Schedule
: strategy = everyMinute, everyChange, restoreOn-
Startup
```

Una volta che tutto l'ambiente è stato configurato, con i propri *item*, il *widget* deve essere configurato con i parametri dell'ambiente di destinazione.
In **Fig. 6** vedete un esempio di configurazione.

CONCLUSIONI

Bene, con questo esempio applicativo termina la nostra rassegna sulla piattaforma CM3-Edu, nata e sviluppata su core Raspberry Pi Compute Module 3 per rendere facile e didattica la realizzazione di applicazioni in ambito domotico, dalle più semplici a quelle complesse, anche dotate di interfaccia grafica utente con pannello LCD interattivo, eventualmente connesse a Internet.

Cosa occorre?

I componenti utilizzati in questa presentazione sono disponibili presso Futura Elettronica.
La board CM3-Home (cod. CM3HOME) è venduta a Euro 169,00, il modulo Raspberry Pi CM3 LITE (cod. CM3LITE) costa Euro 36,00, la scheda WiFi (cod. RT5370N) è disponibile separatamente a Euro 9,00.
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

Listato 12

```
rule "Thermostat Action"
when
    Member of  grpTemp changed or
    Member of  grpTargetTemp changed
then
    grpTargetTemp.members.forEach [ targetItem |
    {
        var switchItem = grpTargetSwitch.members.findFirst[ t | t.name == targetItem.name + "_Switch" ] as
NumberItem
        var tempItem   = grpTemp.members.findFirst[ t | t.name == targetItem.name + "_Temp" ] as NumberItem
```

Futura GROUP srl

UN TEAM DI PROFESSIONISTI
PER IL TUO PROGETTO

oltre 1500 progetti sviluppati
26 anni di attività
più di 15 progettisti

L'area R&D di Futura Group studia, progetta e realizza prototipi per differenti settori utilizzando tecnologie innovative. Nel team di Futura Group fa parte il personale interno e un numero elevato di collaboratori esterni, ciascuno specializzato nel proprio settore (Disegno CAD – Raspberry Pi – Arduino – Audio – Sviluppo firmware – Linux – ecc). La disponibilità di oltre 5000 prodotti elettronici e componenti consente di ridurre i tempi di test e sviluppo e di realizzare facilmente il primo prototipo. La decennale esperienza nel settore e la necessità di ottimizzare i tempi e i costi, ha portato Futura Group a selezionare una serie di terzisti per la produzione di piccola o grande serie.

I SERVIZI CHE TI OFFRIAMO

Progetto Completo

Partendo dalla tua idea progettiamo per te la soluzione.

Sviluppo PCB

Ottimizziamo il tuo prototipo disegnando un PCB compatto secondo le tue specifiche.

Scrittura Codice

Scriviamo il codice del core per dare vita alla tua applicazione.

Produzione Progetto

Portiamo il prototipo alla fase di produzione seguendo per te tutte le fasi.

Maggiori informazioni e richiesta di preventivi su **design.futuragroupsrl.it**

UNI-T® STRUMENTI DI MISURA PER OGNI ESIGENZA!

PINZA AMPEROMETRICA PROFESSIONALE

Misura correnti AC e DC fino a 100 A, tensioni AC e DC fino a 600 volt, resistenze fino a 20 Mohm, capacità fino a 20 mF, continuità e diodi. Dispone di display LCD retroilluminato a 2000 conteggi, individuazione di tensione senza contatto (NCV), autospegnimento, indicatore di batteria scarica e data hold. Alimentazione con 2 batterie tipo AAA 1,5V (incluse).

€68,00

Cod. TP187

OSCILLOSCOPIO PALMARE 16 MHZ CON MULTIMETRO INTEGRATO

€289,00

Cod. TP174

Oscilloscopio palmare con multimetro integrato dalle caratteristiche professionali con larghezza di banda di 16MHz. Dispone di ampio display monocromatico (60x60mm) con una risoluzione di 160x160 pixel molto facile da leggere grazie alla funzione di retroilluminazione di cui è dotato. Integra una memoria interna che consente di salvare e visualizzare sul display fino a 10 segnali; pulsante "Auto-Set" pel lavorare in modo semplice e veloce. È dotato di porta USB che consente di trasferire i dati dei segnali dall'oscilloscopio al computer. Completo di software di gestione per PC. Alimentazione tramite batterie oppure adattatore di rete.

MULTIMETRO DIGITALE CON MISURA DI TEMPERATURA, CAPACITÀ, FREQUENZA, E TRANSISTOR

€31,90

Cod. TP0060

Misura correnti continue e alternate fino a 20 A, tensioni continue fino a 1000 V e alternate fino a 750 V, resistenze fino a 200 Mohm, capacità da 2nF a 20μF, frequenze fino a 20 kHz, temperatura da -40°C a +1000°C con termocoppia inclusa, diodi, transistor e continuità elettrica. Alimentazione con batteria a 9 V (inclusa). La confezione comprende: manuale, puntali, batteria, guscio di protezione e sonda di temperatura (termocoppia tipo K).

MINI FONOMETRO DIGITALE

€ 25,00

Cod. TP0070

Fonometro con risoluzione di 0,1 dB dotato di display LCD retroilluminato con indicazione digitale della misura. Rileva intensità sonore comprese tra 30 e 130 dB. Memorizza i valori minimi e massimi rilevati. Leggero, compatto, semplice da usare, ideale per monitorare i livelli di rumore negli hotel, nelle sale conferenze, nell'home theater, negli ospedali e in qualsiasi altro ambiente sensibile al rumore. Funziona con 3 batterie tipo AAA (incluse).

TACHIMETRO DIGITALE CON LASER E INTERFACCIA USB PER PC

€78,00

Cod. TP0030

Tachimetro digitale con puntatore laser; dispone di porta USB per la gestione e il salvataggio dei dati su PC. Indicato per la misurazione, senza contatto, della velocità angolare di alberi motore, ruote dentate e pulegge, può anche essere utilizzato come contapezzi. Funziona con 4 batterie tipo AA (incluse).

TERMOMETRO DIGITALE PORTATILE PER TERMOCOPPIE TIPO K E J

€ 19,90

Cod. TP0062

Rileva temperature fino a 1300 °C. Dispone di ampio display LDC retroilluminato a 4 cifre che permette di leggere facilmente i valori. Visualizzazione in °C e in °F selezionabile. Mostra la temperatura massima, minima e media rilevata. La termocoppia tipo K inclusa permette di rilevare una temperatura massima fino a 260°C. Funziona con 3 batterie tipo AAA (incluse).

LUXMETRO PORTATILE 0 - 199900 LUX

€ 34,00

Cod. TP0063

Misura l'illuminazione prodotta da LED (luce visibile), lampade fluorescenti, lampade ad alogenuri metallici, lampada al sodio ad alta tensione o lampada ad incandescenza. Dispone di display LCD retroilluminato a 4 digit, unità di misura selezionabili LUX / FC, indicazione di min e max, indicazione di batteria scarica, spegnimento automatico, data hold. Funziona con 3 batterie tipo AAA (incluse).

ANEMOMETRO DIGITALE

€ 37,00

Cod. TP0064

Misura la velocità e la temperatura dell'aria. Lettura della velocità dell'aria in m/s, km/h, ft/min, knots, mph. Lettura della temperatura in gradi Centigradi e gradi Fahrenheit, display LCD retroilluminato a 3 cifre, memorizzazione dei valori minimi, massimi e medi rilevati, funzione di autospegnimento, indicatore di batteria scarica. Funziona con 3 batterie tipo AAA (incluse).

Prezzi IVA inclusa

CIRCUITO GIRATORE

di PIER-ALESSANDRO AISA

Simula elettronicamente le induttanze pur senza farne uso, grazie a un componente attivo.

La progettazione di alimentatori in ambito audio, per il filtraggio dell'alimentazione degli amplificatori di potenza, vede spesso in gioco l'utilizzo di celle di filtro di tipo LC realizzate con induttanze e condensatori. Il filtraggio dell'alimentazione è necessario per abbattere l'ondulazione residua dal raddrizzamento della tensione alternata in ingresso (ripple) allo scopo di attenuare le armoniche generate durante il funzionamento dell'amplificatore e gli eventuali picchi di extra-tensione provenienti dalla rete elettrica. L'amplificatore audio, per poter riprodurre fedelmente le ampie variazioni d'ampiezza presenti in un brano e dovute alla dinamica, richiede all'alimentatore una grande potenza istantanea; in tale condizione si verifica la richiesta di elevate correnti impulsive, che se non opportunamente filtrate si traducono

$$\begin{cases} i_1(t) = \gamma \cdot v_2(t) \\ i_2(t) = -\gamma \cdot v_1(t) \end{cases}$$

Fig. 1
Schematizzazione del quadripolo e relative equazioni, corrispondenti al circuito giratore.

in un calo di prestazione e resa dell'amplificatore audio. In alcuni casi la tensione di ripple si ritrova direttamente accoppiata sul trasformatore d'uscita, come ad esempio nel caso di un amplificatore valvolare Single Ended realizzato con un triodo funzionante senza controreazione (circuitazione non-NFB, ossia senza Negative Feed Back).
Per limitare questi effetti si ricorre spesso alla progettazione di filtri con configurazione a pi-greco, ossia realizzati con celle di tipo *C-L-C*, dove bisogna porre particolare attenzione al dimensionamento dei componenti, nei confronti del picco che si verifica in risonanza e del valore di impedenza che ne deriva.
Difatti, in presenza di notevoli variazioni della corrente assorbita dallo stadio finale o di variazioni rapide della tensione di rete, la variazione della tensione d'uscita è accompagnata da una tensione oscillante (sovraelongazione) smorzata nel tempo, la cui durata e ampiezza dipende fortemente dal fattore di merito Q del circuito risonante.
Le induttanze presentano un'alta impedenza nei confronti delle componenti a frequenza più elevata, ovvero la loro reattanza cresce linearmente con l'aumentare della frequenza; i condensatori costituiscono dei serbatoi di energia, che contribuiscono all'azione di filtraggio in quanto si oppongono alle brusche variazioni di tensione ai loro capi.
Combinando in una cella di filtro LC questi due componenti, in modo che l'induttanza sia in serie al segnale e il condensatore in parallelo, avremo un'attenuazione crescente all'aumentare della frequenza, in quanto l'induttore attenua la corrente e il condensatore la tensione d'uscita.
Solitamente le induttanze vengono costruite su ingombranti e pesanti nuclei magnetici ed inoltre in fase di test possono servire induttanze di diverso valore. Usando il circuito "giratore" presentato in questo articolo potrete sostituire le induttanze, con un semplice circuito elettronico configurabile e dotato di pochi componenti, in grado di presentare ai suoi morsetti il comportamento reattivo tipico di un'induttanza, con la possibilità di regolare a piacimento la precisione del valore di induttanza ed il suo fattore di merito.

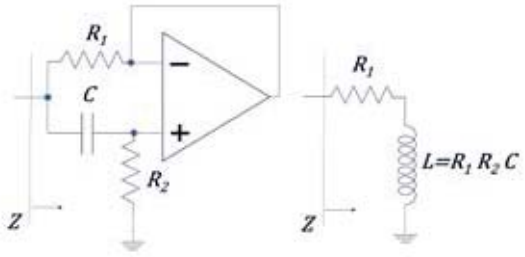


Fig. 2
Schema del circuito giratore con amplificatore operazionale ed induttanza equivalente

PRINCIPIO DI FUNZIONAMENTO

Il "giratore" fu inventato negli anni '50, da Bernard Tellegen un ricercatore dei laboratori della Philips e prende il suo nome dalla proprietà del circuito di effettuare un'inversione di impedenza, trasformando una reattanza capacitiva in una reattanza induttiva.
Con riferimento alla **Fig. 1**, da un punto di vista elettrico lo si può vedere come un doppio bipolo, caratterizzato dalla funzione di trasferimento, che lega tensione e corrente relative alla porta di ingresso ed alla porta di uscita, tramite il coefficiente Υ che è detto "*rapporto di girazione*".
La **Fig. 2** illustra una possibile implementazione del circuito giratore, tramite l'impiego di un amplificatore operazionale di un condensatore *C* e di due resistenze R_1 e R_2.
Dal punto di vista elettrico questo circuito assume lo stesso comportamento di una induttanza di valore $L = R_1 \times R_2 \times C$, come dimostreremo di seguito con i calcoli dell'impedenza equivalente.
Il circuito giratore è molto utile specialmente nella tecnologia dei circuiti integrati, dove è necessario avere un induttore miniaturizzato ma di elevato e preciso valore di induttanza. Infatti, il circuito può essere tarato finemente sulla precisione del valore, tramite la selezione di componenti ad alta precisione per i condensatori ed i resistori, a differenza di un induttore reale, che presenta una più ampia dispersione dei parametri e quindi una minor precisione.
Il massimo valore di induttanza che si può ottenere ed il relativo fattore di merito Q sono comunque limitati e non si può superare un limite operativo come di seguito descritto.
Con riferimento allo schema di **Fig. 2**, l'impedenza

che si vede dall'ingresso del circuito è data dal parallelo di due termini:

$$Z = (R_1 + j\omega R_1 R_2 C) // \left(R_2 + \frac{1}{j\omega C}\right)$$

Se la resistenza R_2 è molto più grande di R_1, l'espressione precedente si può approssimare con:

$$Z = (R_1 + j\omega R_1 R_2 C)$$

In questa espressione riconosciamo una parte resistiva ed una parte reattiva, equivalenti alla serie di una resistenza di valore R_1 e di un'induttanza di valore $L = R_1 \times R_2 \times C$.
Rispetto a una vera induttanza compaiono due termini: un termine *R2C* in parallelo ed un termine R_1 in serie, che limitano il massimo valore di induttanza ottenibile e anche il fattore di merito *Q* e di conseguenza limitano la selettività dei filtri che si possono realizzare.
Inoltre, bisogna tenere presente che a differenza di un'induttanza reale questo circuito non immagazzina energia sotto forma di campo magnetico e quindi non può essere usato in applicazioni come convertitori di potenza quali DC\DC, ad esempio in topologia *Flyback*, *Buck* o *Boost* o anche alimentatori per elevate tensioni/correnti o applicazioni che lavorano a radiofrequenza.
Il circuito giratore viene realizzato solitamente in forma di circuito a componenti discreti o integrati, con valori tipici di induttanza che possono arrivare anche fino a 50 henry e con correnti tipiche nel range 50÷200mA.
In questo articolo mostriamo una realizzazione su circuito stampato, con due circuiti giratori a bordo.
La forma circuitale che abbiamo selezionato fa l'impiego di un MOSFET per la trasformazione di impedenza, in modo che si possa sopportare una elevata differenza di potenziale fra ingresso e uscita del circuito. Questo è molto utile se si volesse, ad esempio, realizzare un filtro passa-basso di tipo LC per un alimentatore che opera ad alta tensione, come ad esempio in ambito valvolare, dove non è difficile doversi trovare di fronte a tensioni dell'ordine dei 400V.

SCHEMA ELETTRICO

In queste pagine riportiamo lo schema elettrico del circuito, che simula un'induttanza equivalente del valore di 22 henry e corrente di lavoro massima di 400 mA. Lo schema elettrico (che trovate nella pagina seguente) comprende due istanze del circuito giratore, con geometrie pensate per utilizzare condensatori di tipo differente, così da permettere una maggiore flessibilità di utilizzo.

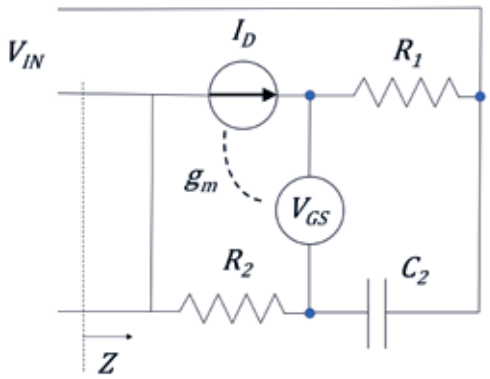


Fig. 3 Circuito equivalente del giratore.

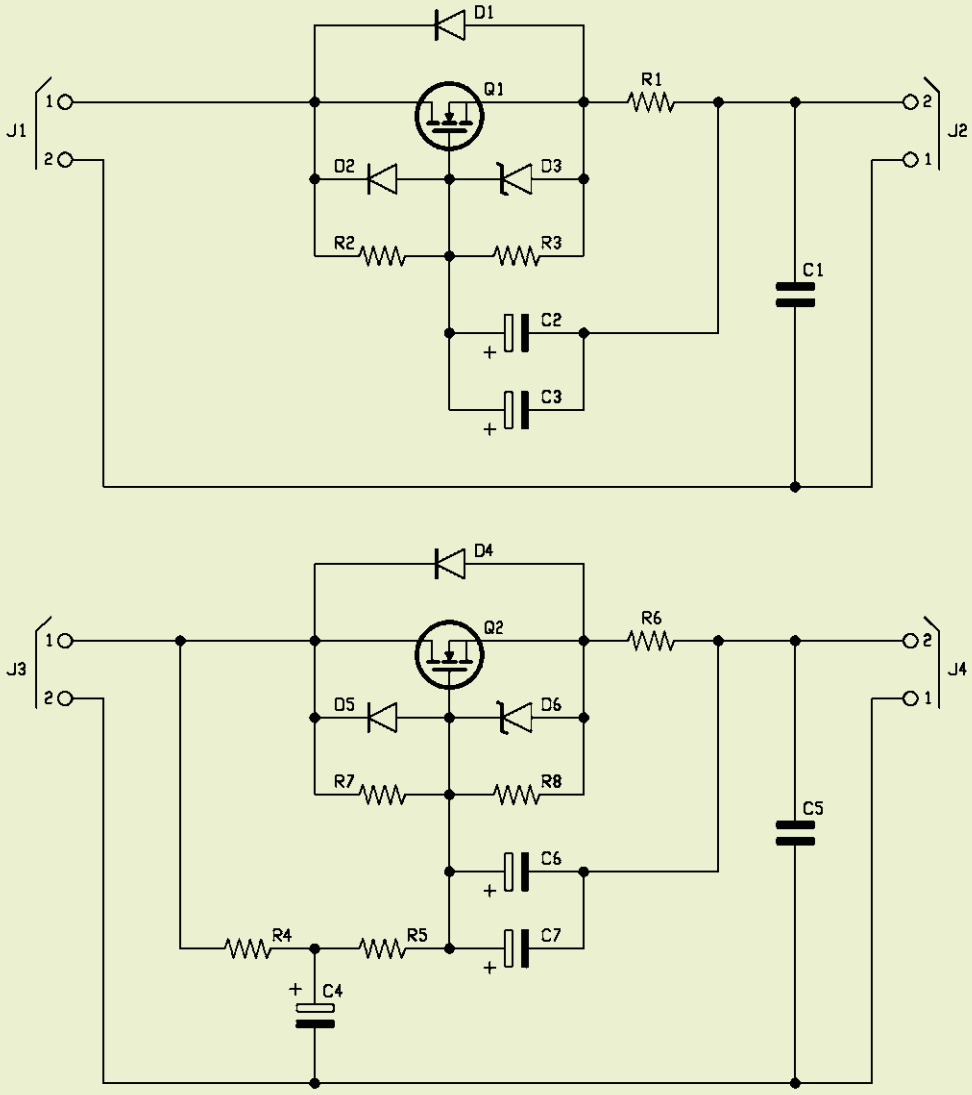
In pratica, nella porzione in alto dello schema elettrico il MOSFET Q1 è retroazionato in alternata dal parallelo dei condensatori C1 e C3, che prendono il segnale dal filtro RC passa-basso formato da R1 e C1; in quello in basso, il solito MOSFET, Q2 è retroazionato dal solito parallelo di condensatori (stavolta sono C6 e C7) ma in più il gate è accoppiato all'ingresso (che poi coincide con il drain) mediante un filtro a pi-greco composto da R4-R5-C4.
Per verificare che anche il circuito con MOSFET effettivamente si comporti come un'induttanza, è necessario risolvere le equazioni del circuito equivalente mostrato in **Fig. 3**, estraendo il valore dell'impedenza Z vista dalla porta di ingresso dove si applica la tensione V_{IN}.
Nel circuito, il MOSFET è rappresentato come un generatore di corrente, il cui valore è controllato dalla tensione presente fra gate e source V_{GS}, tramite il parametro di transconduttanza g_m caratteristico del MOSFET.
Siccome la tensione V_{GS} è condizionata in frequenza, a sua volta dal condensatore C_2, si ottiene la trasformazione di impedenza, che cambia la reattanza di tipo capacitivo in tipo induttivo, come visto nel precedente esempio dove si faceva utilizzo di un amplificatore operazionale.Per ricavare il valore dell'impedenza *Z* si consideri che:

$$Z = \frac{V_{IN}}{I_{IN}}$$

con:

$$I_{IN} = I_D + I_{R1}$$

Il circuito equivalente si può esprimere tramite le



seguenti equazioni che mettono in relazione la corrente di Drain I_D e la tensione V_{GS}, tenendo in considerazione il partitore resistivo formato dal condensatore C_2 e dalla resistenza R_2 e la corrente che scorre nella resistenza R_1.

$$\begin{cases} I_D = g_m V_{GS} \\ V_{GS} = V_{IN} \cdot \dfrac{X_{C2}}{R_2 + X_{C2}} - I_D R_1 \end{cases}$$

Sostituendo l'espressione di I_{IN}, si ottiene l'espressione dell'impedenza Z, dove si possono riconoscere i due contributi di tipo resistivo R e reattivo X_L:

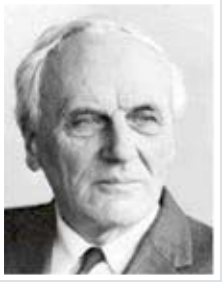
$$Z = \underbrace{\left(\frac{1}{g_m} + R_1\right)}_{R} + \underbrace{j\omega C R_2 \left(\frac{1}{g_m} + R_1\right)}_{X_L}$$

In particolare, l'induttanza è il termine che viene moltiplicato per $j\omega$ e che ha un valore determinato dalla formula:

$$L = C R_2 \left(\frac{1}{g_m} + R_1\right)$$

Bene, stabilito ciò possiamo andare a vedere come si effettua la simulazione del circuito.

L'INVENTORE DEL ... GIRATORE



Bernard D.H. Tellegen (Winschoten, 24 giugno 1900 – Eindhoven, 30 agosto 1990) è stato un ingegnere olandese specializzato in elettrotecnica.
Viene ricordato soprattutto per essere stato l'inventore del pentodo e del giratore, oltre ad aver introdotto il teorema di conservazione delle potenze virtuali, noto anche come teorema di Tellegen. Dopo essersi laureato in ingegneria elettrica all'Università di Delft nel 1923, iniziò a lavorare presso i laboratori di ricerca della Philips a Eindhoven. Nel 1926 inventò il pentodo, mentre il giratore lo sviluppò nel 1948.

Ha registrato in tutto 41 brevetti negli Stati Uniti d'America. Dal 1946 al 1966 Tellegen fu professore aggiunto di teoria dei circuiti all'Università di Delft. Dal 1942 al 1952 è stato presidente e membro onorario della Nederlands Elektronica- en Radiogeno-otschap (Società Elettronica e Radio dei Paesi Bassi). Nel 1953 l'Australian Institute of Radio Engineers conferì all'ingegnere olandese la carica di membro onorario a vita. Tellegen fu eletto membro dell'Accademia Reale delle Arti e delle Scienze dei Paesi Bassi nel 1960. Nel 1970 l'Università di Delft gli diede una laurea honoris causa. Inoltre, Tellegen è stato socio della IEEE e vinse la Medaglia Edison nel 1973 con la seguente motivazione: per una creativa carriera di risultati significativi nella teoria.

SIMULAZIONE DEL CIRCUITO

Per valutare quanto il circuito giratore riesca ad approssimare bene un'induttanza ideale è stata eseguita una simulazione nell'ambiente gratuito LTSpice, su un filtro passa-basso LC, con un valore di induttanza di 22 henry ed una corrente di lavoro di 100 mA. Nelle **Fig. 4** e **Fig. 5** sono rappresentati gli schemi simulati.
Dal diagramma di **Fig. 6** si vede come le due curve del filtro passa basso con induttanza simulata dal circuito giratore (curva in verde) e con induttanza ideale (curva in blu) siano praticamente identiche fino alla frequenza di circa 1 kHz con il picco di risonanza a 7 Hz. Con i valori utilizzati nella simulazione, la componente di frequenza a 100 Hz derivante dal raddrizzamento è attenuata di almeno 40 dB.

Utilizzando una simulazione parametrica sul valore della capacità C_1 si può valutare come cambia il comportamento della risposta in frequenza del filtro passa basso. Un'ottima strategia è quella di evitare l'uso di condensatori con capacità troppo elevate e distribuire diversi condensatori di valore più piccolo nei punti dove l'utilizzatore assorbe maggiore potenza. Ad esempio, nel caso di un amplificatore valvolare si potranno piazzare nelle vicinanze delle valvole finali.

REALIZZAZIONE PRATICA

Bene, a questo punto possiamo passare alla costruzione del circuito giratore, per il quale abbiamo disegnato un apposito circuito stampato, da noi utilizzato anche per le verifiche in laboratorio.

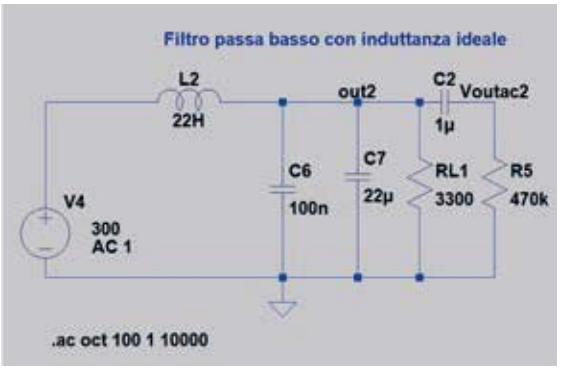


Fig. 4
Schema di simulazione di un filtro passa basso ideale con valore L=22H.i.

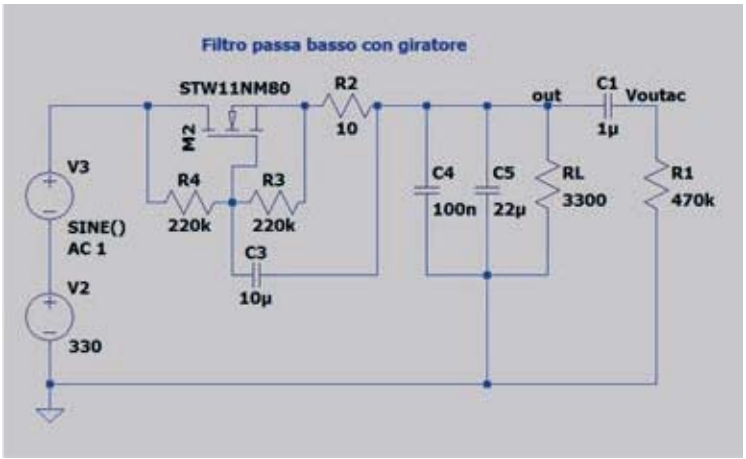
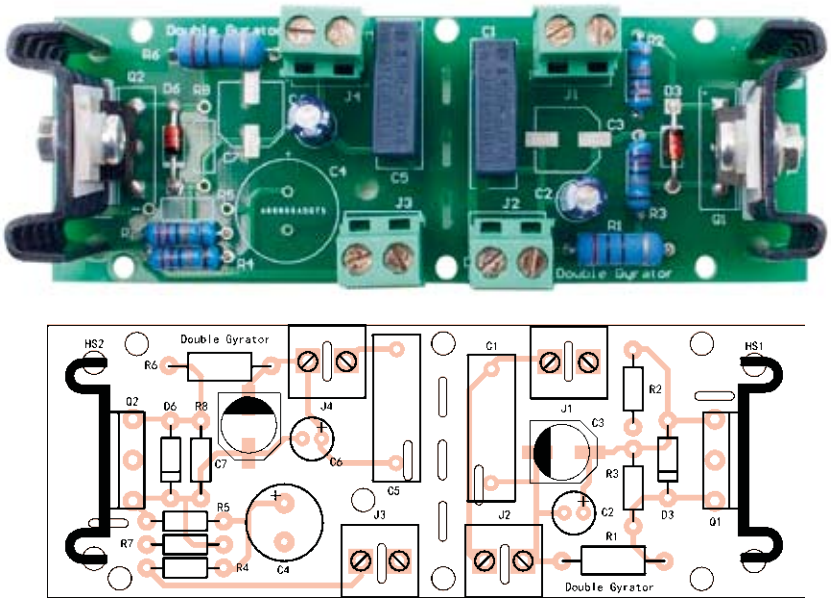


Fig. 5
Schema di simulazione di un filtro passa basso con giratore e valore di induttanza L=22H.



Elenco Componenti:

R1, R6: 10 ohm 1W
R2, R3, R7, R8: 220 kohm 1/2W
R5, R8 -
C2, C6: 10 microFarad 50 Volt elettrolitico
C1, C5: 100 nF 630 Vdc poliestere
C3, C4 -
Q1, Q2: IRF840
D1, D2, D4, D5: SM4007
D3, D6: Zener 15V 1W

Varie
- Morsetto 2 vie passo 5.08mm (4 pz.)
- Dissipatore TO 200 19°C/W (2 pz.)
- Vite 12 mm 3 MA (2 pz.)
- Dado 3 MA (2 pz.)
- Mica TO200 (2 pz.)
- Rondella isolante Nylon 3 mm (2 pz.)
- Circuito stampato S1505 (100x36 mm)

La basetta è facilmente ottenibile per fotoincisione dopo aver ottenuto le pellicole da stampe delle tracce lato rame dai file scaricabili dal nostro sito web *www.elettronicain.it*; una volta incisa e forata, si devono realizzare le poche "vie" mediante corti spezzoni di filo in rame saldati nelle piazzole di interconnessione, quindi montare i componenti iniziando dalle resistenze e dai diodi (per i quali è necessario rispettare l'orientamento indicato nel disegno di montaggio visibile in queste pagine) e proseguendo con i condensatori (rispettate la polarità degli elettrolitici) per finire con i MOSFET.
Per quanto riguarda il dimensionamento dei componenti, dopo aver stabilito la risposta in frequen-

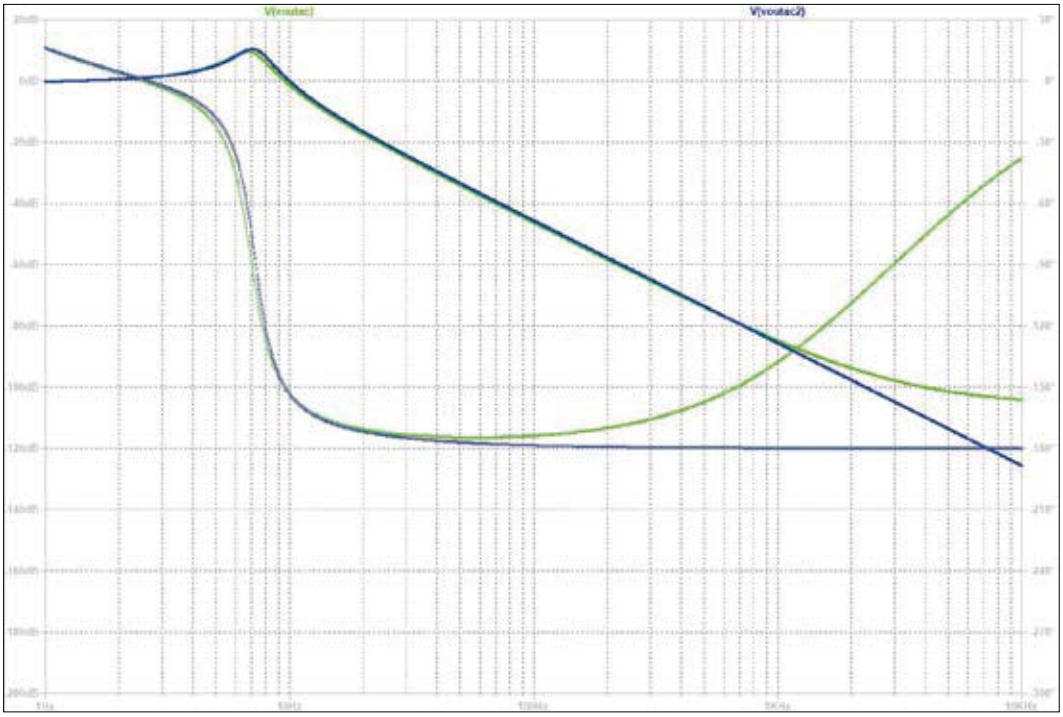
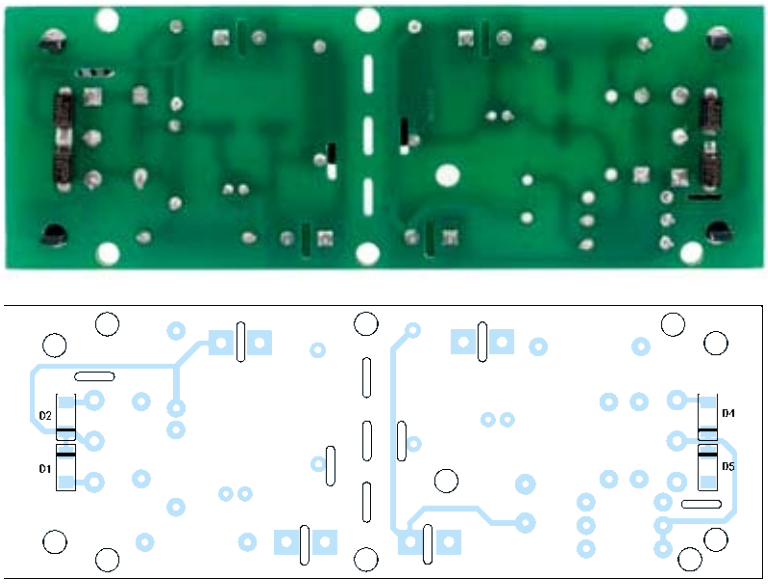


Fig. 6
Curva di risposta del filtro passa basso con circuito giratore (verde) e induttanza ideale (blu).



za del filtro passa basso realizzato con il giratore, tramite la selezione dei valori del condensatore e delle resistenze, utilizzando le formule viste in precedenza è necessario decidere la massima corrente di lavoro del circuito.
Bisogna quindi effettuare la scelta della potenza che i componenti devono gestire e per il MOSFET deve essere fatto il dimensionamento termico per il dissipatore.
Per stabilire la massima corrente di lavoro del circuito è necessario calcolare la potenza dissipata P_D dal MOSFET, considerando le differenze di temperatura e le resistenze termiche in gioco, tramite la formula:

$$P_D = \frac{T_J - T_{AMB}}{R_{T(J-c)} + R_{T(c-d)} + R_{T(d-a)}}$$

con:

- T_j temperatura di giunzione massima del MOSFET. Valore da datasheet T_j = 150°C.
- T_{AMB} = temperatura ambiente. Valore di progetto impostato a 60°C
- $R_{T(j-c)}$ = resistenza termica fra giunzione e case del MOSFET impostata a 1 °C/W dai dati del MOSFET selezionato
- $R_{T(c-d)}$ = resistenza termica fra case e dissipatore

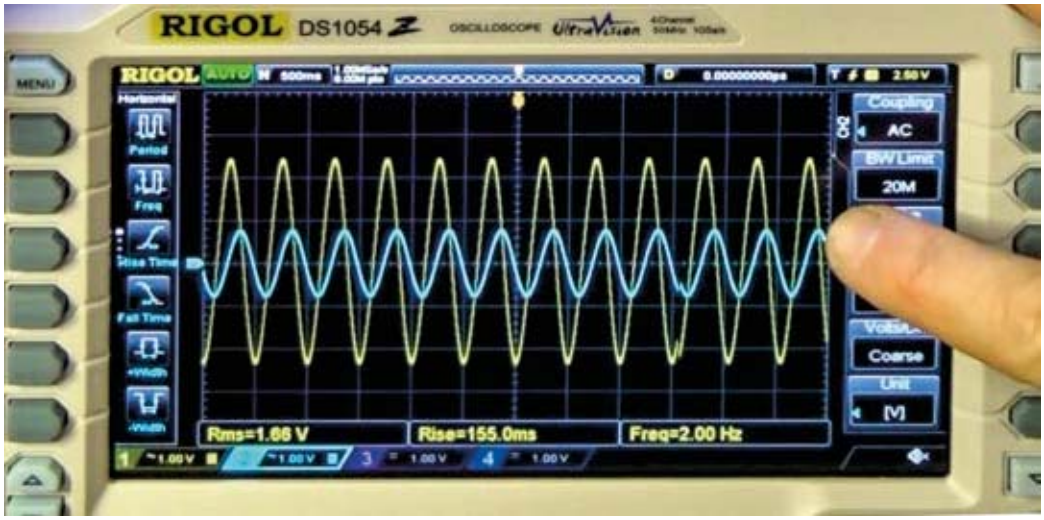


Fig. 7 Misure di attenuazione su primo prototipo alla frequenza di 2 Hz.

Fig. 8
Misure di attenuazione su primo prototipo alla frequenza di 50Hz.

impostata a 0,61 °C/W nel caso di utilizzo di un isolatore di tipo SILPAD.

- $R_{T(d-a)}$ = resistenza termica del dissipatore.

Considerando una resistenza termica del dissipatore di 19 °C /W calcoliamo la massima potenza dissipata P_D:

$$P_D = \frac{150° - 60°}{1 + 0.61 + 19} = 4.4\ W$$

Per operare un filtraggio del ripple con un margine accettabile si ipotizza una tensione di lavoro ai capi del MOSFET di 15V e quindi la massima corrente I_{MAX} vale:

$$I_{MAX} = \frac{4.4\ W}{15\ V} = 300\ mA$$

Questo è il limite superiore della corrente, considerando una temperatura di 60 °C all' interno del contenitore. Se si seleziona un dissipatore avente resistenza termica inferiore, ad esempio di 15 °C/W come per il modello KL105/50/sw, si otterrebbe una massima corrente di lavoro pari a 400mA. L'isolamento del MOSFET non è strettamente necessario viene messo solo per evitare shock elettrici toccando il dissipatore. Sul circuito stampato è previsto l' uso di due dissipatori per i MOSFET, con una resistenza termica di 19 °C/W, come ad esempio il modello Seifert KL105/29/sw.

COLLAUDO E MISURE SUI PRIMI PROTOTIPI

Per il collaudo del prototipo è stato impiegato un setup di misura composto da:

- un generatore di funzione Agilent 332210 A
- un carico attivo SIGLENT SDL1030X
- un oscilloscopio RIGOL DS 1054Z.

Sono state effettuate le misure per ricostruire la curva di risposta in frequenza sperimentale del filtro passa basso LC realizzato con il giratore. Sollecitando il filtro alle diverse frequenze si è verificata la perfetta rispondenza del filtro passa basso LC realizzato a giratore rispetto alla curva di risposta teorica. La **Fig. 7** mostra la tensione di ingresso in regime sinusoidale alla frequenza di 2Hz, con ampiezza di 1,66V RMS (curva gialla) e la tensione in uscita del filtro passa-basso (curva blu). La **Fig. 8** riporta le misure della tensione di ingresso in regime sinusoidale alla frequenza di 50Hz e la tensione in uscita del filtro passa-basso (curva blu), l'attenuazione è di oltre 20dB.

CONCLUSIONI

Il circuito giratore si presta ad essere personalizzato per adattarsi ad ogni impiego e in varie condizioni; provatelo in diverse configurazioni e vi sorprenderà.

Cosa occorre?

Il materiale utilizzato in questo progetto è di facile reperibilità. Il gerber del circuito può essere scaricato dal sito *www.elettronicain.it* nella sezione download di questa rivista e può essere realizzato con il servizio di stampa PCB *www.futuragroup4makers.com*.

Il materiale va richiesto a:

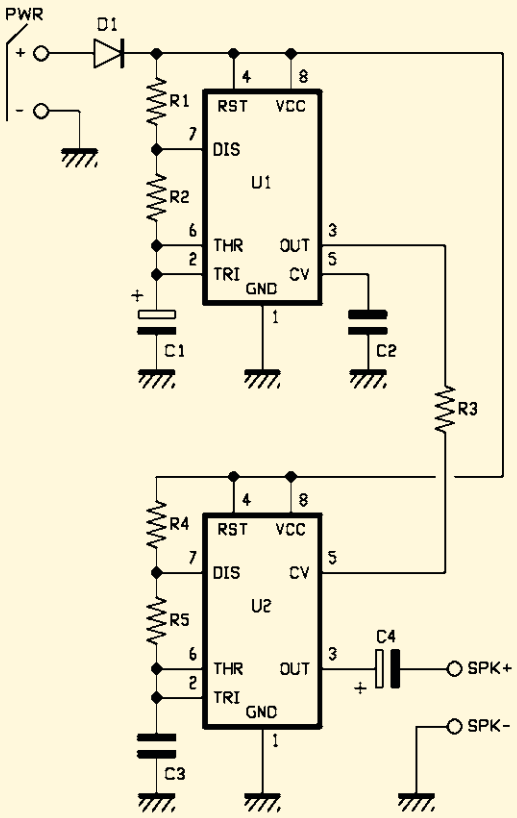
Futura Elettronica srl, Via Adige 11, 21013 Gallarate (VA)
Tel: 0331-799775 - http://www.futurashop.it

SIRENA BITONALE

di **DAVIDE SCULLINO**

Simuliamo il suono prodotto dalle sirene delle auto della polizia o dei mezzi di soccorso, utilizzando due comuni 555 configurati in cascata.

Sintetizzare dei suoni e anche delle voci, oggi, è una cosa facilissima, perché l'elettronica offre chip tecnologicamente avanzati e per di più alla portata di tutti ed anche degli sperimentatori; la sintesi può essere fatta anche sfruttando i computer. Suoni come, ad esempio, quelli delle sirene delle auto giocattolo o quelli utilizzati per colonne sonore ed effetti speciali, ma anche i versi degli animali. Eppure cose del genere non richiedono necessariamente tecnologie avanzate, prova ne è che fino a qualche anno fa la sintesi di tanti suoni e rumori si otteneva con circuiti essenziali basati su componenti discreti, porte logiche o su generatori di funzioni basilari



come il collaudatissimo NE555, che riproponiamo nel progetto di queste pagine in una configurazione canonica che permette di ottenere, nello specifico e a riprova di quanto appena affermato, il suono della sirena di molti mezzi di soccorso.
Il timer 555 è un circuito integrato capace di implementare varie funzioni basate su temporizzazioni singole (monostabile) o periodiche (multivibratore astabile) e in questo progetto ne impieghiamo due funzionanti da astabile: il primo in ordine genera la temporizzazione base e va a modulare in frequenza la forma d'onda prodotta dal secondo, ottenendo così una nota che periodicamente cambia di frequenza. Il secondo NE555 pilota un altoparlante che riprodurrà il suono risultante.

SCHEMA ELETTRICO

Analizziamo dunque il circuito che realizza la sirena e che, come vedete dal relativo schema elettrico in queste pagine, consta sostanzialmente di due integrati NE555; comprendere come funziona diventa abbastanza semplice sapendo cosa contiene l'NE555 e allo scopo è utile riferirsi alla **Fig. 1** che riporta lo schema a blocchi interno dell'integrato.
Il 555 contiene due comparatori aventi in comune un partitore di tensione multiplo (a scala) che polarizza per l'operazionale superiore l'ingresso invertente e per quello inferiore, il non-invertente; dei comparatori vengono resi accessibili dall'esterno gli ingressi non-invertente di quello superiore (i termini superiore e inferiore si riferiscono al potenziale di riferimento che ricevono dal partitore multiplo, quindi quello superiore riceve la tensione più elevata), corrispondente al piedino 6 (THLD, ossia Threshold) e l'invertente di quello superiore, corrispondente al piedino 2 (TRG, ossia Trigger).
Il nodo tra la prima e la seconda (considerate dall'alimentazione positiva) resistenza del partitore multiplo di riferimento dei due comparatori (alimentato, come il resto degli stadi interni al 555, dal piedino 8) viene portato all'esterno tramite il piedino 5 (CTRL, Control Voltage) e ciò permette di alterare le tensioni di riferimento dei comparatori, così da controllare con una tensione esterna la frequenza di lavoro nella configurazione astabile (operando lo shift di frequenza come si fa nei VCO, ossia gli oscillatori controllati in tensione) oppure la durata degli impulsi in quella da timer (monostabile). Più esattamente, nella configurazione astabile, abbassando la tensione sul piedino 5 si accorcia il periodo della commutazione dell'uscita e quindi si incrementa la frequenza generata, viceversa lasciando salire la tensione fino al livello caratteristico (a vuoto, ossia senza alcun circuito collegato al piedino 5) o incrementandola, si allunga il periodo e si riduce la frequenza prodotta. Nella configurazione monostabile, invece, la tensione altera la durata degli impulsi, il che permette, pilotando il trigger con impulsi a frequenza costante, di ottenere un modulatore PWM.
Le uscite dei comparatori entrano una nel RESET (quella del comparatore superiore) e nel SET (comparatore inferiore) di un flip-flop di tipo RS, che è un circuito logico la cui uscita diretta (Q) va a livello alto (corrispondente a circa il potenziale del piedino 8) quando il SET è a livello alto, ovvero assume lo zero logico se SET è posto a circa zero volt; l'uscita presenta un comportamento esattamente opposto quando ad essere stimolato è l'ingresso RESET, il quale, posto a livello logico alto, resetta il flip-flop, intendendo con ciò che ne azzera lo stato di Q. Il flip-flop ha anche un'uscita negata (/Q), la cui condizione logica è sempre l'inverso di quella Q; nel 555 è collegata, tramite una resistenza, alla base di un transistor NPN configurato ad open collector il cui emettitore è connesso a massa e il collettore al piedino DISCHARGE (7). Il transistor può commutare una corrente di collettore di 200 mA, utilizzato in modo sink (ad assorbimento di

corrente). L'uscita diretta, Q, è invece collegata al piedino OUT (3) dell'integrato attraverso un buffer interno push-pull, capace di erogare un massimo di 200 mA. Il negativo di alimentazione, vale a dire la massa di riferimento del 555 corrisponde al piedino 1, mentre al 4 è connesso il reset a logica invertita del flip-flop: questo piedino permette di resettare forzatamente il circuito dall'esterno applicando un livello logico basso. La massima frequenza di funzionamento del 555 configurato come stabile è 500 kHz.

Ciò detto, vediamo che nel circuito ciascun NE555 è configurato come astabile; per farvi capire cosa accade in ciascuna sezione analizziamo il funzionamento di quella facente capo a U1 partendo dalla condizione in cui C1 è scarico, ovvero presenta una tensione di 0 volt: trascurando le vicende di C2, che si carica quasi subito e quindi non disturba la tensione di riferimento applicata dal partitore multiplo ai comparatori interni al 555 (C2 serve per filtrare la tensione dei partitori e del piedino Control Voltage), la condizione iniziale del circuito vede i piedini 2 e 6 a circa zero volt e quindi l'uscita del comparatore inferiore a livello alto. Ciò determina il livello logico alto sull'ingresso SET del flip-flop e lo stesso sull'uscita Q (la /Q si porta a zero logico lasciando interdetto il transistor collegato al DISCHARGE e quindi il condensatore libero di caricarsi).

A questo punto C1 inizia a caricarsi attraverso la corrente che fluisce nella serie R1 - R2 (il pin 7 assorbe praticamente nulla), fin quando supera la tensione di soglia del comparatore inferiore, che è pari a 1/3 di quella applicata al piedino 8; ora l'uscita del suddetto comparatore si porta a zero logico e libera il SET del flip-flop, la cui uscita rimane comunque a livello alto fin quando la tensione ai capi del C1, continuando a crescere, non si porta a un valore superiore ai 2/3 della tensione di alimentazione del 555, ovvero alla soglia di commutazione del comparatore superiore.

Quando ciò accade, l'uscita del comparatore superiore si porta a livello logico alto e resetta il flip-flop, la cui uscita diretta passa a zero logico mentre la /Q si porta a livello alto, mandando in saturazione il transistor interno collegato al piedino 7, il quale attraverso R2 scarica C1.

La situazione rimane questa fin quando la tensione sui condensatori C1 e C2 non scende al disotto della soglia di commutazione del comparatore inferiore, allorché l'uscita di quest'ultimo torna a livello logico alto e setta il flip-flop, il quale riporta Q a 1 logico e /Q a zero. In questa condizione il condensatore riprende a caricarsi perché il piedino 7 torna aperto, ovvero il transistor ad esso collegato ritorna in stato di interdizione.

La carica del condensatore avviene nuovamente fino a 2/3 della tensione applicata tra i piedini 1 e 8 del 555, quindi il ciclo ricomincia.

CARATTERISTICHE TECNICHE

Tensione di alimentazione:
5÷9Vcc

Corrente assorbita:
0,5A

Funzionamento:
Bitonale

Frequenze:
350Hz - 600 Hz

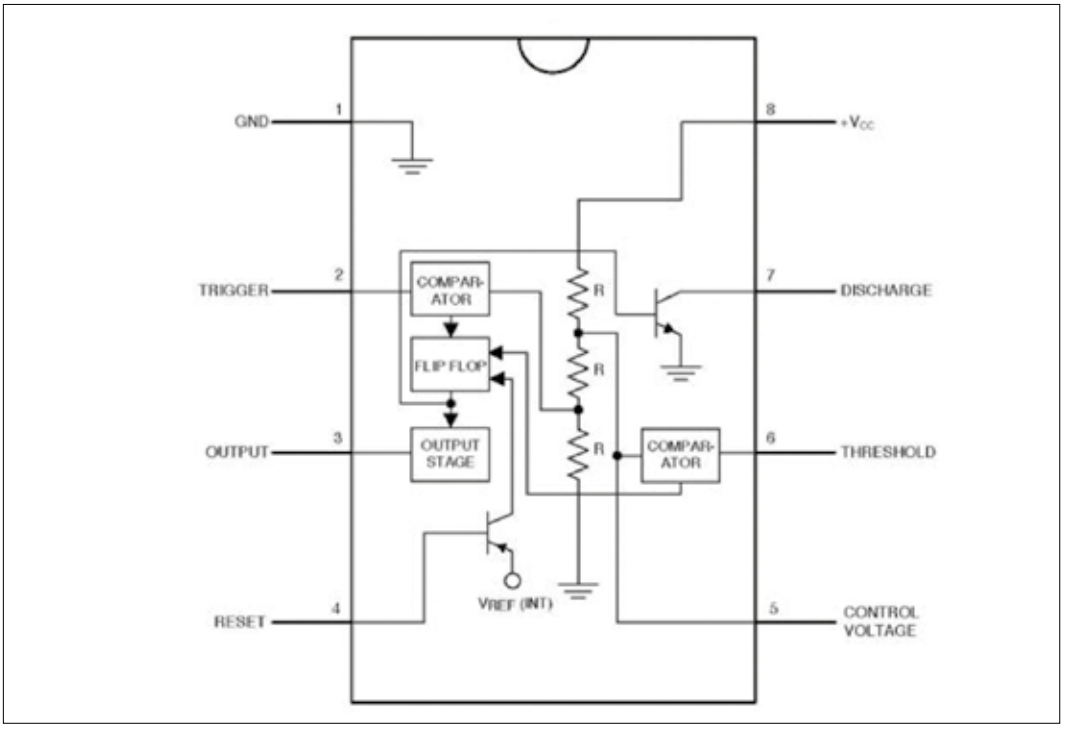
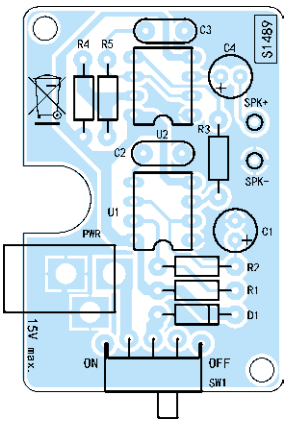


Fig. 1
Schema a blocchi del 555: la massa è il pin 1, il TRIGGER è il 2, OUT è il 3, RESET è il 4, CONTROL VOLTAGE è il 5, THRESHOLD il 6, DISCHARGE è il 7 e il +Vcc è l'8.



Elenco Componenti:

R1: 1 kohm
R2: 68 kohm
R3, R4, R5: 10 kohm
C1, C4: 10 µF 63 VL elettrolitico
C2, C3: 100 nF ceramico
SW1: Deviatore a slitta 90°
U1, U2: NE555

SPK: Altoparlante 8 ohm 0.3W

Varie
- Zoccolo 4+4 (2 pz.)
- Plug alimentazione
- Circuito stampato S1489 (35x51 mm)

Per effetto di ciò abbiamo l'uscita OUT dell'integrato che si mantiene a livello alto nel periodo che serve al C1 affinché la tensione ai loro capi passi da 1/3 a 2/3 della tensione di alimentazione (se supponiamo di alimentare il 555 a 9Vcc, il passaggio è fra 3 e 6 volt, corrispondenti rispettivamente a 1/3 e 2/3 della tensione fra i pin 8 e 1) e a livello basso (zero volt circa) nel tempo che trascorre da quando i condensatori, in fase di scarica, vedono calare la tensione ai loro capi da 2/3 della tensione di alimentazione ad 1/3.
I tempi dipendono dai valori delle resistenze R1 e R2 e sono tanto più simili quanto più la R2 è grande rispetto alla R1: infatti la carica avviene attraverso R1 e R2 mentre la scarica avviene solo attraverso R2.
La durata del periodo T è pari alla somma **th + tl** e il duty-cycle vale:

dc = th/T

La durata dell'impulso (livello alto) è pari a:

th = 0,693 x R1 x C1

Invece la durata della pausa vale:

tl = 0,693 x (R1+R2) C1

La frequenza vale 1/T e si determina con la seguente formula:

f = 1,44/ C (R1 + 2xR2)

Stabilito come funziona il 555, possiamo proseguire l'analisi del circuito e spiegare come si ottiene il suono della sirena bitonale.

IL SUONO DELLA SIRENA

Per ottenere una nota modulata e quindi il suono bitonale desiderato, il segnale fornito dal piedino OUT del primo NE555 (l'U1) pilota, attraverso la resistenza R3, il piedino CONTROL VOLTAGE (5) del secondo timer, anche questo configurato per funzionare come multivibratore astabile, seppure (come desumibile dai valori dei componenti di

Il circuito inserito nel contenitore TEKO SC700.

temporizzazione) a una frequenza ben più elevata; infatti U2 genera la nota base, mentre U1 scandisce il salto di frequenza che serve a generare il suono bitonale della sirena. La nota base, per poter essere modulata in frequenza deve, chiaramente, essere a una frequenza molto più elevata, altrimenti la cosa non funziona.
Lo spostamento di frequenza si ottiene perché portando alternativamente a livello alto (circa la tensione di alimentazione degli NE555) e a livello basso (circa 0V) l'ingresso relativo al piedino 5 dell'U2, si alterano le soglie di commutazione dei comparatori interni all'integrato e questo determina un anticipo o un ritardo nella commutazione degli stessi e del flip-flop, a parità di componenti (resistenze e condensatore) di temporizzazione. In pratica si va a operare una modulazione di frequenza a gradino sul secondo astabile, la cui uscita, mediante il condensatore elettrolitico C4 (indispensabile a bloccare la componente continua e lasciar passare solo le variazioni di tensione) va a pilotare l'altoparlante cui affidiamo il compito di riprodurre la nota modulata.
La nota viene modificata ogni 1,05 secondi, ovvero passa dalla sua frequenza più bassa (circa uguale a 600 Hz) ottenuta quando l'uscita dell'U1 è a livello alto a quella più bassa (circa uguale a 350 Hz) corrispondente a quando l'uscita dell'U2 è a zero. Si verifica, sostanzialmente, uno slittamento di frequenza ad opera dell'onda rettangolare prodotta dall'U1, la quale altera le tensioni di soglia dovute ai partitori interni all'NE555, intervenendo sul piedino 5 (nell'1 tale pin è collegato solo a un condensatore perché ci serve che le tensioni di riferimento dei comparatori siano stabili).
Concludiamo l'analisi del circuito e del suo funzionamento con la sezione di alimentazione: il tutto si alimenta con una pila o batteria, comunque con una tensione continua del valore di 5÷9V applicata fra i contatti + e – PWR; sulla linea di alimentazione che porta dal contatto + ai pin 4 e 8 dei due integrati è interposto un comune diodo al silicio che protegge il circuito dall'inversione di polarità, giacché conduce solo nel verso giusto.

REALIZZAZIONE PRATICA

Passiamo adesso alle note costruttive, premettendo che si tratta di un progetto prettamente didattico (anche se molto utile in pratica...) e quindi concepito per essere realizzato anche da chi ha poca esperienza di montaggi elettronici. partiamo dal circuito stampato, che è un singola faccia e va ottenuto per fotoincisione seguendo la traccia lato rame scaricabile dal nostro sito Internet *www.elettronicain.it*; inciso e forato il c.s. dovete iniziare il montaggio dei componenti, che in questo progetto sono tutti a montaggio tradizionale, per la gioia di chi non familiarizza con gli SMD e rimpiange l'elettronica tradizionale.
Iniziate con le resistenze, il diodo al silicio D1 e i condensatori, quindi procedendo con gli integrati 555, che dovete montare preferibilmente su uno zoccolo dip 4+4 contatti (in tal caso inserirete gli integrati solo dopo aver saldato tutti gli altri componenti).
Per l'orientamento dei componenti polarizzati (integrati e diodo) fate riferimento al disegno di montaggio visibile nella pagina qui accanto; se i 555 li montate su zoccolo, per aiutarvi fate in modo da saldare gli zoccoli con già la tacca di riferimento sul loro corpo orientata come mostrano nel predetto disegno di montaggio.
Completato il tutto potete saldare nelle piazzole +/- PWR i fili di una presa a strappo per pile, in modo da facilitare l'applicazione dell'alimentazione; connettete anche l'altoparlante alle piazzole SPK+ ed SPK- (rammentando che ne serve uno da 8÷16 ohm di impedenza e da circa 300 mW di potenza).
Fatto questo avrete quindi un circuito pronto all'uso e funzionante, giacché non è richiesta alcuna regolazione o taratura preliminare. Per l'alimentazione potete utilizzare sia la classica pila da 9 volt, sia un alimentatore in continua da 6÷9Vcc - 0,5A.
Il nostro generatore del suono della sirena bitonale potrà essere utilizzato per mixarlo in una colonna sonora di un filmato, per campionarlo, ma anche per integrarlo in un giocattolo (ad esempio una macchinina della polizia più o meno grande) o in un plastico. Gli impieghi sono comunque molti e limitati praticamente dalla vostra fantasia.

Cosa occorre?

I componenti principali utilizzati in questo progetto sono disponibili presso Futura Elettronica. Il timer NE555 (cod. 555) è disponibile a Euro 0,40, il contenitore Teko SC700 (cod. SC-700) è in vendita a Euro 1,30. Il gerber del circuito può essere scaricato dal sito *www.elettronicain.it* nella sezione download di questa rivista e può essere realizzato con il servizio di stampa PCB *www.futuragroup4makers.com*.

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

IL PIÙ VENDUTO

COD. DSO72
€ 325,00

Disponibile anche nella versione a 200MHz COD. DSO202 **€ 519,00**

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

COD. DSOFG102 **€ 420,00**

- ✓ 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. DSOFG202 **€ 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. PSO2020
€ 95,00

Oscilloscopio USB per PC 2 Canali 20 MHz

- ✓ 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

COD. DSOPC22 **€ 94,00**

Disponibile anche nella versione a 100MHz COD. DSOPC102 **€ 325,00**

Generatore di funzioni e forme d'onda arbitrarie

€ 206,00
COD. FAW1025G

USCITA FORME D'ONDA ARBITRARIE
- ✓ Frequenza onda DC~25 MHz
- ✓ 1 canale
- ✓ Frequenza di campionamento 200 MSa/s
- ✓ Risoluzione verticale 12 bit

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

€ 278,00 COD. TPO073

- ✓ 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)

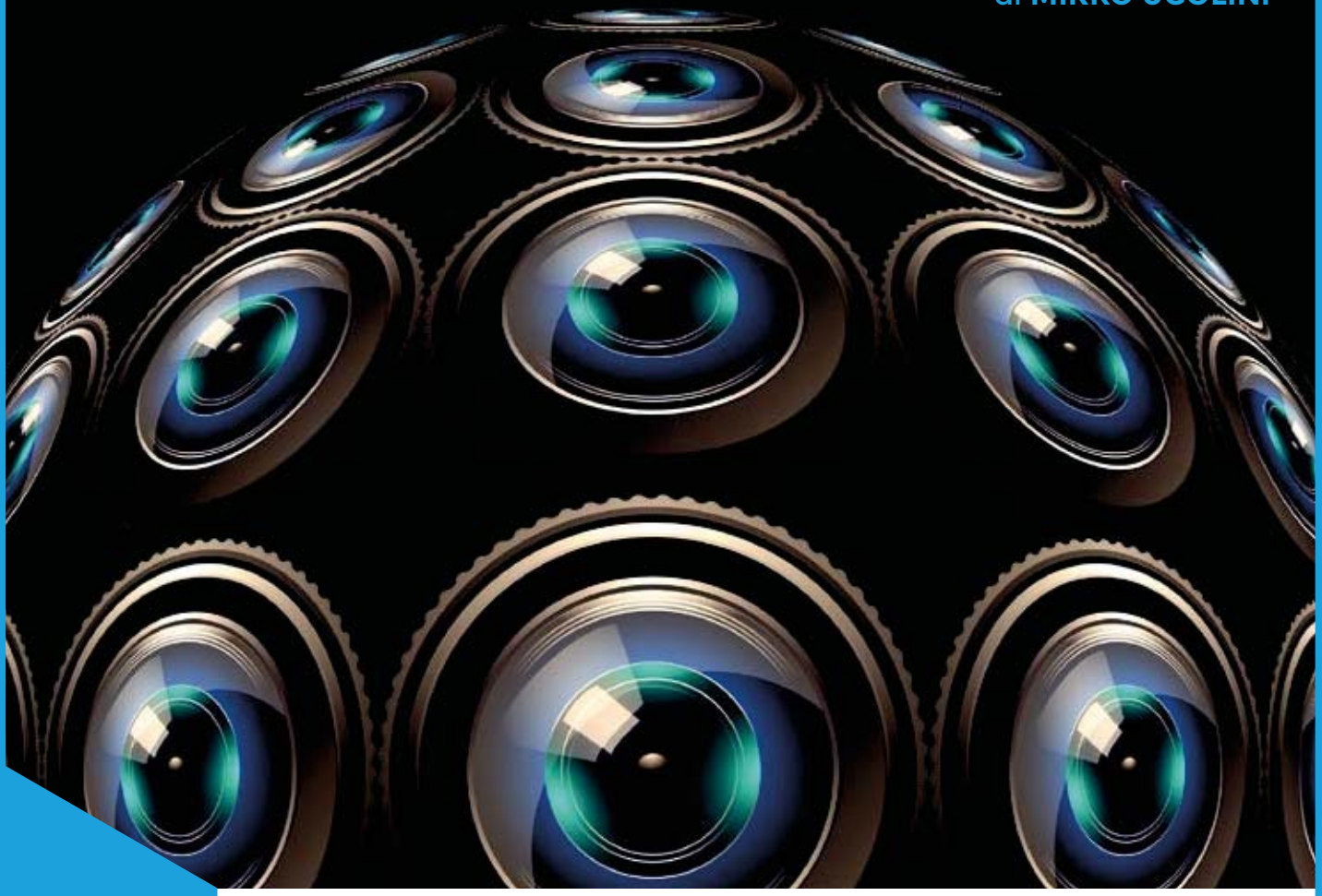
Prezzi IVA inclusa.

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

Caratteristiche tecniche e vendita on-line su: **www.futurashop.it**

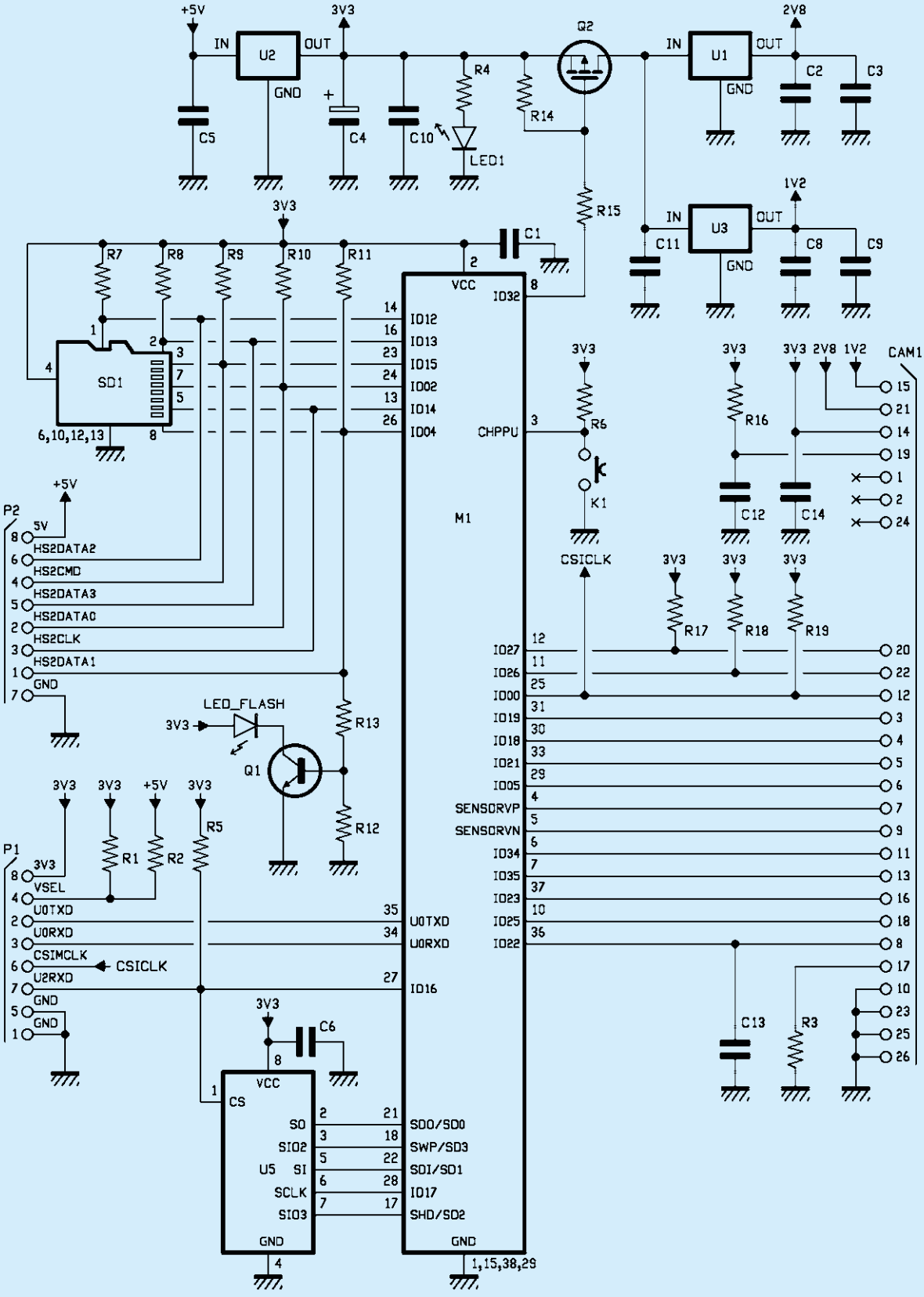
ESP32-CAM

di MIRKO UGOLINI



Realizziamo una microcamera wireless con tanto di controllo PAN/TILT, utilizzando il modulo ESP32 e una telecamera da 2Mpx.

L'ESP32-CAM Development Board è una scheda elettronica basata su un SoC (System On Chip) ESP32S, equipaggiata con una videocamera da 2MPixel (OV2640) e uno slot per una memoria miniSD fino a 4GByte.
Sulla scheda è presente un LED ad alta luminosità che può essere impiegato come flash o come illuminatore per la scena da riprendere e alcune linee GPIO per interfacciarsi con il mondo esterno.
L'elettronica permette di impostare diversi frame-rate, ma per ottenere dei risultati accettabili bisogna arrivare, come vedremo dettagliatamete nel corso dell'articolo, ad un compromesso. Con questo modulo (**Fig. 1**) realizzeremo un'applicazione sfruttando le librerie che la Espressif ha da poco rilasciato per la programmazione con l'IDE di Arduino; prima occorreva il sistema di sviluppo (toolchain) dedicato.


+5V
3V3
2V8
1V2
U2
U1
U3
IN
OUT
GND
C1
C2
C3
C4
C5
C8
C9
C10
C11
R4
R14
R15
Q2
LED1
SD1
R7
R8
R9
R10
R11
M1
VCC
IO32
IO12
IO13
IO15
IO02
IO14
IO04
CHPPU
R6
K1
CSICLK
R16
R17
R18
R19
C12
C14
CAM1
P1
P2
5V
HS2DATA2
HS2CMD
HS2DATA3
HS2DATA0
HS2CLK
HS2DATA1
VSEL
U0TXD
U0RXD
CSIMCLK
U2RXD
LED_FLASH
Q1
R1
R2
R5
R12
R13
C6
U5
CS
SO
SIO2
SI
SCLK
SIO3
SDO/SD0
SWP/SD3
SDI/SD1
IO17
SHD/SD2
IO27
IO26
IO00
IO19
IO18
IO21
IO05
SENSORVP
SENSORVN
IO34
IO35
IO23
IO25
IO22
C13
R3

Elenco Componenti:

R1: 0 ohm
R2, R7: -
R3, R4: 1 kohm
R5, R6: 10 kohm
R10, R11: 47 kohm
R12: 10 kohm
R13, R15: 1 kohm
R14, R19: 10 kohm
R8, R9: 47 kohm
R16, R17, R18: 4,7 kohm
C1: 100 nF ceramico
C2: 10 µF ceramico
C3: 100 nF ceramico
C4: 100 µF tantalio
C5: 10 µF ceramico
C6: 1 µF ceramico
C7, C9, C12, C14: 100 nF ceramico
C8, C10, C11: 10 µF ceramico
C13: 15 pF ceramico
K1: Pulsante NO
LED_FLASH: LED bianco
LED1: LED verde
Q1: S8050
Q2: P-MOS
SD1: Slot micro SD-Card
CAM1: Connettore 24 vie per camera
M1: ESP32S
U1: XC6206-2.8V
U2: AMS1117-33
U3: XC6206-1.2V
U5: PSRAM
- Camera OV2640
- Strip maschio 8 vie (2 pz.)
- Circuito stampato

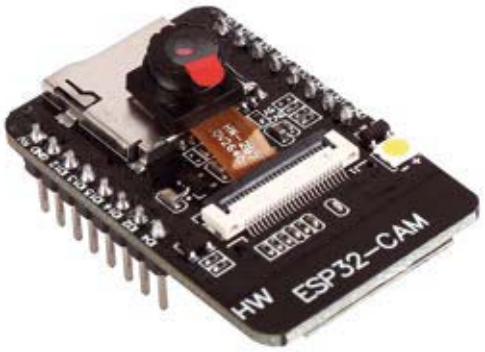


Fig. 1 La scheda ESP32-CAM completa di telecamera.

SCHEMA ELETTRICO ESP32-CAM

Abbiamo ricavato e pubblichiamo lo schema elettrico del modulo, che descriviamo per comprenderne il funzionamento. Il modulo ESP32-CAM non è dotato di porta USB per ragioni di costo e dimensioni, quindi per poter caricare il nostro Sketch sul modulo dovremo dotarci di un convertitore USB/TTL che fornisca l'alimentazione a 5V e permetta di impostare i livelli logici di RX e TX a 3,3 V. Ricordiamo che il modulo SOC ESP32 non è 5V tolerant, quindi i segnali da applicare ai pin GPIO devono essere tassativamente tutti nei limite dei 3,3V.
Nello schema sono presenti tre regolatori lineari: un AMS1117-3.3 che si occupa di convertire i 5V provenienti dall'alimentazione esterna ai 3,3V necessari per il modulo ESP32S; a valle è presente un MOSFET comandato dal segnale CAM_POWER il quale controlla (tramite la libreria Espressif) l'applicazione dei 3,3V ad altri due regolatori, uno da 2,8V ed uno da 1,2V (usati entrambi per alimentare la videocamera). Notiamo inoltre la presenza di un chip PSRAM impiegato per poter utilizzare il modulo con delle risoluzioni più elevate.
È presente anche uno slot per la miniSD che può essere utilizzato per memorizzare immagini acquisite dalla camera. Per il funzionamento della microSD

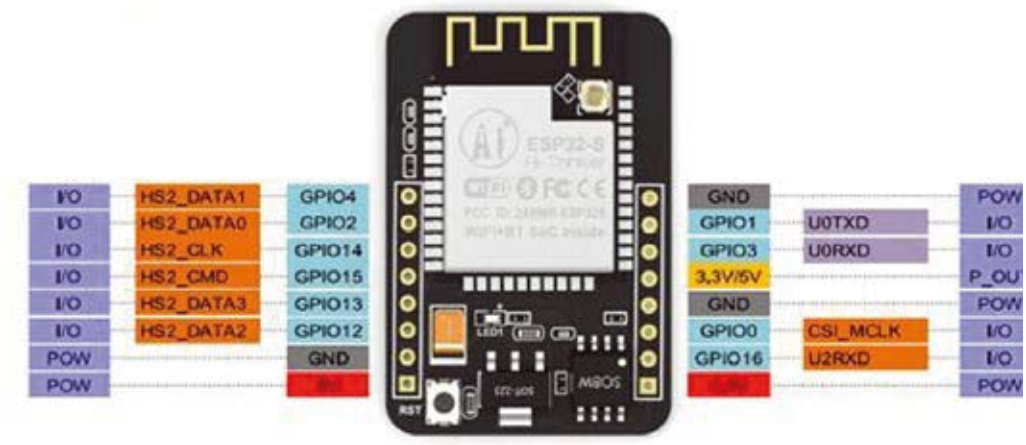
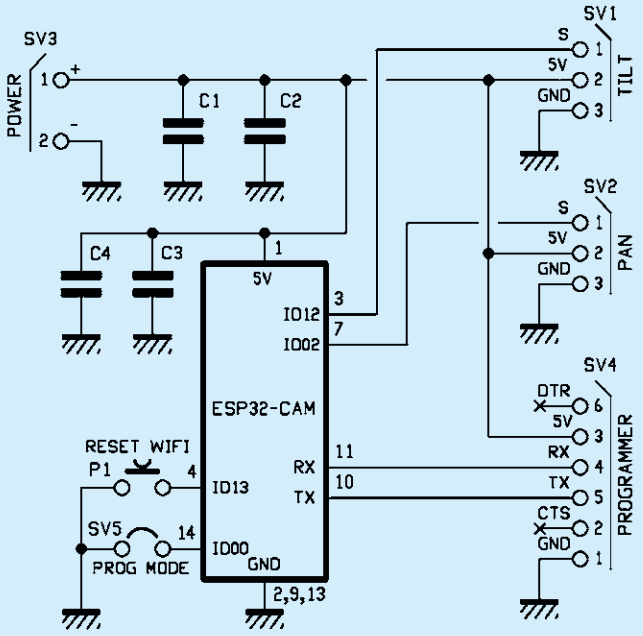


Fig. 2 Pin-out del modulo ESP32-CAM.



servono diversi segnali, alcuni dei quali sono esposti sul connettore del modulo ESP32-CAM, ma se impiegati per la card non si possono utilizzare per altro.
Quanto ai LED, il primo (indicato come LED) è collegato al pin GPIO33 e si trova sul lato opposto rispetto alla camera; impegna un pin che non è neppure esposto esternamente.
L'altro LED è posizionato frontalmente ed è ad alta luminosità (LED_FLASH) gestito dal GPIO4 e controllato tramite un transistore NPN (Q1). Il GPIO33 tuttavia è condiviso (HS2_DATA1) con la scheda microSD e quindi lampeggerà mentre la SD sarà in attività. E possibile utilizzare questo LED per illuminare la scena e lo useremo nel nostro progetto.
A bordo del modulo ESP32-CAM è presente un pulsante di reset del ESP32S collegato a pin E32_RST.

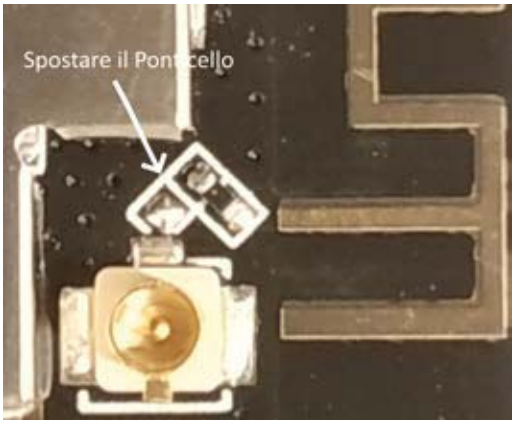


Fig. 3 Ponticello da fare.

Purtroppo questo pin non è stato reso accessibile dall'esterno.
I due pin U0TXD (TX) e U0RXD (RX) che ci permettono di caricare il nostro codice sul modulo ESP32-CAM sono, invece, esposti sul connettore esterno, sul quale è presente anche una seconda seriale U2RXD (RX) utilizzabile nel caso in cui serva ricevere dati tramite seriale. La piedinatura completa del modulo è illustrata nella **Fig. 2**.
A bordo si trova un'antenna WiFi integrata nel PCB, ma c'è la possibilità di collegare un'antenna esterna tramite un micro connettore IPEX (U.FL) che si trova sul lato inferiore del PCB, in prossimità dell'antenna integrata (**Fig. 3**). Per attivare l'antenna esterna è necessario spostare un ponticello realizzato con un resistore SMD da 0 Ohm che deve essere spostato dal collegamento con l'antenna integrata a quello con il connettore IPEX.
Questa operazione (viste le dimensioni del resistore SMD e delle piazzole) è molto delicata, quindi se non avete mano ferma, una lente di ingrandimento ed un buon saldatore con punta sottile, evitatela.
Per l'alimentazione si applica una tensione di 5V all'ingresso 5V, che fa capo al regolatore AMS1117. Notate che il pin 4 (P_OUT) è marcato come 3,3V/5V, infatti nello schema elettrico potete notare due resistenze da 0 ohm collegate tra il pin 4 e la tensione 5V e 3,3V.
Naturalmente sarà possibile collegare solo un resistore alla volta per definire quale sia la tensione presente sul pin, pena un cortocircuito tra le due tensioni e la distruzione del SoC.
Questa è una modifica che difficilmente dovrete fare, ma è bene comprendere lo schema. Normalmente il resistore presente è R1, quindi sul P_OUT è presente la tensione a valle del regolatore AMS1117, ossia 3,3V.
Il segnale di reset della microcamera è derivato da una rete RC composta da R16 e R12 e quindi la camera si resetta automaticamente nella fase di applicazione della tensione di alimentazione.
Potrebbe succedere che la porta del nostro PC non sia in grado di alimentare in modo corretto il modulo stesso e quindi una volta caricato lo sketch è preferibile rimuovere il collegamento con il PC e alimentare il modulo tramite un alimentatore che possa erogare 5V con una corrente massima di almeno 1A (un buon caricatore USB per mobile è più che sufficiente) oppure tramite un PowerBank.
Quello che può accadere alimentando il modulo dalla porta USB è che, a causa della poca corrente erogabile, presenti un comportamenti inaspettato con ripetuti riavvii, blocchi o malfunzionamenti.

IL PROGETTO

Ci proponiamo di realizzare con il modulo ESP32-CAM una piccolissima webcam che può essere utilizzata per il monitoraggio remoto, eventualmente mossa da un sistema PAN-TILT. L'immagine può essere vista in tempo reale in un browser web e quindi anche da un dispositivo mobile; tramite il mouse oppure in modo touch è possibile orientare la webcam. È anche possibile integrare la camera in progetti di domotica come Home Assistant *https://www.home-assistant.io*.
Naturalmente si possono realizzare delle webcam molto più performanti tramite Raspberry Pi Zero W, ma il "lampone" richiede dei tempi di boot sicuramente maggiori, soffre di corruzione del file-system (nel caso di improvvisa interruzione ripetuta dell'alimentazione) e sicuramente il costo di un modulo ESP32-CAM non è paragonabile con quello della Raspberry Pi Zero.
Il modulo ESP32-CAM costa poco, non ha un file-system su SD che può corrompersi, ha un tempo di avvio molto basso: insomma, è il candidato perfetto per realizzare il nostro progetto.
Inoltre, ha ulteriori funzionalità che qui non abbiamo implementato, come il riconoscimento facciale (offline), la possibilità di riconoscere QR-Code.
Per utilizzare in modo agevole il modulo abbiamo realizzato una scheda base che può essere utilizzata per programmare e alimentare l'ESP32-CAM.
Come mostra lo schema elettrico, sulla nostra scheda abbiamo predisposto i due connettori femmina per innestare il modulo, il connettore per il collegamento del programmatore tramite USB, un jumper per settare la modalità programmazione, dei condensatori di disaccoppiamento e due connettori per comandare i due servomotori per il controllo PAN e TILT della camera. Abbiamo previsto anche un pulsante per il reset della configurazione WiFi che non useremo in questo progetto ma che riguarda la modalità di configurazione del WiFi tramite SmartConfig.

IL MODULO ESP32-CAM NELL'IDE ARDUINO

Per programmare l'ESP32-CAM occorre installare il core dei moduli SoC ESP32; allo scopo consigliamo di utilizzare una versione recente dell'IDE di Arduino: diciamo almeno una 1.8.x.
Apriamo quindi il menu File e clicchiamo su

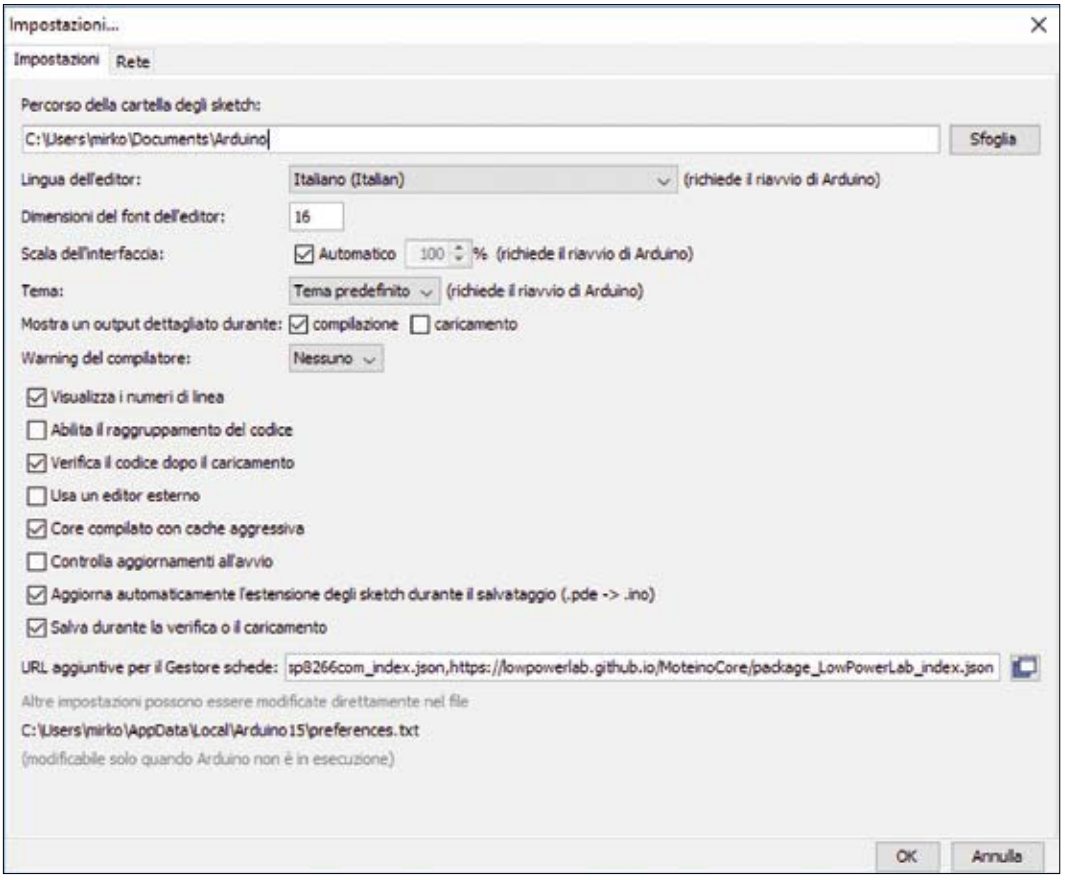


Fig. 4 Finestra di dialogo Impostazioni.

Fig. 5 Inserimento dell'URL aggiuntiva per caricare il modulo ESP32-CAM.

Digitiamo ESP32 nella casella di testo e dopo una breve attesa dovrebbe apparire la voce visibile in **Fig. 6**; selezioniamo la versione più recente del pacchetto (nel nostro caso 1.0.3) e clicchiamo su **Installa**. Una barra di progressione apparirà per informarci sull'avanzare dell'installazione, al termine della quale potremo riavviare L'IDE e alla prossima apertura troveremo una nuova serie di schede

Fig. 6 Caricamento della ESP32-CAM.

Impostazioni (**Fig. 4**) poi individuiamo la casella di testo "**URL aggiuntive per il gestore di schede**" e clicchiamo sul pulsante a destra per aprire una finestra (**Fig. 5**) nella quale inserire l'url di definizione dei core SoC fornito da **Espressif** *https://dl.espressif.com/dl/package_esp32_index.json* ed inseriamolo (aggiungendolo ad altri eventualmente presenti) nella casella di testo e salviamo con OK.
Riavviamo quindi L'IDE di Arduino e poi dal menu Strumenti selezioniamo la voce **Gestore schede**.

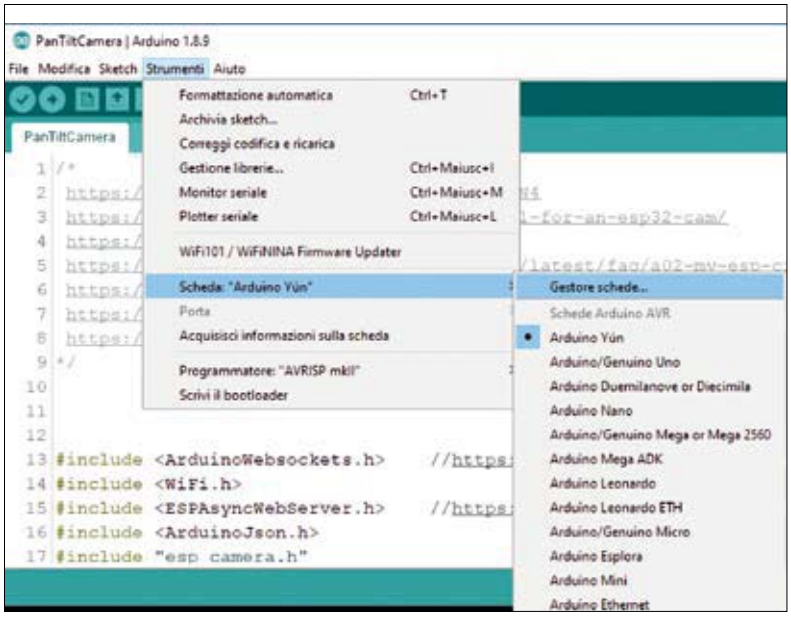


Fig. 7 Scelta della scheda.

basate sul chip SoC ESP32 installate sul nostro ambiente di sviluppo (**Fig. 7**). Il modulo che dobbiamo selezionare per il progetto corrente dovrà essere "**ESP32 Wrover Module**", che adesso sarà visualizzato nel sotto-menu.
Per il progetto di questo articolo avremo necessità di aggiungere le seguenti librerie aggiuntive:

- **ArduinoWebsockets**: *https://github.com/gilmaimon/ArduinoWebsockets*
- **ESPAsyncWebServer**: *https://github.com/me-no-dev/ESPAsyncWebServer*
- **ArduinoJson**: *https://github.com/bblanchon/ArduinoJson*
- **AsyncTCP**: *https://github.com/me-no-dev/AsyncTCP*

Nel caso le abbiate già presenti nel vostro ambiente, assicuratevi di aggiornarle all'ultima versione. Per inserire le librerie, utilizziamo lo strumento **Gestione Librerie** dell'IDE per **ArduinoWebsockets** (**Fig. 8**) e **ArduinoJson** (**Fig. 9**) mentre per **ESPAsyncWebServer** (non essendo tuttora disponibile nel repository di Arduino) dovremo inserirla a mano nella cartella **Libraries** di Arduino.
Dovremo quindi scaricare il file zippato **ESPAsyncWebServer-master.zip** dal rispettivo repository GitHub, scompattarle la cartella, rimuovere il suffisso **–master** dal nome della cartella e spostare la stessa dentro la cartella che solitamente si trova in: **C:\Program Files(x86)\Arduino\libraries** a questo punto siamo pronti per testare il codice.

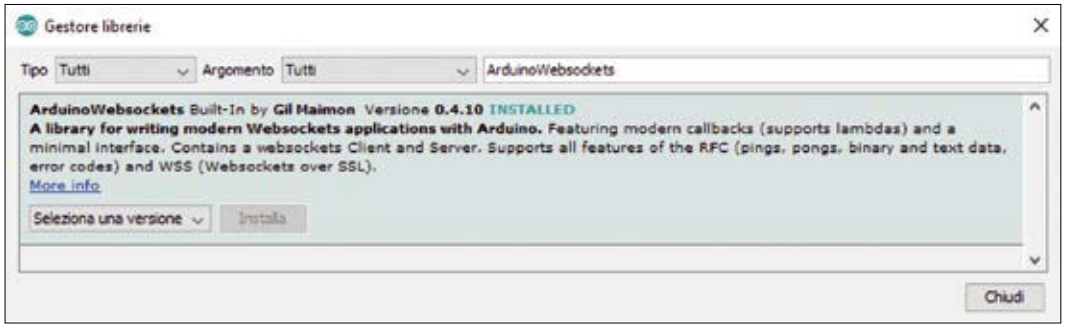


Fig. 8 Gestore librerie.

Vi anticipiamo che in questo articolo non useremo lo schetch **CameraWebServer.ino** di esempio che solitamente viene usato per presentare il funzionamento del modulo ESP32-CAM che si può trovare normalmente nel percorso *C:\Users**NomeUtente**\AppData\Local\Arduino15\packages\esp32\hardware\esp32\1.0.3\libraries\ESP32\examples\Camera\CameraWebServer.*
Lo sketch standard è molto generico, contiene del codice per gestire sia la camera **ov2640** che **ov3660** in diverse risoluzioni e modalità di funzionamento, permettendo anche di sfruttare il riconoscimento facciale. Però è carente da altri punti di vista: per esempio non permette di gestire l'accensione del LED ad alta luminosità dall'interfaccia web integrata e presenta qualche instabilità del funzionamento. Per brevità di spazio non possiamo descrivere anche questo sketch, che vi invitiamo comunque a installare per testare altre funzionalità interessanti del modulo ESP32-CAM; qui utilizzeremo una versione realizzata da noi anche nell'interfaccia, che utilizza sempre la libreria base di Espressif **esp_camera.h**. ma per il resto il funzionamento è completamente diverso. Non riporteremo lo sketch completo ma solo le parti salienti: nel **Listato 1** vediamo la parte di importazione delle librerie. Il cuore della nostra applicazione è chiaramente la libreria **esp_camera.h** resa disponibile da Espressif che espone il funzionamento della scheda ESP32-CAM. Le altre librerie degne di nota sono **ESPAsyncWebServer.h** un webserver che gestisce le chiamate asincrone, poi a seguire i **ArduinoWebsockets** per utilizzare i WebSocket, la Libreria **WiFi.h** per la gestione del Wi-Fi, **ArduinoJson.h** per interpretare i comandi provenienti dalla pagina Web. Mentre la libreria **ESPmDNS.h** permette di implementare il protocollo mDNS (ZeroConfig); in breve, sarà possibile assegnare al dispositivo un nome che potrà essere utilizzato in alternativa all'indirizzo IP assegnato dal Router tramite servizio DHCP. Vedremo poi come configurare e utilizzare questa caratteristica.
Le ultime due inclusioni **index.html.h** e **error404.html.h** in effetti non sono delle librerie ma dei file Html che devono trovarsi nella stessa cartella dello schetch e che sono la versione compressa (Gzipped) e codificata in binario della interfaccia web

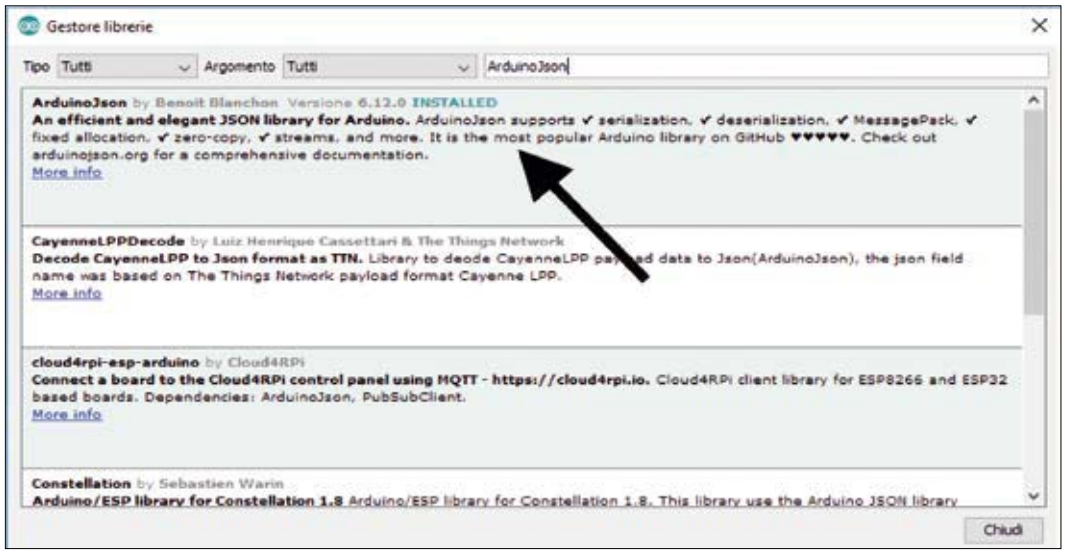


Fig. 9 Installazione ArduinoJson.

Listato 1

