NUMERO 1 IN ITALIA SU ELETTRONICA EMBEDDED E SEMICONDUTTORI

Firmware 2.0

#19
Dicembre 2021

OPEN SOURCE IOT EMBEDDED SYSTEMS DESIGN

elektor ARTICLES INSIDE
design > share > sell

IOT

IN QUESTO NUMERO:

it.emcelettronica.com

Elettronica Open Source

PROFESSIONAL & MAKER ZONE

Firmware 2.0 EDITORIAL PLAN 2021

COSA LEGGERAI NEL 2021?

TOPICS	MAKERS ZONE	DATA DI PUBBLICAZIONE
Open Source IoT	Sensors	1 Febbraio
Automation	Industry4.0	1 Marzo
Energy Management	Energy Harvesting	1 Aprile
Wireless/RF	Smart Projects	1 Maggio
Automotive	Smart Mobility	1 Giugno
Open Source IoT	Blockchain	1 Luglio
Artificial Intelligence	Robotics	1 Settembre
LED/Optoelectronics	Smart Lighting	1 Ottobre
Power/Motor	Power Management	1 Novembre
Open Source IoT	Embedded Systems Design	1 Dicembre

GLI ARTICOLI

Elettronica Open Source

&

elektor

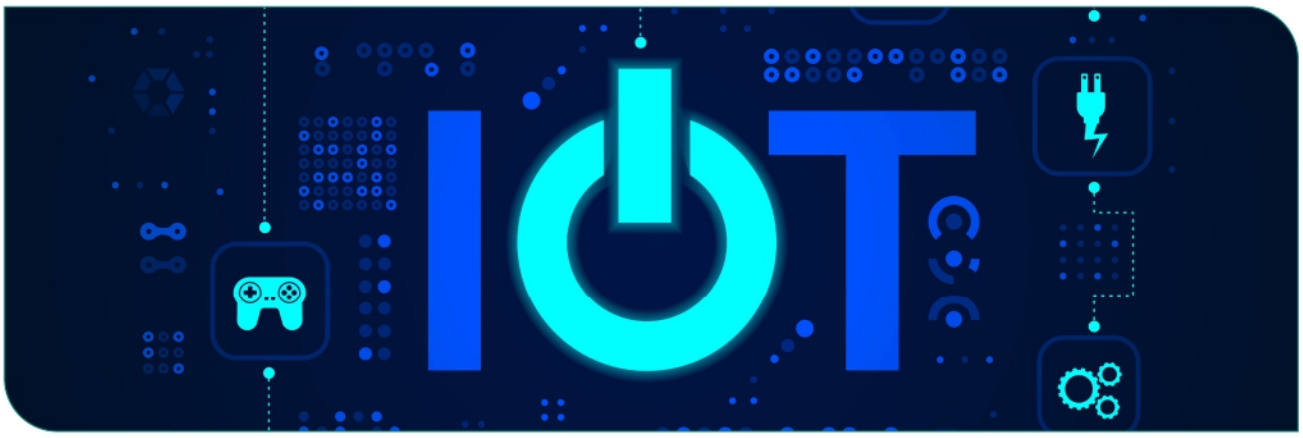
design › share › sell

IN UN SOLO MAGAZINE!

OGNI MESE SU FIRMWARE 2.0

2 ARTICOLI DI ELEKTOR

OPEN SOURCE IOT EMBEDDED SYSTEMS DESIGN



Founder&Editor
Emanuele Bonanni

CFO
Lidia Balica

Editorial Assistant
Maria Pisani

Maker in Chief
Giordana Francesca Brescia

Advertising & Marketing
Cristian Balica
cristian@contangosl.com

Graphic Designer
Marilde Mirra

Circulation
Users - 143.165
Social Network - 129.870

SISTEMI EMBEDDED E INTELLIGENZA ARTIFICIALE: UN BINOMIO POSSIBILE

Cari lettori,
questo nuovo numero di Firmware 2.0 è dedicato al settore dell'Open Source IoT-Embedded Systems Design. Come ben sapete, la costante innovazione nel settore embedded, sia dal punto di vista dello sviluppo di nuove piattaforme hardware sia delle applicazioni basate su microcontrollori, ha reso disponibili sul mercato nuovi dispositivi integrati e kit di espansione in grado di ampliare le possibilità applicative aggiungendo sempre più funzionalità ai progetti elettronici. Le applicazioni di controllo richiedono l'utilizzo di microcontrollori con elevati livelli di efficienza e affidabilità e che siano in grado di garantire al contempo elevate prestazioni. Nel vasto panorama delle tecnologie emergenti, l'AI Embedded è ad oggi un settore in crescita esponenziale. L'Intelligenza Artificiale applicata al mondo embedded si basa su algoritmi di auto-apprendimento da implementare su sistemi con capacità di calcolo limitata, quali sono appunto i dispositivi a microcontrollore, ed è sicuramente un settore sul quale l'industria elettronica sta investendo grandi risorse. Le tecnologie dell'Intelligenza Artificiale rappresentano infatti uno strumento fondamentale per l'innovazione in tantissimi settori, da quello industriale alla ricerca scientifica. Dal punto di vista computazionale i dispositivi embedded presentano delle configurazioni hardware limitate in termini di prestazioni di calcolo e risorse per l'analisi e l'elaborazione dei dati, oltre ad essere ovviamente dei sistemi special purpose. Per tale ragione, questa tipologia di sistemi non potrebbe garantire un supporto adeguato alle applicazioni di Intelligenza Artificiale, le quali, per definizione, sono energivore e ad alta intensità di calcolo, richiedendo elevate performance. In base a ciò, i dispositivi embedded sembrerebbero quindi incompatibili con le tecnologie dell'Intelligenza Artificiale. La realtà però mostra uno scenario ben diverso. Gli embedded device non sono più da considerare come semplici oggetti IoT connessi alla rete Internet e in grado di raccogliere dati da inviare ai server cloud. Grazie all'IA, infatti, i sistemi e le applicazioni embedded sono in grado di operare in modo adattivo e in totale autonomia, proprio come una classica applicazione autonoma basata su algoritmi di Intelligenza Artificiale, con elevati livelli di automatizzazione dei processi per gestire gli eventi senza la necessità di dover pre-programmare i dispositivi. Le architetture embedded possono creare sistemi adattivi e autonomi, non particolarmente complessi e dai costi accessibili, in grado di raccogliere ed elaborare informazioni, monitorare dati ambientali e implementare modelli per l'auto-apprendimento, senza rinunciare alla miniaturizzazione ed al fattore di forma tipici dei dispositivi embedded. Attraverso le tecnologie dell'IA i sistemi embedded possono raccogliere informazioni localmente e pre-elaborarle prima dell'invio al Cloud o a terminali intermedi.
L'Intelligenza Artificiale applicata al settore embedded consente ai dispositivi di valutare il proprio funzionamento e le prestazioni erogate, ma anche di monitorare i consumi energetici, aspetto che è di rilevante importanza per un'applicazione in campo embedded. Un passo in avanti notevole per il mondo dei microcontrollori. La diffusione capillare dell'Embedded AI, secondo gli analisti, porterà a una crescita di questo segmento di mercato oltre i 100 miliardi di dollari entro il 2025.
Buona lettura!

Giordana Francesca Brescia

PROGETTARE CON LA MCU ESP32

di **Daniele Valanzuolo**

La maggior parte dei nuovi progetti digitali necessita della presenza di connettività wireless per poter dar vita a dispositivi interconnessi e capaci di scambiare informazioni sia su brevi che su lunghe distanze. La filosofia dell'IoT (Internet of Things) è la rivoluzione del decennio che sta interessando qualsiasi ambiente o prodotto: consumer, industriale, retail, automotive, wearable e domotica. Con i chip della famiglia ESP32, Espressif ha dato una risposta efficace ed economica alle esigenze dei progettisti di qualsiasi settore, offrendo le potenzialità della connettività Wi-Fi e Bluetooth ad un costo contenuto senza trascurare aspetti come la potenza di calcolo, il consumo energetico e l'espandibilità delle funzionalità.

INTRODUZIONE

Sin dagli albori dell'avvento di Arduino, la fabbrica di semiconduttori Espressif si è distinta nel fornire chip e moduli per l'estensione delle funzionalità Wi-Fi all'interno della community open source dei makers Arduino. Ovviamente, parliamo del mitico **ESP8266** che ognuno di noi ha usato almeno una volta (e se non lo avete fatto vi consiglio caldamente di leggere l'articolo "Primi test con ESP8266 la nuova Wi-Fi Low-Cost dell'Internet delle Cose"). Con il passare del tempo e la crescita esponenziale delle esigenze dei makers interessati sempre di più al mondo IoT, Espressif si è impegnata per realizzare l'evoluzione dell'ESP8266 in grado di rispondere alle esigenze dei makers: è così nata la **famiglia ESP32**.

Questi processori sono stati concepiti e progettati per essere versatili e robusti in modo da adattarsi anche ai contesti industriali e automotive. Particolare attenzione è stata posta al consumo energetico per poter garantire la massima flessibilità in applicazioni IoT alimentate a batteria. Questo obiettivo è stato raggiunto grazie anche ad algoritmi software di gestione dinamica della potenza di calcolo. La società Espressif ha risposto alle esigenze dei progettisti e dei makers con il rilascio di moduli specifici capaci di adattarsi alle specifiche esigenze grazie all'integrazione a bordo di circuiti come antenne per le comunicazioni, amplificatori di potenza e di ricezione a basso rumore, moduli di gestione dell'alimentazione, etc. Infine, punto di forza, le dimensioni molto ridotte sia dei chip che dei moduli.

LE FAMIGLIE TRA CUI SCEGLIERE

Quando si parla genericamente dei **chip ESP32** in realtà si sta menzionando non un prodotto in particolare, ma una famiglia di microcontrollori (MCU) con architettura a 32bit, connettività Wi-Fi (2,4 GHz) e Bluetooth/BLE. Il contenuto tecnologico di questi chip può variare notevolmente a seconda del modello esatto, e di seguito andremo a vedere le principali caratteristiche della famiglia.

Il chip base ESP32 (**Figura 1**) è dotato di 48 pin e presenta le seguenti caratteristiche:

- **MCU principale single-core o dual-core Xtensa® LX6**
  - 520 kB SRAM
  - 448 kB ROM
  - 16 kB RTC SRAM
- Diverse modalità di risparmio energetico con modalità Sleep current inferiore a 5uA
- Periferiche di comunicazione high-speed SPI, UART, I2S e I2C
- GPIO con funzionalità touch e PWM
- Canali analogici di ingresso e di uscita
- Sensore di temperatura interno
- Completamente certificato con antenna integrata e stack software

La famiglia si arricchisce di chip più veloci (frequenze di clock fino a 240MHz) e un numero superiore di GPIO con i chip **ESP32-S2** (single-core) e **ESP32-S3** (dual core). All'interno di questi chip, anziché il processore Xtensa® LX6, troviamo il modello **Xtensa® LX7**. I processori delle famiglie S2 ed S3 presentano inoltre funzionalità di sicurezza aggiuntive quali: eFuse, crittografia flash, avvio sicuro, verifica della firma, algoritmi AES, SHA e RSA integrati. Infine, la serie **ESP32-C3** raggruppa chip dotati

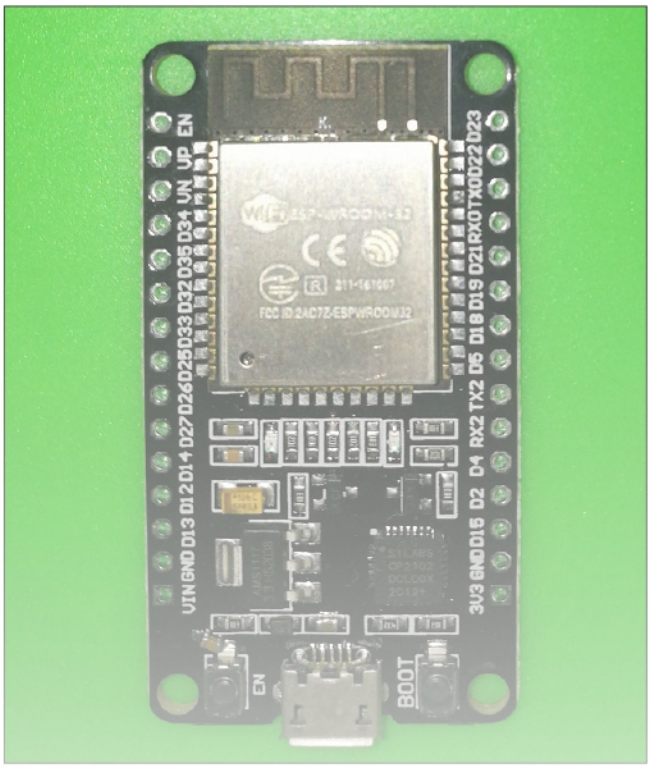


Figura 1: Scheda di sviluppo con ESP32 a bordo

di **un processore 32bit single-core con architettura RISC-V** e frequenze di funzionamento fino a 160 MHz. Con questa famiglia vengono garantite prestazioni elevate dei circuiti a RF.