```
#include <ESPAsyncWebServer.h>
#include <ArduinoWebsockets.h>
#include <ArduinoJson.h>
#include <WiFi.h>
#include <ESPmDNS.h>
#include “esp_camera.h”

#include “soc/soc.h”
#include “soc/rtc_cntl_reg.h”

#include “index.html.h”
#include “error404.html.h”
```

Listato 2

```
const char* ssid          = “Nome_Rete”;
const char* password      = “Password_Rete”;
const char*  mDNS_NAME    =  “esp32-cam”;
#define LED               4
#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
#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
camera_fb_t * fb = NULL;
using namespace websockets;
WebsocketsServer WSserver;
AsyncWebServer webserver(80);
int n_attempts;
```

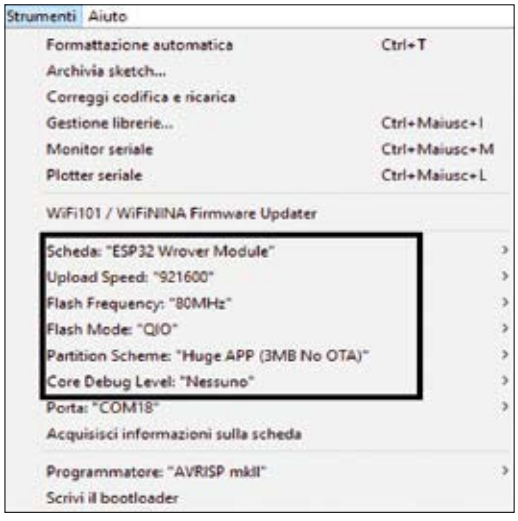


Fig. 10
Impostazioni per compilare il codice.

che abbiamo realizzato per il progetto.
Nel **Listato 2** vediamo la definizione della connessione WiFi a cui il dispositivo ESP32-CAM dovrà collegarsi, quindi sostituite il valore di “Nome_Rete” e “Password_Rete” con quelle della vostra rete WiFi. Poi abbiamo il nome (“esp32-cam”) con cui vogliamo identificare il dispositivo tramite servizio **mDNS**. A seguire viene elencata una serie di parametri legati ai pin GPIO che definiscono la configurazione della scheda e dei collegamenti SoC ESP32-S con la videocamera e miniSD. Questi parametri non devono essere alterati altrimenti lo sketch non funzionerà. Nelle ultime righe vediamo la definizione degli oggetti Frame Buffer della camera **camera_fb, WebSocket** e del server web Asincrono, notiamo anche la definizione della porta canonica (80) del server web.
Nel **Listato 3** è possibile vedere la sezione di setup del codice Arduino. La prima istruzione disabilita il ***Brown_Out***, in pratica rimuove il controllo sulla sotto-alimentazione del chip SoC che potrebbe condurre al reset lo stesso se la tensione di alimentazione accidentalmente scendesse sotto un certo livello. Questo per prevenire reset accidentali nel caso in cui la tensione di alimentazione subisse anche per poco tempo delle piccole variazioni. A seguire abbiamo l’inizializzazione della porta seriale, la configurazione come del GPIO del LED ad alta intensità, la configurazione di due porte GPIO come pseudo PWM per il comando dei due servocontrolli PAN e TILT.
Poi abbiamo la definizione della variabile interna di configurazione della camera che dovrà essere valorizzata assegnando i rispettivi valori assegnati ai parametri definiti precedentemente e di cui abbiamo omesso la trascrizione per esigenze di spazio. Viene poi verificato se l’inizializzazione della camera nel caso in cui questo non succeda, viene inviato un messaggio di debug sulla seriale e vien resettato dopo 500 ms il chip per re inizializzarlo.
Se tutto è andato a buon fine, viene settato il ***Frame Size*** della camera. Poi abbiamo l’inizializzazione con i parametri di accesso della connessione Wi-Fi. Qui abbiamo un Loop ogni 500 ms che viene terminato quando la connessione è stata completata con successo oppure dopo 5 tentativi viene eseguito il Restart del SoC. Se la connessione è terminata con successo viene inviata sulla porta seriale un messaggio indicando anche L’indirizzo IP acquisito. Troviamo poi l’inizializzazione del servizio mDNS. Subito dopo viene inizializzato il server web asincrono che a fronte di una richiesta della Homepage (“/”) invia il contenuto del file memorizzato in binario nella variabile “index_html_gz” specificando che

Listato 3

```
void setup() {
 WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);
 Serial.begin(115200);
 pinMode(LED, OUTPUT);
 ledcSetup(2, 50, 16);
 ledcAttachPin(2, 2);
 ledcSetup(4, 50, 16);
 ledcAttachPin(12, 4);
 camera_config_t config;
config.ledc_channel = LEDC_CHANNEL_0;

 esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
     Serial.printf("Camera init failed with error 0x%x", err);
     delay(500);
      ESP.restart();
  }
  sensor_t * s = esp_camera_sensor_get();
  s->set_framesize(s, FRAMESIZE_QVGA);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
    n_attempts++;
    if (n_attempts > 5) {
        Serial.println("Restarting");
        ESP.restart();
    }
  }
  Serial.println("");
  Serial.print("Connected to ");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
 if (!MDNS.begin("esp32")) {
      Serial.println("Error setting up MDNS responder!");
         while(1) {
           delay(1000);
      }
  }
  Serial.println("mDNS responder started");
  webserver.on("/", HTTP_GET, [](AsyncWebServerRequest * request) {
    AsyncWebServerResponse *response = request->beginResponse_P(200, "text/html", index_html_gz,
sizeof(index_html_gz));
     response->addHeader("Content-Encoding", "gzip");
     request->send(response);
  });
  webserver.begin();
  WSserver.listen(82);
 MDNS.addService("http", "tcp", 80);
}
```

si tratta di un contenuto Gzipped. Al termine della procedura di setup troviamo l'inizializzazione del Webserver Asincrono e del WebSocket sulla porta **82**. L'ultimo blocco di codice Specifica il tipo di servizio, il protocollo e la porta per la risoluzione del nome del dispositivo tramite mDNS.
Nel **Listato 4** vediamo la procedura contenuta nel Loop del codice di Arduino. Seguendo l'ordine del listato abbiamo la definizione del Client WebSocket, la definizione della procedura **handle_message** che si occupa di interpretare i comandi ricevuti dal browser tramite il canale WebSocket. E poi il loop vero e proprio che fintanto che il client è disponibile esegue il ciclo acquisendo il frame dalla camera tramite la funzione **esp_camera_fb_get()**; ed inviandoli (in formato binario) sempre tramite il canale WebSocket (che ricordiamo essere bidirezionale) al browser Web che si occuperà di aggiornare l'immagine della camera. Tutto questo con poche righe di codice.
Nel **Listato 5** abbiamo la procedura che si occupa di interpretare i comandi ricevuti tramite il canale WebSocket provenienti dalla interfaccia del browser web e che abbiamo "agganciato" nella precedente

Listato 4

```
void loop() {
  auto client = WSserver.accept();
  client.onMessage(handle_message);
  while (client.available()) {
    client.poll();
    fb = esp_camera_fb_get();
    client.sendBinary((const char *)fb->buf, fb->len);
    esp_camera_fb_return(fb);
    fb = NULL;
  }
}
```

Listato 5

```
void handle_message(WebsocketsMessage msg) {
String PinName;
  int PinValue;
  int panValue ;
  int tiltValue;
  String message= msg.data();
  StaticJsonDocument<200> doc;
  DeserializationError error = deserializeJson(doc, message);
  if (error) {
     Serial.println("deserialization Failed");
     return;
  }
  Serial.println("deserialization OK");
  if(message.startsWith("{xy:")){
    Serial.println("XY");
    panValue = doc["xy"][0];
    tiltValue = doc["xy"][1];
    Serial.print("pan:");
    Serial.print(panValue);
    Serial.print("tilt:");
    Serial.print(tiltValue);
    panValue = map(panValue, -90, 90, 0, 180); // 0-180
    tiltValue = map(tiltValue, -90, 90, 180, 0); // 0-180 reversed
    ledcAnalogWrite(2, panValue);
    ledcAnalogWrite(4, tiltValue);
  }
  if(message.startsWith("{pin:")){
    Serial.println("PIN");
    const char* pin = doc["pin"];
    PinValue =  doc["value"];
    Serial.println("PinName:");
    Serial.println(PinName);
    Serial.println("PinValue:");
    Serial.println(PinValue);
    digitalWrite(LED, PinValue);
  }
}
```

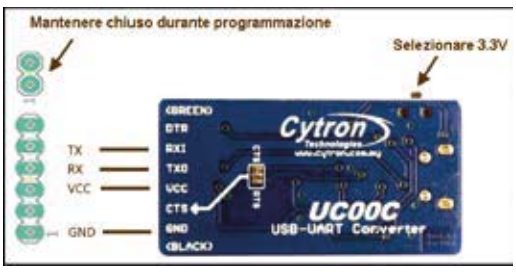


Fig. 11
Il modulo convertitore.

sezione di Loop. In particolare abbiamo l'interpretazione dei comandi PAN e TILT che servono ad orientare la posizione della camera e il controllo della accensione del LED ad alta luminosità che possiamo accendere per illuminare la scena da riprendere se non vi è sufficiente illuminazione ambientale.
Nell'interfaccia web abbiamo predisposto altri controlli web oltre al PAN, TILT e accensione del LED; se volete estendere le funzionalità dovete modificare la funzione **handle_message.**
Notate che il formato dei dati ricevuti tramite il canale WebSocket è JSON; se non lo conoscete, potete approfondire all'indirizzo web *https://arduinojson.org/*. Nel **Listato 6** trovate il dettaglio della funzione **ledcAnalogWrite ()** che permette di implementare una sorta di **PWM** sul modulo ESP32-CAM; il controllo dei servomotori avviene tramite un segnale PWM.

CARICHIAMO LO SKETCH SUL MODULO

Come primo passo compiliamo il codice: clicchiamo su Strumenti nel menu e settiamo le varie voci evidenziate come in **Fig. 10** e la porta seriale assegnata.
Chiaramente dovremo poi mantenere le stesse impostazioni quando avvieremo il processo di caricamento. Terminata la configurazione cliccate sul pulsante con il segno di spunta. Se il processo di compilazione si è concluso con successo, possiamo passare al caricamento dello sketch. Per caricare lo sketch sul modulo, dobbiamo per prima cosa collegare (prestando attenzione alla piedinatura e alle impostazioni dei livelli di tensione) il convertitore USB/seriale alla porta USB del nostro PC e al connettore che abbiamo predisposto sulla piccola motherboard; quindi occorre settare in modalità programmazione il modulo chiudendo con un ponticello il connettore SV5, che fa capo al GPIO0.
Il modulo convertitore USB/seriale, che proponiamo nella **Fig. 11**, ha una pin-out che si adatta al connettore della nostra scheda, ma non è scontato che sia sempre così; quindi se avete un convertitore diverso prestate attenzione.
Poi dobbiamo cliccare sul pulsante di upload dell'IDE di Arduino e dopo qualche secondo pigiare il pulsante di reset che si trova sotto il modulo.
Se tutto procede in modo corretto vedremo apparire nella parte inferiore della IDE dei messaggi di log che ci informano dello stato del caricamento. Terminato il quale saremo invitati a cliccare nuovamente sul pulsante di reset, dobbiamo però prima rimuovere il ponticello sul GPIO0 che setta la modalità d programmazione.

Listato 6

```
void ledcAnalogWrite(uint8_t channel, uint32_t value, uint32_t valueMax = 180) {
  uint32_t duty = (8191 / valueMax) * min(value, valueMax);
  ledcWrite(channel, duty);
}
```

Rimosso il ponticello, apriamo il monitor seriale e pigiamo il pulsante di reset: dopo qualche secondo dovrebbe apparire un messaggio di log, come quello visibile nella **Fig. 12**.
L'ultima riga del messaggio ci informa che possiamo aprire il browser utilizzando l'indirizzo IP (naturalmente il vostro indirizzo IP sarà diverso da quello indicato nell'esempio) oppure impiegare un indirizzo letterale molto più semplice e non soggetto a modifiche dovute ad eventuali variazioni di assegnazione dell'IP da parte del DHCP del Router.

L'INTERFACCIA WEB DELLA CAMERA

Una volta che il modulo ha terminato la procedu-

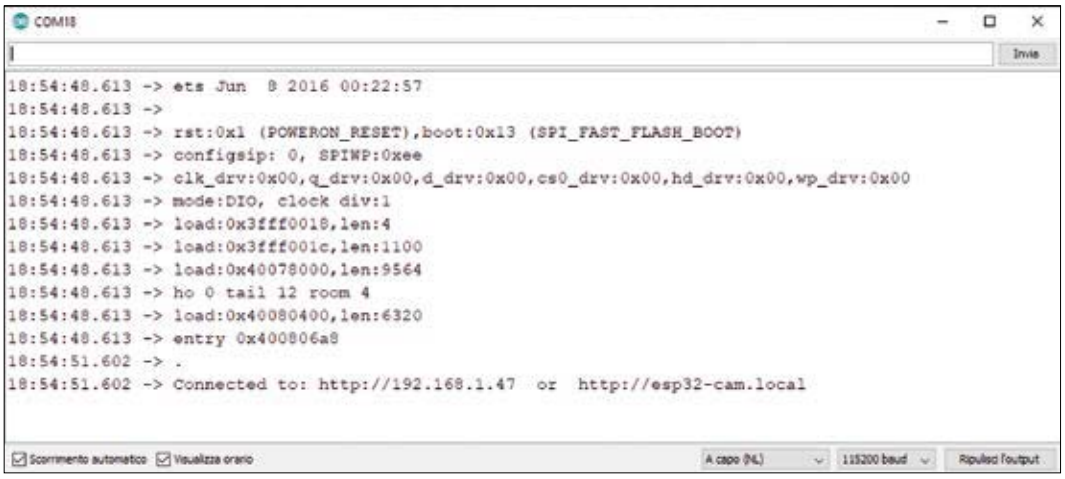


Fig. 12 Messaggio di log.

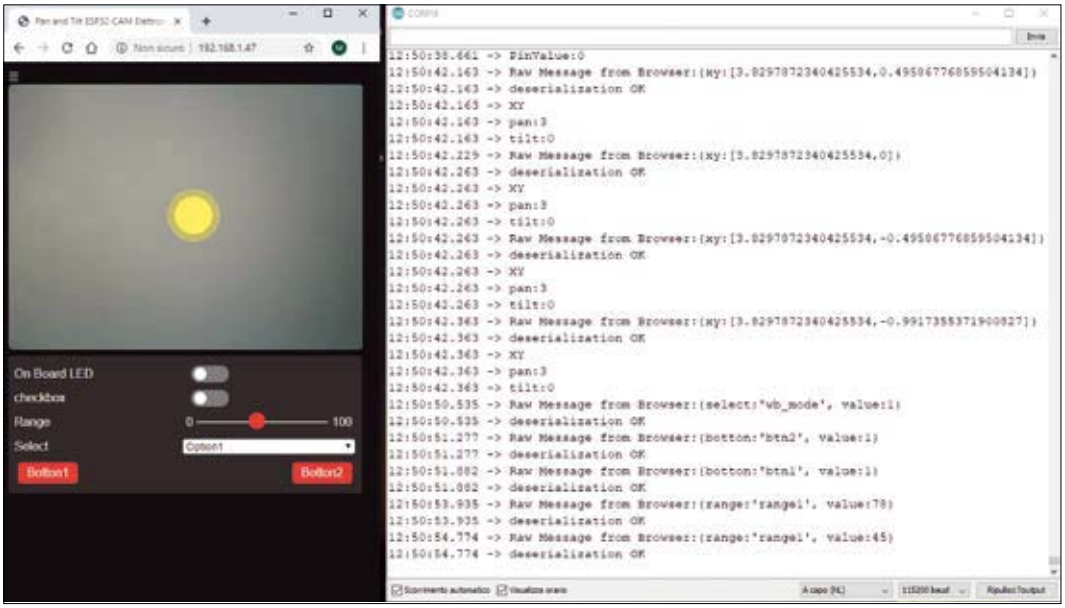


Fig. 13 Interfaccia web e schermata di Serial Monitor.

Listato 7

```
<nav id="menu">
<div class="input-group" id="flash-group">
    <label for="flash">On Board LED</label>
     <div class="switch">
     <input id="flash" type="checkbox" class="default-action" onclick="sentCheckbox(this.id,this.value);">
      <label class="slider" for="flash"></label>
 </div>
</div>
<div class="input-group" id="vflip-group">
    <label for="checkbox1">checkbox</label>
     <div class="switch">
     <input id="checkbox1" type="checkbox" class="default-action" oninput="sentCheckbox(this.id,this.
value);">
     <label class="slider" for="checkbox1"></label>
     </div>
</div>
<div class="input-group" id="saturation-group">
    <label for="saturation">Range</label>
    <div class="range-min">0</div>
    <input type="range" id="range1" min="0" max="100" value="0" class="default-action"
oninput="sentRange(this.id,this.value);">
     <div class="range-max">100</div>
</div>
<div class="input-group" id="wb_mode-group">
   <label for="wb_mode">Select</label>
   <select id="wb_mode" class="default-action" oninput="sentSelect(this.id,this.value);">
       <option value="0" selected="selected">Auto</option>
        <option value="1">Option1</option>
        <option value="2">Option2</option>
    </select>
</div>
<section id="buttons">
    <button id="btn1" onclick="setBotton(this.id);">Botton1</button>
    <button id="btn2" onclick="setBotton(this.id);">Botton2</button>
</section>
</nav>
```

ra di avvio, potremo quindi accedere alla pagina web inserendo sul browser l'url (ad esempio *http://192.168.1.47*) e ci apparirà istantaneamente la scena inquadrata dalla fotocamera (**Fig. 13**). Oppure, grazie al server mDNS potremmo usare *http://esp32-cam.local.* Se utilizzate un router **Fritzbox** è probabile che non sia possibile risolvere il nome del dispositivo con suffisso **local**, in tal caso dovreste potrete raggiungere il dispositivo con il seguente url: *http://esp32-cam.*

Se clicchiamo con il mouse sulla superficie della immagine ripresa, ci apparirà il cursore di controllo di **TILT** e **PAN** che è possibile trascinare sulla superficie per muovere l'inquadratura della camera. Se rilasciamo il cursore, dopo qualche secondo il controllo **TILT** e **PAN** scomparirà per lasciare spazio alla inquadratura, e riapparirà nell'ultima posizione lasciata se clicchiamo nuovamente sulla superficie. Nella zona inferiore della camera abbiamo un piccolo pannello di controllo.

Tuttavia l'unico controllo che attualmente è funzionante è quello di controllo del LED di bordo. Abbiamo lasciato gli altri per darvi la possibilità di sperimentare in modo semplice con la vostra scheda ESP32-CAM. Nel seguito dell'articolo vi spiegheremo anche come modificare l'interfaccia web per esempio per lasciare il solo controllo del LED. In **Fig. 13** abbiamo affiancato a destra dell'interfaccia Web la finestra di debug di Arduino per vedere l'avvicendarsi dei comandi che vengono inviati mentre controlliamo la posizione del PAN e TILT sulla immagine della camera oppure quando accendiamo e spegniamo il LED di bordo. Potrete provare ad usare anche gli altri controlli, vedrete che raggiungeranno il modulo ESP32-CAM, verranno "parsate" in modo corretto dal parser JSON, ma non sortiranno alcun effetto. Per gestire questi ulteriori comandi dovrete intervenire nella procedura **handle_message** dello sketch Arduino. Nel bordo superiore sinistro della camera è presente un pulsante a sandwich in funzionamento toggle che permette di nascondere o visualizzare il pannello di controllo inferiore che

all'apertura della pagina web che risulta di default nascosto. Va fatta una precisazione per quanto riguarda la fruibilità della interfaccia web. Non è possibile usufruire della stessa da più di un browser web contemporaneamente. L'ESP32 non è così potente da poter gestire client concorrenti, ma il lavoro che fa questo piccolo SoC è veramente strabiliante.

MODIFICARE IL FILE DELL'INTERFACCIA WEB

La pagina **index.html** che implementa l'interfaccia web della nostra camera è troppo grande (circa 500 righe di codice) per essere riportato integralmente in queste pagine; riporteremo nel **Listato 7** solo la parte riguardante il pannello di controllo. Se vorrete cimentarvi nella modifica è consigliabile aprire lo stesso non con semplice editor di testo ma è preferibile un editor HTML che vi assista nella fase di editing del codice e vi evidenzi eventuali errori. Il **Listato 7** riporta la parte della interfaccia web della camera che si trova in basso e che rappresenta il pannello di controllo. Nel listato abbiamo riportato diversi controlli. Ma se volete lasciare solo quello relativo al LED (blocco in grassetto) niente di più semplice. Rimuovete la parte di codice HTML relativa ai controlli che non desiderate e poi seguite la procedura che riportiamo nel prossimo paragrafo per rigenerare dal file index.html il file index.html.h da includere nello sketch Arduino. Il SoC ESP32 per poter renderizzare una pagina Web ha necessità che la stessa sia inserita nel codice di Arduino come una array di Byte. Quindi quello che bisognerebbe fare partendo dal file HTML della pagina che vogliamo visualizzare è prima di tutto sottoporre il file ad un processo di compressione tramite Gzip, poi convertire il risultato della compressione in un array di Byte. Per agevolarvi nella procedura, abbiamo realizzato una piccola utility in C# che permette con pochi clic di ottenere il risultato desiderato. Troverete l'eseguibile (FileConverter.exe) nella stessa cartella del codice Arduino insieme con il file index.html.

Aprite l'eseguibile e, nella finestra di dialogo che appare, cliccate sul pulsante Gzip, selezionate il file da convertire. Questo primo passo genererà un file index.html.gz. Cliccate sul pulsante "Convert to ByteArray", selezionate il file Gzip appena creato ed otterrete nella stessa cartella il file index.html.h che lo sketch del nostro progetto si aspetta di trovare nella propria cartella.

Come potete verificare dalla **Fig. 14,** il file index.html.h in effetti è stato convertito in un array di Byte della lunghezza (in questo caso) di 3.638 elementi.

Per agevolare chi voglia modificare l'interfaccia Web della camera, abbiamo predisposto due server web realizzati in NodeJS. Il primo server web è quello che si occupa di "inviare" la pagina index.html verso il

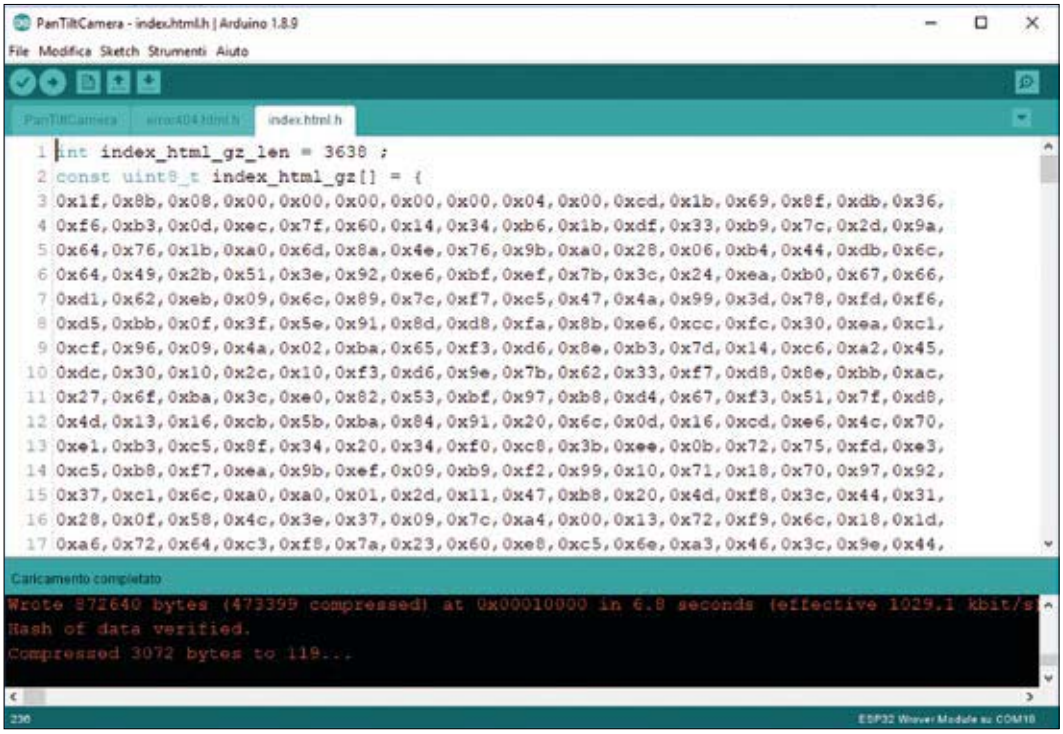
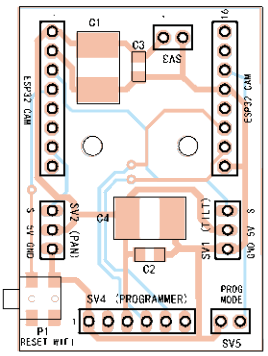


Fig. 14
File convertito in un array.



Elenco Componenti:

C1: 220 µF multistrato
C2, C3: 100 nF ceramico
C4: 220 µF multistrato
P1: Pulsante da CS
- Strip maschio 2 vie (2 pz.)
- Strip maschio 3 vie (2 pz.)
- Strip maschio 6 vie
- Strip femmina 8 vie (2 pz.)
- Jumper
- Circuito stampato S1502

browser. Il secondo server è un server Winsock che si occupa di ricevere i comandi dalla pagina Web. Non abbiamo previsto l'invio di immagini verso la pagina web per simulare la camera, ma riteniamo che sia sufficiente l'ambiente che abbiamo predisposto per potersi cimentare sulla modifica della pagina web della camera e verificare che non ci siano errori Javascript prima di trasformare il file html per caricarlo sul modulo ESP32-CAM. Nella cartella del progetto troverete una cartella NodeJS entro la quale sono presenti due sottocartelle: "server-html" e "server-winsock" all'interno della cartella "server-html" troverete una ulteriore cartella "www" all'interno della quale è presente il file index.html che potrete modificare a vostro piacimento e che poi una volta testato potrà essere convertito come indicato nei paragrafi precedenti. Il codice dei rispettivi server si trova nei file "server.js" delle rispettive cartelle. Se andate a curiosare troverete che i due server sono realizzati con una decine di righe di codice... Magia di NodeJS. Per mandare in esecuzione i due server sarà sufficiente aprire le rispettive cartelle e cliccare sui due file "run.bat", l'esecuzione dei quali aprirà due finestre shell-DOS. Se le due finestre DOS non riportano errori ma si presentano come in **Fig. 15**. Potrete aprire il browser all'indirizzo 127.0.0.1 (localhost) per visualizzare l'interfaccia web della Webcam. Le cartelle dei due server contengono già le librerie NodeJS che servono alla loro implementazione. Se non avete mai installato NodeJS sul vostro PC dovrete installarlo. La procedura è molto semplice. Potrete trovare un interessante tutorial alla pagina seguente: *https://www.mrwebmaster.it/javascript/introduzione-node-js_12432.html*.

COME FUNZIONA

Sul codice che abbiamo installato sul modulo ESP32-CAM coesistono sia un server Web che è in grado di fornire ad un browser una pagina HTML, sia un server Winsock che si occupa di ricevere dei comandi dalla pagina Web della camera che è stata aperta sul browser, sia inviare dal modulo ESP32 verso il browser web il flusso binario relativo alle immagini che vanno a comporre il video in tempo reale della ripresa. Tutto questo senza dover ricaricare il contenuto della pagina web. Ricordiamo infatti che la comunicazione Websocket è bidirezionale. Nel codice Javascript abbiamo inserito una funzione che verifica lo stato della connessione Winsock e nel caso questa sia per qualche motivo interrotta la ripristina mantenendo nel tempo la funzionalità di visualizzazione e controllo della pagina Web. Come potrete vedere analizzando il codice HTML della pagina **index.html** non sono presenti immagini e non sono stati utilizzate librerie particolari ma puro Javascript e del CSS per assegnare l'aspetto grafico agli elementi della pagina. La maggior parte del codice HTML presente nella pagina index.htm è relativo alle funzioni Javascript per instaurare la connessione Winsock, gestire la posizione del controllo PAN e TILT e inviare i dati tramite Winsock verso

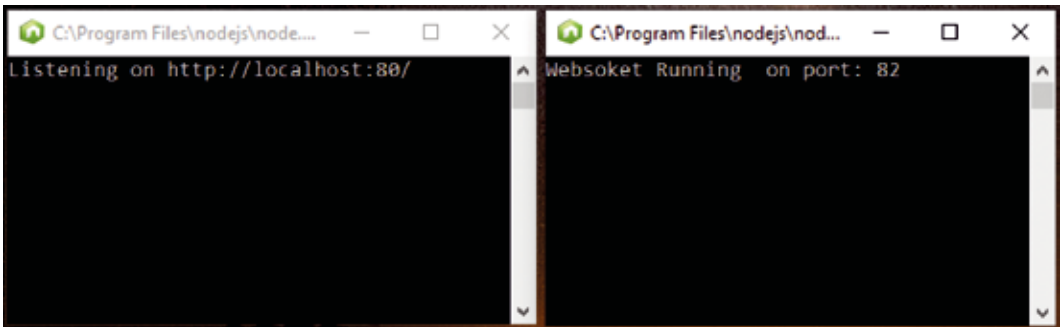


Fig. 15
Finestre DOS.

il modulo ESP32-CAM. Si noti anche il fatto che i comandi inviati tramite Websocket verso il modulo ESP32-CAM sono nel formato Json. Per esempio il comando relativo alla posizione Pan e Tilt viene inviato tramite la stringa Json: **'{xy:[' + panDegrees + ',' + tiltDegrees +']}';** che viene poi parsato tramite la ormai famosa libreria Json.Arduino lato ESP32.

COLLEGHIAMO I SERVO ALLA SCHEDA

Se vogliamo utilizzare la camera con il controllo PAN e TILT è necessario acquistare un dispositivo meccanico (**Fig. 16**) che contenga due piccoli servo e che permetta lo spostamento orizzontale (PAN) e verticale (TILT) della camera. Tale dispositivo è comunemente detto "brandeggio".
Normalmente il pinout dei servomotori è quello indicato in **Fig. 16** dove il pin rosso deve essere collegato ai +5V il marrone alla GND, mentre il pin arancio è il segnale di controllo PWM. Per il collegamento dei cavetti dei servomotori è preferibile rimuovere i connettori e saldare direttamente gli stessi sul PCB della motherboard, assicurandovi di aver lasciato un lunghezza dei cavi che permettano il movimento completo della camera senza ostacolare lo stesso.

REALIZZAZIONE PRATICA:

Passiamo ora alla costruzione pratica: è necessario ottenere il PCB scaricando il file da *www.elettronicain.it* e ricavando da esso la pellicola necessaria alla fotoincisione. Il PCB è a doppia ramatura ed essendo i componenti utilizzati di tipo SMD, la realizzazione pratica richiede un minimo di esperienza e una certa attrezzatura, vale a dire:

- Una stazione saldante che abbia un potenza almeno di 40÷50W. Una punta che, contrariamente a quello che si potrebbe essere portati a pensare non è necessario sia sottilissima. La cosa importante e che sia in buono stato (non ossidata) e che abbia la parte terminale corta, questo consente di trasferire velocemente calore dalla punta alla piazzola ed evita la realizzazione di saldature fredde.
- Della lega di stagno (preferibile Sn-Ag) con flussante, da 0.5 mm e 1 mm.
- Una pinzetta per prelevare e posizionare i componenti.
- Una trecciola dissaldante. Molto importante nel caso in cui vengano a crearsi dei ponticelli tra i pin più ravvicinati dei componenti.
- Una lente di ingrandimento.
- Per ultimo ma non meno importante... tanta pazienza ed una mano salda.

La camera OV2640 presente sul modulo ESP32-CAM normalmente è già collegata al connettore e assicurata con del nastro biadesivo alla parte metallica dello slot micrSD. Per evitare di danneggiarla, non flettete in modo ripetuto il sottile connettore flessibile di cui è dotata e sincerateVi che sia fissata con il biadesivo. Al solito, iniziate il montaggio dai componenti a basso profilo e proseguite in ordine di altezza. Per chi non se la sente di realizzare da sè il modulo Motherboard ESP32-CAM abbiamo previsto una versione montata e collaudata acquistabile su *www.futurashop.it.*

Fig. 16

CONCLUSIONI

Bene, per il momento abbiamo terminato. Risulterà chiaro che il modulo ESP32-CAM è molto versatile. Per il momento abbiamo appena iniziato a sfruttarne le enormi potenzialità.
Sperimentate con il progetto che vi abbiamo introdotto in questo articolo cercando di prendere confidenza con il modulo ESP32-CAM e con la relativa Motherboard. In un articolo successivo vedremo come utilizzare la Microcamera in un altro progetto molto interessante.
Il codice e le librerie utilizzate nell'articolo sono disponibili al seguente url di Github, *https://github.com/open-electronics/ESP32-CAM.*

Cosa occorre?

I componenti principali utilizzati in questo progetto sono disponibili presso Futura Elettronica. Il modulo telecamera con ESP32 (cod. ESP32-CAM) è in vendita a Euro 16,50 mentre il modulo convertitore USB/UART (cod. UC00A) è disponibile a Euro 15,50.
Il PCB della scheda base può essere realizzato con il servizio di stampa PCB *www.futuragroup4makers.com*.
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

Sensori per tutte le esigenze!

Kit di 37 moduli con sensori per Arduino, Raspberry, robotica ecc.

€ 29,90
Cod. **SENSORKIT37**

Modulo con igrometro per terreno

€ 11,50
Cod. **U20SMS**

Interruttore fotoelettrico con LASER a luce visibile

€ 28,00
Cod. **VLPSS**

Sensore di movimento a microonde amplificato

€ 14,00
Cod. **HB100AQ**

VASTA SCELTA DI SENSORI DI ULTIMA GENERAZIONE PER REALIZZARE I TUOI PROGETTI, ANCHE QUELLI PIÙ COMPLESSI.

Sensore qualità dell'aria per particolato PM2.5, PM10

€ 45,00
Cod. **PMSENSOR**

Trasduttore di pressione 0-1,2 MPa

€ 22,00
Cod. **PTS0161**

Modulo con sensore per rilevazione PH e temperatura

€ 29,50
Cod. **YB427**

Sonda per la misurazione del PH

€ 13,50
Cod. **PHSENSOR**

Sensore di Flessione 2,2"

€ 15,50
Cod. **FLEXSENSOR2**

Sensore analogico per misurare la torbidità dell'acqua

€ 14,50
Cod. **TORBISENS**

Sensore fotoelettrico

€ 7,50
Cod. **IRFOTOSENS**

Sensore di forza

€ 9,90
Cod. **SENSOREFORZA**

Prezzi IVA inclusa.

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

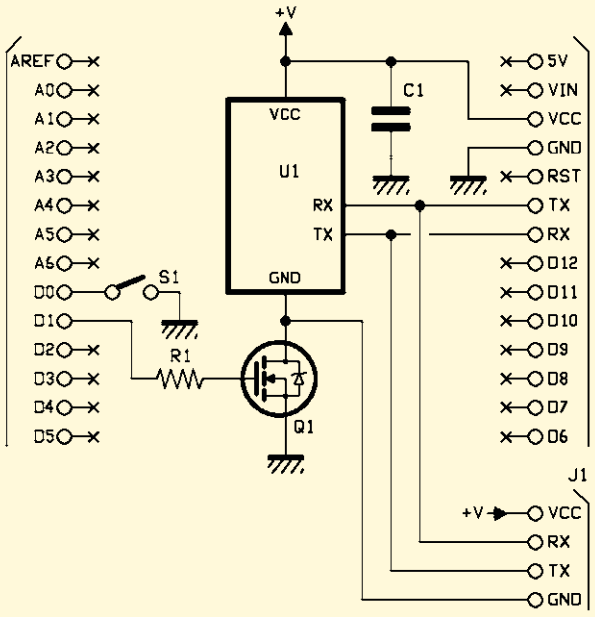
Scopri la gamma completa **su www.futurashop.it**

BIKE TRACKING

dell'ING. MIRCO SEGATELLO

Localizzatore GPS funzionante senza SIM né connessione cellulare, utile come antifurto per biciclette o altri oggetti di valore: sfrutta la rete mobile SigFox per comunicare a distanza i dati di localizzazione e la condizione di allarme.

Con l'avvento del GPS (Global Positioning System) ad uso civile e la riduzione dei costi dei componenti elettronici ad esso associati, i dispositivi di localizzazione a basso costo offerti dal mercato si sono diffusi esponenzialmente. Basti pensare al micro Localizzatore Satellitare GPRS/GSM MiniA8 venduto dalla Futura Elettronica (codice prodotto 1606-A8TRACK) alla cifra irrisoria di 14 Euro. Eppure si tratta di un dispositivo GPRS/GSM con microfono integrato, alimentato da una batteria LiPo (compresa) in grado di fornire a richiesta le coordinate rilevate dalla rete GSM e l'audio catturato nell'ambiente in cui si trova: davvero sbalorditivo! Per l'utilizzo di questi dispositivi è però necessario considerare due aspetti non da poco: la durata della batteria e l'utilizzo della SIM. La durata della batteria



dichiarata per questo dispositivo è di 30 ore ovvero poco più di un giorno, dopodiché è necessario provvedere alla ricarica.
Il secondo aspetto è la necessità dell'utilizzo di una SIM abilitata al traffico dati e voce necessaria al sistema per ricevere i comandi e per comunicare le coordinate. Questo significa spendere di più per mantenere attivo il sistema rispetto a quanto lo avete pagato. Ma la tecnologia avanza e sempre nuovi sistemi di comunicazione sono ora disponibili, soprattutto per quei dispositivi che vanno sotto il nome di "Internet delle cose", IoT in breve. A tal proposito, sempre su questa rivista, è stato proposto un corso in quattro puntate (dal fascicolo n° 237 al n° 240) riguardante l'IoT, partendo dal suo significato sino a giungere a diverse implementazioni.
Prendendo spunto da quanto studiato in quegli articoli, nasce l'idea di questo progetto che andiamo ora a presentarvi. L'idea è quella di sfruttare la tecnologia SigFox per far comunicare il nostro dispositivo superando i limiti del prodotto commerciale presentato prima. Ricordiamo brevemente che la tecnologia SigFox mette a disposizione un'infrastruttura hardware e software che permette ad appositi dispositivi di comunicare con Internet sfruttando comunicazioni a bassa potenza ma lungo raggio, senza la necessità di usare alcuna SIM. La tecnologia SigFox si appoggia infatti ad una rete LPWAN (Low Power Wide Area Network) per far comunicare i suoi dispositivi utilizzando pochissima energia. Per un approfondimento sull'argomento si legga la lezione n°4 del corso sull'IoT pubblicato sul numero 240 della rivista.

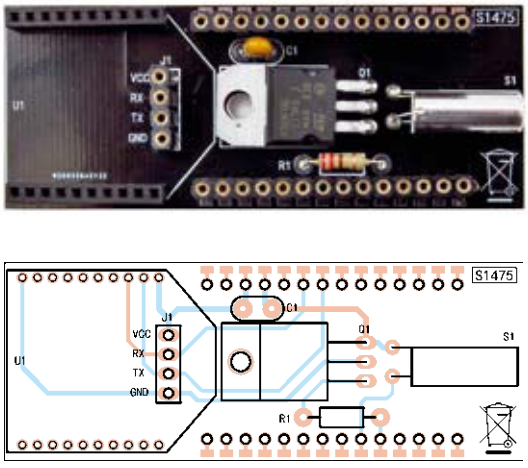
Il vantaggio di non dover acquistare una SIM e relativo abbonamento, associato al fatto di poter contare su componenti elettronici a bassissimo consumo, ci ha permesso di realizzare un localizzatore GPS dalle caratteristiche molto interessanti. La base per il nostro lavoro sarà una scheda Arduino MKRFOX1200 già equipaggiata con un modulo ricetrasmittente certificato SigFox alla quale sarà interfacciato un modulo GPS per la lettura delle coordinate, davvero essenziale ma efficace. Per essere del tutto esaustivi occorre ricordare che già la tecnologia SigFox offre un sistema di localizzazione SigFox Network Location basato su un calcolo di triangolazione tra il dispositivo in oggetto e le basi di ricezione del segnale. La metodologia non usa il calcolo del tempo di volo del segnale nè sfrutta l'effetto Doppler ma semplicemente si basa sull'analisi dell'intensità (RSSI) del segnale scambiato in una comunicazione.
Il dispositivo in oggetto invia regolarmente un messaggio e le stazioni base nelle vicinanze che raccolgono questo messaggio, di cui è nota la posizione, elaborano la possibile posizione della fonte dall'analisi dell'intensità del segnale ricevuto. Il vantaggio di questa metodologia è quello di non richiedere ulteriore hardware se non il solo modulo SigFox, con un abbattimento significativo dei costi, inoltre l'invio dei dati è veramente irrisorio permettendo il funzionamento per lunghi periodi con batterie molto piccole. Di contro il sistema non è molto accurato, infatti la stessa SigFox dichiara una precisione variabile in un range tra 1 e 10km ed è soggetta dal numero di stazioni base che ricevono il segnale. Cercare una bicicletta smarrita con l'indicazione di 1km non sarebbe proprio facilissimo, pertanto il nostro progetto prevede l'integrazione con un modulo GPS in grado di rilevare le coordinate nel raggio di pochi metri.
Il principale utilizzo di questo circuito è come localizzatore in caso di furto, nel nostro caso di una bicicletta. Non è inusuale lasciare incustodita la propria bicicletta mentre ci assentiamo per una commissione o semplicemente l'abbiamo lasciata in stazione prima di recarci al lavoro. Il circuito è più piccolo di una pacchetto di sigarette e va posizionato in modo da non essere visto, nel nostro esempio all'interno della borraccia di cui è fornita la bici. Un piccolo interruttore anch'esso in zona nascosta permette di accendere e spegnere il sistema. Se la bici è al sicuro e non prevediamo possa incappare in qualche malintenzionato il circuito rimane completamente spento e le batterie possono rimanere in questo stato anche per anni

senza necessità di manutenzione. Due comunissime ed economiche batterie formato stilo sono più che adatte per alimentare il nostro circuito. Quando invece riteniamo che la nostra bici possa essere a rischio accendiamo il circuito, il quale segnala l'entrata in funzione tramite l'accensione del LED di sistema della scheda MKR. Ad un lampeggio lungo seguiranno da uno a quattro lampeggi brevi a seconda dello stato di carica della batteria, utile per sapere per quanto tempo potrà funzionare il sistema. Dopo alcuni secondi, necessari affinché la bicicletta venga parcheggiata, il sistema entra in una modalità a bassissimo consumo (circa 100µA) salvaguardando la durata della batteria.
Se nessuno ha manomesso la bicicletta al ritorno spegnerete il circuito consci del fatto che la batteria avrà mantenuto quasi tutta la sua carica. Infatti una normale batteria formato AA da circa 2000mA/h potrebbe funzionare in questa modalità per 20000 ore; considerando l'autoscarica e l'assorbimento durante l'accensione (13mA) si garantirebbe il funzionamento per almeno un anno.
Se per qualsiasi motivo la bicicletta fosse manomessa il sensore di movimento a sfera lo rileverà attivando il microcontrollore tramite un segnale di interrupt, a questo punto verrebbe acceso anche il modulo GPS e non appena saranno rilevate delle coordinate valide, queste saranno inviate al portale SigFox che è configurato per inviare una e-mail di allerta con le coordinate appena acquisite. A questo punto il sistema ritorna in uno stato a basso consumo per risvegliarsi ad intervalli regolari (impostabili) rileggere le coordinate GPS e rispedirle, il tutto finché non spegnere il circuito o finché la batteria sarà completamente scarica. Nella modalità di ricerca satelliti (GPS FIX) l'assorbimento si attesta sui 120mA, mentre durante l'invio dei dati alla rete SigFox l'assorbimento è di 40mA, il tutto per pochi secondi ad intervalli di diversi minuti, garantendo una lunga durata della batteria.

SCHEMA ELETTRICO

Per facilitare l'assemblaggio dei componenti abbiamo predisposto un piccolo circuito stampato atto ad accogliere tutto ciò che è necessario al funzionamento: in pratica questa scheda è la base su cui montare la scheda SigFox e il modulo GPS.
Dall'analisi del relativo schema elettrico potete notare la scheda MKRFOX1200 interfacciata con un modulo GPSBee; abbiamo comunque previsto il connettore J1, per chi volesse utilizzare il GPS NEO6M (codice 2846-GPSNEO6M compreso di antenna) al posto del modulo predefinito.
Per il sensore di movimento a sfera S1 potete utilizzare indifferentemente uno dei due modelli disponibili sempre da Futura Elettronica (*www.futurashop.it*): quello contraddistinto dal codice 8300-YK010, ovvero il modello 6710-MERS4.
Siccome nel nostro sistema c'è la necessità di togliere alimentazione al modulo GPS durante la fase di funzionamento a basso consumo, abbiamo previsto il MOSFET Q1 in formato I-Pak, ma co-

piano di MONTAGGIO



Elenco Componenti:

R1: 220 ohm
C1: 100 nF ceramico
S1: Sensore di movimento
Q1: STP60N06
U1: Modulo GPS GPSBEE

Varie
- Strip femmina 10 vie 2 mm (2 pz.)
- Strip femmina 4 vie
- Strip femmina 14 vie 2,54 mm (2 pz.)
- Circuito stampato S1475 (69x27 mm)

Listato 1