IL GIUSTO RUOLO NELL'ARCHITETTURA

La ricca disponibilità di interfacce di comunicazione rende gli oggetti ESP32 (ma se preferiamo i chip che li mo-

Questo elemento è molto utile quando si desidera progettare applicazioni a basso consumo, in cui la CPU principale sia spenta e venga accesa solo quando si verificano determinati eventi. Inoltre, gli eventi di accensione possono essere gestiti tra RTC, interrupt esterni, ingressi capacitivi e tanto altro. Per il corretto funzionamento, soprattutto nelle diverse modalità sleep, il coprocessore ULP è dotato di un RTC da 8MHz e di una memoria (da 8kB) che può essere utilizzata sia per memorizzare le istruzioni da eseguire sia per condividere dei dati con la CPU principale. Inoltre, attraverso il bus, il coprocessore può acquisire i dati dalle periferiche del processore principale (ADC, sensore effetto Hall, input capacitivi, etc.) ed eseguire funzioni quali: operazioni aritmetiche, memorizzazione di dati, gestione dell'esecuzione dell'applicativo, comunicazioni, acquisizione di segnali analogici e digitali. I moduli ESP32 forniscono 10 pin per la funzionalità di touchpads: ossia, **pin sensibili alle variazioni di capacità** che possono essere utilizzati per realizzare (attraverso le pads sul PCB) degli interruttori/pulsanti di tipo touch. Questa è una tecnologia che oggigiorno troviamo largamente nell'elettronica di consumo in quanto facile da implementare e molto economica data l'assenza di componenti meccanici. L'acquisizione dei sensori tattili avviene direttamente da una periferica dedicata degli ESP32, che può essere interrogata dal software (andando a leggere i relativi registri) oppure può generare degli interrupt, molto utile quando si utilizza la CPU nelle diverse modalità di risparmio energetico. Nell'ultimo caso sarà il coprocessore ULP (Ultra Low Power processor) a gestire il rilevamento del tocco e la dell'attivazione del core principale. All'interno

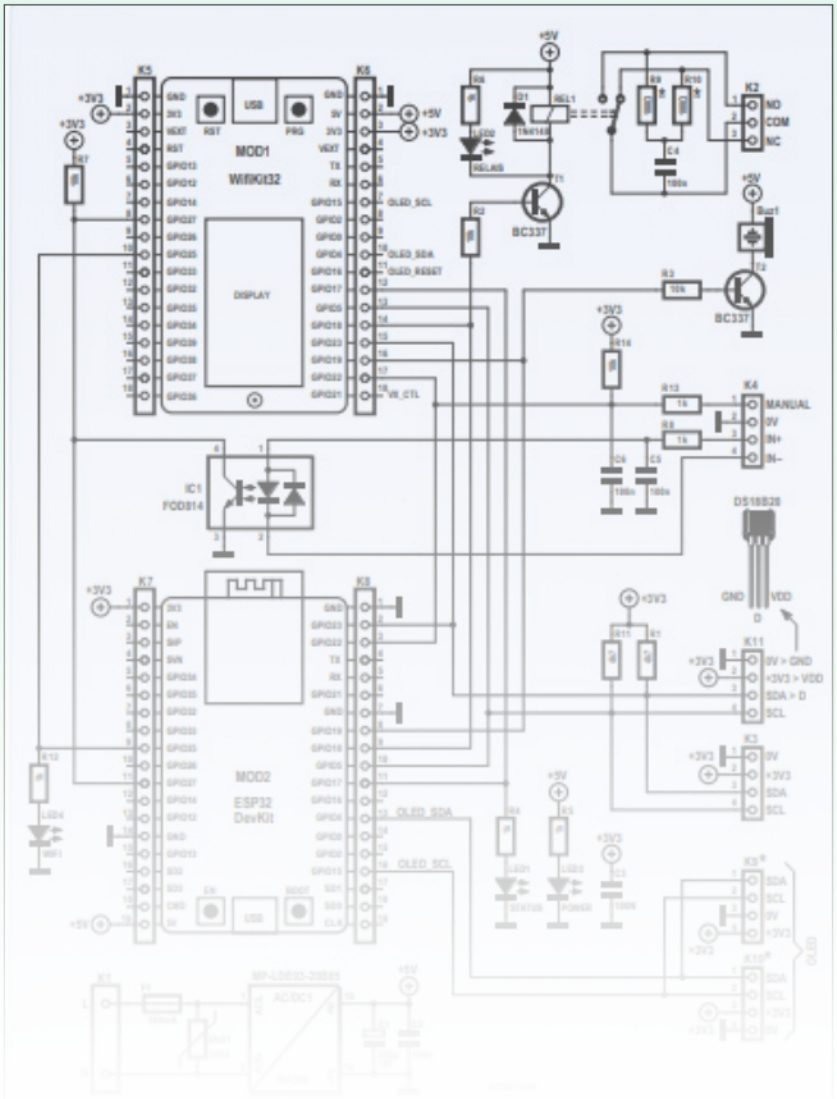
PROGETTO CON ESP32- TERMOSTATO CONNESSO – CONSERVA IL TUO VINO ALLA GIUSTA TEMPERATURA!

Per assicurarsi un adeguato invecchiamento del vino è importante conservarlo in un luogo fresco e poco illuminato. Nonostante molti specialisti si trovino in disaccordo su quale sia la temperatura ideale di conservazione del vino, tutti concordano sul fatto che questa debba essere costante. Il termostato presentato in questo articolo non solo vi permette di conservare il vino a una temperatura costante, ma è anche capace di avvisarvi via email in caso di sbalzi di temperatura.

L'idea di questo progetto mi venne durante la realizzazione del **termostato Wi-Fi di Elektor** [1], progettato con l'**ESP8266** di **Espressif**. Ho provato a convertire il software da quel progetto a uno basato su **ESP32**, ma ho lasciato perdere per via della mole di lavoro che comportava. Ho quindi deciso di ricominciare da capo prendendo ispirazione da ciò che era già stato fatto (interfaccia web, orologio NTP, controllo temperatura con isteresi programmabile, etc.) migliorando l'interfaccia web e correggendone i difetti (in particolare la User Interface, che non veniva aggiornata in tempo reale). Ho aggiunto anche un piccolo **display OLED**, molto pratico, che permette di utilizzare il termostato anche se la connessione Wi-Fi viene interrotta (in condizioni di degrado). Questo progetto è facilmente adattabile ad altre applicazioni. Per fare un esempio, ho usato lo stesso hardware per un rilevatore di umidità DHT11 (o BME280) per controllare il sistema di ventilazione meccanico del mio bagno. L'input optoisolato rileva l'accensione della luce (punto luce 12 V) e spegne temporaneamente l'estrattore d'aria, piuttosto rumoroso, mentre si è in bagno.

IL CIRCUITO

Lo schema elettrico del termostato con l'ESP32 è rappresentato in **Figura 1**. Per ottimizzare costo, affidabilità e dimensioni, ho deciso di utilizzare un modulo Wi-Fi Kit 32 di Heltec (**Figura 2**) come cuore del termostato. Questo modulo basato su ESP32, acquistabile facilmente online, include un **display OLED integrato**, blu, 0.96", 128x64 pixel, dotato di LED bianco programmabile dall'utente. È collegato a K5 e K6. Da notare che il modulo ESP32 DevKitC (connettori K7 e K8) assieme al display OLED (0.96" 128x64 px) separato, possono essere utilizzati al



5 V dato che possiede un suo regolatore 3.3 V. Può anche essere alimentato direttamente mediante bus micro-USB.

RELÈ

Il carico del termostato viene commutato dal relè RE1 che, a sua volta, viene controllato da un piccolo transistor NPN (T1) connesso alla porta IO18. LED2 indica lo stato del relè. Il diodo D1 evita che le altre unità elettroniche commutino la corrente indotta dal relè, mentre il network snubber C4/R9 o C4/R10 (montare solo un resistore) evita che i contatti del relè scarichino scintille di commutazione in caso di carico elevato. Il relè usato può sopportare carichi di 250-VAC fino a 10 A. Entrambi i contatti, NO (normally open) e NC (normally closed), sono reperibili (K2). Da notare che K1 e K2 possono essere combinati in una morsettiera a 5 vie.

BUZZER

Il buzzer con oscillatore integrato (Buz1), guidato da T2 attraverso la porta IO19 è disponibile per feedback acustici. Dato che possiede già il suo oscillatore integrato, dovrete solo collegarlo alla corrente per sentire il "suo bip". Potete utilizzare anche il resistivo Buzzer AC ma, in

CLASSIFICAZIONE DI IMMAGINI CON L'ESP32-CAM

di **Andrea Garrapa**

In alcuni precedenti articoli abbiamo affrontato il problema dell'implementazione e distribuzione di modelli di Machine Learning su microcontrollori a bassissimo consumo elettrico. In particolare, abbiamo addestrato la scheda Arduino Nano 33 BLE Sense a discriminare il suono di due o più parole. Tutto ciò è stato possibile grazie alla piattaforma Edge Impulse, che mette a nostra disposizione un ambiente intuitivo e allo stesso tempo potente per creare modelli ML e poi distribuirli su dispositivi con forti vincoli in termini energetici e di risorse. In generale, la distribuzione del modello si può fare anche senza Edge Impulse con un procedimento molto più lungo e laborioso. In questo articolo faremo un piccolo passo avanti, continuando ancora ad avvalerci della piattaforma, ma cercando di distribuire il modello su un dispositivo non ancora ufficialmente supportato dalla stessa.

INTRODUZIONE

In questo articolo andremo a descrivere come distribuire un modello di Machine Learning per la classificazione di immagini su un dispositivo embedded con pesanti vincoli in termini di risorse. Distribuire un modello di Machine Learning in grado di eseguire l'inferenza su un dispositivo a microcontrollore così limitato è conosciuto in gergo come **TinyML**. Rispetto a precedenti esperienze di TinyML descritte in questo blog, la differenza con la nuova applicazione che andremo a trattare sta nel fatto che il modulo che utilizzeremo, ovvero l'**ESP32-CAM**, non è ancora ufficialmente supportato dalla piattaforma **Edge Impulse**.

Scriviamo "ufficialmente" perché alcuni sviluppatori hanno pensato di implementare una soluzione temporanea per supplire a questa mancanza. Detto questo, il procedimento che andremo a descrivere risulterà più laborioso rispetto alla formula tradizionale.

Per implementare l'applicazione utilizzeremo:

- un modulo **ESP32-CAM**, nel nostro caso particolare si è trattato della scheda **AI-Thinker** dotata di videocamera **OV2640**. Inoltre, un convertitore seriale UART-USB può facilitare la vita durante il caricamento del firmware;
- un account **Edge Empulse**, gratuito;
- un account **Google**, gratuito;
- l'editor **Arduino IDE**, gratuito.

Data la complessità dell'implementazione, non descriveremo nel dettaglio il modulo ESP32-CAM né tutte le fasi di configurazione e interfacciamento con l'editor **Arduino IDE**. Il modulo è stato ampiamente descritto in un **precedente articolo di Fulvio De Santis**, quindi **rimandiamo il lettore alla lettura dello stesso, per preparare l'ambiente di lavoro prima di procedere con l'implementazione odierna**.

Anche la piattaforma di sviluppo Edge Impulse è stata ampiamente trattata in un altro **articolo precedente**. In questo articolo forniremo comunque le istruzioni necessarie per muoversi all'interno dell'ambiente, ma per chi ne volesse sapere di più rimandiamo al link. In **Figura 1** vengono indicate sommariamente le fasi attraverso le quali ci muoveremo per implementare l'applicazione. I dati per l'addestramento verranno trasferiti da un repository verso la piattaforma Edge Impulse, utilizzando Google Colab. All'interno di Edge Impulse avverrà l'addestramento del modello e la preparazione di quest'ultimo per la distribuzione su microcontrollore. Infine, utilizzeremo l'IDE Arduino per caricare lo sketch comprensivo del modello sul modulo ESP32-CAM.