```
float batteryVolt = analogRead(ADC_BATTERY) * voltageReference / ( 1024.0 * voltageDivider);
float VBatMin=2.2; //batteryLevel=1
float VBatMax=3.2; //batteryLevel=4
int batteryLevel= (batteryVolt-VBatMin)*(4.0-1.0)/(VBatMax-VBatMin) + 1.5;
constrain(batteryLevel, 0, 4);
Serial.print(batteryVolt,2);
Serial.print("V  level=");
Serial.println(batteryLevel);

  digitalWrite(LED_BUILTIN, HIGH);
tone(buzzerPin, 1000, 2000);
delay(2000);
digitalWrite(LED_BUILTIN, LOW);
delay(1000);

for (int i = 0; i < batteryLevel; i++)
{
  digitalWrite(LED_BUILTIN, HIGH);
  tone(buzzerPin, 1000, 300);
  delay(300);
  digitalWrite(LED_BUILTIN, LOW);
  delay(300);
}
```

munque va bene un qualsiasi MOSFET a canale N, anche nel classico formato TO-220.
La resistenza R1 serve a limitare la corrente di gate nel MOSFET che fluisce, per quanto il gate sia elettricamente isolato, a causa della sua natura capacitiva nei momenti in cui varia bruscamente la Vgs, quindi nei passaggi da 1 a zero logico e viceversa.

REALIZZAZIONE PRATICA

Bene, descritto lo schema elettrico possiamo vedere qualche dettaglio costruttivo: iniziamo con la scheda base, che è l'unico circuito da autocostruire, perché la MKRFOX1200 e il modulo GPS sono prodotti che si acquistano già pronti; lo stampato per la scheda base va ottenuto per fotoincisione, ricavando le necessarie pellicole dalle tracce lato

Listato 2

```
void loop()
{
  String positionData = GetGPSposition();

  if ( positionData != "NULL")
  {
    if (DEBUG) Serial.println("GPS Data OK -> send position data");
    SendSigFox(positionData);
  }
  else
  {
    if (DEBUG) Serial.println("GPS time out!");
    delay(2000);
  }

  if (DEBUG)
  {
    Serial.println("Go to sleep...\n");
    delay(1000*UPDATE_SIGFOX_TIME);  // use this line for continuos debug
  }
  else
    LowPower.deepSleep(1000*UPDATE_SIGFOX_TIME);
}
```

rame scaricabili dal nostro sito *www.elettronicain.it*. Una volta inciso e forato il PCB, potete disporvi pochi componenti occorrenti a partire da quelli a più basso profilo, quindi R1 e C1, quindi montando l'interruttore S1 e il Mosfet sdraiati sulla superficie dello stampato per ridurre lo spessore del circuito. Completate il montaggio inserendo e saldando gli strip femmina laterali per ospitare la MKRFOX1200 e il modulo GPSBEE; se optate, invece, per il GPS NEO 6M, montate anche lo strip corto J1 (4 poli). Aiutatevi con le foto riportate nell'articolo per la corretta disposizione dei componenti. Per completare il progetto, oltre al materiale in elenco (PCB, modulo GPS e MKRFOX1200), serve un portabatterie per alloggiare le due batterie formato AA ed un piccolo interruttore o deviatore.

IL SOFTWARE

Vediamo ora i passaggi da seguire per rendere operativo il circuito; è consigliato leggere la lezione n°4 del corso IoT presentata sul numero 240 della rivista, che descrive più in dettaglio la tecnologia SigFox ed il suo impiego pratico.
Aprite l'IDE di Arduino ed installare il supporto hardware alle schede SAMD Boards (32-bits ARMS Cortex-M0), successivamente installate le librerie Arduino SigFox for MRKFOX1200, Arduino Low Power e per finire TinyGPS++.
La prima parte dello sketch viene usata per configurare alcuni parametri di funzionamento. Di seguito sono riportate le tre righe di configurazione, rispettivamente il tempo in secondi massimo impiegato dal GPS prima di abortire l'operazione di

Listato 3

```
String GetGPSposition()
{
  String pos="";
  digitalWrite(GPSpowerPin, HIGH); //Turn GPS on
  delay(2000);  //wait for GPS power on

  unsigned long int GPSWakeUpTime = millis();

  while (true)
  {
    if (millis() > GPSWakeUpTime+GPS_FIX_TIMEOUT*1000)
    {
      digitalWrite(GPSpowerPin, LOW); //GPS turned off
      return "NULL";
    }

    while (Serial1.available() > 0)
      if (gps.encode(Serial1.read())) // A new sentence is correctly encoded.
      {
          if (gps.location.isValid())  // There is a valid coordinates
          {
            digitalWrite(GPSpowerPin, LOW); //GPS turned off

            //convert the coordinates in a string floatToFourChar
            float lat= gps.location.lat();

            byte *blat = (byte *)&lat;
            for (int i=4-1; i>=0; i--)
            {
              pos.concat(char(blat[i]));
            }

            float lng=gps.location.lng();
            byte *blng = (byte *)&lng;
            for (int i=4-1; i>=0; i--)
            {
              pos.concat(char(blng[i]));
            }
            return pos;
          }
      }
  }
}
```

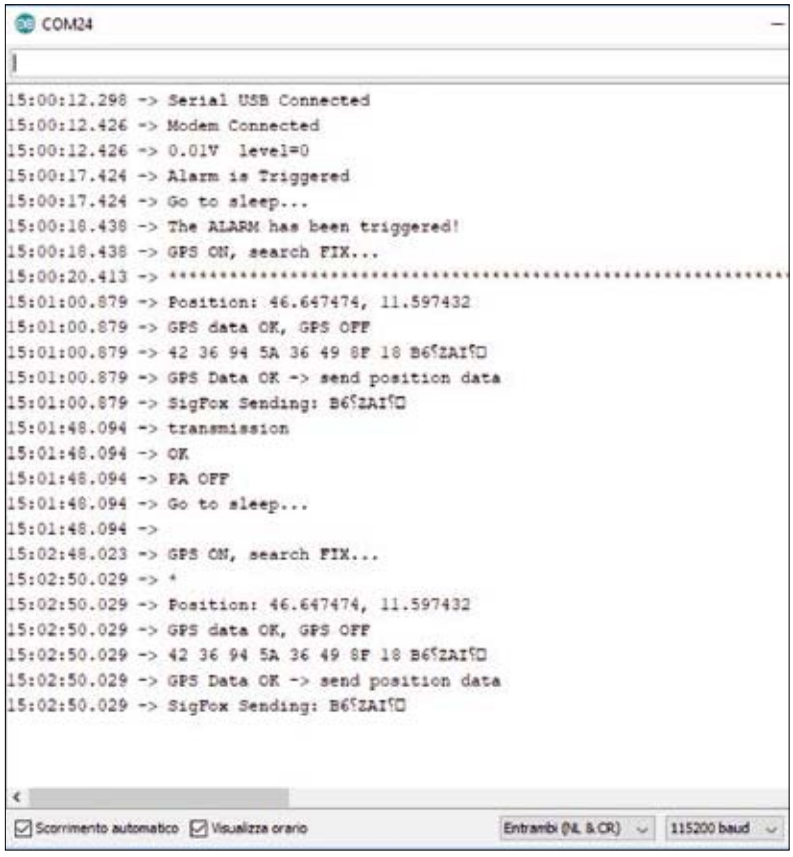


Fig. 1
Serial monitor per lo sketch Bike_Tracking.ino.

ricerca satelliti:

```
#define GPS_FIX_TIMEOUT 60      //seconds
#define UPDATE_SIGFOX_TIME 1800  //seconds
#define TIME_TO_PLACE_BIKE 2    //seconds
```

Infatti può accadere che la bicicletta possa essere stata parcheggiata all'interno di un' abitazione dove il segnale GPS è troppo debole (si può trattare, per esempio, di un seminterrato o un box, ovvero un garage interrato) per poter venire captato e la ricerca continua dei satelliti finirebbe per scaricare rapidamente la batteria. La ricerca sarà nuovamente tentata trascorso il tempo specificato dal parametro UPDATE_SIGFOX_TIME, valore che scandirà anche le operazioni di invio alla rete SigFox. Per finire, il parametro TIME_TO_PLACE_BIKE che imposta il tempo tra l'accensione del circuito e il momento in cui il sistema sarà armato abilitando il sensore di movimento.
Altro punto saliente dello sketch è la lettura della tensione della batteria e la successiva visualizzazione usufruendo del LED interno alla scheda MKR. All'accensione il LED di sistema si accende per due secondi per poi lampeggiare da 1 a 4 volte a seconda del livello di tensione della batteria. I due livelli minimo e massimo sono impostati con le variabili VBatMin e VBatMax (**Listato 1**). Se, come nel caso nostro, il LED non fosse visibile, potete optare per aggiungerne uno esterno collegato al pin6 con in serie una resistenza da 220 ohm, oppure potete connettere un piccolo buzzer passivo tra il pin2 e il GND lasciandolo all'interno della borraccia, il beep emesso fornirà le stesse informazioni del LED.
Terminata la sequenza di accensione in cui il GPS e il modulo SigFox sono spenti il microcontrollore viene posto in deep sleep al fine di ridurre al massimo la corrente assorbita. Solo l'interrupt dal sensore di movimento lo potrà risvegliare facendo eseguire le istruzioni contenute nella sezione **loop()**, riportate nel **Listato 2**.
La procedura **GetGPSposition()** provvede ad atti-

Listato 4

```
void SendSigFox(String data)
{

  SigFox.begin();   // Initializes the SigFox library and module
  delay(100);       // Wait at least 30mS after first configuration (100mS before)
  SigFox.status();  // Clears all pending interrupts
  delay(1);
  SigFox.debug();   // see (https://forum.arduino.cc/index.php?topic=478950.0)
  delay(100);

  SigFox.beginPacket(); //Begins the process of sending a packet
  SigFox.print(data);// send buffer to SIGFOX network

  int ret = SigFox.endPacket(true);