GOOGLE COLAB E KAGGLE

La prima cosa da fare è creare un nuovo progetto all'interno dell'account Edge Impulse. Appena creato, comparirà una procedura guidata di cui però non avremo bisogno,

Figura 1: Breve descrizione delle fasi per giungere alla distribuzione del modello ML sul modulo ESP32-CAM

quindi chiudiamola. Poiché il modulo **ESP32-CAM** non è supportato da Edge Impulse, non potremo utilizzare la scheda per acquisire i dati come invece abbiamo fatto nei **precedenti articoli**. Per tale motivo, importeremo i dati direttamente in Edge Impulse, attraverso Google Colab. **Colaboratory**, per chi non lo conoscesse, è uno strumento gratuito messo a disposizione da Google, che permette di scrivere codice Python, condividerlo ed eseguirlo sfruttando risorse remote messe a disposizione dalla compagnia. Per utilizzare Colab occorre possedere un account Google. Una volta autenticati nel proprio account Google sarà possibile aprire il seguente link: **https://colab.research.google.com/drive/1n-VFqIye7h16IOj3OmcfYcY4Bm_QEzJz**

Si aprirà un file con cui scaricare i dati di addestramento nel nostro progetto. Analizziamo i blocchi in cui è suddiviso il codice.

```
import os
```

occupa di data science, e dalla quale scaricheremo il dataset di addestramento. Le due righe successive permettono l'autenticazione. La quarta riga, invece, si occupa di scaricare il dataset, mentre l'ultima esegue la decompressione del file zip. Il dataset contiene 4242 immagini di fiori. La raccolta dei dati si basa su dati Flickr, immagini Google, immagini Yandex. È possibile utilizzare questo set di dati per riconoscere i fiori dalle foto. Le immagini sono divise in cinque classi: margherita, tulipano, rosa, girasole, dente di leone. Per ogni classe ci sono circa 800 foto. Le foto non sono ad alta risoluzione, circa 320x240 pixel.

```
!npm cache clean -f
!npm install -g npm
!n stable
!npm install -g --unsafe-perm edge-impulse-cli
```

Il terzo blocco, senza addentrarci in particolari, consente di utilizzare i comandi della CLI (Command Line Interface) di Edge Impulse, come ad esempio il comando `edge-impulse-`

RILEVATORE DI PRESENZE CON L'ESP32-CAM

di **Andrea Garrapa**

Con l'evoluzione delle reti neurali convoluzionali, è diventato semplice costruire programmi in grado di vedere. Le CNN filtrano i complessi input mappandoli in modelli e forme noti. La precisa combinazione di questi pezzi ci dice quali entità sono presenti in un'immagine digitale. In questo articolo andremo a descrivere un esempio di applicazione delle CNN nell'ambito della visione artificiale. La nostra applicazione utilizzerà la camera di un ESP32-CAM per rilevare la presenza o meno di una persona nel suo raggio visivo, e terrà il conto delle persone avvistate visualizzando il numero su un Display LCD.

INTRODUZIONE

Fino a poco tempo fa, il senso della vista non era disponibile alle macchine. La maggior parte dei robot testavano il mondo circostante grazie a sensori di prossimità e tattili, ottenendo informazioni sulla forma delle strutture dalle collisioni. La vista permette di ottenere informazioni come forma e utilità di un oggetto senza la necessità di interagire con esso. Purtroppo, un robot non aveva questa fortuna, poiché le informazioni visive erano troppo disordinate, non strutturate e difficili da interpretare. Oggi, grazie ai **modelli di visione artificiale basati su CNN**, il senso della vista è disponibile anche alle macchine, ed il numero di applicazioni è in costante crescita nei settori più diversi, come nei veicoli autonomi per evitare gli ostacoli, nei robot industriali per rilevare i difetti dei prodotti e nelle immagini medicali per la diagnosi di malattie. I modelli di visione artificiale basati su CNN possono anche essere distribuiti su dispositivi a microcontrollore che presentano vincoli stringenti in termini di risorse, come memoria e consumo di potenza. Questi dispositivi possono essere alimentati con piccole batterie per mesi o addirittura anni, rendendoli perfetti per applicazioni embedded che non necessitano di una connessione ad Internet. Ad esempio, si potrebbero piazzare nella giungla o in una barriera corallina, per tenere il conto degli animali in via di estinzione.

DESCRIZIONE DELL'APPLICAZIONE

In questo articolo andremo a descrivere un'applicazione embedded che utilizza un modello ML per classificare immagini catturate da una camera. Il modello è addestrato per riconoscere quando una persona è presente nell'input della camera. Questo significa che la nostra applicazione sarà in grado di rilevare la presenza o l'assenza di una persona e produrre un output in accordo. Ad esempio, il nostro codice potrebbe accendere una luce LED quando una persona viene rilevata o controllare altri oggetti. Nel nostro caso abbiamo deciso di tenere il conto delle persone rilevate e di aggiornare questo numero visualizzandolo su un **display LCD**.

Per realizzare l'applicazione abbiamo utilizzato i seguenti componenti:

- un modulo **ESP32-CAM**, nel nostro caso particolare si è trattato della scheda **AI-Thinker** dotata di videocamera **OV2640**. Inoltre, la presenza di un convertitore seriale UART-USB può facilitare la vita durante il caricamento del firmware;
- un classico **Display LCD** 16x2 con retroilluminazione blu e dotato di interfaccia I2C;
- alcuni **cavetti jumper**;
- una **breadboard**.

COLLEGAMENTO DEI COMPONENTI

In **Figura 1** vengono riportati i collegamenti da realizzare tra i vari componenti per l'applicazione in esame.

Il modulo **ESP32-CAM AI-Thinker** è una scheda di sviluppo ESP32 con una fotocamera OV2640 e diversi GPIO per collegare le periferiche. Tuttavia, non ha un programmatore integrato. Si rende necessario un programmatore FTDI per collegarlo al computer e caricare il codice. Molti