  SigFox.end();   // shut down module
}
```

vare il modulo GPS ed attende la corretta lettura delle coordinate relative alla posizione. Se questa operazione dovesse durare più del tempo massimo GPS_FIX_TIMEOUT la funzione ritornerebbe NULL (riferitevo al **Listato 3**).
SigFox impone stringenti limiti al massimo numero di byte inviabili (12) e comunque non più di 140 messaggi al giorno (uno ogni 10 minuti circa), pertanto è necessario codificare le coordinate GPS di latitudine e longitudine in formato float in soli 8 byte. Le coordinate GPS fornite dalla procedura **GetGPSposition()** vengono poi passate alla procedura **SendSigFox(positionData)** che provvede ad inviarlo al portale SigFox dopo aver aggiunto in automatico il valore RSSI e l'header del messaggio (**Listato 4**).
Il listato dello sketch completo di nome *Bike_Tracking.ino* è disponibile assieme ai file di questo numero della rivista. Ricordiamo che prima di tutto è necessario avviare lo sketch *FirstConfiguration.ino* che permetterà di visualizzare l'hardware ID e PAC del modulo SigFox che dovranno essere comunicate al portate, inoltre per queste prime prove potete lasciare la scheda MKRFOX1200 alimentata dalla porta USB del PC.
Nel testare il progetto occorre considerare che sia il modulo GPS che il modulo SigFox, utilizzano segnali radio a bassissima potenza facilmente schermabili, quindi ad esempio in interni non funzionerà. Anche la vicinanza con strutture metalliche potrebbe interferire con i segnali radio.
Altro dato importante è che quando il microcontrollore è in deep sleep, interromperà il funzionamento della USB virtuale, facendo cadere la comunicazione con il Personal Computer; tale connessione non potrà essere automaticamente ripristinata al risveglio, problematica non presente

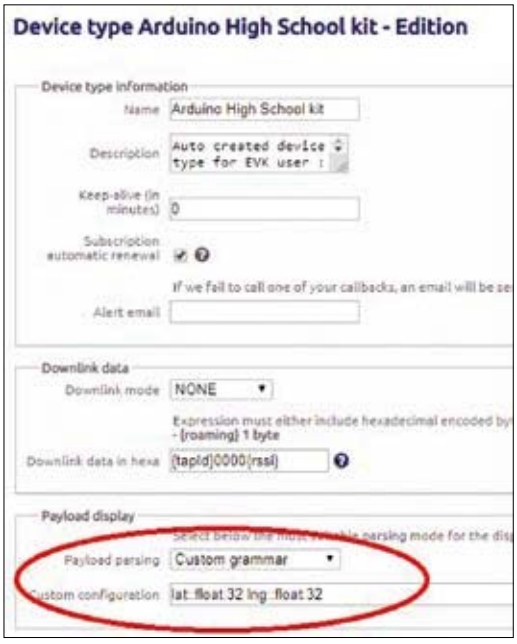


Fig. 2 Impostazione del payload display in SigFox.

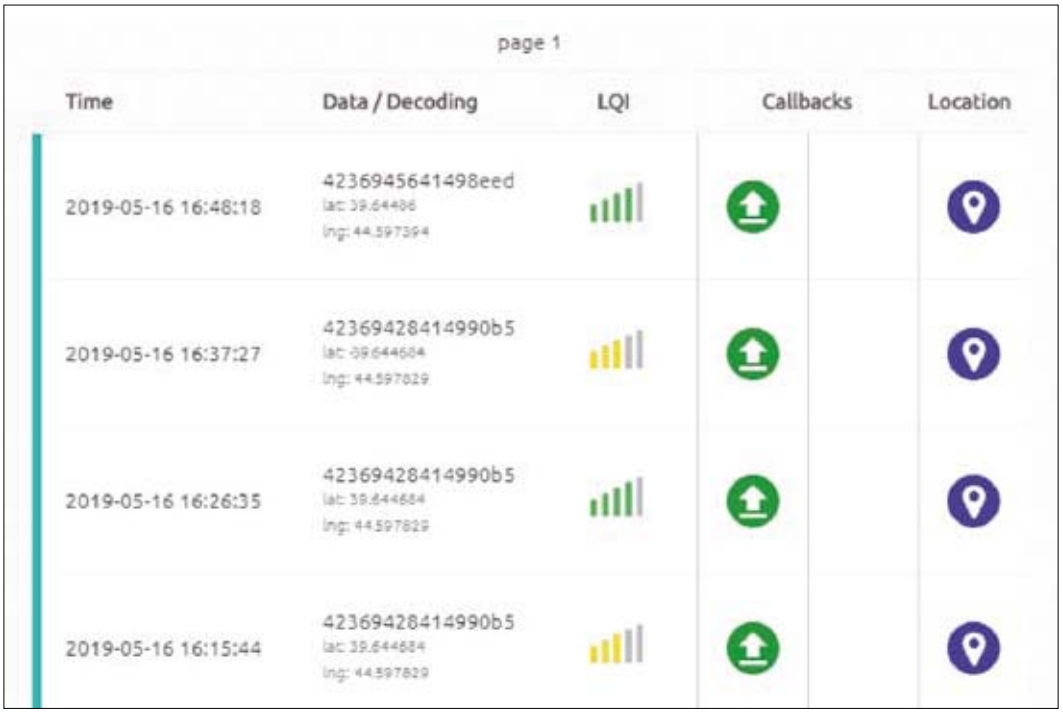


Fig. 3 Visualizzazione dei dati ricevuti.

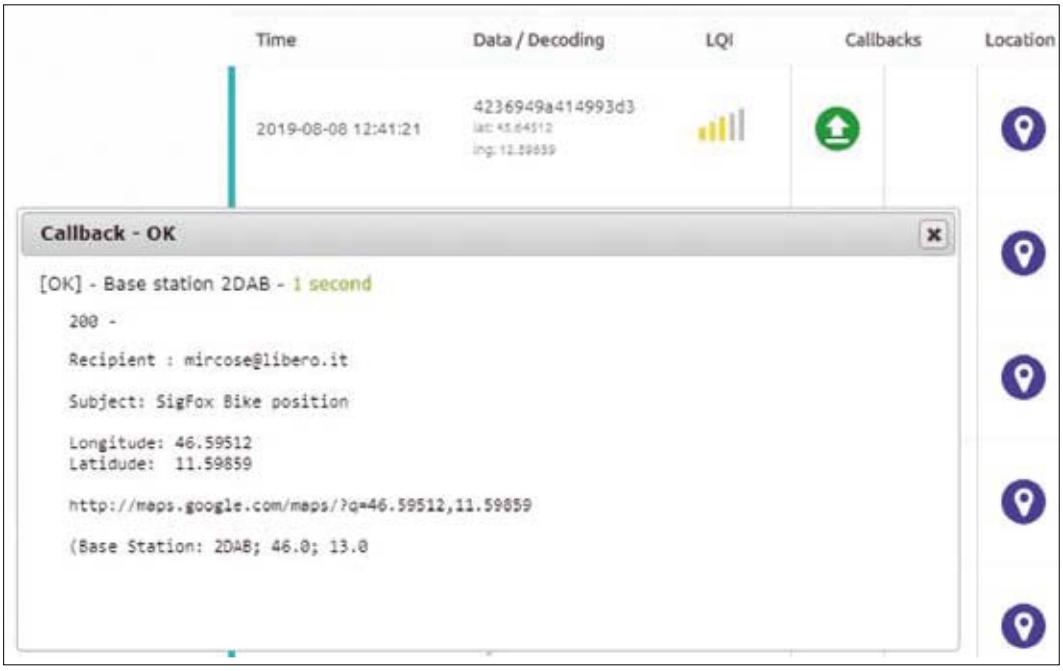


Fig. 4
Messaggio di Callback.

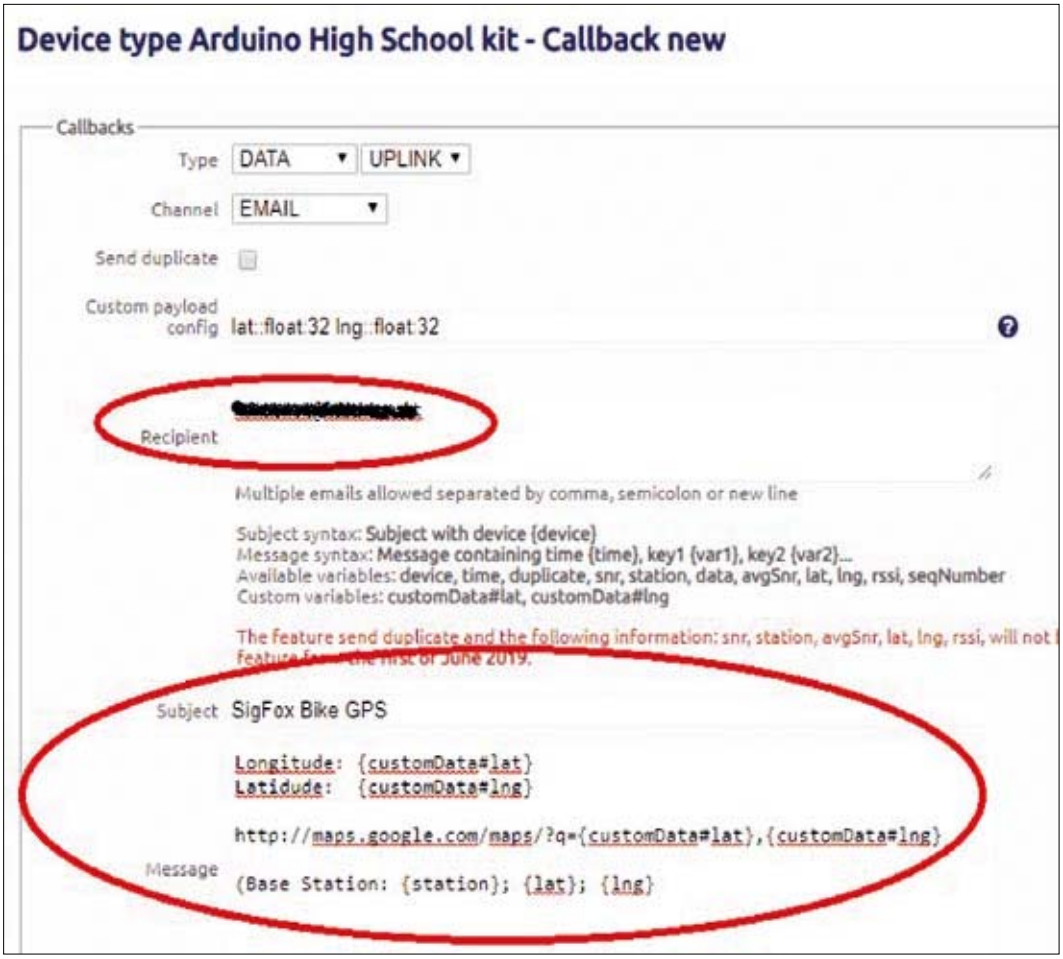


Fig. 5
Configurazione del Callback.

con Arduino UNO, perché dispone a bordo di un convertitore USB/Seriale separato dall'MCU.
Per eseguire un debug dei vari passaggi è quindi possibile impostare la variabile *DEBUG = true* attivando l'invio di messaggi sulla seriale relativamente alle varie operazioni in corso, la scheda non andrà mai in modalità deep sleep, assicurando così il funzionamento ininterrotto con Serial Monitor di Arduino (riferitevi alla **Fig. 1**).
A questo punto dobbiamo impostare il portale web di SigFox in maniera opportuna, allo scopo di fargli ricevere correttamente i dati che verranno inviati dalla nostra scheda, ovvero dal nostro localizzatore GPS basato sulla board MKRFOX1200; per i dettagli sulla procedura da svolgere allo scopo, vi rimandiamo alla puntata del corso IoT dove è stata descritta la tecnologia SigFox, vale a dire la quarta ed ultima puntata.
Nella tab **Device Type** dovete impostare il payload display, come mostrato nella **Fig. 2**; questo garantirà di riuscire a visualizzare correttamente le coordinate relative alla posizione del nostro localizzatore, sulla tab MESSAGE del portale (un esempio di visualizzazione è quello visibile nella **Fig. 3**).
Se ora cliccate sull'icona a forma di POI aprirete una cartina geografica, ovvero una mappa stradale, con indicata la posizione (approssimativa) del dispositivo, la quale sarà stata calcolata con il metodo precedentemente descritto; se invece cliccate sull'icona a forma freccia di colore verde, potete visualizzare il messaggio CALLBACK associato all'evento (**Fig. 4**) che lo ha scatenato.
Consigliamo di fare alcune prove lasciando la scheda connessa alla porta USB del PC, mantenendo Serial Monitor di Arduino aperto e attivo, con l'opzione DEBUG settata su *true*, per verificare che tutto funzioni correttamente e comunque nella maniera che ci si aspetta.
A questo punto possiamo completare la configurazione del portale web SigFox facendo in modo che ad ogni coppia di coordinate di localizzazione che viene ricevuta consegua l'invio di una e-mail di notifica contenente le coordinate stesse; per fare questo occorre accedere alla tab CALLBACK, creare un nuovo messaggio di CALLBACK (questa operazione permette di creare il cosiddetto "custom callback") ed editarlo come visibile nella **Fig. 5**. Qui, Il campo **recipient** contiene l'indirizzo e-mail al quale volete che sia inviata la segnalazione, Subject è l'oggetto della e-mail e **Message** contiene il corpo della e-mail, vale a dire il testo della notifica che esso dovrà contenere.
Quando tutto diverrà operativo ed il sistema sarà

Fig. 6
E-mail inviata da SigFox con le coordinate del dispositivo.

Fig. 7
Schermata di Google maps con la posizione del dispositivo..

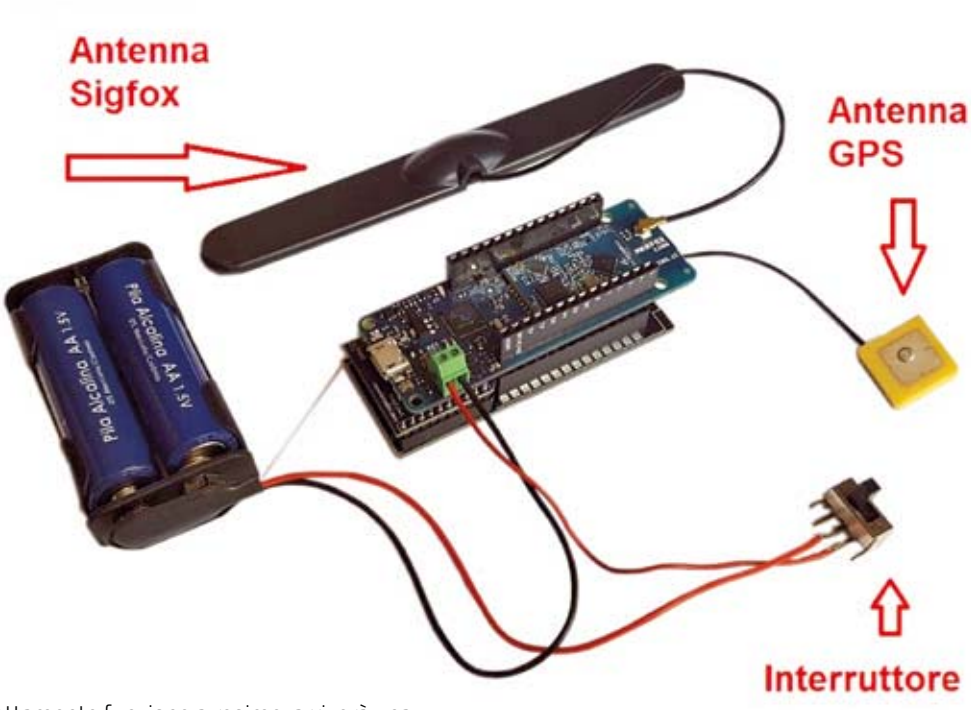


Fig. 8
Il circuito completamente assemblato con modulo GPSBee.

perfettamente funzione a regime, arriverà una e-mail all'indirizzo indicato nell'impostazione del messaggio di CALLBACK custom, con la cadenza (periodicità) specificata dal parametro UPDATE_SIGFOX_TIME; il messaggio sarà fatto come quello visibile in **Fig. 6** che riporta l'e-mail ricevuta su uno smartphone.
La e-mail potrà essere letta da qualsiasi dispositivo mobile o da Personal Computer e, se nel dispositivo mobile (tablet o smartphone) sarà installata l'applicazione Google Maps, cliccando sopra il link contenuto nel messaggio di posta elettronica ricevuto sul dispositivo si aprirà la mappa con evidenziata la posizione del nostro dispositivo, come viene mostrato nello screenshot che vedete nella **Fig. 7**.

INSTALLAZIONE DEL SISTEMA

Il localizzatore GPS/SigFox qui proposto, una volta assemblato innestando il modulo GPS e la board MKRFOX1200 sulla scheda base, dev'essere cablato in modo volante disponendo opportunamente le antenne e i loro cablaggi; anche il portabatterie per l'alimentazione rimarrà volante, purché

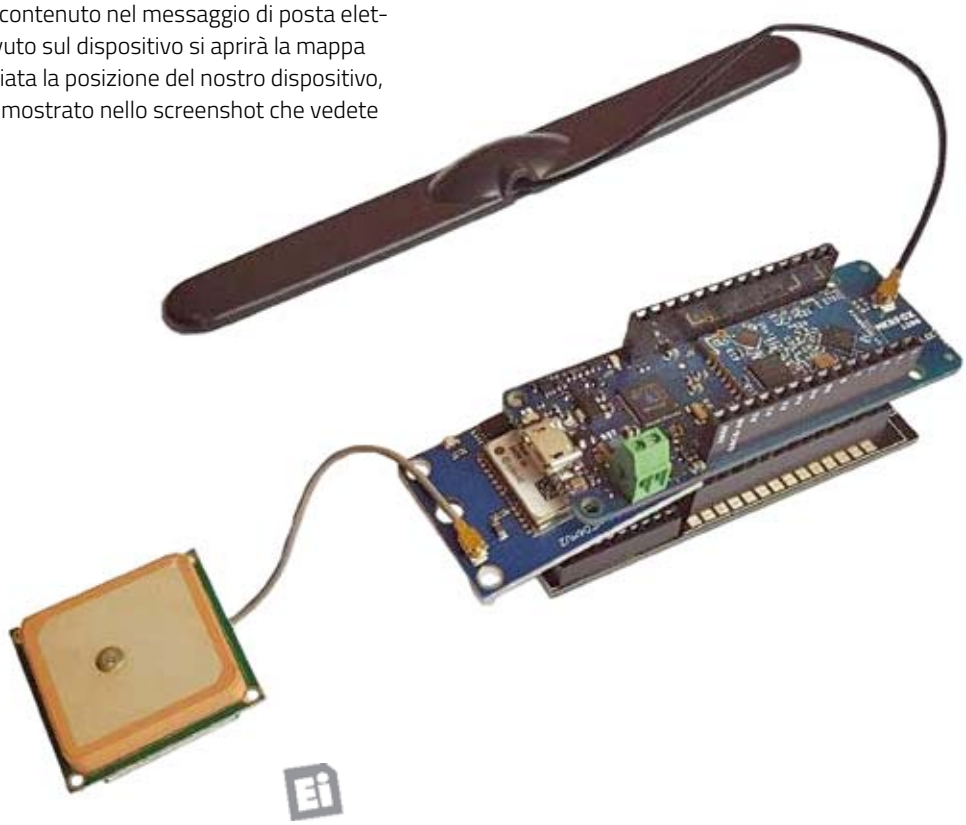


Fig. 9
Il circuito con il modulo GPS NEO-6M.

distanziato dai circuiti stampato.
Per rendere il circuito completamente operativo ed autonomo dovete procuravi un piccolo interruttore ed un porta batterie per due batterie tipo stilo (formato AA) che forniranno alimentazione al piccolo morsetto a vite disponibile nella scheda MKRFOX1200. L'interruttore si rende necessario per poter spegnere e accendere il sistema al bisogno, ovvero spegnerlo quando siamo in giro in bicicletta o quando quest'ultima è parcheggiata in un luogo sicuro, e accenderlo quando si pensa di aver bisogno del localizzatore.
L'insieme cablato con le batterie e l'interruttore ON/OFF è visibile nella **Fig. 8** che riporta una fotografia del sistema con il GPSBee: noterete la presenza dell'antenna per il modulo SigFox (che è quella nera e lunga) e l'antenna per il modulo GPS (quella quadrata tipo mattonella, con il cavo saldato in posizione centrale).
Lo stampato è predisposto per accogliere sia il modulo GPSBee, sia il NEO-6M della U-Blox (nella **Fig. 9** vedete il localizzatore SigFox nella versione proprio con il modulo GPS della U-Blox); rammentiamo che quest'ultimo, essendo più lungo, sporgerà leggermente dallo stampato della scheda base. Inoltre, siccome si innesta nel connettore più piccolo e non completamente a bordo (quindi si fissa alla scheda base da un lato solo, il che lo espone all'oscillazione) può essere necessario bloccarlo applicandogli sotto il corpo un pezzetto di spugna e fermandolo poi con del nastro adesivo.
Il circuito è piccolo ma non minuscolo, pertanto la sua naturale collocazione è occultarlo all'interno di qualcosa che normalmente è presente in una bicicletta e che pertanto non desta sospetti; a noi è venuta l'idea di collocarlo all'interno della borraccia porta-bibite presente nella bicicletta. Il piccolo interruttore di accensione dovrà essere posizionato in un punto non visibile. Della gomma piuma risulta ottima per fissare il tutto all'interno della borraccia, assicurando protezione da eventuali scossoni e danni correlati. Utilizziamo l'accortezza di mantenere le antenne il più lontano possibile da qualsiasi elemento metallico, batterie comprese (**Fig. 10**).
Terminata la preparazione della borraccia, quest'ultima completa di tutta l'elettronica prevista potrà essere collocata nella sua naturale posizione nella bicicletta e, naturalmente, chiusa affinché non sveli il proprio contenuto, altrimenti i malintenzionati come prima cosa la getteranno via, ovvero ne svuoteranno il contenuto.
Dopo diverse prove effettuate sul nostro prototipo, vale la pena di sottolineare il fatto che sia il segnale GPS che il segnale della rete SigFox, essendo molto deboli possono essere facilmente ostacolati: alberi ad alto fusto, palazzi o strutture metalliche importanti, possono quindi compromettere il funzionamento del circuito, ovvero la possibilità di comunicare la posizione.
Però, non appena la bicicletta si troverà in un punto favorevole, le sue coordinate saranno presto note, permettendo di ritrovarla, quindi anche il limite suddetto non è poi così critico né può pregiudicare sensibilmente l'affidabilità del sistema localizzatore.

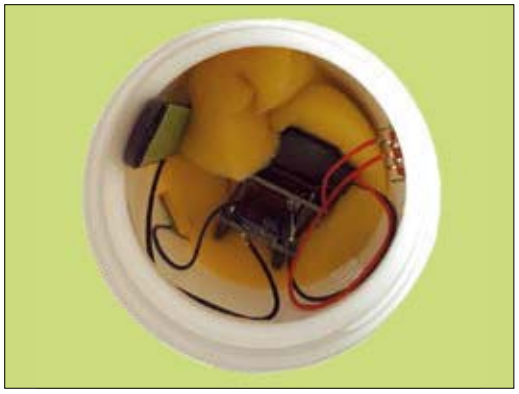


Fig. 10 Circuito sistemato all'interno della borraccia.

CONCLUSIONI

Bene, giunti a questo punto abbiamo terminato.
Il progetto qui descritto nasce come generico localizzatore attivato da un sensore di movimento a contatto e può essere applicato a qualsiasi veicolo o anche a pacchi in viaggio a bordo di veicoli per tracciarne gli spostamenti.
Piccole modifiche consentono molti adattamenti a situazioni anche diverse da quella per cui il sistema è stato progettato.

Cosa occorre?

I componenti utilizzati in questa presentazione sono disponibili presso Futura Elettronica.
La board Arduino MKR FOX 1200 (cod. MKRFOX1200) è in vendita a Euro 43,90, la relativa antenna (cod. ANTENNADIPOLO) costa Euro 6,00, il GPS con UBLOX NEO-6M (cod. GPSNEO6M) è disponibile separatamente a Euro 21,00 infine il sensore di vibrazione a sfera (cod. YK010) costa Euro 1,00.
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

Costruisci e programma un ROBOT!

cod. KSR21
€ 42,00

cod. KSR19
€ 35,00

Robot RICCIO - in kit

Bellissimo robot riccio completo di tutte le parti meccaniche ed elettroniche. È ideale per avvicinare i ragazzi al mondo dell'elettronica e della meccanica. Il robot riccio è dotato di sensore sonoro che gli consente di interagire con il battito delle mani. In funzione del numero di battiti delle mani il robot riccio risponde con una sequenza di movimenti differente. Completo di manuale d'istruzioni in italiano per il montaggio organizzato in step e riccamente illustrato. Funziona con 4 batterie ministilo tipo AAA.

Robot 5 in 1 con codifica meccanica - in kit

Robot didattico con codifica STEM (Science, Technology, Engineering, Mathematics). Ideale per principianti, permette di avvicinarsi all'ingegneria di costruzione delle macchine e della codifica manuale. Stimola i bambini a pensare e a sviluppare la loro capacità nella risoluzione dei problemi. Dispone di ruota di codifica con cui è possibile programmare i movimenti. Tanto maggiore saranno le capacità e l'esperienza dell'utente, quanto più interessanti e impegnativi diventeranno i movimenti del robot. Funziona con 2 batterie ministilo tipo AAA.

Robot TOBBIE II con MICRO:BIT in kit

Robot a 6 gambe con corpo libero di ruotare, dotato di tutti i componenti elettronici necessari, tra cui un motore, un sensore a infrarossi e un cicalino per renderlo attivo e interattivo. Ideale in combinazione con Micro:bit. È possibile programmarlo in tutta semplicità con Javascript Blocks Editor o Python Editor sul tuo PC portatile o smartphone. Funziona con 4 batterie ministilo tipo AAA.

cod. KSR20
€ 89,00

Robot TOBBIE II in kit

Robot in kit a 6 gambe. Può funzionare in due differenti modalità: "Seguimi" e "Esplora". Nella modalità "Seguimi" Tobbie è in grado di seguire qualsiasi cosa gli si avvicini. Quando funziona in modalità "Esplora", grazie al rilevamento automatico tramite sensori infrarossi, è in grado di evitare gli ostacoli che trova davanti a sé trovando sempre un nuovo percorso e continuare a muoversi. Adatto dai 14 anni in su. Funziona con 4 batterie ministilo tipo AAA.

cod. KSR18
€ 37,00

Prezzi IVA inclusa

di DAVIDE SCULLINO

I BUS PER AUTOMOTIVE

Le automobili contengono sempre più elettronica, coordinata e interfacciata mediante link di comunicazione che possono essere bus o vere e proprie reti locali. Studiamole nella prospettiva dell'avvento dell'ADAS.

Viste da sotto il cofano, le automobili dei nostri tempi sono molto diverse da quelle prodotte anche solo 30 anni fa e chi crede che l'elettronica riguardi solo l'auto elettrica, guardando bene dentro un'automobile a motore endotermico di ultima generazione arriverebbe a ricredersi. Infatti oggigiorno una parte della meccanica viene soppiantata dall'elettronica e sensori, attuatori e microcalcolatori sono distribuiti a bordo in grande quantità; diverranno ancor di più con il crescere dei livelli di assistenza alla guida definiti da quello che ormai conosciamo come ADAS.

Le ragioni dell'elettronica a bordo delle auto sono molteplici e pesano in varia misura: qualcuna è necessaria e dettata da ragioni di sicurezza, di ecologia ed economia, qualcun'altra da questioni di marketing.

La prima comparsa dell'elettronica a bordo delle auto si deve a due fattori: ottimizzazione del funzionamento del motore e miglioramento della vita di bordo. Vediamo coinvolta l'elettronica in questi campi:

- gestione motore, per ottimizzare i consumi e ridurre le emissioni inquinanti;

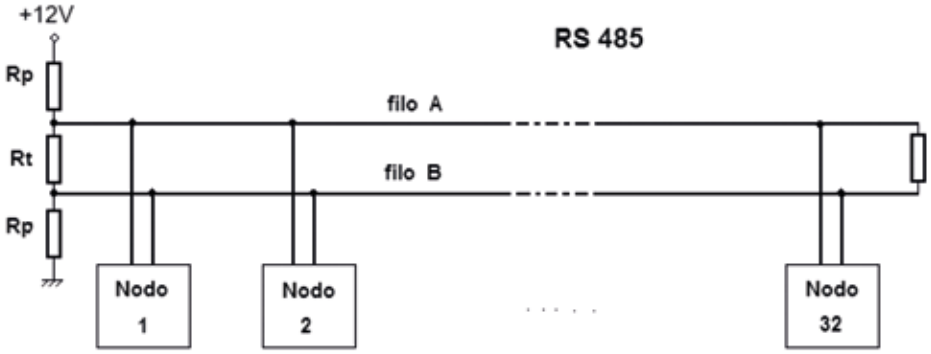


Fig. 1
Tipico bus fisico del CAN.

- gestione di trasmissione, sterzo e sospensioni, per accrescere il livello di sicurezza e guidabilità;
- automazione della climatizzazione e delle funzioni normalmente manuali (chiusura porte, alzavetri ecc.);
- realizzazione di sistemi di infotainment sempre più sofisticati;
- connessioni via rete mobile.

L'elettronica dei primi tempi, limitata essenzialmente alla gestione del motore e della vita a bordo, era stand-alone, nel senso che ogni sistema viveva di vita propria, funzionava in maniera specifica ed era confinato all'ambito per il quale era stato progettato. Al crescere della complessità dei sistemi è sorta l'esigenza di creare delle connessioni standard e soprattutto versatili ed espandibili, capaci sia di realizzare soluzioni scalabili (ad esempio per aggiungere sensori, attuatori e funzionalità) sia di integrare, all'occorrenza, i vari sottosistemi. Sono nati così i bus per automotive, che nelle loro formulazioni iniziali erano abbastanza semplici, ma che oggi sono approdati a vere e proprie reti informatiche: un esempio per tutti è la 100BASE-T1. In questo articolo faremo una carrellata sui principali link utilizzati nell'automobile moderna conoscendone peculiarità, ambiti applicativi, passato, presente e sviluppi futuri.

IL CAN-BUS

Se si parla di link dati per l'automotive viene in mente prima di tutto il CAN-Bus, nato in casa Bosch per armonizzare i sistemi di gestione motore, i quali soprattutto per rispettare le sempre più stringenti normative ambientali sono cresciuti di complessità, al punto che la più elevata concentrazione di dispositivi elettronici in un'automobile è nel gruppo motopropulsore ed è lì per cercare di fare l'impossibile, ossia annullare le emissioni nocive dei motori endotermici, siano essi a ciclo Otto o a ciclo Diesel. Se pensiamo a come è complesso oggi un motore Diesel rispetto a solo 35 anni fa, ne abbiamo un'idea; infatti il motore Diesel base era puramente meccanico e non aveva impianto elettrico, il che, abbinato alle caratteristiche intrinseche di quel tipo di propulsore, lo rendeva nettamente più affidabile del motore a benzina. Oggi attorno a un motore Diesel c'è così tanta elettronica che non solo è diventato più complesso, ma anche meno affidabile: solo per migliorare la combustione si è passati dall'iniezione meccanica a quella elettronica common-rail o iniettore-pompa aggiungendo poi tutta una serie di sensori per rilevare i parametri operativi del motore (regime di giri, fase della distribuzione, istante di iniezione, temperatura combustibile e liquido di raffreddamento del motore, temperatura e volume d'aria aspirata) e attuatori per il controllo dell'iniezione, dell'aspirazione ecc. Per non tirare in ballo quelli riguardanti il contenimento delle emissioni inquinanti, che vanno dai sistemi per abbattere CO e CO2, a quelli per la riduzione degli NOx (EGR e catalizzatore a riduzione selettiva) e soprattutto del particolato. Simile è il discorso che riguarda il motore a benzina.
Tornando sul CAN-Bus, anche noto come ISO 15765, è stato sviluppato nel 1986 ed è un bus seriale progettato per reti interne al veicolo; la comunicazione avviene su un doppino ritorto allo scopo di offrire un'alta immunità alle interferenze elettromagnetiche. In un veicolo possono essere presenti più istanze di CAN-Bus: per esempio il controllo dei finestrini e dei sedili è normalmente eseguito da un CAN a bassa velocità, mentre la gestione del motore ed il controllo dei freni richiedono un CAN ad alta velocità.
CAN è acronimo di Controlled Area Network e offre un data-rate tipico di 1 Mbps per distanze fino a 40 m; le evoluzioni subite dal CAN-Bus hanno portato a migliorie in fatto di data-rate e oggi abbiamo anche CAN FD (che opera fino a 5 Mbps) e CAN Partial Networking. Sul piano fisico, il CAN-Bus sfrutta un link seriale a due fili riferiti a massa,

quindi una linea differenziale che come livello dei segnali è un RS485 (**Fig. 1**); è proprio questo che conferisce immunità nei confronti dei disturbi, inevitabilmente presenti a bordo degli autoveicoli.
Il CAN è un bus che viene impiegato nella comunicazione tra la ECU e le altre centraline di bordo, come ad esempio quelle dell'ESP e dell'ABS, il body computer (BSI), le centraline di governo del cambio automatico e del servosterzo elettroidraulico (GEP) o elettrico.
Il CAN-Bus è un protocollo che definisce sia lo scambio dei dati, sia il mezzo fisico (PHY) che lo permette; quest'ultimo consiste in una connessione seriale a linea bilanciata, vale a dire che i dati viaggiano su due fili riferiti a massa, ma in opposizione, quindi quando su un filo è presente il livello logico alto sull'altro si trova quello basso; l'unità che riceve i dati ha in ingresso un amplificatore differenziale (un operazionale), il quale fornisce alla propria uscita la somma algebrica dei segnali agli ingressi invertente e non invertente. Ne deriva che se un disturbo elettrico investe i fili dei dati e si presume che induca in entrambi lo stesso livello di tensione, il differenziale annulla il disturbo sovrapposto ai fili, mentre amplifica correttamente i dati. Infatti se il disturbo entra nei due input del differenziale, quello che passa dal non-invertente viene sommato all'invertente e la somma dà zero, mentre siccome i livelli logici sono opposti, all'uscita abbiamo la somma, che è un impulso di ampiezza doppia e di polarità dipendente da quale, tra gli ingressi invertente e non-invertente è positivo.
Lo standard utilizzato nel CAN-Bus è tipicamente l'interfaccia RS-485, che è una seriale bilanciata in cui le unità trasmittente e ricevente sono collegate

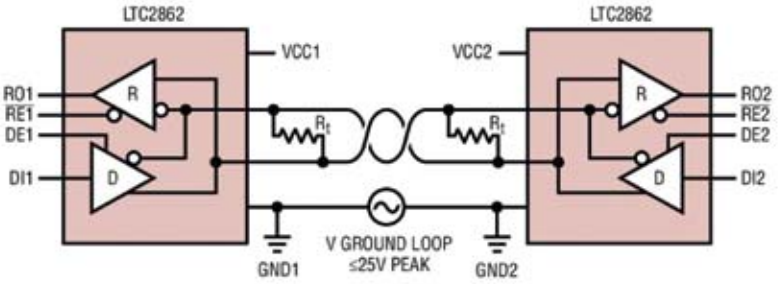


Fig. 2
Esempio di transceiver RS485.

con cavi di tipo twisted pair (doppinoritorto) per aumentare l'immunità ai disturbi. La linea RS-485 prevede dispositivi trasmittenti e riceventi e può funzionare in modo sia unidirezionale che bidirezionale; normalmente nel CAN Bus si adotta una sola linea bidirezionale e i dispositivi affacciati al doppino twistato sono transceiver, vale a dire che contengono sia un'unità trasmittente che una ricevente, attivate una alternativamente all'altra mediante segnali di controllo. La **Fig 2** mostra un esempio di linea basata su transceiver RS485.
Sul piano del protocollo, ossia del formato dei dati e la loro codifica (modello ISO/OSI), lo standard descrive principalmente lo strato (layer) di scambio dati (Data Link Layer), composto dal sottostante strato (sublayer) logico Logical Link Control (LLC) e dallo strato sottostante questo che è il Media Access Control, (MAC). I protocolli di tutti gli altri layer sono lasciati alla libera scelta del progettista della rete di bordo. La **Fig. 3** propone il modello ISO/OSI del CAN-Bus.
Il CAN permette di realizzare delle reti (per questo si può definirne uno stack secondo il modello ISO/OSI, che definisce l'architettura delle reti dati),

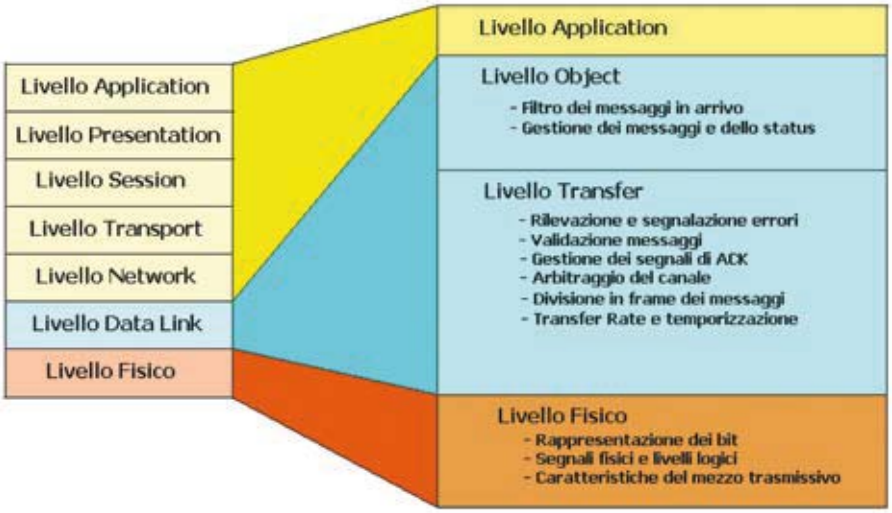


Fig. 3
Architettura CAN-Bus.

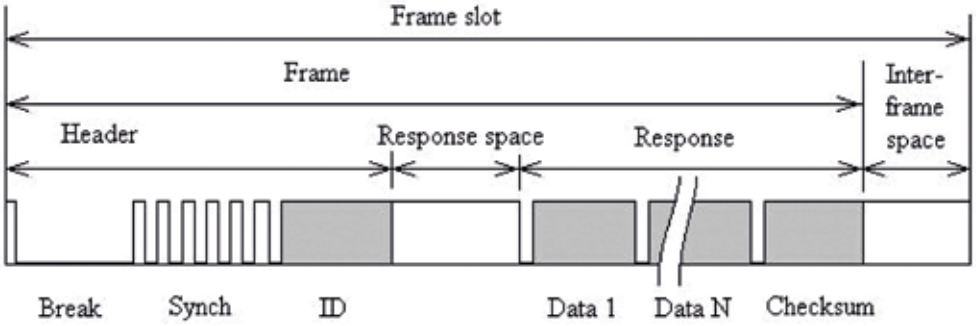


Fig. 4
Tipico frame del protocollo LIN.

quindi con dispositivi (nodi) che comunicano tra loro e non con un master che interroga le periferiche e le coordina; questo impone che il protocollo detti alcune regole utili a evitare che in caso di trasmissione simultanea, i dati vengano a sovrapporsi e quindi vadano perduti.
La regola principale è che i dati viaggiano secondo un modello basato su bit dominanti (0 logico) e bit recessivi (1 logico); se un nodo trasmette un bit dominante e un altro un bit recessivo, il bit dominante prevale, come avviene dell'operazione logica AND vista in questo stesso capitolo. Quando vengono a trovarsi contemporaneamente sulla rete un bit recessivo e uno dominante, si verifica una collisione e solo il bit dominante va in rete (le collisioni sono invisibili). In pratica il bit dominante determina l'impulso sulla linea bilanciata e quello recessivo non produce effetto, quindi ogni volta che si presenta una differenza di potenziale tra i due fili di segnale, tutta la rete la interpreta come bit dominante.
Nel caso di un bus a linea bilanciata come l'RS485, per stabilire la priorità di comunicazione si applica lo schema CSMA/BA (Carrier Sense Multiple Access/Bitwise Arbitration): se due o più dispositivi trasmettono i loro pacchetti dati contemporaneamente, si applica un meccanismo di arbitrato basato sulla priorità per decidere a quale dispositivo permettere di proseguire la trasmissione. Durante la trasmissione, ogni nodo in trasmissione controlla lo stato del bus e confronta il bit ricevuto con il bit trasmesso: se riceve un bit dominante mentre viene trasmesso un bit recessivo, interrompe la trasmissione (perde l'arbitrato). L'arbitrato viene eseguito durante la trasmissione del pacchetto dei dati di identificazione del nodo; i nodi che incominciano contemporaneamente a trasmettere inviano un ID dominante a 0 binario, che incomincia con il bit alto. Non appena il loro ID è rappresentato da un numero più grande (quindi a priorità minore) i nodi stessi inviano un bit 1 (recessivo) e aspettano la risposta di uno 0 (dominante), quindi interrompono la trasmissione. Al termine dell'invio degli ID, tutti i nodi sono tornati allo stato di OFF, e il messaggio con la priorità corrente massima può liberamente transitare.
I messaggi scambiati in rete, detti "frame" iniziano con un bit di start (Start Of Frame, SOF); possono essere di quattro tipi:

- Data frame; sono i dati che il nodo trasmette;
- Remote frame; richiede la trasmissione di un determinato identificatore.
- Error frame; trasmesso da un nodo che ha rilevato un errore, per comunicarlo alla rete;
- Overload frame: introduce un ritardo fra data frame e/o remote frame.

Fig. 5
Connessione LIN di un alternatore.

Per quanto riguarda i Data frame, ossia i dati scambiati sul CAN, i messaggi possono avere due formati:

- Base frame format, caratterizzato da 11 bit di ID;
- Extended frame format, con 29 bit di ID.

Lo standard CAN deve riconoscere almeno il primo, mentre il secondo è opzionale; se è presente ma non gestito, deve essere comunque tollerato.
Il frame base permette 2^{11} (2048) tipi di messag-

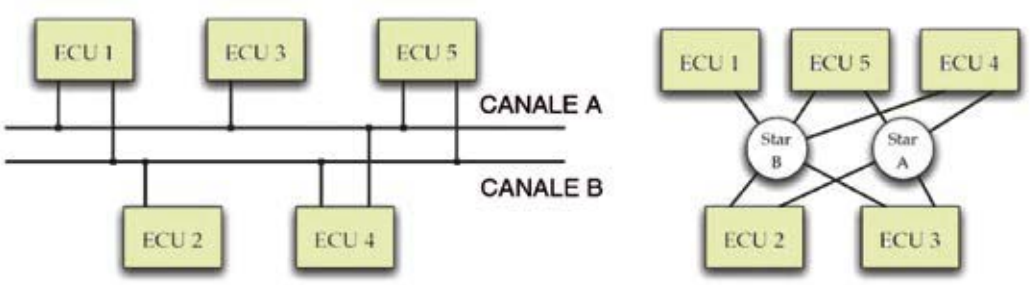


Fig. 6
La topologia di rete di un cluster FlexRay può essere di tipo a bus oppure a stella.

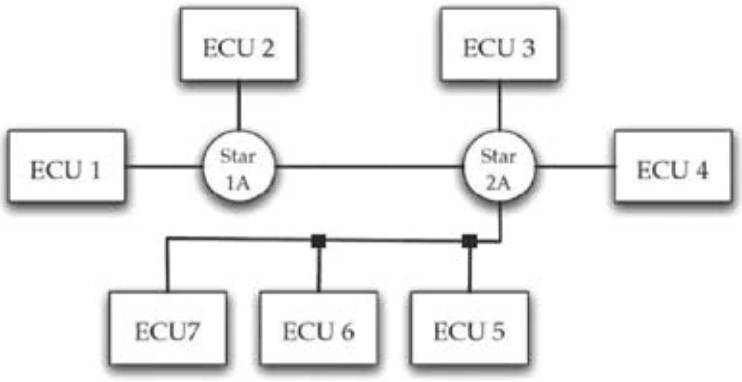
gio, ma da specifiche Bosch se ne possono usare solo 2031; l'extended frame format arriva a 2^{29} (536.870.912) tipi di messaggo.
La connessione fisica non è universale ma può cambiare in base al dispositivo; ha comunque sempre due contatti per i dati e una massa di riferimento. I livelli in gioco sono di ±12V.
Il CAN-Bus è normalmente presente sul connettore OBDII, che permette la diagnosi della centralina (ECU o Body Computer) principale del veicolo; le relative connessioni sono le seguenti:

- pin 6 = CAN High;
- pin 14 = CAN Low;
- pin 4 = massa impianto veicolo;
- pin 16 = positivo di batteria veicolo.

IL BUS LIN

Il Local Interconnection Network è un bus utilizzato per connessioni locali di sensori e attuatori di sottosistemi di bordo con le centraline principali, ovvero la ECU o il body computer; per esempio interfaccia la ECU con il modulo regolatore di carica dell'alternatore. Il bus è a bassa velocità di comunicazione, pensato per collegare i sensori e gli attuatori di un singolo sottosistema del veicolo in modo semplice e soprattutto economico. Alcuni esempi sono: dispositivi di controllo degli automatismi delle porte (alzavetri e chiusura centralizzata), regolazione specchietti elettrici, lettura dei sensori del motore. Quindi il controller del LIN è un dispositivo di interfaccia tra la centralina principale e i sensori e attuatori, che rappresentano le parti periferiche del sistema.
La comunicazione avviene su un solo filo rispetto a massa, come nell'One-wire; un dispositivo Master gestisce la comunicazione sul bus e deve avere una temporizzazione precisa, mentre i dispositivi Slave si sincronizzano con le stringhe di dati inviate dal master. La comunicazione è quindi asincrona e le unità slave utilizzano l'intestazione (header) dei messaggi inviati dal master per sincronizzare il proprio baud-rate; ecco perché il master deve avere un clock preciso, mentre per gli slave è ammessa una tolleranza del 14%.
La velocità di comunicazione (data-rate o bit-rate che dir si voglia) sul bus LIN non supera i 19.200 bps e questo limita il massimo numero di dispositivi gestibili, almeno se si desidera una risposta in tempo breve, tuttavia il LIN nasce per l'interrogazione di sensori e il comando di attuatori che non richiedono tempi di reazione particolarmente stretti. il LIN supporta anche la comunicazione a 2.400 e 9.600 bps.
Nello standard LIN ogni dispositivo connesso al bus è chiamato *nodo* e il complesso dei nodi su un bus prende il nome di *cluster*. La specifica prevede un numero massimo di 16 nodi per cluster e una lunghezza del bus non superiore a 40 metri, condizione che non viene mai raggiunta in un automezzo, anche perché di norma il sottosistema è abbastanza vicino ai sensori o attuatori correlati.
Per quanto riguarda il protocollo di comunicazione, il bus LIN ha molte caratteristiche in comune con il CAN, in particolare la modalità di comunicazione che è di tipo broadcast. Non esiste una comunicazione punto-punto tra master e slave, ogni messaggio inviato è ricevuto da tutti i nodi della rete. Si elimina così il problema di conflitti e di sincronizzazione. Il bus è perciò deterministico, poiché l'invio di un messaggio dipende esclusivamente dalle

Fig. 7
Topologia di rete Flexray ibrida single-channel.



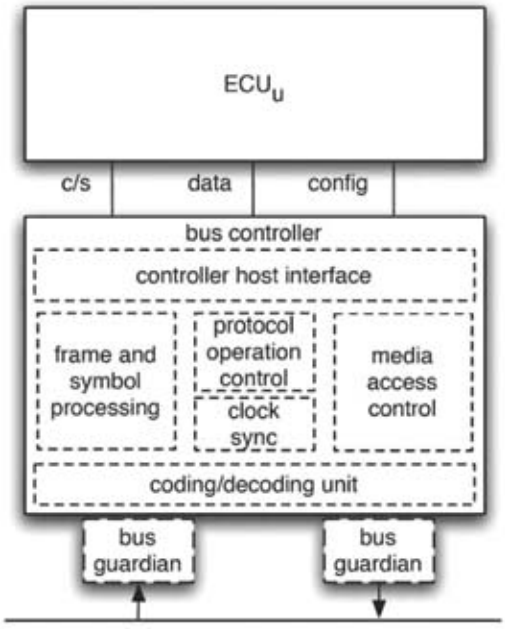


Fig. 8
Struttura completa di un nodo FlexRay: l'unità di controllo elettronico è collegata al controller del bus tramite le linee di configurazione, controllo e dati.

temporizzazioni del master. Anche qui il messaggio si chiama frame ed ha un proprio identificativo PID che ha la struttura di **Fig. 4**: la specifica prevede 64 PID (0x00...0x3F) di cui gli ultimi quattro riservati per comandi specifici del protocollo.
La tipica connessione del LIN è un connettore a due poli, dei quali spesso è collegato solo quello di segnale, perché come massa viene utilizzata quella dell'involucro del sensore o attuatore da leggere; almeno quando il tipo di segnale lo permette perché non soffre del cammino di corrente sulle masse dell'automobile nel punto dove si va a effettuare la connessione LIN (**Fig. 5**). Il connettore può anche avere tre poli.
Il segnale è tipicamente un'onda rettangolare unidirezionale dell'ampiezza di 12V e a vuoto, ossia senza collegamento alle unità Master, sulle Slave si rileva una frequenza dell'ordine di 4,8 kHz (assenza di dati). Il bus LIN è basato su un'interfaccia UART a 8 bit e supporta un'architettura di tipo Single Master/Multi Slave, vale a dire che un'unità master può gestire più unità slave, che nello specifico sono tipicamente sensori o attuatori.

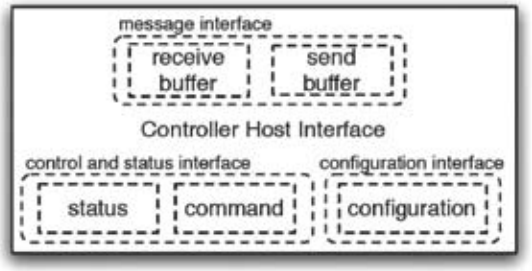


Fig. 9
Controller host interface.

Con questa architettura, l'aggiunta o la rimozione di un nodo comporta delle conseguenze soltanto sul nodo Master: i nodi Slave non vengono influenzati in alcun modo. Un'altra caratteristica del bus LIN è che il nodo Master è in grado di sincronizzare tutti gli Slave tramite un segnale di clock incorporato, pertanto non è necessario un filo di clock o un oscillatore esterno.
Ciò che rende pratico ed economico il LIN è che a livello fisico richiede soltanto un filo alimentato a 12 V (la massa del veicolo costituisce il conduttore di ritorno) e questo semplifica i cablaggi.

RETE FLEXRAY

Questo bus di comunicazione è una vera e propria rete nata per sistemi distribuiti. È stato sviluppato dal consorzio FlexRay fondato nel 1999, da BMW, Bosch, DaimlerChrysler, Freescale Semiconductor, General Motors, NXP Semiconductors e Volkswagen, cui si sono aggiunti Fiat, National Instruments, Renault e Fujitsu Microelectronics Europe GmbH.
Il FlexRay dispone di due canali di comunicazione, ciascuno in grado di fornire una velocità pari a 10 Mbps, che possono essere utilizzati insieme in modo da introdurre un meccanismo di ridondanza tale da garantire velocità e affidabilità (tolleranza verso i guasti) oppure indipendentemente l'uno dall'altro, permettendo di raggiungere una velocità aggregata pari a 20 Mbps. Il FlexRay è un protocollo di comunicazione flessibile (adattabile a molte situazioni) e fault-tolerant (ossia capace di recuperare gli errori senza che il sistema si blocchi).
Un sistema FlexRay è costituito da alcune ECU, ciascuna con un controller che gestisce l'accesso a uno o due canali di comunicazione. Tale sistema è il cluster FlexRay.
Anche per il FlexRay è definito il concetto di cluster e un cluster consiste in uno (*single-channel*) o due canali (*dual-channel*) e può essere realizzato tramite un bus, una rete a stella o con una configurazione ibrida. Nella topologia dual-channel, ciascuna ECU può essere connessa sia ad un solo canale che ad entrambi.
Alcuni esempi di configurazione dei cluster FlexRay sono illustrati in **Fig. 6**.
Nella "stella" il punto di connessione centrale è detto *star coupler*. Una rete ibrida è una combinazione di una rete a bus e di una a stella, pertanto un canale è realizzato come un bus e l'altro come una stella. Un esempio di tale configurazione è mostrato in **Fig. 7**.
Possono essere aggiunti i cosiddetti Guardian Bus, che consentono di incrementare la tolleranza

ai guasti, giacché permettono di fronteggiare le principali cause di guasti che si possono presentare in una topologia a bus o a stella. I Guardian Bus sono dispositivi posizionati tra il controller del bus e il canale nel caso di configurazione a bus, mentre sono posizionati nel centro stella nella topologia a stessa; essi disconnettono la ECU dal canale quando non è consentito loro di trasmettere dati.
Nelle ECU che supportano il FlexRay, il controller del bus è collegato all'interfaccia di I/O del microntrollore; il controller si interfaccia mediante:
- porta di controllo/stato (c/s);
- porta dati (data);
- porta di configurazione (config).

La struttura è mostrata nella **Fig. 8**. Il controller del bus è strutturato in sei componenti principali:
- interfaccia con l'host, la quale interfaccia la ECU con il bus controller;
- controllo operativo protocollo, che gestisce tutti i comandi del procollo FlexRay e relativi stati;
- controllo di accesso al mezzo; ha il compito di programmare l'assemblaggio dei messaggi e la trasmissione;
- elaborazione simboli e frame, che gestisce i messaggi ricevuti e separa l'header dai dati effettivi;
- unità di codifica e decodifica, che esegue fisicamente l'accesso in scrittura e lettura al bus e inoltre decodifica i messaggi ricevuti e codifica quelli da trasmettere;
- sincronizzazione del clock, che ha il compito di fornire un orologio locale per l'intero sistema, che deve essere sincronizzato con tutti gli altri.

Il blocco controller host interface (CHI) fornisce l'interfaccia tra il controller del bus e la ECU e consta di (**Fig. 9**):
- interfaccia messaggi per la porta dati
- interfaccia di configurazione per la porta config
- interfaccia di controllo e stato per la porta c/s.

Nell'interfaccia messaggi ci sono due buffer: uno di trasmissione (SB) e uno di ricezione (RB). Nel primo, l'host deposita il payload dei messaggi che devono essere inviati sul bus scrivendo tramite la porta dati. Oltre a questo buffer, è utilizzato un puntatore a questa area di memoria (SBP) ossia un contatore utilizzato come indirizzo per accedere all'SB. Quando l'host scrive dalla porta dati, i dati sono memorizzati all'indirizzo a cui punta l'SBP e quest'ultimo è incrementato di un'unità.
Dal secondo buffer l'host legge i dati che sono

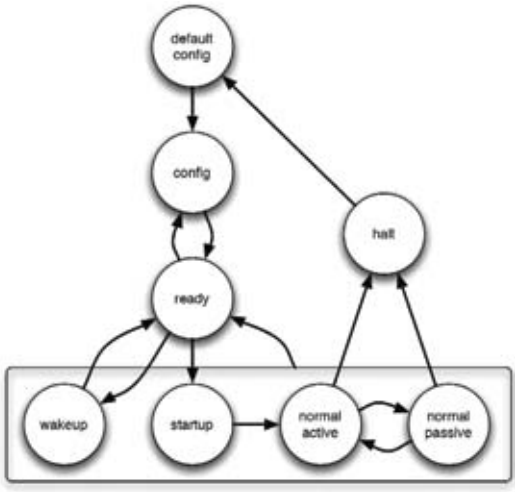


Fig. 10
La macchina a stati che descrive il funzionamento di un nodo FlexRay.

stati ricevuti; anche qui è previsto un puntatore (RBP) all'area di memoria RB. L'interfaccia di configurazione conserva alcuni parametri fondamentali scritti solo durante la fase di avvio, attraverso la porta di config. Infine, l'interfaccia c/s fornisce all'host utili informazioni circa lo stato di ricezione dei frame. Inoltre, l'host può interagire con lo stato del bus controller inviando dei comandi tramite la porta c/s.
Passiamo al Protocol Operation Control (POC), scopo del quale è rispondere ai comandi dell'host o a particolari condizioni, come stati di errore. Il POC imposta gli altri componenti nella condizione operativa appropriata. All'avvio, il POC parte dallo stato di default config, per poi passare alla configurazione del nodo nello stato config; il controller può essere configurato tramite la porta config solo in tali stati. Dopo la configurazione, il POC entra nello stato ready. A seconda dell'attività del canale di comunicazione, il controller andrà in wake-up o in startup; in quest'ultimo stato, il controller si integra nel cluster. Finchè non si verifica alcun errore, esso rimane nello stato normal active, mentre in caso contrario si porta in normal passive e cerca di reintegrarsi nel cluster. Invece se si verifica un fatal error, il controller blocca qualsiasi attività del nodo, mandandolo nello stato halt. Inoltre, all'host è con-

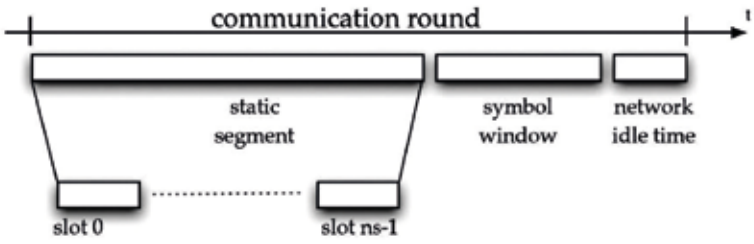


Fig. 11
Struttura di una communication round FlexRay: la presenza di un segmento statico con un numero fisso di slot garantisce una comunicazione con tempi deterministici.

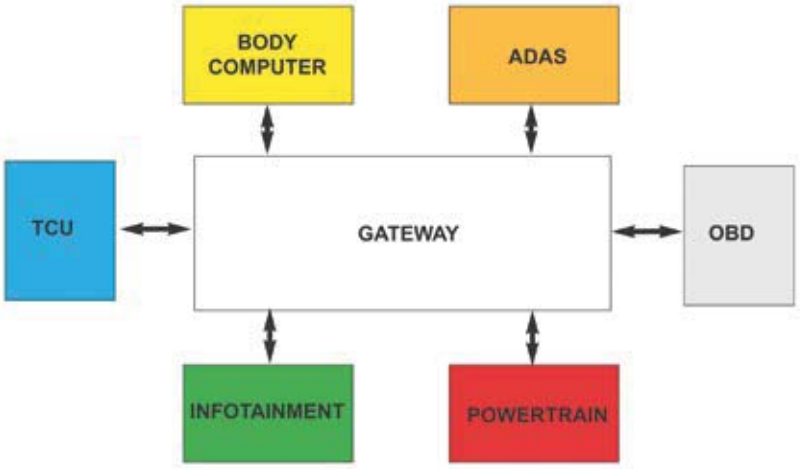


Fig. 12 Schema a blocchi di un sistema 100BASE-T1 con TCU.

cesso di modificare lo stato del controller. In ogni caso, il controller deve decidere se tale comando può o non può essere eseguito, in base allo stato attuale; per esempio, un comando di halt inviato dall'host è consentito solo se il nodo si trova in normal active o normal passive.
Il funzionamento del nodo FlexRay è definito dalla macchina a stati che vedete in **Fig. 10**.
Passiamo al controllo di accesso al mezzo, ossia alla rete: il media access control (MAC); questo componente gestisce la trasmissione dei dati dall'host verso il bus, programma l'accesso in scrittura al bus e aggiunge l'header al payload.
Nel protocollo FlexRay, a ciascuna ECU è concesso di trasmettere un messaggio solo all'interno di un certo intervallo temporale, chiamato slot. In ogni slot, il MAC verifica se la relativa ECU può trasmettere; in caso affermativo, il MAC importa il payload dal buffer di trasmissione (SB) e genera l'header del messaggio, poi l'unità di codifica/decodifica segnala quando il messaggio può essere scritto.
Quanto al componente frame and symbol processing (FSP), gestisce la ricezione dei dati; l'unità di codifica/decodifica segnala se è ricevuto un frame valido ricevuto. Dopo la ricezione, il messaggio viene suddiviso in header e payload, che viene copiato nel buffer di ricezione (RB). Inoltre l'FSP fornisce lo stato all'host dopo ogni slot, ossia l'informazione relativa all'ora locale per la sincronizzazione e allo stato di ricezione del frame (per esempio, se è stato ricevuto un frame errato nello slot precedente).
L'unità coding/decoding (CODEC) è il componente che fisicamente accede al bus. Quando il MAC segnala al CODEC di volere trasmettere un messaggio, quest'ultimo aggiunge il campo CRC per il controllo degli errori. A questo punto il messaggio viene codificato con gli opportuni valori di tensione.
Per la ricezione dei frame, il CODEC è in ascolto sul bus. Quando arriva un messaggio, innanzitutto viene verificata l'integrità dei dati, tramite controllo del CRC e se tutto risulta corretto, è notificato ad FSP l'arrivo del messaggio.
Infine, il componente di sincronizzazione del clock esegue due modi di sincronizzazione e genera un'unità per la scansione del tempo locale, detta macrotick. In questo modo il controller dispone di una base temporale per la schedulazione dei messaggi e degli interrupt verso l'host

Sincronizzazione del clock FlexRay

Il componente di sincronizzazione del clock prevede due modi di sincronizzazione e genera un'unità per la scansione del tempo locale, detta *macrotick*, utlizzata dal controller per schedulare i messaggi e gli interrupt verso l'host. La sincronizzazione è finalizzata a rendere uguale la lunghezza di tutti i macrotick nel cluster, evitando così che due nodi trasmettano nello stesso momento; l'oscillatore di clock può derivare ad esempio a causa del calore e lo standard FlexRay prevede una massima deviazione di ±0,15% il che implica che il clock di due nodi possa differire al massimo dello 0,3%.
La sincronizzazione viene effettuata tramite messaggi di *sync* e la correzione è effettuata secondo due criteri: correzione dell'offset e del rate.

Scheduling

Il protocollo FlexRay utilizza una schedulazione dei messaggi di tipo time-triggered: diversamente da un sistema event-triggered (come ad esempio il CAN-Bus) questo garantisce tempi di trasmissione deterministici, il che è fondamentale per i sistemi critici. L'accesso al bus è programmato tramite uno schema TDMA (Time-Division Multiple Access), in cui la scrittura sul bus è consentita solo entro gli slot (a ciascuna ECU ne è assegnato un certo numero). Ogni controller conosce i propri slot di trasmissione ma non quelli degli altri controller. La schedulazione di tutti i nodi è detta *communication round* (**Fig. 11**) e contiene un segmento statico con un numero fisso di slot, una symbol window e un tempo di inattività della rete. La symbol window è utilizzata per trasmettere messaggi speciali come i simboli di wakeup. Durante il periodo di inattività della rete, il componente per la sincronizzazione del clock calcola ed esegue la correzione.
Dopo l'accensione di un controller è richiesta una strategia di avvio perché il suo clock non risulta sincronizzato con gli altri. Inoltre, è richiesta una

procedura per il riavvio dopo un fatal error, poiché lo stato halt può essere lasciato solo passando per il default config. Per soddisfare tali condizioni, all'interno di un cluster FlexRay sono previste alcune ECU, dette *coldstart node*, capaci di inviare messaggi broadcast contenenti l'informazione di sincronizzazione. Dopo l'accensione, un algoritmo di elezione sceglie il nodo leader fra tutti i coldstart node, che invia in broadcast l'informazione del tempo agli altri controller, che così possono integrarsi nei cluster sincronizzando i loro clock.

ETHERNET IN AUTO

Nell'automotive ha preso piede la tendenza a utilizzare la rete Ethernet. Ideata dalla Xerox nel 1976, attualmente costituisce lo standard (IEEE 802.3 e sue varianti) più utilizzato e più semplice, tanto che viene adottato anche per realizzare reti tra computer sfruttando il protocollo di comunicazione TCP/IP (Transmission Control Protocol/Internet Protocol) cioè lo stesso di Internet, nel quale ogni elemento è identificato da un indirizzo composto da gruppi di tre cifre separati da punti (per esempio 192.168.1.2 nello standard IPv4).
Dagli iniziali 3 Mbit/s la velocità di comunicazione della ethernet è stata portata a 10 Mbit, poi a 100 Mbit/s e successivamente a 1 e 10 Gigabit. La più recente implementazione è la 100BASE-T1, specifica per l'automotive; classificata dall'IEEE come 802.3bw e nota anche come 100BASE-T1 (in passato era BroadR-Reach) è una Ethernet a 100 Mbps che consente di aumentare la quantità effettiva di dati attraverso l'utilizzo dei principi di base della sovrapposizione e degli schemi specifici di codifica e scrambling, che permettono di ridurre le interferenze elettromagnetiche (EMI), il peso del cablaggio, i costi e le dimensioni rispetto agli standard Ethernet 10BASE-T e 100BASE-TX.
Il 100BASE-T1 è un nuovo protocollo di comunicazione a livello fisico (PHY) che consente velocità di 100 Mbps su un cavo a singolo doppino ritorto non schermato, come quello del CAN-Bus; fino al suo avvento, l'ethernet non era molto diffusa nelle auto, anche se alcuni veicoli utilizzano il 100BASE-TX per gli strumenti diagnostici di bordo (OBD). Tuttavia quest'ultimo non ha avuto successo perché richiede due cavi a doppino ritorto e non soddisfa i severi limiti per le emissioni irradiate di Classe 5 del Comité international spécial des perturbations radioélectriques (CISPR). Invece il 100BASE-T1 consente data-rate effettivi di 100 Mbps con cavi lunghi anche 15 metri. Il profilo di emissione 100BASE-T1 è conforme al metodo stripline CISPR 25 Classe 5 Allegato G e ad altri standard di emissione automobilistici come il TC8 di Open Alliance.
Il 100BASE-T1 è in grado di abilitare la comunicazione di dati audio, video, connected car, firmware/software e dati di calibrazione all'interno di veicoli utilizzando la raccolta AVB (Audio Video Bridging) di protocolli Ethernet su cavo a singolo doppino ritorto non schermato. La raccolta di standard AVB presenta latenza ridotta e deterministica, nodi sincronizzati e shaping del traffico, tutti aspetti importanti per scambiare vari tipi di informazioni

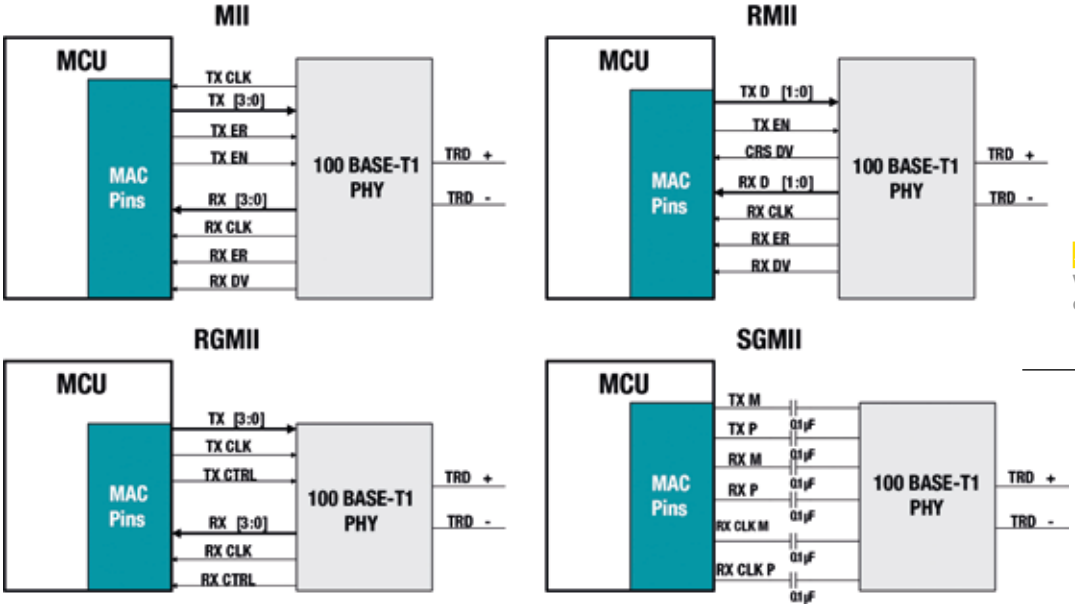


Fig. 13
Vari tipi di MII disponibili.

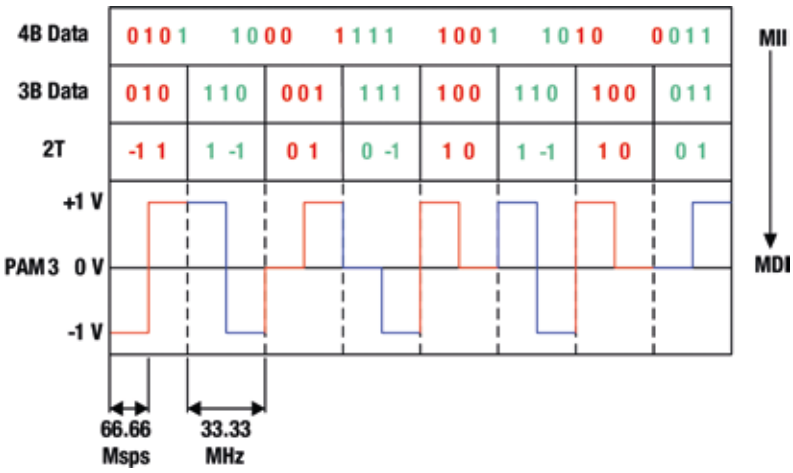


Fig. 14
Conversione di dati GPHY da MII a MDI.

nei sistemi automotive, e consentire il trasporto di diverse tipologie di dati con varie priorità (bassa velocità di trasmissione dati e alta priorità oppure velocità elevata e bassa priorità, nonché sincronizzazione temporale). La possibilità di trasportare, oltre ai dati per la gestione elettronica del veicolo, informazioni audio e video consente di integrare nella gestione di sistema sia l'infotainment, sia l'ADAS.
La ripartizione della larghezza di banda è gestita dall'AVB; l'impostazione predefinita assegna 75 Mbps ai flussi audio e video e i restanti 25 Mbps ai flussi di dati. Tuttavia, nelle applicazioni ADAS, i dati video grezzi forniti dalle telecamere possono richiedere larghezze di banda superiori a 1 Gbps, a meno che il video non venga compresso prima della trasmissione; ciò viene ottenuto integrando a bordo delle telecamere dei CODEC in grado di comprimere la banda, spesso integrati in microcontrollori dotati di interfaccia Ethernet sia per eseguire la compressione video sia per fornire un

Bit rate (Mbps)	Sampling rate (kHz)	Profondità campionamento (bit/sample)
1.4112	44.1	16
2.1168		24
2.8224		32
1.536	48	16
2.304		24
3.072		32
3.072	96	16
4.068		24
6.144		32

Tabella 1
Larghezza di banda audio per due canali.

layer MAC (Media Access Control) per le comunicazioni Ethernet. In generale tutti i dispositivi che si affacciano alla rete 100BASE-T1 devono avere un MAC address. Il 100BASE-T1 in combinazione con l'AVB può trasmettere sia dati audio che video, aprendo quindi opportunità per l'Ethernet nel campo dell'infotainment e dell'ADAS.
La **Tabella 1** mostra la larghezza di banda necessaria per l'audio in base alla frequenza di campionamento e alla profondità espressa bit per due canali; appare come il 100BASE-T1 risponde ai requisiti di larghezza di banda per audio a 44,1 kHz, 48 kHz e persino a 96 kHz campionati con profondità in bit fino a 32 campione. Con un'ampiezza di banda di 75 Mbps di AVB abbinata al 100BASE-T1 dovrebbe essere possibile gestire una coppia di canali video nel campo dell'infotainment.

Applicazioni Connected Car

Ma il 100BASE-T1 non si limita alle applicazioni AVB: una connessione fondamentale all'interno del veicolo è l'unità di controllo telematica (TCU), che controlla la comunicazione wireless da e verso il veicolo. La comunicazione dalla TCU al gateway (che interconnette i sottosistemi dell'autoveicolo, come si vede nella **Fig. 12**) consente l'accesso al web, facilitando gli aggiornamenti software/firmware via etere per varie ECU, ma anche interagendo con le infrastrutture stradali.
La maggior parte delle auto dispone di una porta OBD 100BASE-TX per la lettura dei dati diagnostici e l'aggiornamento o il flashing del software/firmware. Collegando varie ECU in un veicolo tramite 100BASE-T1 a un gateway centrale con una porta OBD, gli aggiornamenti possono essere eseguiti più rapidamente rispetto a bus di comunicazione come CAN, CAN FD e FlexRay. Il 100BASE-T1 può inoltre facilitare la calibrazione di una ECU in prossimità del completamento della produzione.

Codifica dei dati

Il 100BASE-T1 opera con un esclusivo schema di codifica 4-bit to 3-bit (4B3B), 3-bit to 2-ternary (3B2T) e a modulazione di ampiezza degli impulsi a tre livelli (PAM3) per ottenere emissioni ridotte rispetto al Fast Ethernet. Il 100BASE-T1 prescinde dal MAC in quanto la Media Independent Interface (MII) esistente non è cambiata rispetto all'ethernet comune. Al momento sono disponibili quattro xMII principali per 100BASE-T1:

- **MII**; è un'interfaccia dati a 4 bit (controlli di ricezione e trasmissione, clock di ricezione e trasmissione;

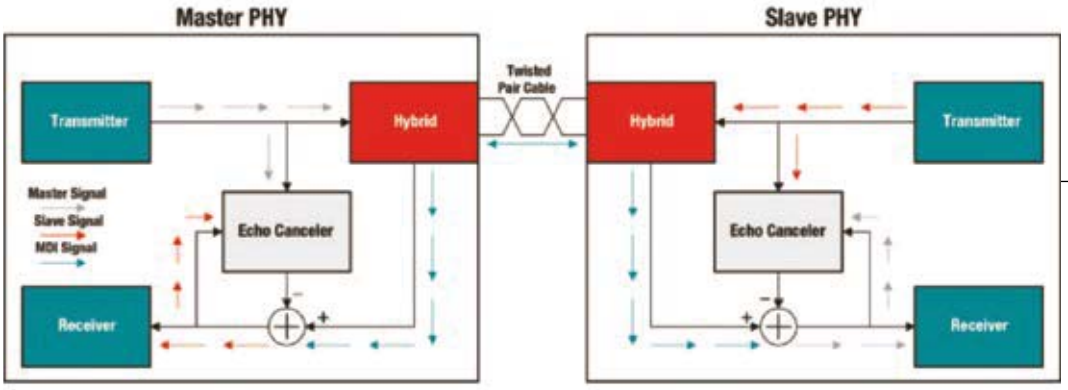


Fig. 15
Cancellazione dell'eco e schema a blocchi ibridi.

- **Reduced Media Independent Interface (RMII)**; è un'interfaccia dati a 2 bit, controlli di ricezione e trasmissione, riferimento clock singolo;
- **Reduced Gigabit Media Independent Interface (RGMII)**; interfaccia dati a 4 bit, con controlli di ricezione e trasmissione, clock di ricezione e trasmissione;
- **Serial Gigabit Media Independent Interface (SGMII)**; prevede un percorso di ricezione a due fili Low-Voltage Differential Signaling (LVDS) nonché un percorso di trasmissione anch'esso a due fili LVDS.

Per LVDS si intende l'utilizzo di impulsi di bassa tensione ma differenziali, ossia su linea bilanciata, che permettono di elevare l'immunità ai disturbi e il data-rate raggiungibile; per il funzionamento vale quanto spiegato sull'interfaccia fisica del CAN Bus. La **Fig. 13** mostra i vari tipi di comunicazione tra MAC e PHY a seconda dell'interfaccia.
Dopo aver ricevuto i dati dal MAC, il PHY (transceiver) esegue la codifica, lo scrambling e la serializzazione dei dati. Questi processi preparano i dati per il front-end analogico del PHY, che li trasmette quindi sul doppino verso l'unità destinataria. Ad esempio, un PHY 100BASE-T1 che comunica con un MAC tramite RGMII riceverà quattro bit paralleli con clock a 25 MHz (100 Mbps in totale). Il PHY converte questi quattro bit in tre bit e aumenta la frequenza di clock a 33 1/3 MHz per mantenere la velocità in bit di 100 Mbps (se un frame non è divisibile per tre, il PHY aggiunge bit di riempimento per abilitare la conversione corretta; il partner di collegamento rimuove questi bit di riempimento prima di trasferirli al MAC). Usando ciascun gruppo di tre bit, il PHY genera una coppia ternaria (2T) in base alla specifica mappa dei simboli. Infine, il vettore della coppia ternaria (TA, TB) viene trasmesso utilizzando la modulazione di ampiezza dell'impulso a tre livelli (PAM3) a una frequenza fondamentale di 66 2/3 MHz. La **Fig. 14** mostra i dati convertiti da MII alla Medium Dependent Interface (MDI) attraverso il PHY: osservando il segnale PAM3 si nota come ogni periodo di 33 1/3 MHz rappresenti tre bit di dati e quindi trasferisca i dati a 100 Mbps. Questo segnale viene inviato utilizzando tre livelli di tensione (+1 V, 0 V e -1 V) con meno di 2,2 V da picco a picco (se misurati con terminazione differenziale 100 O). Il PAM3 viene trasmesso sull'MDI, che include il connettore per il cavo e il cavo a singolo doppino ritorto stesso, nonché tutti i componenti passivi esterni utilizzati per il filtraggio passa-basso aggiuntivo e la reiezione in modalità comune (CM). Il connettore MDI e i cavi a doppino ritorto non schermato devono soddisfare specifiche come l'attenuazione di riflessione, la perdita per conversione e la tolleranza ai guasti, definite nella Sezione 96.8 della specifica MDI dello standard 100BASE-T1.
I protocolli Ethernet hanno specifiche differenti per lo scrambling, la serializzazione e la codifica dei dati in base alla loro applicazione di destinazione. Per confronto, il 100BASE-TX trasmette i dati a 125 MHz utilizzando il Multi-Level Transmit (MLT-3). La frequenza fondamentale è superiore a quella del 100BASE-T1 (66 2/3 MHz) e richiede un doppino ritorto dedicato per la trasmissione e la ricezione. Il 100BASE-T1 richiede solo circa 33,3 MHz di larghezza di banda, quasi la metà del 100BASE-TX. In pratica, è possibile utilizzare un cavo di qualità inferiore (riduzione dei costi) offrendo al contempo un filtraggio migliore per migliorare emissioni e immunità, entrambi fondamentali nelle applicazioni automobilistiche. Inoltre, l'efficienza spettrale aumenta quando si utilizza il metodo di 100BASE-T1 di mappatura del segnale 3B2T e la modulazione PAM3 sulla codifica 4B5B di 100BASE-TX in MLT-3, diminuendo quindi le emissioni attraverso

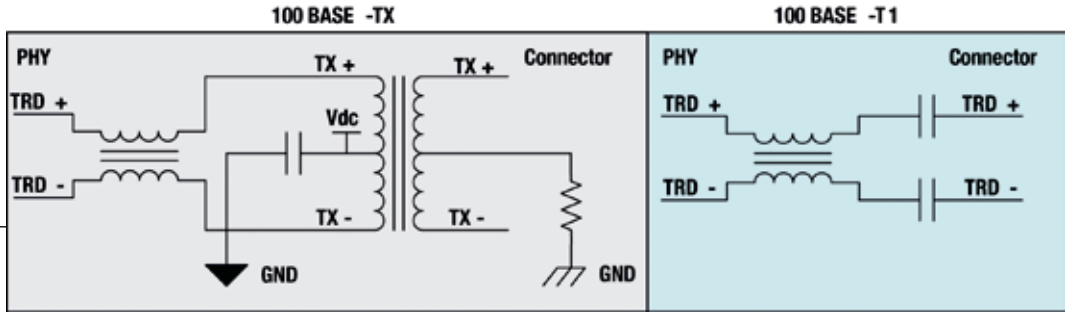


Fig. 16
Confronto fra isolamento CC 100BASE-TX e 100BASE-T1.

la riduzione della larghezza di banda necessaria per inviare la stessa quantità di dati.

Interfaccia TX/RX a singolo doppino

Il 100BASE-T1 è un'interfaccia fisica full duplex che consente la trasmissione e la ricezione sullo stesso doppino, a differenza di 10BASE-T e 100BASE-TX, dove trasmissione e ricezione avvengono su doppini dedicati. I supporti condivisi riducono il peso complessivo dei cavi nel veicolo, riducendo i consumi. Il full duplex fisico è realizzato attraverso i principi di sovrapposizione. I PHY 100BASE-T1 presentano ibridi integrati e utilizzano la cancellazione dell'eco per rimuovere il proprio segnale trasmesso ed estrarre le informazioni ricevute da un partner di collegamento; allo scopo, un PHY funge da master e l'altro da slave. Quando due PHY 100BASE-T1 si connettono tra loro avviano un processo di addestramento che porta sia il dispositivo in prova (DUT, Device Under Test) che il partner di collegamento a trasmettere informazioni alla stessa frequenza e con la stessa fase. La **Fig. 15** schematizza la cancellazione ibrida e dell'eco all'interno di ciascun PHY.

Prima che la segnalazione PAM3 esca dalla scheda o vi entri, vengono implementati dei condizionamenti mirati a isolare l'MDI per evitare loop di massa e offset CC del driver, ridurre il rumore CM (modo comune) e ridurre le emissioni irradiate, mantenendo un'elevata immunità.

Per quanto riguarda il disaccoppiamento in CC, i PHY 100BASE-TX utilizzano in genere trasformatori con presa centrale (sul lato PHY) collegata a una tensione CC dipendente dal PHY. I trasformatori utilizzano anche la terminazione Bob Smith (presa centrale, sul lato del connettore, collegata a massa attraverso un resistore) per contribuire a migliorare il filtraggio del rumore CM.

Il 100BASE-T1 ha un approccio più semplice, basato sull'utilizzo di soli due condensatori che forniscono l'isolamento CC e riducono le dimensioni della soluzione rispetto a un'applicazione con un trasformatore.

La **Fig. 16** mostra un'implementazione tipica del circuito 100BASE-T1. Invece nella **Fig. 17** è possibile vedere la schematizzazione del tipico sistema 100BASE-T1 al completo, partendo dal microcontrollore e arrivando al doppino ritorto che consente la comunicazione con i sottosistemi di bordo.

CONCLUSIONI

Bene, con questo abbiamo terminato la nostra carrellata sui bus di comunicazione utilizzati nelle automobili moderne e che certamente equipaggeranno anche le auto del futuro, a prescindere da quale ne sarà la motorizzazione.

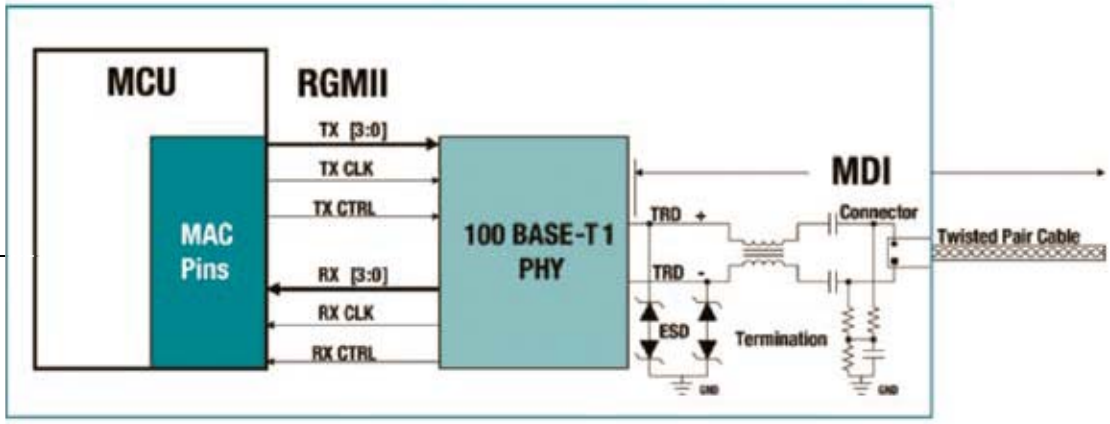


Fig. 17
Implementazione tipica della 100BASE-T1.

STAMPANTE 3D4040

oggi ancora più ricca di accessori!

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 montaggi in italiano con illustrazioni

Versione montata
cod. 3D4040/M
€ 1.299,00 IVA inclusa.

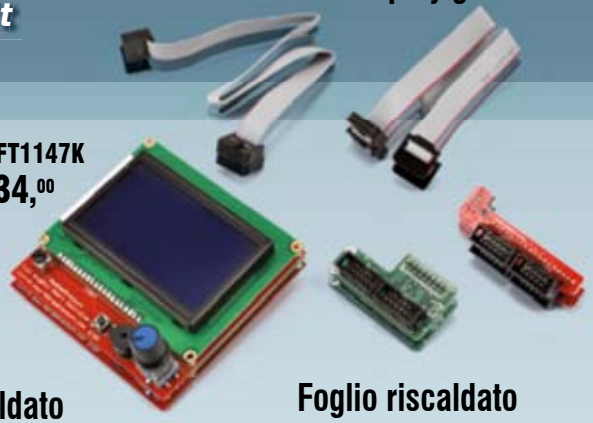
VASTA GAMMA DI ACCESSORI!
SCOPRILI TUTTI SU
www.futurashop.it

Box in plexiglass per stampante 3D4040

NOVITÀ!

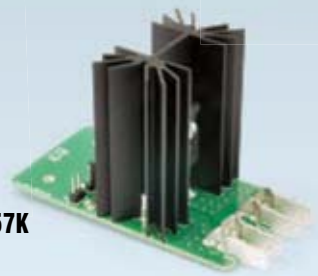
Cod. BOX3D4040
€ 198,00

Controller per stampa autonoma con display grafico



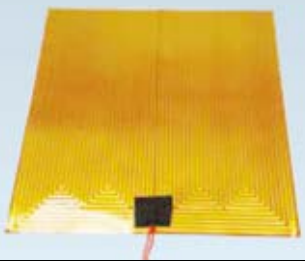
Cod. FT1147K
€ 34,00

Relè allo stato solido per piatto riscaldato



Cod. FT1357K
€ 15,00

Foglio riscaldato in kapton 40x40cm



Cod. 3D4040HPLATE
€ 65,00

Hai comprato questa rivista e ti è piaciuta?

Abbonati oggi stesso
e ti sconteremo il prezzo
di questo numero
sul tuo nuovo
abbonamento!

Hai capito bene!

Per poter avere il prezzo scontato manda una mail
a ***community@elettronicain.it*** indicando i tuoi dati e se preferisci ricevere
la rivista in formato digitale o cartaceo, riceverai un codice coupon del valore
di 6 € per sottoscrivere il tuo nuovo abbonamento a Elettronica In

e ricorda che abbonandoti avrai :

- La **rivista puntuale** ogni mese direttamente a casa tua
  (il primo giorno del mese per l'abbonamento digitale)
- **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"
  (solo con l'abbonamento cartaceo)
- La possibilità di leggere **Gratis** la versione digitale di Elettronica In
  (solo con l'abbonamento cartaceo)

PER ABBONARTI COLLEGATI AL SITO

www.elettronicain.it

Futura Group Edizioni

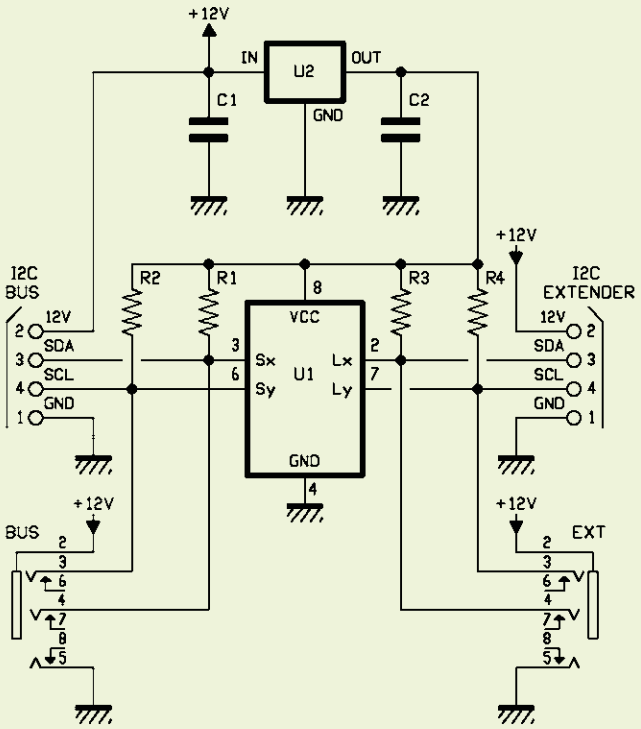
FUTURA GROUP srl
Via Adige 11
21013 Gallarate (VA)
TEL.: 0331 752668

I²C EXTENDER

Estendiamo la distanza di collegamento tra dispositivi a interfaccia I²C-Bus, superando i limiti imposti da capacità parassite e resistori di pull-up.

di **BORIS LANDONI**

Per quanto sia molto, anzi moltissimo utilizzato in elettronica per interconnettere dispositivi a bordo della stessa scheda o sensori e componenti nello stesso apparato, il bus I²C (e il suo derivato, l'SM-Bus) soffre di un paio di limiti, che derivano essenzialmente da come funziona: la velocità di comunicazione e la distanza massima alla quale possono trovarsi i device Slave rispetto al Master. La copertura, se così vogliamo chiamarla, è limitata dal fatto che nell'I²C-Bus, per avere una linea dati bidirezionale si sfruttano dispositivi di output open-collector



(o open-drain) che consentono di portare la linea a livello basso senza danneggiarsi reciprocamente, mentre lo stato logico alto vene mantenuto, in assenza di conduzione da parte dei transistor, grazie a un resistore di pull-up. Il valore di quest'ultimo forma, insieme alle capacità parassite (tipicamente 400 pF) dei dispositivi affacciati sul bus (nel caso di input a MOSFET si tratta della capacità di gate, che è rilevante), una costante di tempo parassita tale da limitare il massimo data-rate implementabile.

Il prototipo con montate le morsettiere, senza le prese jack.

La situazione viene peggiorata dalla distanza di collegamento e dalla lunghezza delle linee.
Questo limite può essere superato, almeno se basta collegare tra loro solamente due apparati (un Master e uno Slave), interfacciando i dispositivi mediante una coppia di transceiver come quello descritto in queste pagine, che nasce per irrobustire il segnale quanto basta a superare distanze interessanti per molte applicazioni domotiche o IoT; unico vincolo è l'utilizzo, per la connessione, di un doppino ritorto, ovvero di un cavo dati a coppia twistata o un cavo di rete ethernet cat. 5e.
Un transceiver avrà come ingresso/uscita il dispositivo Master I²C-Bus e l'altro il dispositivo Slave.

SCHEMA ELETTRICO

Il circuito è molto semplice e lo potete vedere dal relativo schema elettrico che trovate in queste pagine; il merito è tutto dell'integrato P82B715, prodotto dalla NXP (ma anche dalla Texas Instruments) che contiene un doppio driver in grado di elevare la corrente assorbita a livello basso dalla linea, la quale ha ovviamente la possibilità di erogarla dall'alimentazione tramite i resistori di pull-up. Sostanzialmente opera come una sorta di driver a loop di corrente, capace di convertire i 3mA tipici di partenza in 30 mA sui cavi di collegamento. Nella **Fig. 1** potete vedere lo schema a blocchi dell'integrato, che nello schema elettrico abbiamo siglato U1.
Questo componente, in un package SMD SO-8 a quattro piedini per lato integra due buffer in grado di erogare, pilotati dai segnali applicati alle linee di ingresso, una corrente che arriva a 30 mA e perciò garantisce connessioni fino a 20 metri con ottima immunità ai disturbi, tanto più se realizzata tramite cavo dati schermato come, ad esempio, quello per reti a 100 Mbit (cat. 5e).
Ciascun buffer ha un piedino di ingresso ed uno di uscita, perciò nell'integrato abbiamo due linee di input (SCL ed SDA) facenti capo ai piedini Sx (SDA) e Sy (SCL) che si interfacciano al dispositivo I²C-Bus da affacciare sul "bus potenziato" e altrettante di output localizzate ai piedini Lx (corrispondente a Sx, quindi relativa all'SDA) e Ly (relativa a Sy e quindi all'SCL) che affacciano il bus sul cavo che lo interconnetterà al dispositivo gemello.
In verità non è esatto parlare di input e output perché la linea dati del bus I²C è bidirezionale, quindi a stretto rigore si dovrebbe definire il canale Sx (pin 3)/Sy (piedino 6) come lato dispositivo I²C-bus e quello Lx (pin 2)/Ly (pin 7) come lato cavo o interfaccia long-range o extender.

In ogni caso, come vedete nello schema elettrico, sono presenti i canonici resistori di pull-up sia sul lato device sia su quello del cavo di interconnessione con il circuito gemello; però R1 e R2 (lato device I²C-Bus) sono di valore relativamente elevato e standard per la connessione I²C-Bus, mentre quelli dal lato del cavo, dovendo il driver erogare correnti più elevate, hanno valore inferiore. Il tutto funziona comunque a 5 V, sia lato device, sia lato cavo.
I resistori di pull-up servono perché di fatto i buffer contenuti nell'integrato non sono altro che dei ripetitori degli stati logici, ovvero del funzionamento dei transistor (sovente MOSFET...) presenti sui pin SDA ed SCL dei canonici dispositivi a interfaccia I²C-Bus: quindi, prendendo in considerazione la linea SDA, in presenza dello zero logico su Sx la linea Lx si porta in modalità sink e chiude a massa il piedino 2, mentre con 1 logico (open) da parte del device I²C-Bus la stessa rimane open e consente al dispositivo che si trova dall'altro lato del cavo di condizionare lo stato della linea SDA.
Ogni buffer è bidirezionale, quindi se quando il device collegato al lato I2C BUS del circuito è in stato open su una certa linea, per esempio SDA, quello dall'altra parte invia lo zero (ossia chiude a massa) SDA dal lato I2C EXTENDER del circuito, l'ingresso Lx va a condizionare in tal senso Sx. In breve, con Sx a zero logico Lx va a livello basso e non può essere mutata dal lato I2C EXTENDER, mentre se Sx è a livello alto (open) l'eventuale zero logico su Lx la porta anch'essa a zero.
Lo stesso vale per la linea SCL, che essendo il clock, tuttavia, è unidirezionale, cioè se il segnale di clock viene applicato ad SCL e quindi ad Sy, Ly lo segue e il piedino Sy dell'integrato che si trova sulla scheda gemella dall'altra parte del bus I²C esteso fa altrettanto.
Ovviamente il buffer della linea di clock è bidirezionale non perché SCL ammetta il passaggio in entrambe le direzioni, ma perché si utilizza lo stesso integrato nei due circuiti posti alle estremità del cavo di estensione dell'I²C-Bus e pertanto deve poter condizionare il livello o subirlo, a seconda di qual è il lato connesso al Master del bus; infatti il Master dell'I²C-Bus è il dispositivo che non solo può avviare la comunicazione, ma anche quello che genera il clock e lo invia ai device Slave.
Sul piano delle connessioni con l'esterno, il circuito che vi proponiamo prevede due possibilità di connessione con la linea estesa e i dispositivi a interfaccia I²C-Bus: tramite morsettiere o mediante spinotti, ovvero prese jack da 3,5" a quattro poli (opzionali).
Nello schema elettrico, i jack sono siglati BUS ed EXT: il primo si collega al dispositivo I²C-Bus da estendere (corrisponde all'ingresso I2C) e il secondo al cavetto che termina sulla scheda gemella (equivale al lato I2C EXTENDER).
Dato che in questa connessione transita anche l'alimentazione di ingresso del circuito, se si utilizzano le prese jack è consigliabile effettuare la connessione e disconnessione degli spinotti quando ancora non è stata applicata l'alimentazione.
In tema di alimentazione, il circuito accetta in ingresso tensioni continue di valore compreso fra 8 e 12Vcc e il regolatore integrato U2 (un comune 7805...) ottiene, a partire da tale valore, 5 volt ben stabilizzati e filtrati dal condensatore C5, che alimentano sia l'integrato U1, sia i resistori di pull-up delle linee I²C-Bus e linea estesa.

CARATTERISTICHE TECNICHE

- **Tensione di alimentazione:** 12Vcc
- **Corrente max assorbita:** 40 mA
- **Distanza superabile:** 20 m
- **Frequenza clock supportata:** 400 kHz

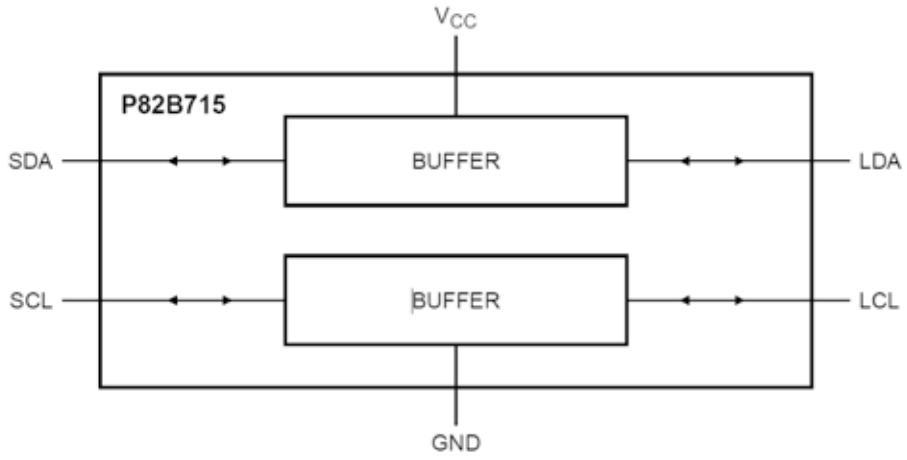
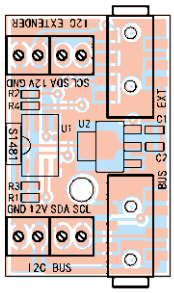


Fig. 1 Schema a blocchi dell'integrato.

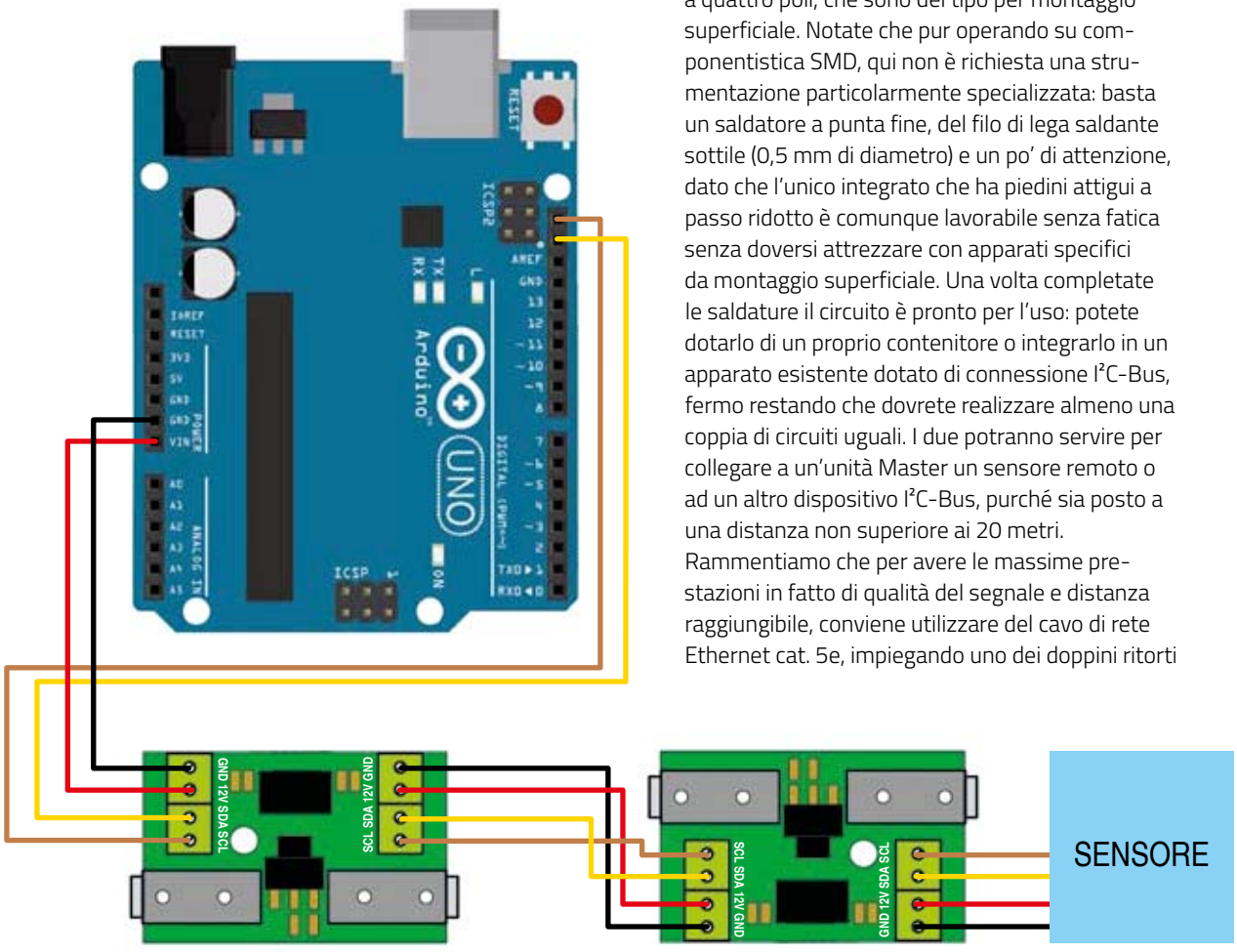


Elenco Componenti:

R1, R2: ,7 kohm (0603)
R3, R4: 470 ohm (0603)
C1, C2: 100 nF ceramico (0603)
U1: P82B715TD,112
U2: LD1117S50TR

Varie:
- Morsetto 2 vie passo 2.54mm (4 pz.)
- Circuito stampato S1481 (22x35 mm)

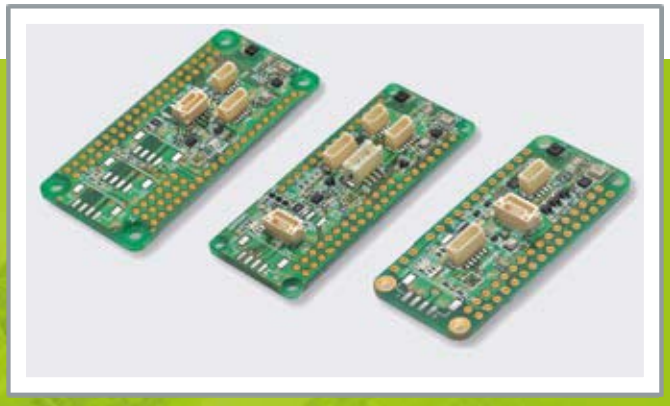
Schema di cablaggio



REALIZZAZIONE PRATICA

Bene, con questo abbiamo concluso la spiegazione dello schema elettrico e possiamo passare alle note costruttive.
Il circuito occorrente per realizzare ciascun modulo di interfaccia è piccolo e compatto, nonché di facile realizzazione, essendo composto da pochissimi elementi, quasi tutti SMD; è comunque realizzato su una basetta a doppia ramatura, per costruire la quale occorre scaricare le due tracce lato rame dal nostro sito www.elettronicain.it e creare da esse le due pellicole con cui procedere per fotoincisione.
Una volta inciso e forato il circuito stampato bisogna montare dapprima i componenti passivi (resistenze e condensatori) e poi gli attivi (i due integrati) seguendo, per questi ultimi, l'orientamento indicato nel piano di montaggio che trovate in queste pagine. Completate il tutto con le morsettiere miniatura (sono gli unici componenti del circuito a montaggio tradizionale, cioè a foro passante) e, se pensate di utilizzare anche la connessione con cavetti dotati di jack, con le prese jack da 3,5 mm a quattro poli, che sono del tipo per montaggio superficiale. Notate che pur operando su componentistica SMD, qui non è richiesta una strumentazione particolarmente specializzata: basta un saldatore a punta fine, del filo di lega saldante sottile (0,5 mm di diametro) e un po' di attenzione, dato che l'unico integrato che ha piedini attigui a passo ridotto è comunque lavorabile senza fatica senza doversi attrezzare con apparati specifici da montaggio superficiale. Una volta completate le saldature il circuito è pronto per l'uso: potete dotarlo di un proprio contenitore o integrarlo in un apparato esistente dotato di connessione I²C-Bus, fermo restando che dovrete realizzare almeno una coppia di circuiti uguali. I due potranno servire per collegare a un'unità Master un sensore remoto o ad un altro dispositivo I²C-Bus, purché sia posto a una distanza non superiore ai 20 metri.
Rammentiamo che per avere le massime prestazioni in fatto di qualità del segnale e distanza raggiungibile, conviene utilizzare del cavo di rete Ethernet cat. 5e, impiegando uno dei doppini ritorti

SENSORI IoT SEMPRE CONNESSI



Omron ha recentemente proposto la sua nuova piattaforma di sviluppo 2JCIE-EV01 composta da tre schede, delle quali la host monta a bordo 6 sensori di parametri ambientali come temperatura, umidità, pressione, luce, suono e accelerazione; si tratta di una scheda di sviluppo nata per offrire ai programmatori che lavorano sui sistemi di acquisizione per IoT un rapido percorso verso la creazione di un prototipo "proof of concept" ospitato su board Raspberry Pi, Arduino o Adafruit Feather. La scheda è supportata da un software di acquisizione dati e integra connettori di espansione per interfacciarsi con il sensore termico D6T, il sensore digitale di pressione differenziale MEMS D6F-PH, il sensore analogico di flusso D6F-P, il sensore ottico B5W-LB o il rilevatore di qualità dell'aria/polvere B5W-LD, tutti della Omron.
Un ulteriore connettore a bordo consente di collegare alla scheda qualsiasi sensore compatibile Qwiic.
Il sensore interno della scheda di sviluppo si basa sul modulo di rilevamento ambientale USB 2JCIE-BL01 Omron, che supporta la registrazione dati on-board per circa 3 mesi; oltre che alla scheda host, l'unità può connettersi a più dispositivi, per esempio gli smartphone, tramite Bluetooth 5.0. Sulla scheda possono essere impostati i valori di soglia personalizzati per generare degli alert che informano del verificarsi di eventuali letture anomale.

in esso contenuti per i dati e, possibilmente, collegare da entrambi i capi la maglia di schermo del cavo alla massa di ciascuna delle schede.
Collegare lo schermo consente di elevare l'immunità ai disturbi, contribuendo, insieme al funzionamento in corrente degli extender, a rinforzare la comunicazione sulle lunghe distanze.
L'alimentazione a 12 volt non è necessario trasportarla, o meglio, serve farlo solamente se il dispositivo remoto cui collegherete la scheda non ha una propria alimentazione ma va alimentato con il dispositivo locale: per esempio nel caso di sensori remoti per sistemi domotici e di monitoraggio ambientale, ovvero applicazioni IoT che prevedano il rilevamento di parametri ambientali o condizioni di funzionamento di macchinari e apparati di vario genere. Insomma, quando non è possibile o risulta sconveniente trovare una fonte di alimentazione per ciò che si trova dal lato Slave.
Analogamente, potrebbe essere necessario disporre di un'unità Master remota e di sensori o comunque dispositivi Slave locali e avere la Master in un luogo privo di fonti di elettricità: è il caso di unità a microcontrollore che elaborano sul posto o comunicano in wireless.

CONCLUSIONI

Il progetto descritto in queste pagine è un ottimo traslatore di livelli di corrente che permette di rendere immune ai disturbi la comunicazione su I²C-Bus a distanze che normalmente non potrebbero essere coperte da un protocollo che, per sua natura, è riservato alla comunicazione dati su scheda tra dispositivi (tipicamente microcontrollori) Master periferiche quali sensori, memorie, I/O Expander a interfaccia parallela.
Ricordiamo che per ciascuna linea si deve utilizzare una coppia di circuiti, uno da connettere all'unità Master e l'altro alla Slave; in teoria si potrebbero connettere anche più Slave sulla stessa linea I²C a valle del dispositivo periferico, in quanto nel bus I²C esiste un unico Master e ciascuno Slave impegna il canale dati solo quando il comando inviato dal Master è diretto ad esso, pertanto a comunicare ci sarà solo uno Slave alla volta.

Cosa occorre?

I componenti utilizzati in questo progetto sono disponibili presso Futura Elettronica. La breakout board I2C extender (cod. FT1481) è in vendita a Euro 14,00 mentre la scheda Arduino UNO (cod. ARDUINOUNOREV3) è disponibile a Euro 24,50. 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

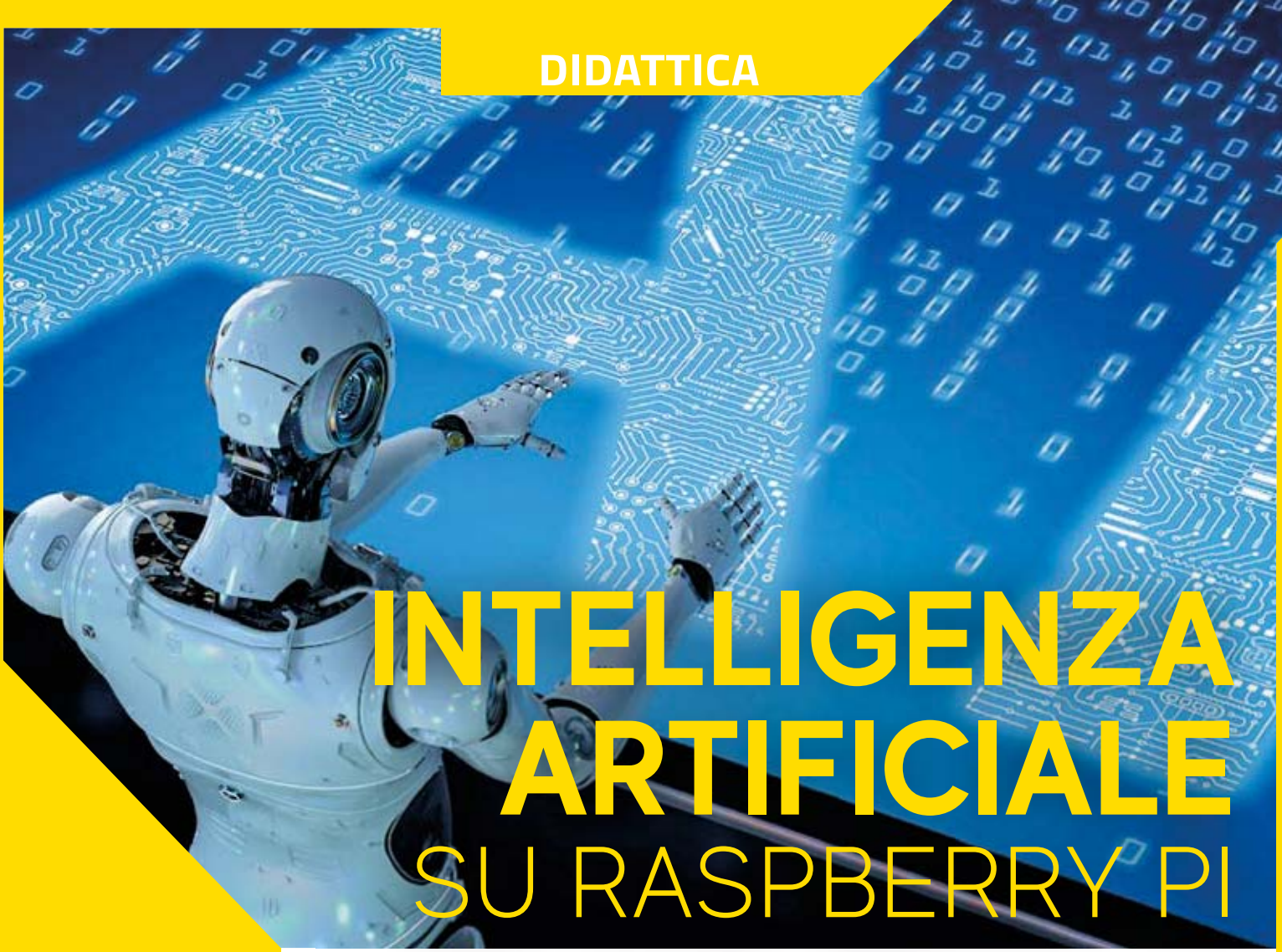
Sfoglia on-line

FUTURA ELETTRONICA®

i nostri cataloghi

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

www.futurashop.it



INTELLIGENZA ARTIFICIALE SU RASPBERRY PI

dell'Ing. **DANIELE DENARO**

Spingiamoci oltre le Neural Network di base e sperimentiamo l'AI sulla scheda Raspberry Pi e su micro-hardware specializzato.

Nella serie di articoli dedicati all'AI proposta nei numeri passati di Elettronica In, abbiamo visto come sia possibile utilizzare la tecnologia delle Reti Neurali anche in microcontrollori limitati come quelli delle schede Arduino e simili. Naturalmente in tali contesti applicativi il loro utilizzo è confinato ai compiti semplici, dove le soluzioni del genere possono vantaggiosamente sostituire la programmazione classica. Ricordiamo, infatti, che l'approccio tramite il paradigma "Machine Learning", di cui le Reti Neurali sono l'implementazione più importante e promettente, è quello di lasciar perdere il tentativo di descrivere una procedura logica che possa affrontare il problema da risolvere mediante i classici linguaggi di programmazione. Al contrario, si lascia ad un algoritmo generale di apprendimento il compito di trovare da solo la corrispondenza funzionale tra i valori di input e di output, mediante una serie di esempi da sottoporre all'algoritmo stesso (**Fig. 1**).
Occorrono quindi una struttura e un metodo di apprendimento; una struttura basilare può essere quella delle Reti Neurali Feedforward a due strati (chiamate anche Multilayer Perceptron) schematizzata nella **Fig. 2**. La libreria proposta per

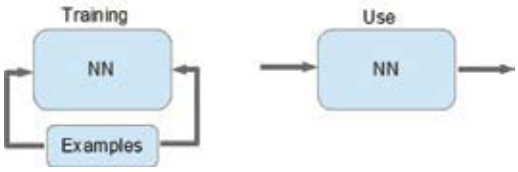


Fig. 1 Machine Learning: struttura che impara.

Arduino (*github.com/open-electronics/Artificial_Intelligence*) permette di definire semplicemente una rete di questo tipo e di imparare dagli esempi mediante il classico metodo della "discesa del gradiente" (SGD: Stochastic Gradient Descent). La caratteristica principale, che permette l'utilizzo su Arduino, è di poter addestrare velocemente la NN su un hardware potente come un PC e poi portare la struttura addestrata e operativa su Arduino, utilizzando la memoria Flash di programma. Così una qualunque corrispondenza tra input sensoriale ed azione di output può essere velocissima a prescindere dalla sua complessità funzionale. Per esempio, un rover può essere comandato in base ai sensori di distanza in modo da realizzare un comportamento anche sofisticato.
Tra gli esempi della libreria proposta ce n'è uno che non può girare su Arduino, ma sicuramente su Raspberry Pi, ed è stato inserito per consentire ai curiosi di verificare come anche una semplice rete a due strati possa risolvere, in maniera soddisfacente, il problema del riconoscimento delle cifre scritte a mano. Questo "data-set" di immagini (28x28) in scala di grigi è conosciuto con il nome di MNIST ed è uno dei test standard per i *benchmark* tra i sistemi di *Machine Learning*. Una NN a due strati può risolvere questa operazione di riconoscimento con una percentuale positiva maggiore del 90%, ma strutture più sofisticate possono fare di meglio soprattutto possono affrontare compiti più gravosi. Queste strutture di NN hanno più di due strati e per questo motivo vengono chiamate "deep" (profonde). Il Deep Learning è in realtà più complesso della semplice sovrapposizione di più strati, in quanto in esso intervengono anche modifiche strutturali dei vari strati, che possono non essere strati completamente connessi.

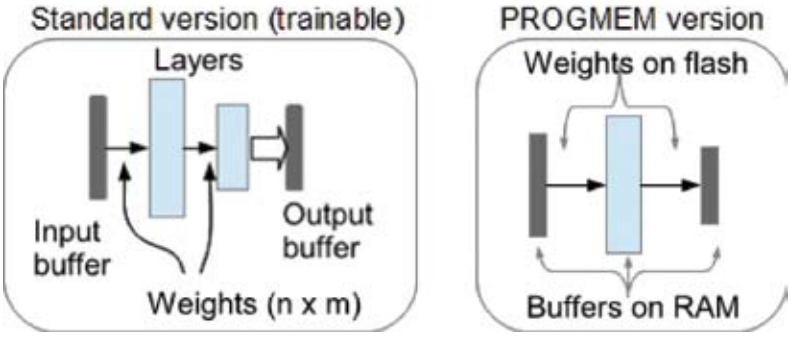


Fig. 2 Rete Neurale feedfoward a 2 strati per Arduino.

DEEP LEARNING E TENSORFLOW

Per esempio, le reti per il riconoscimento di immagini sono attualmente reti convolutive (CNN). In queste reti le immagini vengono filtrate in parallelo con una serie di filtri (che sono anch'essi strati che imparano). Oppure, per esempio, nelle reti ricorrenti (RNN) per l'analisi dei testi o del parlato, sono presenti strutture (addestrabili) che sanno ricordare o dimenticare gli stati passati. Per strutture così complesse occorrono librerie ed ambienti di sviluppo potenti, per poterle descrivere, addestrare ed utilizzare.
Tenete presente che strutture di questo tipo possono contenere anche milioni di parametri, che poi non sono altro che i pesi delle connessioni. Anche i nodi della rete (quelli che fanno le elaborazioni in base al loro input) possono avere diversi tipi di funzioni di attivazione.
Il tutto deve poi automatizzare il metodo di addestramento "SGD" pur nella complessità della struttura. Inoltre si devono poter utilizzare alcuni metodi di ottimizzazione dello SGD che nel frattempo sono stati proposti. Sono nate, allora, librerie complesse come: Teano, Caffè, PyTork, Darknet ecc. Una di queste è diventata, in un certo senso, leader del mercato: TensorFlow.
Sono tutti framework destinati a descrivere strutture anche con decine di milioni di parametri; è naturale quindi pensare di utilizzare hardware spinto con elaborazione parallela come per esempio gli acceleratori grafici (GPU) basati sull'architettura ad alto parallelismo CUDA (Compute Unified Device Architecture) di NVIDIA. Infatti questi framework sono predisposti per poter utilizzare l'elaborazione parallela. D'altronde la struttura stessa delle Reti Neurali, basata sull'elaborazione dei singoli nodi della rete, si presta bene a questo tipo di calcolo distribuito.
Su questo hardware sofisticato, anche le complesse architetture "convoluzionali" destinate all'elaborazione di immagini, riescono a funzionare in tempo reale (ma non per l'addestramento che può comunque richiedere diverse ore).
Ma allora, per l'hardware "normale" non c'è speranza? In realtà a causa dell'aumento di potenza delle CPU e dell'introduzione di reti semplificate (strutture tiny, lite, o Mobilenet) è possibile portare sofisticati sistemi di riconoscimento di oggetti anche su hardware limitato come Raspberry Pi o smartphone. Inoltre sempre più spesso micro-computer o micro-controller nascono con un appendice destinata al

calcolo parallelo. Come per esempio lo Jetson nano di NVIDIA o i microcontroller di Sipeed che utilizzano il modulo K210 (**Fig. 3**).
TensorFlow è gratuito ed *open-source*. È stato implementato da Google per i suoi laboratori di AI e messo successivamente a disposizione di tutti. In realtà le API a basso livello costruiscono un grafico computazionale. Ovvero mentre la struttura software tradizionale si basa su una elaborazione sequenziale delle istruzioni (architettura von Neumann), una modalità alternativa è quella detta Dataflow (**Fig. 4**). In questo ultimo caso un grafo di nodi viene elaborato parallelamente, nel senso che un nodo è computato non appena i dati di input sono acquisiti (elaborazione per flusso di dati).
Cosa si ottiene con questa apparente complicazione?
Si ottengono due importantissimi effetti:

- L'elaborazione può essere automaticamente distribuita su più unità di calcolo.
- Il calcolo del gradiente può essere notevolmente semplificato, generalizzato ed automatizzato (con la regola della catena delle derivate).

Infatti il calcolo del gradiente, fondamento della ***back-propagation***, è il cuore delle reti neurali che apprendono dagli esempi.
Il prefisso Tensor del framework Tensorflow, sta ad indicare che gli input dei nodi del grafo non sono, in genere, semplici variabili ma vettori (*array*) ad una o più dimensioni, chiamati, in questo caso impropriamente, tensori. Infatti i tensori hanno caratteristiche più complesse.
Ma state tranquilli, perché non è necessario descrivere un grafo computazionale con le API di basso livello. Tenete presente che un grafo computazionale non è il grafo della rete neurale intesa come struttura di nodi neurali. Quindi bisognerebbe scendere più in dettaglio. Ma fortunatamente è presente l'insieme di API di alto livello che formano la libreria **Keras**.
Keras è quindi una libreria basata su Tensorflow ed

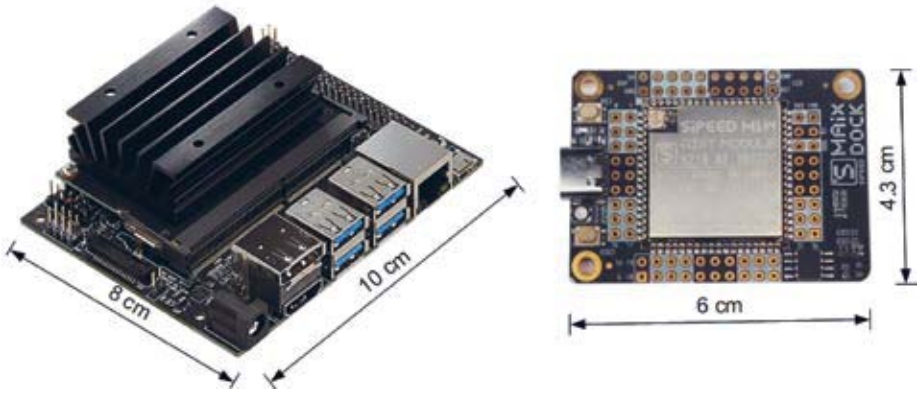


Fig. 3
Jetson nano e Sipeed Maix Dock.

è ormai è stata completamente integrata nell'istallazione di Tensorflow in ambiente Python. Benché le API di Tensorflow siano definite anche per gli ambienti di programmazione C++, Java e Javascript, il linguaggio di programmazione preferito rimane Python. Ed è proprio in questo linguaggio che sono definite le librerie di alto livello come Keras, la quale permette di descrivere la rete neurale in termini di strati (layer), e contempla una numerosa tipologia degli stessi, così da implementare diverse strutture di reti neurali. Le funzioni Keras di alto livello si occupano loro di esplodere questi layer e le loro connessioni nel grafo dataflow gestito da Tensorflow.
Per quanto riguarda Python, per chi non lo conoscesse, esistono in rete diversi tutorial a riguardo. Qui diciamo solo che è un linguaggio interpretato quindi subito eseguibile; è orientato agli oggetti ma può essere utilizzato anche con un approccio funzionale-procedurale. Inoltre è molto sintetico e può trattare in maniera molto semplice variabili-vettori a più dimensioni o liste di valori sia come parametri che come *return* di una funzione; può essere istallato su qualunque computer e si trova già presente nella distribuzioni Raspbian di Rasperry Pi.

$$f(x,y)=x^2\cdot y+y+2 \Rightarrow$$

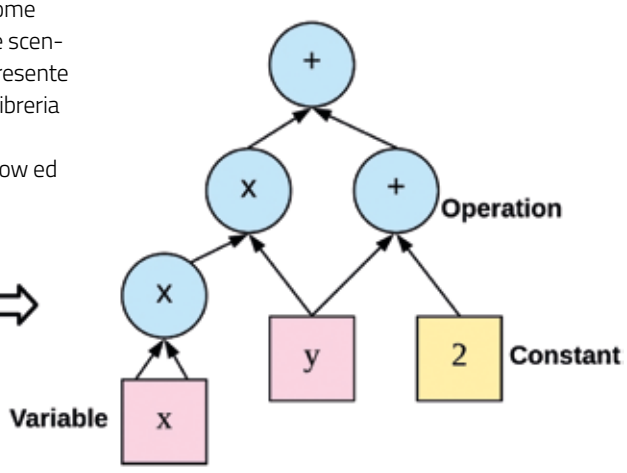


Fig. 4
Esempio di dataflow di una semplicw espressione.

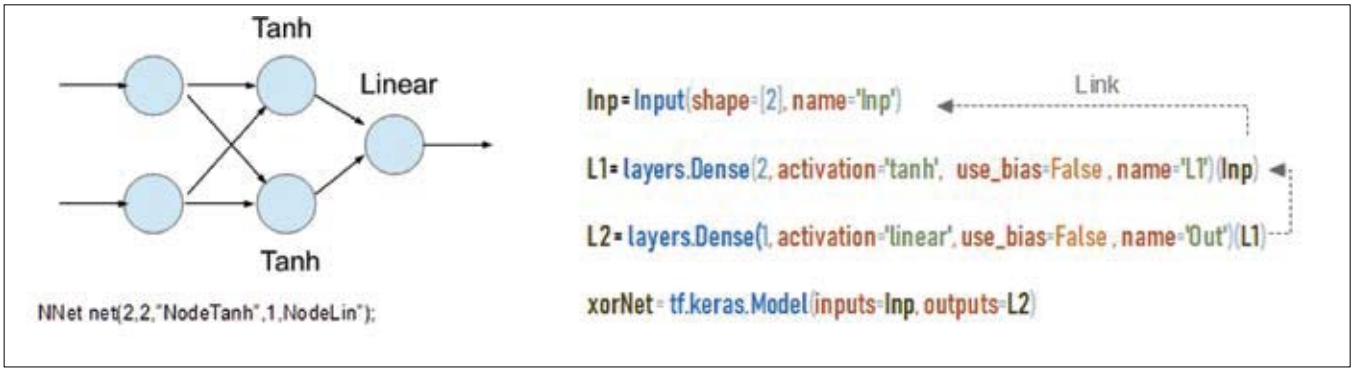


Fig. 5
Neural Network per XOR:
a) con libreria NNetLib;
b) con Keras.

D'ora in poi, salvo casi particolari come Darknet, faremo riferimento a **Python 3.7.4** come ambiente di programmazione. Una volta istallato Python *https://www.python.org/* (se non già presente), si installa Tensorflow tramite l'applicazione "pip" allegata a Python:

```
pip install tensorflow
```

L'applicazione pip la trovate nella cartella script dell'istallazione Python ed è usata per installare i vari moduli (package, librerie) sviluppati per ampliare l'utilizzo di Python. Sono moltissimi ed un elenco lo potete trovare nel Python Package Index: *https://pypi.org/.*
Chiudiamo questa lunga premessa parlando dell'utilità di un buon ambiente di sviluppo per Python. Vista la complessità dell'ambiente che intendiamo utilizzare è opportuno installare un buon IDE Python in modo da semplificare l'editing e la gestione dei progetti. Anche la capacità di completare i comandi con la lista delle possibili opzioni è molto utile data la smisurata dimensione delle funzioni implementate. Ne suggeriamo un paio (i più famosi) : PyCharm di Jetbrains *www.jetbrains.com/pycharm/ oppure Spyder www.spyder-ide.org/.*

RETE NEURALE CON KERAS

Come abbiamo detto, il "Deep Learning" è sostanzialmente basato su strutture di reti neurali feed-forward a molti strati successivi. Inoltre i singoli strati possono essere più complessi del semplice collegamento di tutti-a-tutti con lo strato seguente. Per questo motivo Keras ha predisposto una numerosa tipologia di strati già configurati e pronti per essere collegati uno con l'altro. Il modo migliore di vedere Keras all'opera è partire con una Rete Neurale di struttura semplicissima: ad esempio una rete per risolvere la funzione XOR. Una rete che abbiamo già visto realizzata con la libreria "NNetLib" per Arduino (vedere numeri 238 e 230 della nostra rivista o il sito *github.com/open-electronics/Artificial_Intelligence*). Nella **Fig. 5** è rappresentata la struttura della rete, formata da 5 nodi; nella stessa figura, a sinistra c'è la semplice formulazione con la libreria NNetLib, mentre a destra la sua definizione con Keras. È evidente che in questo caso semplicissimo la definizione con Keras sembra inutilmente complessa, ma, con decine di strati non elementari, la formulazione della rete con Keras diventa chiaramente maneggiabile ed indispensabile. Nell'esempio della figura 5 i layer "Dense" stanno ad indicare semplici strati con i nodi connessi con tutti i nodi dello strato successivo; ma esistono anche layer "Convolutivi", "Polling layer", "Recurrent layer" ecc.
La descrizione della rete, riportata in **Fig. 5**, non è l'unica che si può utilizzare per una struttura NN; esistono anche altre forme più sintetiche e specializzate ma è probabilmente la più chiara e la più generale. In **Fig. 6** potete vedere il listato completo, in Python, che implementa la rete e la addestra fino ad ottenere la funzionalità desiderata.
Come si vede, la rete definita ed addestrata viene salvata in un file : "xorNet.h5'. Questo formato di file con estensione ".h5" (o anche ".hdf5") è una struttura binaria (non testuale) in cui viene salvata la struttura della rete insieme ai suoi pesi.
In realtà è possibile anche salvare separatamente la struttura ed i pesi. In questo caso la struttura viene salvata in formato testuale di tipo "json" mentre i pesi sono ancora salvati nel formato ".h5". A questo punto è necessario fare una precisazione riguardo alla struttura NN definita in Keras. La definizione Keras presentata in **Fig. 5** non è completa, infatti le manca l'istruzione:

```
xorNet.compile(loss='mean_squared_error',
optimizer='RMSprop', metrics=['accuracy'])
```

che invece potete vedere nel listato di **Fig. 6**. Tale istruzione aggiunge alla struttura anche i metodi di addestramento e la misura dell'errore da presentare ad ogni ciclo (epoca) di addestramento.
Questi parametri sono anch'essi contenuti nella

struttura salvata sul file. In questo modo l'addestramento può riprendere in qualunque momento. Per visualizzare il contenuto dei file ".h5" si possono utilizzare dei programmi come "HDFView" disponibili su rete. Su rete sono disponibili anche programmi che visualizzano la struttura ".h5" in formato grafico. Prendiamo ora in considerazione l'istruzione per l'addestramento:

```
xorNet.fit(xtrn, ytrn, batch_size=1, epochs=50, steps_
per_epoch=100, shuffle=True)
```

La funzione fit() riceve come parametri: la sequenza degli input, la corrispondente sequenza degli output corretti, il numero di cicli di ripetizione (epoche) ed il numero di esempi da presentare per ogni epoca. Il parametro "shuffle" richiede il rimescolamento degli esempi nella sequenza di addestramento, mentre il "batch_size=1" sta ad indicare che l'addestramento viene effettuato con l'aggiustamento dei pesi ad ogni esempio. In realtà per definizione con il deep-learning è preferibile utilizzare l'aggiornamento dei pesi mediato su un certo numero di esempi ("mini-batch"). Ma per uniformità con la libreria NNetLib, che non può utilizzare questa tecnica per le note ristrettezze di memoria, si è preferito disabili-

ExeXOR.py

```
# Esempio di rete neurale per simulare funzione XOR realizzata in Python e Tensorflow-Keras

import tensorflow as tf                    #carica la libreria Tensorflow
from tensorflow.keras import layers, Input  #carica le librerie layers e Input di Keras
import numpy as np                         #carica le librerie numpy (gestione array)


#Definizione della rete neurale. Esistono molti modi, ma forse questo è il più chiaro.
#Buffer (tensore) di input
Inp = Input(shape=[2], name='Inp')
#Strato hidden con 2 nodi, con funzione di attivazione tangete iperbolica e senza bias
#Lo strato è applicato (agganciato) al tensore di input in modalità tutti-a-tutti ("dense")
L1 = layers.Dense(2, activation='tanh',   use_bias=False , name='L1')(Inp)
#Strato finale con 1 nodo, attivazione lineare e senza bias
#Lo strato è applicato al tensore L1 (strato precedente) in modalità "dense"
L2 = layers.Dense(1, activation='linear', use_bias=False , name='Out')(L1)
#Viene finalmente definita la rete assegnando il primo e l'ultimo strato
xorNet = tf.keras.Model(inputs=Inp, outputs=L2)

#La rete (modello keras) viene compilata nel grafo computazionale Tensorflow
#In questo frangente vengono definiti anche:
#  -  il tipo di valore da ottimizzare (gradiente)
#  -  l'eventuale metodo di ottimizzazione del gradiente
#  -  i valori da visualizzare nel training (accuracy: errore)
xorNet.compile(loss='mean_squared_error', optimizer='RMSprop', metrics=['accuracy'])

#training set per XOR
xtrn = np.array([[0.0, 0.0], [0.0, 1.0], [1.0, 0.0], [1.0, 1.0]])
ytrn = np.array([0, 1, 1, 0])

#training (in keras chiamato fit)
#per default in keras il training avviene per blocchi di esempi con accumulo del gradiente
#(batch_size); poichè l'applicazione di riferimento era stata fatta con le librerie per Arduin
#che agiscono esempio per esempio si è posto batch_size=1
#durante il training viene stampato automaticamente l'errore ad ogni epoca e il training set
#viene ripetuto con rimescolamento dell'ordine di presentazione degli esempi
xorNet.fit(xtrn, ytrn, batch_size=1, epochs=50, steps_per_epoch=100, shuffle=True)

#visualizzazione del funzionamento della rete addestrata per le 4 combinazioni  (predict(x))
for x in xtrn:
    y= xorNet.predict([[x]])
    print("x= %1.1f, %1.1f    y= %1.1f" % (x[0], x[1] , y[0]))

#salvataggio della rete pesi addestrati compresi (pronta per essere utilizzata)
#la rete viene salvata nella stessa directory dello script
import os
pathsave = os.path.dirname(__file__)+'/xorNet.h5'
xorNet.save(pathsave,save_format='H5')
```

Fig. 6 Listato Python per implementare ed addestrare NN per Xor.

tare questa modalità. Come si vede la funzione fit() realizza da sola tutto il processo di addestramento e questa è una ulteriore caratteristica della complessa funzionalità di Keras. Ad ogni epoca stampa la misura dell'errore. Utilizzando un'ulteriore funzionalità di Tensorflow chiamata Tensorboard è possibile avere il "plotting" dell'andamento dell'errore. Tensoboard è un eseguibile che va scaricato. L'utilizzo consiste in un processo separato che funziona come un server WEB. Lo si lancia passandogli come parametro ("--logdir ...") l'indicazione della directory dove può trovare un file di log che gli serve per visualizzare o plottare le varie informazioni del processo di training. La stampa dei dati di log va abilitata all'interno dello script Python tramite opportune istruzioni tensorflow. Quindi aprendo un browser sull'indirizzo locale: *http://localhost:6006* si possono visualizzare i grafici anche mentre procede l'addestramento. Infine, avendo addestrato la rete, vogliamo verificarla od utilizzarla. Per questa attività useremo la funzione:

```
y= xorNet.predict([[x]])
```

Tenete presente che l'array a due dimensioni per x e per xtrn è necessario perché abbiamo predisposto, per l'addestramento, una lista di array a due valori. Per salvare la rete complessivamente (struttura con pesi) è sufficiente l'istruzione:

```
xorNet.save(nomefile,save_format='H5')
```

oppure per salvare separatamente la struttura ed i pesi bisogna utilizzare le seguenti istruzioni:

```
print(xorNet.to_json(), file=open(pathmodel,'w'),
end='\n')
xorNet.save_weights(pathweights)
```

Per utilizzare una rete addestrata e salvata basta caricarla con l'istruzione:

```
xorNet = tf.keras.models.load_model(nomefile)
```

Se invece sono stati salvati separatamente struttura e pesi si possono usare le seguenti istruzioni:

```
xorNet = tf.keras.models.model_from_
json(open(pathmodel,'r').read())
xorNet.load_weights(pathweights)
```

GRAFO KERAS E GRAFO TENSORFLOW

Keras è una sovrastruttura di Tensorflow utile per definire in modo semplice Reti Neurali complesse. Le sue funzioni sono orientate alla struttura di rete e prescindono dalla loro effettiva implementazione. Infatti Keras può essere utilizzato anche come macro-linguaggio per altri framework come Teano. Noi, però, ci riferiremo a Tensorflow come unico "back-end" (strato implementativo) di Keras. Poiché la struttura definita dalle funzioni Keras deve essere "esplosa" nei grafici computazionali di Tensorflow, questi ultimi sono molto più complessi della struttura a strati della Rete Neurale espressa da Keras. Inoltre, anche la struttura Keras si porta dietro molti parametri in più di quelli strettamente necessari per computare (inferenziare) la rete. Per esempio i buffer per i gradienti ed altro. Ecco che quindi si presenta il problema di ottimizzare la struttura nel caso si voglia utilizzare la rete addestrata come inferenza (funzione input → output). In questo ultimo caso è anche opportuno passare alla versione C++ del grafo computazionale di Tensorflow. Infatti Tensorflow ha le funzioni di base definite in C++.
Per ora tralasciamo i problemi di ottimizzazione e quelli per l'utilizzo di CPU multiple e/o di GPU ad architettura parallela come NVIDIA CUDA. Per il momento possiamo concentrarci su costruzioni di reti di Deep-Learning con Python e Keras anche se possono evidenziare seri problemi di performance nel caso di compiti complessi.

TEMPI DI RISPOSTA DI RETI SOFISTICATE

Nell'utilizzo del deep-learning dobbiamo distinguere due modalità:

- addestramento;
- inferenza (produzione).

Come abbiamo evidenziato più volte, una Rete Neurale è una struttura che alla sua prima implementazione è vergine, ovvero non sa far nulla. Occorre addestrarla in modo che i suoi parametri (pesi) evolvano fino a determinare la funzionalità che si vuole ottenere. Siccome l'addestramento richiede la presentazione ripetuta di molti esempi, che possono essere anche centinaia di migliaia di immagini, nel caso di reti per il riconoscimento di forme, i suoi tempi possono durare anche diverse ore utilizzando potenti architetture hardware parallele. Se all'addestramento è consentito essere molto lungo, perché non necessita di stringenti tempi di risposta; al contrario, l'utilizzo della rete addestrata come inferenza, ovvero come risposta ad un input, può richiedere una risposta immediata. Come nel caso di identificazioni di oggetti da streaming di telecamera. In questo caso l'utilizzo di hardware potente diventa necessario.

Tensorflow, ma anche gli altri framework, può essere indirizzato ad usare hardware multi-processore o GPU con elaborazione parallela come l'architettura CUDA. Una delle applicazioni più conosciute in ambito deep-learning è sicuramente quella chiamata "*Objects Detection*". Per chiarire, è quella dove si vede nello streaming l'apparizione di rettangoli che inquadrano gli oggetti della scena che sono stati riconosciuti. Esistono, in Rete, strutture già addestrate a svolgere questo compito.
La loro risposta inferenziale dipende sia dal tipo di struttura implementata che dall'hardware su cui sono mandate in esecuzione. Tralasciamo, per il momento, la descrizione e il confronto delle diverse strutture. In questo articolo cercheremo di vedere quale è l'impatto sull'hardware di questo tipo di applicazione, prendendo in considerazione la struttura meno pesante, in funzione di un possibile utilizzo su Raspberry Pi.

DARKNET E YOLO

La struttura NN deep-learning più veloce è attualmente quella detta YOLO (You Only Look Once). Si differenzia dai precedenti approcci perché cerca di gestire l'immagine in un'unica passata, ovvero evitando di rilevare prima aree di interesse e successivamente di identificarle. Nonostante ciò è comunque una rete deep-learning di oltre un centinaio di strati per un totale di circa 250 MB di parametri (pesi). Di strutture YOLO esistono 3 versioni, l'ultima delle quali è la 3.
YOLO è stata implementata originalmente su un framework in C chiamato Darknet (*pjreddie.com/darknet*). L'implementazione in C contribuisce sicuramente alla velocità di esecuzione. Purtroppo Darknet non è documentata opportunamente anche se esistono i sorgenti e il "*Makefile*" o il "*Cmakelists*" con le varie opzioni da editare: OPENCV si/no,CUDA si/no ecc.
A causa della "leggerezza" computazionale del framework Darknet e della struttura di rete YOLO, prima di passare ad utilizzare framework più flessibili e documentati come Keras, diamo un'occhiata a questa soluzione in puro C e C++. Come detto, il framework Darknet non è documentato ma può caricare altre strutture di rete, non solo YOLO. Per chi volesse cimentarsi con questo framework l'unica soluzione è quella di esaminare i file sorgenti. Riassumendo, la struttura della rete è descritta in un file (testuale) detto file di configurazione ("*.cfg"), mentre i pesi si trovano in un file separato ma in formato digitale ("*.*weights*").
I file pesi sono prodotti in questo formato dal framework alla fine della fase di addestramento. Darknet è in pratica un programma C che può ricevere diversi parametri di lancio che gli permettono di operare come addestratore o come produttore caricando strutture di rete e pesi. Perciò il file sorgente più indicativo da esaminare, può essere il file "darknet,c" che contiene il "*main(argv,argc[])*".
In ogni caso il framework Darknet è conosciuto sostanzialmente per l'utilizzo con la struttura YOLO sviluppata dagli stessi autori (Redmon, Joseph and Farhadi, Ali) del framework. Mettendo all'operaYOLOv3 su un computer Windows (con processore I7), senza scheda GPU CUDA, si sono rilevati tempi inaccettabili pari a : 2.7 sec/frame ovvero 0.37 frame/s. È chiaro che non sarebbe praticabile su Raspberry Pi; però è stata presentata una versione Lite di YOLO chiamata TinyYOLO. In questa versione sono stati ridotti gli strati convolutivi e i parametri si sono ridotti a circa 35 MB (dagli originali 250 MB), a scapito di una maggiore imprecisione. In questa situazione sono stati rilevate le seguenti performance nella gestione degli "stream" da webcam:

- PC Windows = circa 3 frame/s.
- Raspberry Pi 3 = minore di 0,2 frame/s.

Come si vede anche con TinyYOLOv3, Raspberry pi 3 non ha un funzionamento "propriamente fluido". Si può ricorrere, allora, a un acceleratore "software" come NNPACK (github.com/Maratyszcza/NNPACK). Questa applicazione legge ed ottimizza la struttura della rete in termini di calcolo. Quindi in pratica modifica il cuore del framework Darknet. Con questa

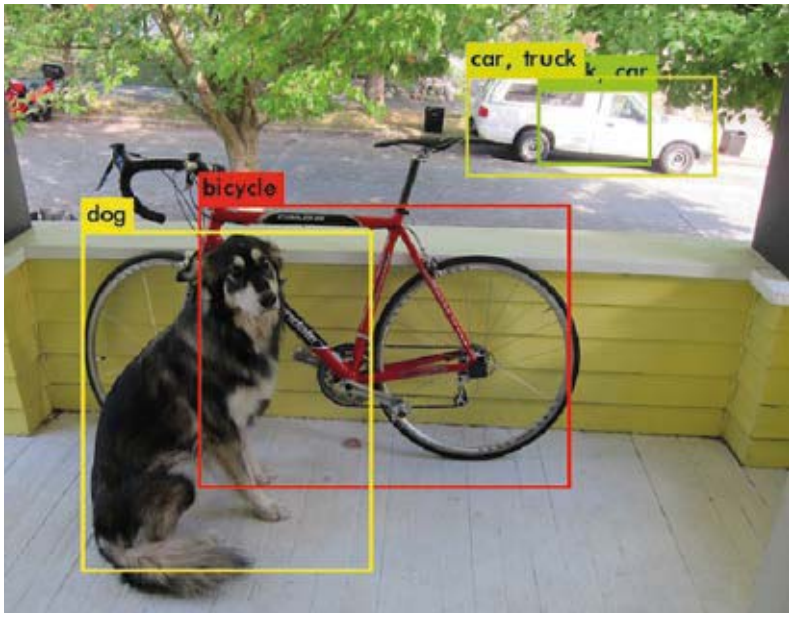


Fig. 7 Object detection.

ulteriore modifica (darknet-nnpack) si è riusciti a salire a circa 1 frame/s. In realtà di versioni TinyYOLO ne sono state sviluppate 3: l'ultima sembra essere più precisa, ma forse la versione 2 può essere appena un po' più veloce. Date le performance (appena sufficienti) dell'ultima soluzione, è comunque possibile realizzare un controllo di presenza umana con Raspberry Pi 3 . Per esempio un allarme esterno, dove Raspberry Pi con Picamera può distinguere movimenti di animali o piante mosse dal vento dalla presenza effettiva di una sagoma umana. Chiaramente in questo utilizzo lo scorrimento a scatti dello "stream" non ha molta importanza così come il tempo di latenza di 1 secondo. Nei prossimi articoli analizzeremo l'utilizzo di acceleratori hardware e sicuramente le cose cambieranno. Intanto per chi volesse cominciare a fare pratica con queste tematiche utilizzando Raspberry Pi 3 (o 4) può fare riferimento al nostro repository *github.com/open-electronics/Artificial_Intelligence/tree/master/Deep_Learning*. In questo sito troverete i programmi per Windows e per Raspberry Pi, già pronti ma che hanno bisogno di OpenCV precedentemente istallato. OpenCV è necessario esclusivamente per il trattamento dello stream da webcam, per cui, nel caso di utilizzo solo con immagini fisse, può non essere presente. Nelle directory operative sono presenti i file comando per lanciare Yolov2 e Yolov3 e soprattutto i file di pesi che la darknet caricherà insieme alla configurazione scelta. L'alternativa è quella di compilarsi da soli l'ambiente. Esistono diversi siti che fanno riferimento al sistema completo "Darknet-NNPACK" ma purtroppo ci sono diversi problemi ed errori; in ogni caso elenchiamo alcuni riferimenti:

- *https://github.com/buptdbj/darknet-windows-linux* (Darknet);
- *https://www.robotwars.nz/robots/display_doc_Final.cgi?sesid=dntbot_545539RVfTi7tvQRA&docno=50000550&crc=50096596* (Darknet-NNPACK).

YOLO è una struttura deep-learning molto utilizzata per l'object detection a causa della sua efficienza ed efficacia. Ma per avere un funzionamento fluido è opportuno utilizzare hardware più potente di Raspberry Pi 3, oppure un acceleratore hardware come la chiavetta USB Intel Movidius che contiene una struttura elaborativa parallela su cui scaricare la struttura. Infine si otterrebbero sicuramente risultati molto interessanti utilizzando hardware predisposto al machine-learning come Nvidia Jetson Nano o Sepeed di cui abbiamo parlato all'inizio. Questo hardware può essere usato da solo o in collegamento con Raspberry Pi. Ne riparleremo nei prossimi articoli. Anche se Raspberry Pi non sembra adatto a compiti così gravosi come l'Objet Detection in real time, nulla vieta di sperimentare altre applicazioni AI dove la velocità di risposta non è importante. Per esempio il riconoscimento di pattern relativamente lenti o fermi, o la produzione di immagini inventate (per estrapolazione), o il riconoscimento del significato delle frasi testuali od altro. Ma per questi compiti probabilmente Keras è più chiaro per l'implementazione.

Fig. 8
La chiavetta USB Intel Movidius.

Cosa occorre?

Il materiale utilizzato in questo progetto è disponibile presso Futura Elettronica. Lo starter kit per Jetson Nano (cod. JNANO) è in vendita al prezzo di Euro 229,00, Raspberry Pi 4 con 4GB di memoria (cod. RPI4-4GB) è disponbile a Euro 69,00, mentre Raspberry Pi 4 con 2GB di memoria (cod. RPI4-2GB) costa 59,00 Euro. 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

L'ADDESTRAMENTO

Nel riconoscimento di oggetti abbiamo utilizzato, finora, strutture già addestrate, caricando pesi messi a disposizione su Internet. Infatti, come accennato, il training di una rete complessa come YOLO richiede quantità di esempi da sottoporre e tempi di addestramento notevoli. Ma con quali esempi si addestra una rete per il riconoscimento di oggetti? Per fortuna esistono in rete dei cataloghi di esempi di addestramento che volenterosi ricercatori di AI hanno messo a disposizione. In questo modo, oltre a semplificare la vita agli altri ricercatori, si ottiene un metodo di misura comune sull'efficacia delle strutture proposte. Un esempio di questi cataloghi lo abbiamo già visto quando abbiamo introdotto il riconoscimento delle cifre scritte a mano, già nella presentazione della semplice libreria NNetLib da noi proposta per Raspberry Pi e addirittura per Arduino, ovvero il catalogo MNIST .
Nel caso del riconoscimento di oggetti sono presenti

in rete i cataloghi di seguito descritti:

- PASCAL VOC (brevemente VOC) *host.robots.ox.ac.uk/pascal/VOC;* ormai non più aggiornato, in esso si possono identificare 20 classi di oggetti (tra cui "person"), più di 5.000 immagini contenenti anche più oggetti, corredate da indicazioni sugli oggetti da rilevare.
- COCO (Common Object in Context) *cocodataset.org*; ben 80 classi di oggetti da identificare (tra cui "person"). Più di 300.000 immagini contenenti anche più oggetti corredate da indicazioni sugli oggetti da rilevare.

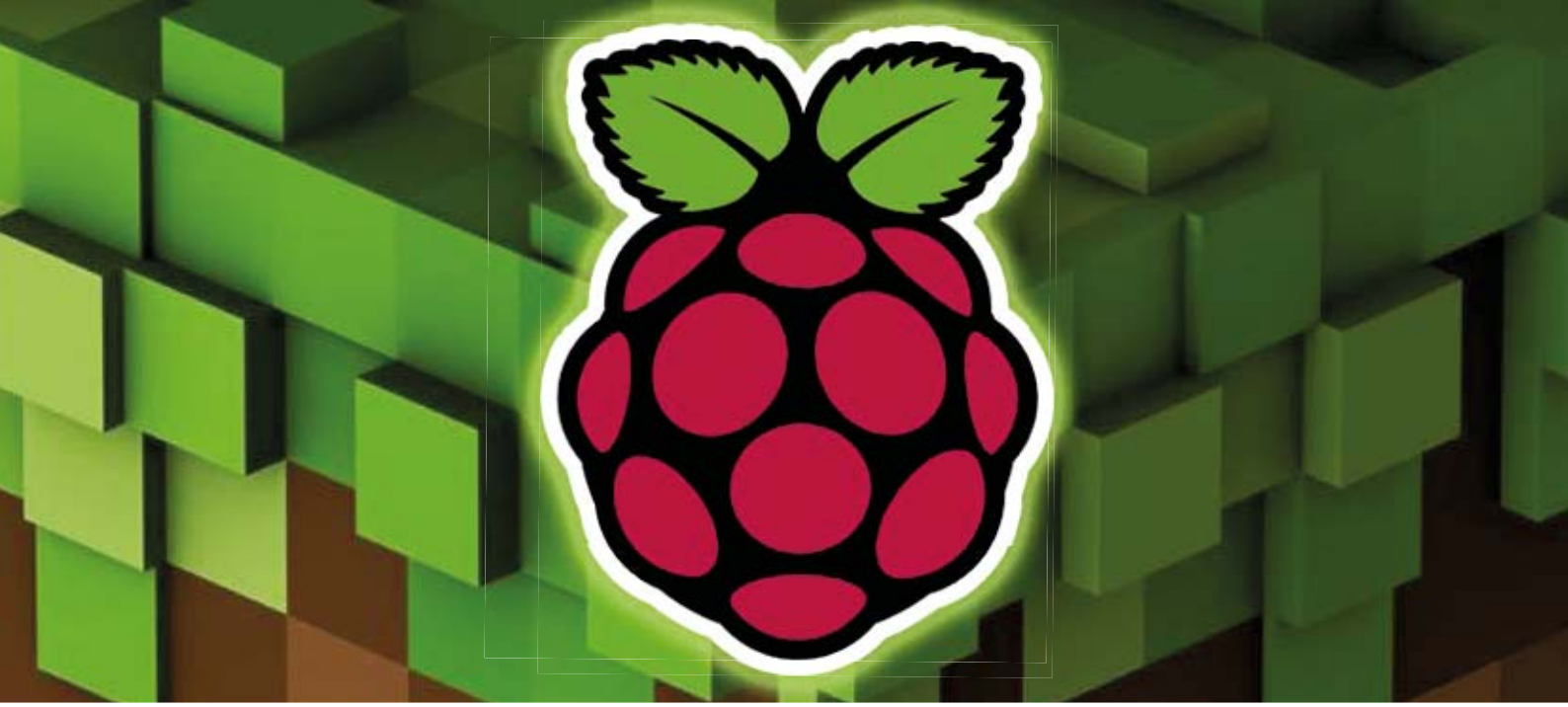
Come vedete, un carico di lavoro consistente per l'addestramento, che necessita quindi di hardware particolarmente prestante per poter limitare in modo ragionevole il tempo necessario (ore, giorni?). Nel nostro sito "github" sono presenti due comandi che lanciano darknet, caricando rispettivamente YOLOv2 addestrata con VOC (che riconosce quindi 20 categorie di oggetti) e YOLOv3 addestrata con COCO (che riconosce quindi 80 categorie di oggetti):

- *TestCam-tiny2-Rasp.sh (in DarkNetRasp.zip\darknet-nnpack\) e TestCam-tiny2-Win.cmd (in DarkNetWinNoGpu.zip\DarkNetWinNoGpu\);*
- *TestCam-tiny3-Rasp.sh (in DarkNetRasp.zip\darknet-nnpack\) e TestCam-tiny3-Win.cmd (in DarkNetWinNoGpu.zip\DarkNetWinNoGpu\).*

Ambedue i programmi fanno uso di OpenCV ed utilizzano una comune webcam su Window e una picam su Raspberry Pi 3. È probabile che su Raspberry Pi 3 possa essere usata anche una webcam senza problemi. Tenete presente che su Windows dovrebbe bastare scaricare la directory e i file dei pesi (che valgono sia per Window che per Raspberry Pi). Quindi è sufficiente lanciare uno dei comandi sopra riportati. Mentre per Raspberry Pi dopo aver scaricato le due directory comprese all'interno del file compresso, bisogna prima copiare le librerie come riportato nel file Readme.txt.
I file dei pesi "yolov2-tiny-voc.weights" e "yolov3-tiny.weights" vanno copiati nella directory dove si trovano i comandi di lancio.

CONCLUSIONI

Per il momento ci fermiamo qui. Prossimamente approfondiremo la materia analizzando sia del software che dell'hardware specializzato, sempre nell'ottica di fare in modo che le tecniche di Intelligenza Artificiale entrino a far parte concretamente nell'operato di hobbisti, sperimentatori o utilizzatori evoluti. A presto e buon lavoro!



RASPBERRY Pi 4

Utilizzala subito con i nuovi set!

CON RASPBERRY Pi 4 DA 4 GB

€ 79,00

Cod. RPI4BULK4GB

Il set comprende:
- Board Raspberry Pi 4 da 4 GB di memoria
- Alimentatore 5V-3A con connettore micro USB
- Adattatore Micro USB a Type C

€ 89,90

Cod. RASPKITV10

Il set comprende:
- Board Raspberry Pi 4 con 2 GB di memoria
- Box trasparente per Raspberry Pi 4 con ventola
- Alimentatore 5V-3A
- Micro sdcard hc da 32GBb + adattatatore
- Adattatore da micro hdmi a hdmi
- Minuteria varia

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**

Approfondiamo la sintassi del linguaggio KV che ci consente di gestire meglio le interfacce grafiche dotate di più pagine. Terza puntata.

Il linguaggio KV, detto anche **kvlang** o **linguaggio kivy**, consente di creare strutture di widget ad albero e di associare tra loro le proprietà dei widget o di richiamarle in maniera più semplice all'interno dello script in Python. Come vedremo nel prosieguo di questo corso, la sintassi semplificata, offerta dal linguaggio KV, permetterà di editare e modificare anche l'interfaccia utente in maniera più agile e veloce. Inoltre, il flusso del programma farà capire meglio la logica dell'applicazione sviluppata o in fase di sviluppo. Vediamo dunque come funziona, con la premessa che nella trattazione utilizzeremo spesso la sigla "KV" come abbreviazione di "linguaggio KV".

CARICARE IL CODICE KV

Partiamo con i metodi per caricare il codice KV in uno script Python, che sono due:

1. **tramite un nome convenzionale del file**: Kivy cerca automaticamente nella stessa directory un file KV con lo stesso nome della classe *App* scritta in minuscolo, senza "App" se termina con "App".
   Per esempio:
   - nome dell'applicazione = **MyApp**, nome del file nella stessa cartella **my.kv**;
   - nome dell'applicazione = **TestApp**, nome del file nella stessa cartella **test.kv**;

2. **tramite il modulo Builder**: Kivy può caricare direttamente in uno script Python una stringa KV interna o un file KV esterno. La stringa KV deve trovarsi all'interno dello script fra tre apici all'inizio e alla fine, come fosse un commento di Python, o in un file esterno con estensione *.kv*.
   Una volta caricata la stringa KV, il nome della classe iniziale diventa un widget *root* (radice).
   Questo widget verrà restituito automaticamente da uno dei seguenti metodi:
   - **Builder.load_file('percorso/file.kv')**, per esempio: **Builder.load_file(test.kv')**.
   - **Builder.load_string(kv_string)** che verrà esemplificato più avanti.

REGOLE SINTATTICHE

Una stringa o un file KV deve essere scritto seguendo delle regole sintattiche, che verranno utilizzate per descrivere il contesto di un widget. Si può avere un widget *root* e un qualsiasi numero di widget al suo interno.
Il widget *root* viene creato dichiarando la classe di un widget senza alcuna indentazione seguita dai due punti : all'inizio della radice (*root*).
Per esempio:

BoxLayout:
oppure
GridLayout:
oppure
FloatLayout:
... e così via.

Una classe widget viene dichiarata con un nome arbitrario tra i segni di maggiore/minore ovvero < > seguita dai due punti : all'inizio della radice (*root*).
Per esempio:

<MyLayout>:

All'interno della classe **<MyLayout>** si possono aggiungere i widget in numero indefinito. Per esempio:

```
<MyLayout>:
        BoxLayout:
        GridLayout:
```

Analogamente, si potrebbero fare altri esempi.
ATTENZIONE: il codice KV usa l'indentazione tipica di Python, per cui bisogna stare attenti a usare le tabulazioni (di solito 4 livelli) per formattare il testo KV in modo che al momento della compilazione non emergano degli errori.

PAROLE CHIAVE

Nel linguaggio KV esistono tre parole chiave specifiche:
- **app**: si riferisce sempre all'istanza dell'applicazione;
- **root**: si riferisce al widget *root* corrente;
- **self**: si riferisce sempre al widget corrente.

Spiegheremo come usare le parole chiave negli esempi che seguono.

DIFFERENZE DI LINGUAGGIO

Per affrontare correttamente le differenze che intercorrono fra il linguaggio KV e Python, diamo un'occhiata al codice seguente. Per dichiarare all'interno del widget root uno o più widget figlio, ovvero istanze di una classe *root*, basta dichiarare il/i widget figlio all'interno del widget root. Per esempio:

```
<MyRootWidget>:
        BoxLayout:
                Button:
                        text: 'LED 1'
```

TABULAZIONI

L'esempio fatto sopra definisce in modo elegante come scrivere codice nel linguaggio KV rispettando i quattro livelli di tabulazione previsti:
1. livello 1: widget *root* **MyRootWidget** (non richiede alcuna tabulazione);
2. livello 2: layout con widget **BoxLayout** (una tabulazione);
3. livello 3: BoxLayout con un widget figlio **Button** (due tabulazioni);
4. livello 4: proprietà del widget **Button** (tre tabulazioni).

Continuando ad aggiungere widget si devono rispettare le tabulazioni descritte sopra.
Per esempio:

```
<MyRootWidget>:
        BoxLayout:
                orientation: 'vertical'
                Button:
                        text: 'LED 1'
                Button:
                        text: 'LED 2'
                Button:
                        text: 'LED 3'
                GridLayout:
                        cols: 2
                        raws: 2
                        padding: [15,15,15,15]
                        Button:
                                text: 'LED 4'
                        Button:
                                text: 'LED 5'
                        Button:
                                text: 'LED 6'
                        Button:
                                text: 'LED 7'
```

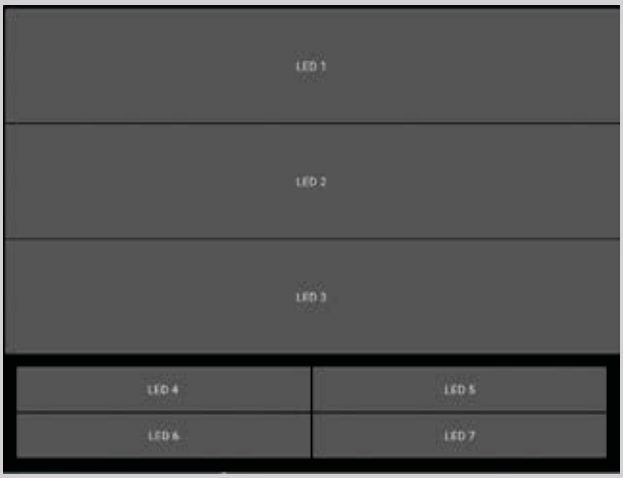


Fig. 1 - Esempio ottenuto dalla stringa in KV.

Si può vedere il risultato nella **Fig. 1**: un BoxLayout con orientamento verticale e tre pulsanti orizzontali con il testo LED 1, LED2 e LED3.
Sotto al BoxLayout viene creato un GridLayout con 2 colonne e altrettante righe con padding di 15 pixel sui quattro lati, contenente altri quattro pulsanti con il testo LED 4, LED 5 LED 6 eLED 7.
Si fa notare che nel linguaggio KV non è necessario importare all'inizio dello script i moduli dei widget da usare, come per esempio, **BoxLayout** e **Button**; è sufficiente utilizzarli all'interno della stringa KV per essere disponibili.
L'equivalente in Python del precedente listato potrebbe essere quello nel **Listato1**, probabilmente meno piacevole, sia da leggere che da scrivere.
Facciamo notare che è necessario aggiungere il widget **GridLayout** come widget al **BoxLayout** e quindi aggiungere il tutto nell'oggetto *root*.

VALORI DELLE PROPRIETÀ

Per scrivere i valori delle proprietà dei widget, abbiamo visto dagli esempi, che il linguaggio KV è molto più semplificato.
In Python, per impostare il numero di colonne di un **GridLayout**, dobbiamo scrivere:

```
grid = GridLayout(cols=3) #valore della proprietà cols
```

Per fare la stessa cosa in KV, basta impostare le proprietà del widget direttamente al suo interno senza il segno di uguale =:

```
GridLayout:
        cols: 3 #valore della proprietà cols
```

Il valore del parametro KV viene valutato comunque come un'espressione di Python e verranno osservate tutte le proprietà utilizzate nell'espressione. Per esempio, in Python bisogna sempre dichiarare il widget con tutte le proprietà dentro le parentesi e separate da virgole:

```
btn = Button(text='LED 1', font_size='50sp', size_
hint=(.25,.5), backgound_color=[0,.5,.2,.8])
```

Invece, con KV, diventa enormemente più semplice perché si scrivono le proprietà del widget una sotto l'altra, rendendo tutto molto più leggibile:

```
Button:
        text: 'LED 1'
        font_size: 50
        size_hint: [.25,.5]
        backgound_color=[0,.5,.2,.8])
```

Questo a dimostrazione del fatto che è sufficiente istanziare un widget per ottenerne i metodi e impostarne le proprietà. Una volta compreso questo, possiamo proseguire con altri esempi.
Ulteriori informazioni sulla sintassi del linguaggio KV sono disponibili al seguente link: *https://kivy.org/doc/stable/guide/lang.html.*

EVENTI

Gli eventi sono una parte importante della programmazione di Kivy. Questo potrebbe non essere sorprendente per i programmatori con esperienza, ma è un concetto importante per il neofita. Una volta capito come funzionano gli eventi e come collegarli, si potranno usare ovunque in Kivy, soprattutto per rendere possibile l'interazione con la GUI e l'hardware di Raspberry Pi.
Per non mettere troppa carne al fuoco, prenderemo in esame solo gli eventi che sono utili agli esempi esposti di volta in volta.

EVENTI CALLBACK

Per "triggerare" eventi di callback, ovvero attivare una certa funzione al tocco di un pulsante, il linguaggio KV si differenzia molto e, fortunatamente, si semplifica.
Abbiamo già visto che in Python, per collegare una funzione *callback* alla pressione di un pulsante, si deve creare la funzione callback e utilizzare il metodo *bind* per il collegamento e il passaggio di valori all'istanza.
Per esempio:

```
btn = Button(text='LED 1')
btn.bind(on_press=callback) #evento on_press per richiamare la funzione callback alla pressione del pulsante

def callback(instance, value): #funzione callback
  #stampa il valore dello stato del pulsante
  print('My button state is <%s>' % (value.state))
```

Alla pressione del pulsante si vedrà apparire sul terminale il

Listato 1

```
from kivy.uix.boxlayout import BoxLayout #necessario importare il modulo BoxLayout
from kivy.uix.button import Button # necessario importare il modulo Button
…
…
class TestApp(App):
def build(self):
root = MyRootWidget()
box = BoxLayout(orientation='vertical')
box.add_widget(Button(text='LED 1'))
box.add_widget(Button(text='LED 2'))
box.add_widget(Button(text='LED 3'))

box2 = GridLayout (cols=2, raws=2, padding = [15,15,15,15])
box2.add_widget(Button(text='LED 4'))
box2.add_widget(Button(text='LED 5'))
box2.add_widget(Button(text='LED 6'))
box2.add_widget(Button(text='LED 7'))
box.add_widget(box2)
root.add_widget(box)

return root
```

seguente messaggio:

My button state is <down>

Nel linguaggio KV si possono utilizzare gli eventi in modo diretto e, come vedremo, richiamare anche funzioni *root* in modo diretto; un esempio è:

```
Button:
  text: "Press me"
  on_press: print("Button pressed")
  on_release: print("Button not pressed")
  on_state:  print("My button state is <{}>".format(self.state))
```

Alla pressione o al rilascio del pulsante si vedranno apparire sul terminale i seguenti messaggi:

My button state is <down>
Button pressed
My button state is <normal>
Button not pressed

API REFERENCE

Come abbiamo fatto nella scorsa puntata, prima di iniziare l'analisi di un esempio pratico, ci sembra doveroso, nonché utile fornire i riferimenti API di alcuni widget che verranno utilizzati.

API di Button

La classe **Button** è, in pratica, una label con azioni associate alla pressione o al rilascio.
Per configurare un pulsante, vengono utilizzate le stesse proprietà per la classe Label (per esempio, *font_size*, *size_hint* e *pos_hint*).
Oltre a queste proprietà comuni a tanti widget, la classe Button dispone di altri metodi per gestire l'aspetto grafico. Potrebbe essere interessante, per esempio, cambiare il colore dei pulsanti quando sono a riposo o quando vengono premuti, così come creare pulsanti con immagini personalizzate:

- **background_color**: colore dello sfondo, nel formato [r, g, b, a], cioè red, green, blue e alpha. I valori fungono da moltiplicatore per il colore predefinito, che è grigio.
  Quindi l'impostazione del colore dello sfondo darà un risultato combinato con il grigio, più o meno scuro, a seconda del trasparenza del canale alfa.
  Il valore predefinito è: *background_color: [1, 1, 1, 1]*; per esempio, un rosso pieno si ottiene con [1,0,0,1], un verde scuro con [0,1,0,.5];
- **background_disabled_down**: immagine dello sfondo del pulsante quando è disabilitato e viene premuto.
  L'impostazione predefinita è: *background_disabled_down: "atlas://data/images/defaulttheme/button_disabled_pressed"* (vedere l'apposito riquadro);
- **background_disabled_normal**: immagine dello sfondo del pulsante qando è disabilitato e non viene premuto.
  L'impostazione predefinita è: *background_disabled_normal: "atlas://data/images/defaulttheme/button_disabled"* (vedere l'apposito riquadro);
- **background_down**: immagine dello sfondo del pulsante quando viene premuto.
  L'impostazione predefinita è: *background_down: "atlas://data/images/defaulttheme/button_pressed"*(vedere riquadro);
- **background_normal**: immagine dello sfondo del pulsante quando è nello stato normale e non viene premuto.
  Il suo valore predefinito è *background_normal: "atlas://data/images/defaulttheme/button"*;
- **Border**: ingrossa il bordo utilizzato del pulsante quando si usa con **background_normal** e **background_down**.
  Il valore predefinito è *border: (16, 16, 16, 16)*.

ATTENZIONE: come spiegato nella prima puntata di questo corso, bisogna ricordarsi di aggiungere all'inizio di ogni script

MODULO ATLAS

Il modulo Atlas di Kivy gestisce i cosiddetti atlas, ovvero gli "atlanti" di texture grafiche: sono delle raccolte di immagini raggruppate in un unico file. Con Atlas, si riduce il numero di immagini in memoria e si accelera il caricamento dell'applicazione. Il modulo contiene sia la classe **Atlas**, sia il comando da terminale per la creazione di un atlas personalizzato da una serie di singoli file PNG; quindi si può creare un proprio atlas e importare singolarmente le immagini in esso contenute. Un atlas è composto da:

- un file json (.atlas) che contiene i nomi e le posizioni delle texture dell'atlante;
- un file PNG che raggruppa una serie di texture a cui fa riferimento il file json (.atlas).

Per esempio, l'atlas predefinito è il file json **defaulttheme.atlas** e il tema predefinito cui fa riferimento è l'immagine **defaulttheme-0.png**, come vedete nell'immagine seguente.

Il file json contiene i riferimenti all'immagine PNG nel formato: *nome della texture, coordinate, dimensioni*. Ecco le prime righe dell'atlas predefinito (**defaulttheme.atlas**):

```
{"defaulttheme-0.png": {"progressbar_background": [392, 227,
24, 24], "tab_btn_disabled": [332, 137, 32, 32], "tab_btn_
pressed": [400, 137, 32, 32], "image-missing": [152, 171, 48,
48], "splitter_h": [174, 123, 32, 7], "splitter_down": [11, 10, 7, 32]
```

Per creare un atlas si può sfruttare il modulo Atlas per creare file di atlanti con un comando di questo tipo:

```
python -m kivy.atlas <nome> <dimensione> <elenco di immagini...>
```

Supponiamo di avere in una cartella chiamata "images" una serie di immagini personalizzate da mettere in un atlante.

All'interno della cartella mettiamo alcuni file PNG che, per comodità, saranno tutti 64x64 pixel. Entriamo nella cartella e leggiamo il contenuto con i tipici comandi da terminale:

```
cd images
ls
button1.png
button1-down.png
button2.png
button2-down.png
…
```

A questo punto, con il comando seguente si possono combinare tutti i singoli file PNG in un unico file PNG e generare il file json:

```
python -m kivy.atlas myatlas 256 *.png
```

… l'output sarà:

```
Atlas created at myatlas.atlas
1 image has been created
```

… con il commando *ls* elenchiamo tutti i file:

```
ls
button1.png
button1-down.png
button2.png
button2-down.png
myatlas.atlas
myatlas-0.png
```

Come si può vedere, sono presenti due nuovi file: **myatlas.atlas** e **myatlas-0.png**. Questi due file andranno spostati nella cartella **/home/pi/kivy/kivy/data/images**.
Per usare il proprio atlas, basterà scrivere lo stato del pulsante con il nuovo percorso dell'atlas. Per esempio:
Primo pulsante:

```
btn1 = Button(background_normal='atlas://images/myatlas/button1',
background_down='atlas://images/myatlas/button1_down')
```

Secondo pulsante:

```
btn2 = Button(background_normal='atlas://images/myatlas/button2',
background_down='atlas://images/myatlas/button2_down')
```

Nella figura sottostante sono illustrati i pulsanti personalizzati caricati dal nostro atlas.

Fig. 2 - Esempio di quattro pulsanti orizzontali.

Python le seguenti righe (togliere il commento alla riga che interessa):

```
import os
#os.environ["KIVY_WINDOW"] = 'sdl2' #uso con Raspberry Pi 4B
#os.environ['KIVY_GL_BACKEND'] = 'gl' #uso con Raspberry Pi 3B+
```

CREARE QUATTRO PULSANTI IN KV

Riprendendo l'esempio applicativo della puntata precedente, possiamo addentrarci ad esplorare il codice in linguaggio KV che ci serve per poter disporre di quattro pulsanti virtuali collocati orizzontalmente sullo schermo (**Fig. 2**) touch e farli interagire con altrettanti LED connessi fisicamente a Raspberry Pi tramite i suoi GPIO collegati tramite una breadboard come mostrato in **Fig. 3**.
Il codice di esempio è *four_buttons_GPIO.py* e lo potete scaricare dal nostro sito Internet *www.elettronicain.it* insieme agli altri file del corso.
In questo esempio impareremo ad implementare il codice KV nel codice Python, per il semplice motivo che dobbiamo gestire la porta GPIO, che sarebbe altrimenti irraggiungibile con le istruzioni e i metodi delle classi di Kivy.
La parte commentata in verde è la stringa KV che costruisce una classe **<LedScreen>**, che a sua volta contiene il widget base **BoxLayout**, che a sua volta contiene i quattro widget figli **ToggleButton**. Vi facciamo notare che all'inizio dello script, oltre ai moduli *App* e *Builder*, abbiamo importato anche i moduli *ScreenManager* e *Screen*.
Leggete i commenti nel **Listato 2** per avere la descrizione esaustiva di tutte le operazioni svolte dal codice.
Come si può vedere, gli eventi *on_press* in KV non hanno bisogno di collegarsi a una funzione di *callback*, ma attivano direttamente le funzioni della classe *root* che nel nostro caso si chiamano:

led_switch1
led_switch2
led_switch3
led_switch4

Una volta creato un oggetto **ScreenManager** è sufficiente aggiungere la classe **LedScreen** per avere l'integrazione con la classe root **LedScreen** che è stata dichiarata all'interno della stringa KV.
A questo punto, si crea la classe derivata da App (nel nostro caso **LedApp**) che ritorna l'oggetto **sm** (ScreenManager).
Con l'istruzione **LedApp().run()** si manda in esecuzione il tutto. Quando si toccheranno i quattro ToggleButton dello schermo tattile, si vedranno i quattro LED corrispondenti accendersi o spegnersi.

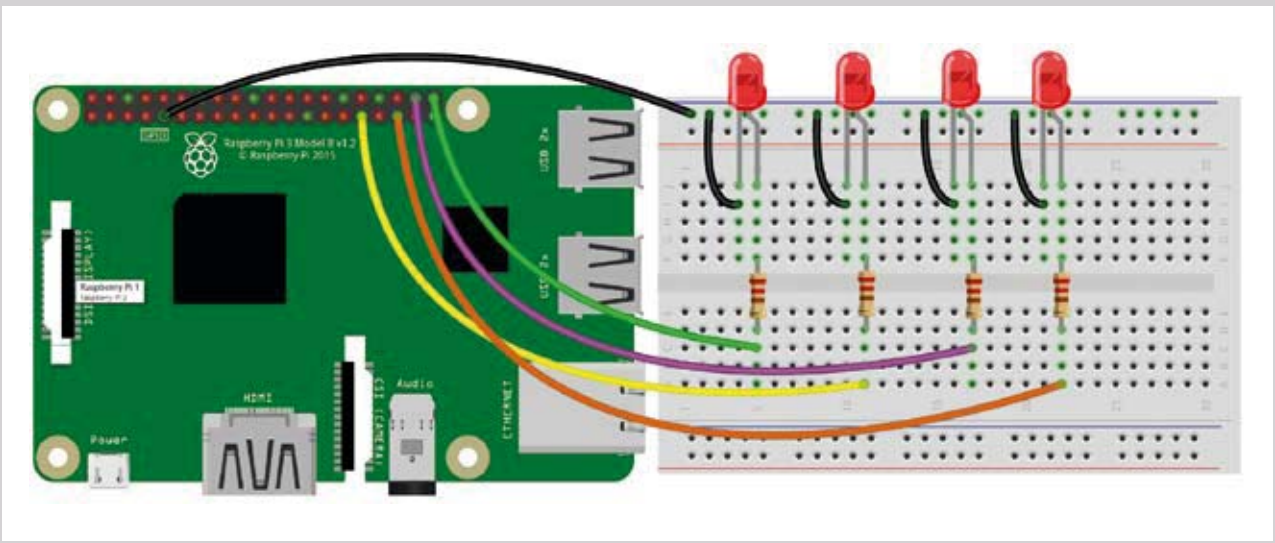


Fig. 3 - Collegamento di quattro LED a Raspberry Pi.

Listato 2

```
from kivy.app import App #modulo App
from kivy.lang import Builder #modulo Builder
from kivy.uix.screenmanager import ScreenManager, Screen #modulo ScreenManager e Screen
import RPi.GPIO as GPIO #modulo RPi.GPIO di sistema

GPIO.setmode(GPIO.BCM) #modalità BCM della piedinatura GPIO
ledPin=[21,6,20,19] #array di LED

for i in range (0,4): #ciclo for per impostare la modalità OUT e spegnere tutti I LED
    GPIO.setup(ledPin[i], GPIO.OUT)
    GPIO.output(ledPin[i],0)

# stringa KV commentata fra apici
Builder.load_string("""
<LedScreen>: #classe root
    BoxLayout: #crea un layout con il widget BoxLayout
        orientation: "vertical" #orientamento verticale del layout
        ToggleButton:   #aggiunge un widget ToggleButton al layout
            text: 'LED 1' #testo del widget ToggleButton
            on_press: root.led_switch1() #evento per triggerare la funzione della classe led_switch1
        ToggleButton: #aggiunge un widget ToggleButton al layout
            text: 'LED 2' #testo del widget ToggleButton
            on_press: root.led_switch2()  #evento per triggerare la funzione della classe led_switch2
        ToggleButton: #aggiunge un widget ToggleButton al layout
            text: 'LED 3' #testo del widget ToggleButton
            on_press: root.led_switch3()  #evento per triggerare la funzione della classe led_switch3
        ToggleButton: #aggiunge un widget ToggleButton al layout
            text: 'LED 4' #testo del widget ToggleButton
            on_press: root.led_switch4()  #evento per triggerare la funzione della classe led_switch4
""")

class LedScreen(Screen): #classe LedScreen derivata da Screen
    def led_switch1(self): #funzione per la gestione del primo LED
        if GPIO.input(ledPin[0])== GPIO.LOW: #se il LED è spento
            print ("LED "+str(ledPin[0]) + " on") #stampa la stringa "LED on" e il pin GPIO
            GPIO.output(ledPin[0],1)            #accende il LED
            return
        if GPIO.input(ledPin[0])== GPIO.HIGH: #se il LED è acceso
            print ("LED "+str(ledPin[0]) + " off") #stampa la stringa "LED off" e il pin GPIO
            GPIO.output(ledPin[0],0)            #spegne il LED
            return

#come per il LED 1
    def led_switch2(self):
        if GPIO.input(ledPin[1])== GPIO.LOW:
            print ("LED "+str(ledPin[1]) + " on")
            GPIO.output(ledPin[1],1)
            return
        if GPIO.input(ledPin[1])== GPIO.HIGH:
            print ("LED "+str(ledPin[1]) + " off")
            GPIO.output(ledPin[1],0)
            return

#come per il LED 1
    def led_switch3(self):
        if GPIO.input(ledPin[2])== GPIO.LOW:
            print ("LED "+str(ledPin[2]) + " on")
            GPIO.output(ledPin[2],1)
            return
        if GPIO.input(ledPin[2])== GPIO.HIGH:
            print ("LED "+str(ledPin[2]) + " off")
            GPIO.output(ledPin[2],0)
            return

#come per il LED 1
    def led_switch4(self):
        if GPIO.input(ledPin[3])== GPIO.LOW:
            print ("LED "+str(ledPin[3]) + " on")
            GPIO.output(ledPin[3],1)
            return
        if GPIO.input(ledPin[3])== GPIO.HIGH:
            print ("LED "+str(ledPin[3]) + " off")
            GPIO.output(ledPin[3],0)
            return

sm = ScreenManager() #crea un oggetto ScreenManager
sm.add_widget(LedScreen()) #aggiunge il widget root all'oggetto sm

class LedApp(App): #classe LedApp dell'applicazione principale
    def build(self): #funzione build all'interno della classe LedApp
        return sm #ritorno della funzione build con lo ScreenManager completo

LedApp().run() #esecuzione dell'applicazione principale
```

Listato 3

```
Builder.load_string(“””
<MenuScreen>: #classe root MenuScreen
    BoxLayout: #widget BoxLayout
        orientation: “vertical” #orientamento vertical del layout
        Button: #widget Button
            text: ‘Goto LED screen’#testo del pulsante
            background_color: [0,0,1,1] #colore dello sfondo
            on_press: root.manager.current = ‘settings’ #richiama la classe LedScreen che ha il nome “settings”
        Button: #widget Button
            size_hint_x: .25 #dimensione x al 25%
            size_hint_y: .25 #dimensione y al 25%
            pos_hint: {‘right’: 1} #posizione a destra e y invariato
            text: ‘Quit’#testo del pulsante
            on_press: root.quit_app() #richiama la funzione per terminare l’applicazione

<LedScreen>: #classe root LedScreen
    BoxLayout: #widget BoxLayout
        orientation: “horizontal” #orientamento horizontal del layout
        Button: #widget Button
            text: ‘LED 1’ #testo del pulsante
            on_press: root.led_switch1() #evento per richiamare la funzione led_switch1
        Button: #widget Button
            text: ‘LED 2’ #testo del pulsante
            on_press: root.led_switch2() #evento per richiamare la funzione led_switch2
        Button: #widget Button
            text: ‘LED 3’ #testo del pulsante
            on_press: root.led_switch3() #evento per richiamare la funzio0ne led_switch3
        Button: #widget Button
            background_color: [0,1,1,1] #colore dello sfondo
            text: ‘Back to main menu’ #testo del pulsante
            on_press: root.manager.current = ‘menu’ #richiama la classe MenuScreen che ha il nome “menu”
“””)

#Il listato della classe LedScreen con le funzioni di controllo GPIO dei LED:
class LedScreen(Screen): #classe LedScreen derivata da Screen
    def led_switch1(self): #funzione per il primo LED
        if GPIO.input(ledPin[0])== GPIO.LOW: #se il LED è spento
            GPIO.output(ledPin[0],1)            #accende il LED
            return
        if GPIO.input(ledPin[0])== GPIO.HIGH: #se il LED è acceso
            GPIO.output(ledPin[0],0)            #spegne il LED
            return
    def led_switch2(self): #funzione per il secondo LED
        if GPIO.input(ledPin[1])== GPIO.LOW: #se il LED è spento
            GPIO.output(ledPin[1],1)            #accende il LED
            return
        if GPIO.input(ledPin[1])== GPIO.HIGH: #se il LED è acceso
            GPIO.output(ledPin[1],0)            #spegne il LED
            return
    def led_switch3(self): #funzione per il terzo LED
        if GPIO.input(ledPin[2])== GPIO.LOW: #se il LED è spento
            GPIO.output(ledPin[2],1)            #accende il LED
            return
        if GPIO.input(ledPin[2])== GPIO.HIGH: #se il LED è acceso
            GPIO.output(ledPin[2],0)            #spegne il LED
            return

#Il listato della classe MenuScreen con la funzione quit_app per terminare l’app:
class MenuScreen(Screen): #classe MenuScreen derivata da Screen
    def quit_app(self): #funzione per terminare l’app
        print (“Exit”) #Stampa “Exit”
        exit() #esce dall’applicazione

# Imposta l’oggetto screen manager con la transizione FadeTransition
sm = ScreenManager(transition=FadeTransition())
sm.add_widget(MenuScreen(name=‘menu’)) #aggiunge il widget root MenuScreen
sm.add_widget(LedScreen(name=‘settings’)) #aggiunge il widget root LedScreen

class MenuApp(App): #classe dell’applicazione chiamata MenuApp
    def build(self): #costruisce lo screen manager
        return sm #ritorna lo screen manager coni widget

MenuApp().run() #esegue l’applicazione MenuApp
```

SCREENMANAGER

Nell'esempio precedente abbiamo importato il modulo **ScreenManager** che è un gestore di schermate. La classe, come valore predefinito visualizza solo una schermata alla volta e, nel caso sia necessario produrre schermate multiple, utilizza una classe *TransitionBase* per passare da una schermata all'altra applicando vari effetti tra quelli previsti:

- **NoTransition**: cambia le schermate senza animazione;
- **SlideTransition**: fa scorrere lo schermo da qualsiasi direzione come le slide;
- **CardTransition**: le nuove slide appaiono come nelle carte;
- **SwapTransition**: scambia le schermate;
- **FadeTransition**: crea una sfumatura in entrata o uscita;
- **WipeTransition**: cancella la schermata corrente da destra a sinistra;
- **FallOutTransition**: la schermata precedente "cade" e diventa trasparente, rivelando quella nuova;
- **RiseInTransition**: la schermata corrente si alza dal centro dello schermo mentre passa da trasparente a opaco.

Per attivare le modalità di transizione è necessario importare anche il modulo relativo alla transizione; per esempio, per applicare la transizione *FadeTransition* bisogna importare il modulo corrispondente:

```
from kivy.uix.screenmanager import FadeTransition
```

Per applicare una transizione allo ScreenManager, basta scrivere la seguente istruzione (vedi il codice di seguito):

```
sm = ScreenManager(transition=FadeTransition())
```

GESTIONE MULTISCHERMO

Per la gestione di più schermi abbiamo pensato di creare una semplice applicazione in grado di accedere a due schermate:

- la prima schermata ha un pulsante orizzontale con la scritta "Goto LED screen" che rimanda alla seconda schermata di controllo LED e un pulsante "Quit" per terminare l'applicazione (**Fig. 4**);
- la seconda schermata ha quattro pulsanti verticali (**Fig. 5**): tre per il controllo di altrettanti LED e un quarto pulsante per tornare alla prima schermata.

Il codice per la gestione della porta GPIO è identico al precedente, per cui ci sembra superfluo riportarlo (il relativo file è *four_buttons_GPIO_screen.py* e potete scaricarlo dal nostro sito web *www.elettronicain.it* insieme agli altri file del corso). Ricordiamo solo che per impostare la transizione *FadeTransition* è necessario aggiungere il modulo all'inizio dello script:

```
from kivy.uix.screenmanager import FadeTransition
```

La parte più interessante da analizzare è la stringa KV che

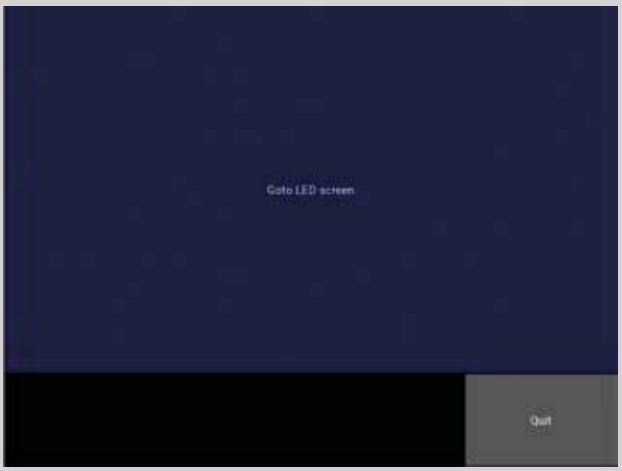


Fig. 4 - Prima schermata dell'esempio, con il pulsante Quit.

Fig. 5 - Seconda schermata dell'esempio.

serve a creare due schermi con due classi che abbiamo chiamato **<MenuScreen>** e **<LedScreen>**.
Abbiamo aggiunto anche la proprietà **background_color** per cambiare il colore dello sfondo del primo pulsante del **MenuScreen**. Il secondo pulsante del **MenuScreen** è stato ridotto al 25%, impostando la proprietà *size_hint* e posizionato in basso a destra con la proprietà *pos_hint*. Nella schermata **LedScreen** i primi tre pulsanti sono collegati ad altrettante funzioni di controllo GPIO dei LED. Il quarto pulsante ha un colore dello sfondo diverso ed è collegato alla funzione per tornare alla schermata precedente. Trovate il codice commentato nel **Listato 2**.

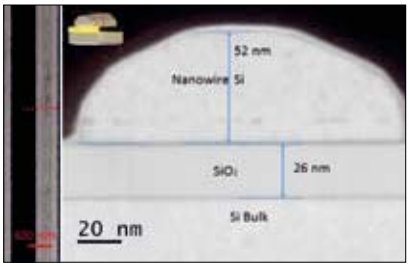
CONCLUSIONI

Per questa puntata ci fermiamo qui: vi abbiamo introdotto il linguaggio KV, che ci consente di gestire meglio le interfacce grafiche composte da più pagine. Con la speranza di essere stati sufficientemente chiari, vi diamo appuntamento alla prossima puntata. A presto! □

Dispositivi elettronici e fotonici sulla stessa piattaforma di silicio

Un team di ricercatori dell'Istituto di fotonica e nanotecnologie (Ifn) e l'Istituto per la microelettronica e microsistemi (Imm) del Consiglio nazionale delle ricerche (Cnr), in collaborazione con le Università di Marsiglia e Dresda, ha dimostrato come, sfruttando una naturale instabilità dei materiali sottili a semiconduttore come il silicio, si possano controllare in modo preciso e indipendente le dimensioni, la posizione, la direzione e le interconnessioni di nano-fili ottenuti direttamente su un substrato isolante. I risultati sperimentali sono stati confrontati con simulazioni teoriche, rendendo chiaro il meccanismo di formazione delle nanostrutture. Nella ricerca, pubblicata su *Nature Communications*, viene poi descritta un'applicazione di questi wires come transistor, la cui importanza risiede nella dimensione di queste strutture che soddisfano perfettamente le esigenze dei dispositivi nanometrici funzionanti nel range spettrale del medio infrarosso.

"*I circuiti realizzati su larga scala con materiale semiconduttore, privi di difetti e con interconnessioni controllate rappresentano il nesso tra componenti elettronici e fotonici. Nanostrutture sottili, simili a fili (denominati nano-wires), realizzate su silicio presentano proprietà elettroniche e ottiche superiori e configurabili rispetto agli stessi materiali depositati su film continui*", spiega **Monica Bollani** ricercatrice Cnr-Ifn e coordinatrice dello studio. "*Il loro dirompente potenziale è stato dimostrato in fotonica (ad es. come componentistica per laser o in ottica quantistica), in elettronica, nelle applicazioni termoelettriche o nella sensoristica per il rilevamento di gas. Per questi motivi, le attività di ricerca per la loro realizzazione sono state affrontate con una moltitudine di tecniche che mirano alla produzione di strutture controllate e ultra lunghe, dovendo rispondere alle esigenze di alto rendimento, di scalabilità (ad esempio, l'integrazione di un gran numero di dispositivi sullo stesso nanowires) e qualità del materiale (ad es. interfacce fluide).*"



Il primo volo del nuovo Boeing 777X

L'aereo col quale Boeing vorrebbe mettersi alle spalle, almeno parzialmente, i tanti problemi provocati dal modello 737 Max 8, ha compiuto con successo il primo volo di test. Basato sul diffuso 777 e con le innovative tecnologie del 787 Dreamliner, il 777X è decollato dalla pista di Everett, Washington, per un volo di tre ore e 51 minuti sullo stato di Washington prima dell'atterraggio al Boeing Field di Seattle.

"*Il 777X ha volato magnificamente e i test di oggi sono stati molto produttivi*", ha dichiarato il capitano **Van Chaney**, capo pilota per Boeing Test & Evaluation.

"*Il nostro team Boeing ha preso il jet a doppio corridoio di maggior successo di tutti i tempi e lo ha reso ancora più efficiente, più capace e più comodo per tutti*", ha affermato **Stan Deal**, presidente e CEO di Boeing Commercial Airplanes.

La flotta di prova, che ha iniziato i test a terra a Everett lo scorso anno, subirà nei prossimi mesi una serie completa di test a terra e in volo per dimostrare la sicurezza e l'affidabilità del progetto.

Membro più recente della famiglia widebody leader di mercato della Boeing, il 777X offrirà un consumo di carburante e emissioni del 10% inferiori e costi operativi del 10% inferiori rispetto alla concorrenza attraverso l'aerodinamica avanzata, l'ala composita in fibra di carbonio di ultima generazione e il motore commerciale più avanzato di sempre, il GE9 di GE Aviation.

In quest'ottica, la gamma di approcci disponibili per la crescita di nanofili è aumentato costantemente nel tempo, sia usando un approccio top-down tramite tecniche di nanolitografia, che al trasferimento di nanowires cresciuti da super-reticoli semiconduttori ottenuti via bottom up. Tuttavia, il pieno controllo sulla loro morfologia, dimensioni, posizione, direzione, interconnessione ma anche isolamento elettrico rimane una sfida poiché le tecniche attuali non sono versatili e spesso richiedono molti step di fabbricazione per l'implementazione dei wires semiconduttori. Con questa ricerca, viene quindi dimostrata una tecnica che permette un'efficace integrazione di dispositivi elettronici e fotonici sulla stessa piattaforma a base di silicio.

www.cnr.it

Il nuovo 777X combina anche il meglio delle cabine Dreamliner 777 e 787 per offrire la migliore esperienza di volo del futuro. I passeggeri potranno godere di una cabina ampia e spaziosa, ampi cassonetti che si chiudono facilmente per un comodo accesso ai loro effetti personali, finestre più grandi per una vista da ogni posto, migliore altezza e umidità della cabina, meno rumore e una guida più fluida.
Boeing prevede di consegnare il primo 777X nel 2021. Il programma ha segnato 340 ordini e impegni dai principali vettori di tutto il mondo, tra cui ANA, British Airways, Cathay Pacific Airways, Emirates, Etihad Airways, Lufthansa, Qatar Airways e Singapore Airlines.

www.boeing.com/777X

777X

Micro condensatori con capacità record ispirati dalla natura

Gli scienziati dell'Università Tecnica di Ilmenau sono riusciti a realizzare micro supercondensatori con prestazioni da record. I risultati del lavoro di ricerca nel campo della nanofisica applicata sono stati pubblicati sulla rivista internazionale "Nature Communications". L'efficiente stoccaggio di energia è la chiave per nuovi sistemi elettronici autonomi e per tecnologie wireless innovative nell'Internet of Things. Imitando la struttura dei favi, gli scienziati di Ilmenau hanno progettato e prodotto una nanostruttura in ossido di alluminio e l'hanno utilizzata come piattaforma per i nanoelettrodi e la costruzione di micro supercondensatori. La densità di energia ottenuta è paragonabile a quella delle moderne microbatterie, ma con una densità di potenza molto più elevata per area - un risultato di ricerca notevole dal campo della nanofisica applicata. Tali dispositivi di accumulo di energia potrebbero essere utilizzati in futuro per nuovi sistemi elettronici autonomi e per tecnologie wireless, per reti di sensori o per dispositivi medici impiantabili. Nell'Internet of Things, le nuove tecnologie consentono di mettere in rete oggetti reali e virtuali tra loro e di farli lavorare insieme. Ma fornire ai dispositivi l'energia necessaria è un problema. Lo sviluppo di dispositivi miniaturizzati di stoccaggio dell'energia che possono essere integrati in un chip è una delle maggiori sfide tecnologiche per i ricercatori di tutto il mondo. Con le loro prestazioni elevate e la durata eccezionale, i micro supercondensatori sono la migliore soluzione per un'elevata energia e densità di potenza su scala ridotta.

www.tu-ilmenau.de/

Tre Guinness World Record per la stampante 3D dell'Università del Maine

Riconosciuta come la più grande stampante polimerica 3D del mondo, ha anche stampato il più grande oggetto solido e la più grande barca.
La nuova stampante 3D è progettata per stampare oggetti lunghi fino a 100 piedi per 22 piedi di larghezza per 10 piedi di altezza ed è in grado di stampare più di 500 libbre all'ora.
Durante l'evento presso il Centro strutture e compositi avanzati dell'Università del Maine, è riuscita a stampare una barca di 7,5 metri di lunghezza per 2 tonnellate di peso in 72 ore; la barca è stata battezzata col nome di 3Dirigo ed è attualmente in fase di test presso l'Alfond W2 Ocean Engineering Laboratory.
La stampante, unica nel suo genere, supporterà diverse iniziative ambiziose, tra cui lo sviluppo di materie prime a base biologica che utilizzano cellulosa derivata da risorse di legno e la prototipazione rapida di applicazioni civili, di difesa e infrastrutturali.

Una collaborazione di ricerca da 20 milioni di dollari con il Oak Ridge National Laboratory (ORNL), il più grande laboratorio scientifico ed energetico del Dipartimento dell'Energia degli Stati Uniti, sosterrà la ricerca fondamentale in aree tecniche chiave nella produzione di additivi su larga scala a base biologica.

https://umaine.edu

Coronato da successo il test finale senza equipaggio della navicella Crew Dragon

Era l'ultimo volo di test, quello decisivo, prima di utilizzare quella navicella – la Crew Dragon – per mandare uomini nello spazio. Un test che simulava un problema al razzo vettore in fase di lancio e che doveva verificare il buon funzionamento delle procedure di emergenza in grado di mettere in salvo la navicella e l'equipaggio. Mentre il razzo vettore veniva fatto esplodere dopo un paio di minuti dal lancio, il sistema di emergenza allontanava rapidamente la navicella per poi fare ricadere la Crew Dragon nell'Oceano al largo della Florida, rallentata nella caduta da quattro enormi paracadute che hanno funzionato perfettamente. Dunque Space X ha superato anche questa impegnativa prova e la Crew Dragon dovrebbe iniziare i voli umani tra qualche settimana, segnando il ritorno degli Stati Uniti alle missioni con equipaggio umano dopo dieci anni, ovvero dalla dismissione dell'ultimo Shuttle. In questo periodo, per mandare nello spazio i propri astronauti, Washington si era affidata alla russa Soyuz, pagando circa 80 milioni di dollari ciascun lancio. Una politica alquanto insolita per una potenza quale gli USA, frutto di gravi errori strategici da parte della NASA, e che si è dovuta rivolgere a organizzazioni private quali Space X di Elon Musk per recuperare il terreno perduto.

www.nasa.gov

Un razzo nel centro di Milano

Milano come Cape Canaveral, un razzo vettore VEGA è pronto ad aprire le porte dello Spazio ai visitatori del Museo della Scienza e della Tecnologia di Milano. In realtà il lanciatore Vega completa il tema dei razzi vettori affrontato nell'esposizione permanente dedicata allo Spazio raccontando, attraverso un impatto fortemente emotivo, come il passato, presente e futuro delle applicazioni per la ricerca aerospaziale passano anche dalle competenze e dalla progettualità italiana, un patrimonio internazionalmente riconosciuto che va preservato e valorizzato.
Questa nuova ed affascinante iniziativa del Museo milanese mette in mostra il lanciatore spaziale VV01 della prima missione **VEGA**, realizzato grazie a un progetto promosso da ESA e ASI, sviluppato in Italia da Avio S.p.A. Date le dimensioni che lo caratterizzeranno, **30 metri di altezza per 3 metri di diametro**, il modello di VEGA è stato collocato negli spazi esterni del Museo accanto a un altro enorme oggetto di grande richiamo come il sottomarino Toti.
L'**esposizione** è accompagnata da immagini suggestive e infografiche che raccontano le caratteristiche tecniche della tipologia dei lanciatori VEGA nonché le varie fasi di lancio.

www.museoscienza.org

Difendi il tuo IP, Brand e Fonti di Entrate

Soluzioni di Sicurezza facili da aggiungere, difficili da violare

Lascia che Microchip ti aiuti a proteggere non solo i tuoi progetti, ma anche il tuo Brand e le tue entrate. Grazie a due decenni di esperienza nel campo della sicurezza, i nostri esperti ti eviteranno la preoccupazione dell'integrare sicurezza, eliminano anche la necessità di costose competenze interne. Combina queste competenze con le nostre fabbriche e i servizi di provisioning sicuri e capirai perché così tante grandi aziende si affidano agli esperti Microchip nel realizzare i loro progetti.

Dalla crittografia sicura agli ambienti di esecuzione affidabili, trova le implementazioni di sicurezza che soddisfano le tue esigenze specifiche nella nostra vasta gamma di soluzioni basate su hardware e software.

SMART | CONNECTED | SECURE

Proteggi il tuo progetto su: www.microchip.com/Secure

MICROCHIP

Diventa operativo il progetto franco-tedesco per la produzione di batterie per autotrazione

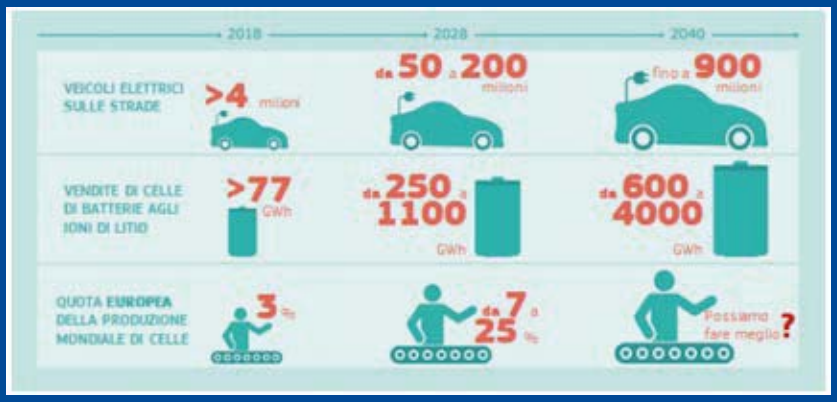
Qualcuno lo ha definito l'Airbus delle batterie perché il progetto assomiglia molto a quello che anni fa vide la nascita dell'Airbus, la società franco-tedesca che sarebbe diventata il primo produttore al mondo di aerei commerciali, spezzando il monopolio di Boeing. In questo caso non si tratta di spezzare il monopolio di qualcuno, semplicemente si tratta di garantire ai produttori europei di autovetture le batterie di cui avranno bisogno. In questi giorni è stato reso noto il nome e la roadmap della joint venture tra PSA-Opel e Total-Saft: la nuova società si chiamerà Automotive Cell Company e realizzerà innanzitutto un impianto a pilota a Nersac, in Francia. L'impianto servirà soprattutto a fare ricerca di prodotto mentre la produzione vera e propria inizierà solo nel 2023 con la prima fabbrica di batterie su larga scala che sorgerà nella regione dell'Hauts-de-France, con una capacità pro-

Sion: in arrivo nel nostro paese l'elettrica con pannelli solari integrati

È a tutti gli effetti una vettura elettrica da città, con un motore da 80 kW e un pacco batterie da 35 kWh che garantisce un'autonomia di circa 250 km con una ricarica e che permette alla vettura di raggiungere i 140 chilometri orari. E come tale può essere ricaricata tramite una presa casalinga Shuko (AC 3,7 kW), ricarica AC a 11 kW con connettore tipo 2 e ricarica rapida in DC a 50 kW (CSS, 80% in 30 minuti).
Ma la vera novità sono i pannelli fotovoltaici incorporati nell'intera carrozzeria (da cui la definizione di *Solar Electric Vehicle*) che sono in grado di ricaricare la batteria aggiungendo, in condizioni ottimali, una carica elettrica equivalente a 34 km al giorno. Un valore superiore alla percorrenza media di un pendolare. Questo significa che per la

duttiva iniziale di 8 GWh che salirà successivamente a 24 GWh. A ruota verrà realizzata una fabbrica gemella in Germania con la medesima capacità produttiva. Complessivamente i due siti produttivi avranno una capacità di 48 GWh annui, pari a un milione circa di pacchi batteria per auto. Il progetto avrà un costo complessivo di circa 5 miliardi di Euro, 1,3 miliardi dei quali verranno garantiti dalla Comunità Europea e dai governi tedesco e francese. Nel 2030 la fabbrica sarà in grado di garantire il 10/15 % del fabbisogno di pacchi batterie per autovetture elettriche.

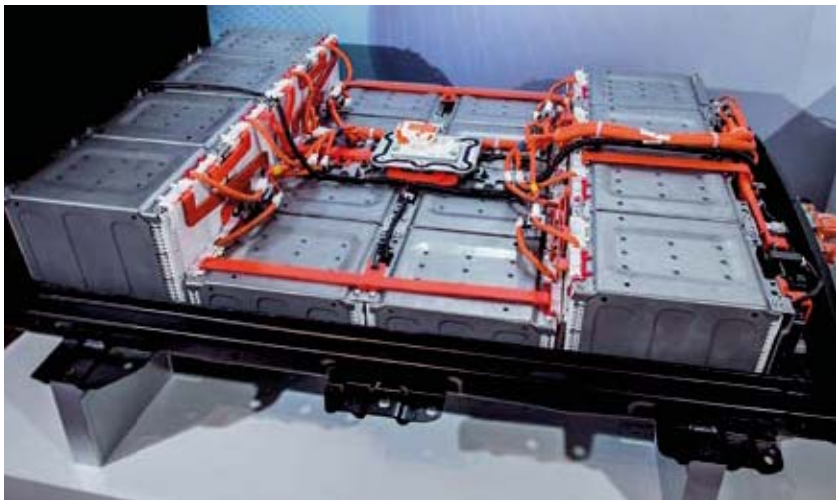
https://ec.europa.eu



stragrande maggioranza degli spostamenti potremo contare sulla energia (gratuita) ricavata dal sole.
Ma le sorprese non finiscono qui: è stato infatti comunicato il prezzo al pubblico, pari a 25.500 Euro, sicuramente un prezzo interessante per una vettura con queste caratteristiche.
Il rovescio della medaglia? Le difficoltà produttive e finanziarie in cui si dibatte questa interessante iniziativa da oltre otto anni, nonostante i moltissimi preordini, con alcuni clienti che hanno pagato per intero la vettura prenotata e la recente raccolta di oltre 50 milioni di dollari con l'ultima campagna di Community Funding.
A questo punto la domanda sorge spontanea: a quando un progetto simile da parte di Tesla?

https://sonomotors.com/it

Enel X: progetto per aumentare vita utile e sicurezza delle batterie agli ioni di litio



Enel X, la linea di business dedicata ai servizi energetici innovativi del Gruppo Enel, sarà una delle 17 aziende che riceveranno finanziamenti approvati dalla Commissione Europea nell'ambito del Progetto paneuropeo IPCEI (Important Project of Common European Interest), volto a sostenere la crescente filiera europea di batterie mediante investimenti in attività di ricerca e sviluppo (R&S) e First Industrial Deployment (FID).
Il progetto proposto da Enel X prevede lo sviluppo di strumenti basati sull'apprendimento automatico per la previsione dei guasti e la modellizzazione del deterioramento delle batterie agli ioni di litio di prossima generazione. Gli strumenti predittivi avranno l'obiettivo di allungare la vita utile delle batterie e aumentarne la sicurezza, ottimizzando al contempo le attività operative e di manutenzione. Il completamento del progetto è previsto entro il 2022.
Nell'ambito dell'IPCEI, sette Stati membri europei (Belgio, Finlandia, Francia, Germania, Italia, Polonia e Svezia) forniranno nei prossimi anni fino a circa 3,2 miliardi di euro di finanziamenti per l'intero progetto, che dovrebbe essere completato entro il 2031. Il progetto coinvolgerà 17 partecipanti diretti, per lo più aziende tra cui Enel X, che collaboreranno strettamente tra loro e con partner esterni, come piccole e medie imprese (PMI) e organizzazioni di ricerca pubblica in tutta Europa.

www.enel.com

Energia: ENEA nel progetto Ue per lo sviluppo sostenibile dell'eolico

Migliorare l'accettabilità sociale e promuovere lo sviluppo sostenibile dell'eolico in alcune regioni di sei Paesi europei (Italia, Germania, Lettonia, Norvegia, Polonia e Spagna), dove la produzione di energia dal vento è diffusa in maniera limitata.
È questo l'obiettivo del progetto europeo WinWind, finanziato dal programma Ue Horizon 2020 e coordinato per l'Italia da ENEA e dalla società Ecoazioni, con il coinvolgimento delle Regioni Lazio e Abruzzo.
Grazie ad un approccio inclusivo sul piano sociale il progetto ha analizzato le specificità socioeconomiche e ambientali delle Regioni partecipanti con lo scopo di sviluppare soluzioni su misura per accelerare lo sviluppo del mercato dell'eolico, anche dove sussistono barriere che ne ostacolano l'accettabilità sociale.
Per superare questi impedimenti WinWind ha condotto un'analisi comparativa di 10 buone pratiche selezionate in tutta Europa e ne ha testate alcune trasferendole in contesti locali, regionali e nazionali differenti rispetto a quelli in cui hanno prodotto risultati incoraggianti.

https://winwind-project.eu/project-it

Primo volo commerciale al mondo con un aereo elettrico

Harbour Air, la più grande compagnia aerea di idrovolanti del Nord America, ha annunciato il successo del primo volo commerciale al mondo realizzato con un velivolo elettrico. Il velivolo impiegato è un de Havilland Beaver DHC-2 da 6 passeggeri trasformato in un velivolo elettrico grazie ad un sistema di propulsione da 560 kW (750 CV) Magni500. Lo storico volo si è svolto sul fiume Fraser presso il terminal Harbor Air Seaplanes di Richmond (YVR South); ai comandi del velivolo il CEO e fondatore di Harbour Air Greg McDougall.
Inizia così, con questo volo, la terza era nel settore dell'aviazione - l'era elettrica.
"Oggi abbiamo fatto la storia", ha dichiarato Greg McDougall "Sono incredibilmente orgoglioso del ruolo di leader di Harbour Air nel ridefinire la sicurezza e l'innovazione nel settore dell'aviazione e degli idrovolanti. Il Canada ha da tempo ricoperto un ruolo iconico nella storia dell'aviazione e far parte di questa incredibile pietra miliare è qualcosa di cui tutti possiamo essere veramente orgogliosi." Il magni500, presentato al Paris Air Show di giugno 2019, è un sistema di propulsione elettrica ad alta densità di potenza che garantisce un modo pulito ed efficiente per alimentare gli aeroplani.

www.harbourair.com

ABB selezionata da IONITY per la seconda fase di espansione della rete di ricarica europea

ABB ha ricevuto da IONITY un ordine per ulteriori 324 stazioni di ricarica per veicoli elettrici ad alta potenza da 350 kW. Le stazioni di ricarica entreranno in servizio in 24 paesi entro la fine del 2020 nell'ambito della seconda fase di espansione della rete.
ABB era già stata scelta nel 2018 come partner tecnologico per la fornitura di 340 stazioni di ricarica ad alta potenza a IONITY ed è stata la prima a commercializzare in Europa stazioni di ricarica da 350 kW con cavi raffreddati a liquido. Un livello di potenza così elevato può consentire di ripristinare 200 chilometri di autonomia in soli 8 minuti. IONITY, una joint venture tra BMW Group, Daimler AG, Ford Motor Company e il Gruppo Volkswagen con Audi e Porsche, ha finora aperto 202 siti di ricarica in 18 mercati europei. IONITY ha recentemente celebrato il lancio della sua 200^ stazione di ricarica a Lully, in Svizzera, con le ultimissime stazioni di ricarica ABB, personalizzate IONITY. Le nuove stazioni di ricarica beneficiano anche di altre caratteristiche migliorate, tra cui un funzionamento più silenzioso, un display su misura e cavi più lunghi con retrazione integrata, che oltre a rendere più facile il raggiungimento della presa di ricarica su tutti i modelli di auto assicura che i cavi siano sempre tenuti lontano da terra e quindi privi di sporcizia.

www.abb.com

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

Interruttore WiFi

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

€7,90
Cod. FR775

Installabile in scatole di derivazione o porta frutto

€19,90
Cod. FR776

Prezzi IVA inclusa.

€24,90
Cod. FR778

Telecomando WiFi, 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.

iOS android
WORKS WITH amazon alexa
WORKS WITH Google Assistant
WORKS WITH IFTTT

Micro interruttore WiFi 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**

Presa smart WiFi

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

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

Caratteristiche tecniche e vendita on-line su: www.futurashop.it

Where are you?

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

Prezzi IVA inclusa.

LOCALIZZATORE GPS/GSM/GPRS CON ASCOLTO AMBIENTALE

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

NEW

cod. FR763

€ 74,00

MICRO LOCALIZZATORE SATELLITARE GPRS/GSM

Tutto sotto controllo con il nuovo micro localizzatore GPRS/GSM. Per attivarlo basta inserire una SIM di qualsiasi operatore; inviando un SMS potrai attivare l'ascolto ambientale oppure ricevere un link con le coordinate che indicano la posizione in tempo reale del localizzatore. Dispone di batteria agli ioni di litio ricaricabile. Ideale per tenere sotto controllo la posizione di autoveicoli, persone, cose, animali domestici, bagagli, ecc. La confezione comprende: il micro localizzatore, il cavetto USB-micro USB per la ricarica della batteria e le istruzioni in italiano.
Dimensioni: 43,2x32x14 mm.
Peso: 20 grammi.

cod. A8TRACK

€ 14,00

cod. TK102

€ 54,00

LOCALIZZATORE TASCABILE GPS/GSM/GPRS

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

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

Caratteristiche tecniche e vendita on-line su: www.futurashop.it