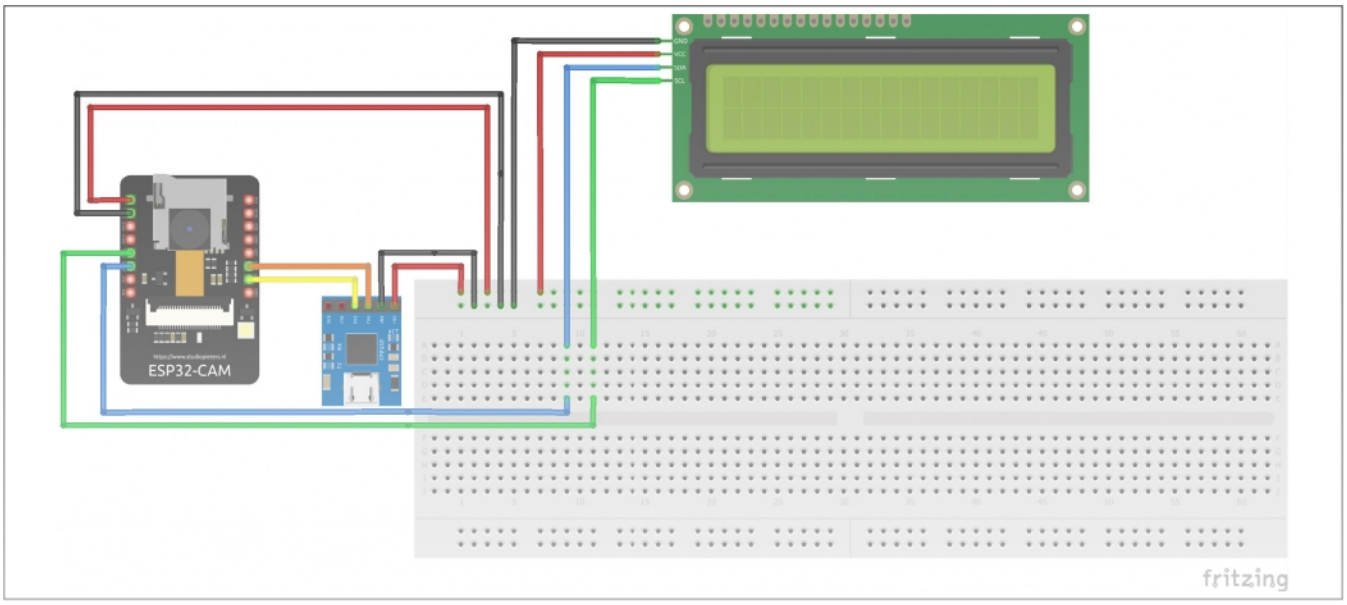


Figura 1: Schema dei collegamenti tra i vari dispositivi coinvolti nell'applicazione

programmatori FTDI hanno un ponticello che consente di selezionare 3.3V o 5V. Meglio assicurarsi che il ponticello sia nella posizione giusta per selezionare 5V. Nella **Tabella 1** vengono descritti nel dettaglio i collegamenti tra i pin dei diversi dispositivi e il colore del cavo associato.
Inoltre, occorre tener presente che in fase di **caricamento del firmware**, il GPIO0 dell'ESP32-CAM deve essere collegato a GND. Questo GPIO è collegato internamente a un resistore di pull-up da 10 kΩ. Quando GPIO0 è collegato a GND, l'ESP32 entra in modalità flash ed è possibile caricare il codice sulla scheda. Per far funzionare l'ESP32 "normalmente", basta disconnettere il GPIO0 da GND e premere il tasto di reset della scheda.

SETUP DELL'IDE ARDUINO

Prima di cominciare con la descrizione dell'applicazione occorre configurare l'IDE Arduino per operare con il modulo ESP32. In un articolo precedente vengono descritte tutte le operazioni per poter usare l'ESP32-CAM all'interno dell'IDE. Invitiamo quindi il lettore a fare click sul link seguente: **https://it.emcelettronica.com/riconoscimento-facciale-per-la-domotica-con-lesp32-cam-parte-1** e seguire i passi descritti nell'articolo al paragrafo "PROGRAMMAZIONE DELLA SCHEDA DI SVILUPPO ESP32-CAM", prima di procedere oltre. A questo punto, all'interno dell'IDE Arduino, occorre scaricare la libreria

PRAM, IL TUO PRIMO ROBOT AUTONOMO MOBILE – PARTE 4

di **Fulvio De Santis**

Nel precedente articolo "PRAM, il tuo Primo Robot Autonomo Mobile – Parte 3" abbiamo realizzato e installato la scheda di controllo di PRAM, eseguito tutte le fasi propedeutiche alla programmazione della scheda di sviluppo uChip, infine, abbiamo creato e salvato lo sketch del codice per il funzionamento di PRAM. In questa quarta ed ultima parte del progetto, descriveremo gli aspetti più rilevanti del codice e lo caricheremo nella scheda di sviluppo uChip, eseguiremo le tarature e il collaudo funzionale del prototipo e, finalmente, vedremo PRAM all'opera.

INTRODUZIONE

In questo articolo tratteremo la parte più interessante della "creazione" del **robot autonomo mobile PRAM**: daremo a PRAM una vita artificiale. PRAM si comporterà come una falena alla ricerca della luce; se la luce è troppo intensa si allontanerà da essa (evitamento della luce), oppure se l'intensità della luce è scarsa, allora la falena PRAM inizierà ad esplorare la stanza alla ricerca di luce e userà i suoi sensori (gli **switch paraurti** e il **sensore di distanza HC-SR04**) per evitare gli ostacoli. L'approccio classico all'Intelligenza Artificiale consiste nell'utilizzare la modellazione del mondo, pianificare l'azione ed eseguire questi piani. Questo tipo di metodo è adatto solo per ambienti strettamente controllati, come nel caso di applicazioni con il braccio robotico utilizzato nell'industria dell'assemblaggio di linee automobilistiche. Per i robot mobili autonomi la modellazione del mondo è diventata molto complessa ed è quasi impossibile mettere tutto l'ambiente imprevedibile e dinamico all'interno di questo modello. Come potremmo pianificare un'azione di evitamento di un ostacolo se viene spostato imprevedibilmente dalla sua originaria posizione?

Fino alla fine degli anni '80, questo metodo classico era ancora l'approccio dominante al robot mobile autonomo; con una complessa equazione matematica per modellare il mondo, il robot mobile diventa inaffidabile perché la sua reazione all'ambiente sarà molto lenta, inoltre, questo metodo di sicuro non saremmo in grado di eseguirlo con semplici microcontrollori. La soluzione è utilizzare l'approccio **bottom-up** (invece dell'approccio top-down utilizzato nell'Intelligenza Artificiale classica). Questo metodo è solitamente chiamato "Intelligenza Artificiale basata sul comportamento", in quanto la teoria asserisce che prima si realizza il programma di comportamento primitivo di base del robot, come l'evitamento degli ostacoli, poi si aggiunge via via un comportamento sempre più avanzato su di esso. Ad esempio, evitare la luce, seguire la luce, esplorare l'ambiente, etc. A ciascuno di questi comportamenti viene assegnata una priorità fissa per assumere il controllo degli attuatori del robot. Al comportamento più importante viene data una priorità più alta, come il comportamento di evitamento degli ostacoli. La combinazione di tutti questi comportamenti produrrà la risposta desiderata all'ambiente del robot.

L'ASSEGNAZIONE DELLE PRIORITÀ

Al fine di rendere il programma più strutturato e facile da modificare nel caso di inserimento di nuovi comportamenti in futuro, viene implementato l'**algoritmo selettore di priorità fissa** che consente a ciascuno dei comportamenti di avere il proprio controllo assoluto della guida e della velocità di PRAM. Come mostrato dallo schema di **Figura 1**, il funzionamento di PRAM si basa su cinque comportamenti ai quali viene assegnato un livello di priorità fisso a partire dal comportamento a priorità più alta (evitamento degli ostacoli con switch paraurti: livello 1) fino alla più bassa priorità (esplorazione: livello 5). Il selettore di priorità è il metodo dell'algoritmo che garantisce che un solo comportamento alla volta possa prendere il controllo dei motori di PRAM; questo selettore di priorità funziona

come un commutatore che collega l'azione del comportamento direttamente al controllo dei motori. Ad esempio, se l'inseguitore di luce rileva la luce nello stesso momento in cui uno switch paraurti colpisce un ostacolo, il selettore di priorità assicurerà che l'evitamento dell'ostacolo venga servito per primo, poiché ha una priorità maggiore rispetto al comportamento dell'inseguitore di luce.
Di seguito riportiamo le parti di codice relative alle funzioni dei cinque comportamenti di PRAM e al selettore di priorità.
Nella funzione *void loop* viene inizialmente eseguita la lettura dei sensori:

```
/* Start Read All the ADC input here */
// The PWM Motor Speed Control
speed=analogRead(A1); // read the input pin D1/A1.

// Bumped Sensor
bump_sensor=analogRead(A6);

// The LDR Sensor
ldr_left=analogRead(A5); // read the input pin D4/A4
ldr_right=analogRead(A4); // read the input pin D5/A5

// PRAM Steering Speed
PRAM_speed(speed*halt_status);
digitalWrite(uC3,0); // Turn Off PRAM Head Light
```

Dopo aver letto il valore di tutti i sensori, il programma continua chiamando le funzioni di comportamento per il relativo controllo dei motori, dalla funzione di evitamento

Occorre evidenziare che ogni funzione di comportamento utilizza il valore rilevato dal rispettivo sensore per inizializzare il generatore di numeri casuali mediante la funzione *srand(valore del sensore)* a cui si riferisce la funzione *random_number()* per generare un numero casuale e usarlo come valore di ritardo nella variabile relativa all'azione di guida di PRAM *steer_delay[ ][ ]=random_number()* che determina la durata di accensione dei motori di PRAM; questo ritardo influirà sul comportamento di guida di PRAM.
Il programma termina con la funzione di selezione delle priorità *PRAM_PrioritySelector(halt_status)* che stabilisce il comportamento prioritario di PRAM. Terminata l'azione di comportamento selezionata dal selettore, la funzione *PRAM_Reset()* resetta tutte le variabili delle cinque funzioni di comportamento.
Riportiamo di seguito il listato completo del codice *pram_1.0.ino*:

```
// ************************************************
********* //
// File Name : pram_1.0.ino
// Version : 1.0
// Description : PRAM is a Robot Autonomous Mobile
//
// Author : fds
// Target : uChip
// Compiler : C++
// IDE : Arduino
// Programmer : Arduino
// Last Updated : 12-08-2021
```

SCRIVERE UN SOFTWARE DI QUALITÀ CON LE GIUSTE REGOLE

di **Daniele Valanzuolo**

Lo sviluppo software e firmware ha assunto negli ultimi anni un ruolo sempre più critico e delicato nella realizzazione dei nuovi prodotti tecnologicamente avanzati. Ciò che prima era demandato all'uomo, ora è sempre più implementato a livello macchina per semplificare l'intervento umano nella gestione di tutte quelle azioni ripetitive, ma al tempo stesso delicate. In particolar modo, quando si parla di macchine legate alla sicurezza, o comunque ambienti critici come l'automotive, il trasporto aereo, ferroviario, etc. è necessario poter dotare i prodotti di un software con i più elevati standard qualitativi.

INTRODUZIONE

Di pari passo allo sviluppo ed alla diffusione del software è nata la branca ingegneristica denominata **"Ingegneria del Software", ossia la disciplina che si occupa del processo di produzione dei sistemi software** con particolare attenzione alle metodologie di sviluppo e verifica degli stessi.

Un aspetto fondamentale che viene introdotto con l'ingegneria del software è la qualità del sistema sviluppato. Il concetto di **qualità del software** non è semplice, ma è complesso in quanto racchiude differenti aspetti che il sistema software finale deve soddisfare sia dal punto di vista delle funzionalità che dal punto di vista della struttura, o meglio, architettura del software stesso.

Le svariate metodologie di sviluppo del software, dette anche cicli di vita, sono tra loro equivalenti e ognuna di esse presenta degli aspetti peculiari che la rendono maggiormente applicabile in determinati contesti soprattutto dove si predilige il riuso del codice o facilitare la manutenibilità del prodotto finale.

Il rapido aumento del ricorso al software sta concentrando sempre più l'attenzione sull'importanza critica di garantire l'affidabilità.

Gli studi e le ricerche nel campo dell'Ingegneria del Software hanno portato a definire dei "parametri di qualità" che maggiormente rispecchiano in maniera significativa cosa si intende per prodotto software di qualità.

Un ruolo fondamentale è occupato anche dalle tecniche di misura di questi parametri di qualità affinché i parametri possano essere misurabili, cioè quantificabili, e confrontabili. I principali obiettivi dei parametri di qualità sono la produzione di un software che, oltre alle garanzie del corretto funzionamento, possa essere (**Figura 1**):

- **testabilità**: ossia bisogna ridurre al minimo le parti di software non raggiungibile e tutto deve poter essere sollecitato per essere provato nel corretto funzionamento;
- **manutenibilità**: cioè al di là del progettista, anche soggetti diversi possono facilmente comprendere il codice del programma e modificare per introdurre migliorie. Questo vuol dire che il codice deve essere scritto in modo chiaro, con l'aggiunta di commenti;
- **portabilità**: cioè deve poter girare su diversi ambienti. Questa è una caratteristica soprattutto dei software che girano su architetture x86 o x64, mentre nella maggior parte dei contesti firmware, il software si adatta all'architettura del microcontrollore di fatto riducendo notevolmente la portabilità a vantaggio delle prestazioni.

Tutti questi aspetti, se correttamente implementati, possono consentire la corretta gestione del ciclo di vita del software, in particolar modo quando questo deve essere sviluppato o modificato successivamente.

IL MODELLO DI SVILUPPO A V

L'Ingegneria del Software definisce differenti modelli di sviluppo del software, che tra di loro si differenziano per alcune peculiarità e dunque sono preferibili in determinati

Figura 1: Parametri di qualità del software

contesti progettuali o meno.

I modelli universalmente riconosciuti sono:

- modello a cascata
- modello a V

L'utilizzo del modello a cascata (waterfall model) fornisce benefici per quanto riguarda la definizione delle attività e del controllo degli obiettivi.
Al contempo, fornisce anche degli effetti collaterali nella

LE REGOLE PER LA PROGRAMMAZIONE

L'aspetto della sicurezza associata ai prodotti tecnologici dotati di software è diventato un elemento cruciale e molto delicato. In ogni settore industriale sono stati definiti già da tempo degli standard ai quali i produttori devono conformarsi affinché il loro prodotto possa ricevere le certificazioni per un dato settore.
Per quanto riguarda il software dei sistemi embedded, scritto principalmente nel linguaggio C, e solo di recente anche in altri linguaggi tra cui il C++, questi devono esse-

UTILIZZO DEI DISPLAY IN PROGETTI CON RASPBERRY PI

Questo numero di Elektor Books presenta un capitolo tratto dal libro di Dogan Ibrahim "Using Displays in Raspberry Pi Projects" pubblicato da Elektor. Fin dalla loro introduzione, i display OLED sono stati favoriti dai Makers e dagli ingegneri dei sistemi embedded. In questo articolo esamineremo diverse modalità per implementare display OLED in vari progetti creando meno confusione possibile e con il minimo sforzo in termini di programmazione. Osiamo affermare che i suddetti requisiti puntano chiaramente verso Raspberry Pi, in quanto candidato perfetto. Scopriamolo in questo articolo!

Un diodo organico a emissione di luce (Organic Light-Emitting Diode, OLED) è un display LED nel quale lo strato elettroluminescente emissivo è costituito da una pellicola di composto organico in grado di emettere luce in presenza di una carica elettrica. Lo strato organico è situato tra due elettrodi, dei quali almeno uno è trasparente. I display OLED sono utilizzati in un gran numero di applicazioni commerciali, schermi televisivi, schermi di computer, smartphone, game console e altri dispositivi digitali. Gli OLED possono essere gestiti con controller a matrice passiva (PMOLED) o a matrice attiva (AMOLED). Solitamente, nei display di tipo PMOLED, ogni colonna e ogni linea sono controllate in maniera sequenziale, una alla volta. Nelle tipologie AMOLED un transistor backplane a film sottile è utilizzato per accedere e controllare direttamente l'accensione e lo spegnimento di ogni singolo pixel.

In generale, gli OLED hanno i seguenti vantaggi se comparati agli LCD:

- Il consumo energetico degli OLED è molto basso, sono dunque ideali per gli schermi dei dispositivi mobile
- I display OLED comportano ottimizzazione della qualità d'immagine, miglior contrasto, maggiore luminosità, angolo di visione più ampio e una frequenza di aggiornamento più rapida
- Gli schermi OLED si caratterizzano per maggior durabilità, ciò li rende appropriati per essere utilizzati a un range di temperatura più esteso
- Gli OLED possono essere flessibili, vengono infatti prodotti OLED pieghevoli, indossabili sul polso e così via dicendo

Gli OLED non sono comunque perfetti e comportano qualche svantaggio:

- Il costo economico è piuttosto elevato
- Si caratterizzano per un breve ciclo di vita (anche se numerose ricerche stanno cercando di ovviare a questo problema)
- Data la loro naturale emissività, l'uso degli OLED può risultare problematico sotto la luce solare diretta

In **Figura 1** sono rappresentati alcuni esempi di display OLED.

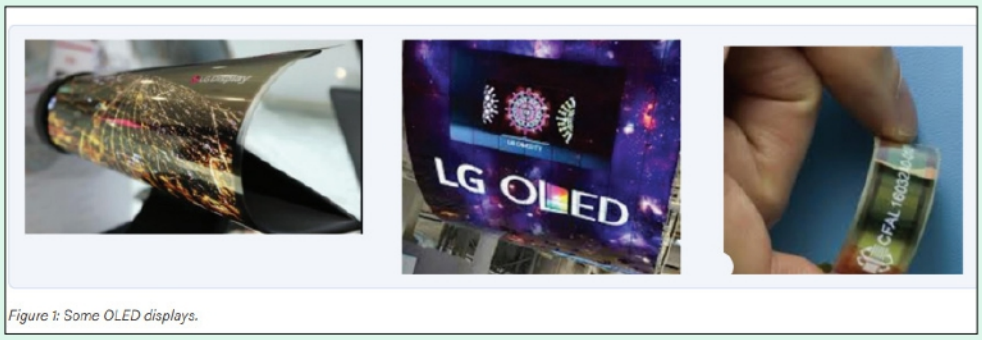


Figure 1: Some OLED displays.

UTILIZZO DEI DISPLAY OLED

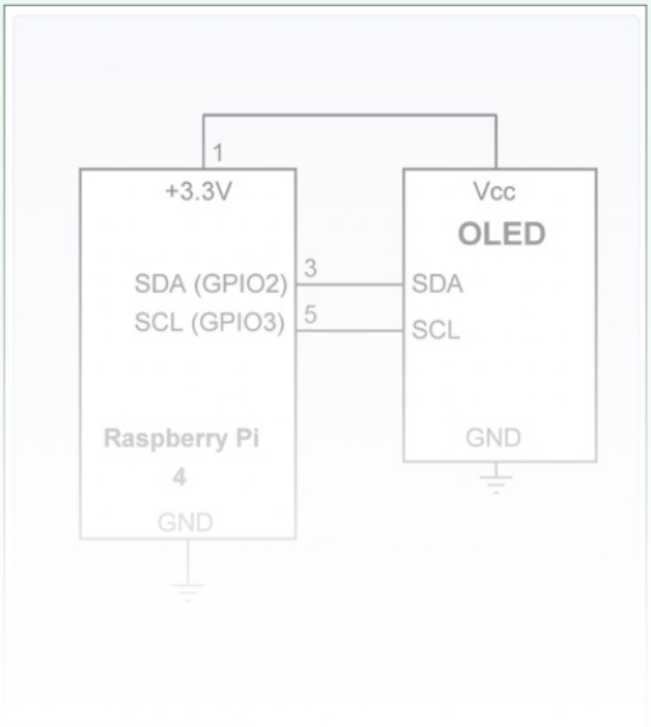
Esiste un gran numero di display OLED disponibili in diverse misure e dimensioni, e con diverse matrici di pixel. È ovviamente impossibile occuparci ora di tutti i tipi di display OLED disponibili sul mercato.

In questo capitolo ci focalizzeremo su un modello piuttosto popolare, un display OLED **128x32 pixel**, controllato con **chip integrato SSD1306**. Un ripasso delle caratteristiche del suddetto display potrebbe rivelarsi utile prima di cominciare a occuparsi del progetto.

Il tipo di OLED che utilizzeremo è rappresentato in **Figura 2**.

Raspberry Pi port	Raspberry Pi pin No.	OLED pin
SDA	(GPIO2)	2
SCL	(GPIO3)	3
GND	39	GND
+3.3V	1	V_{cc}

Nella **Figura 3** è rappresentata la connessione tra il Raspberry Pi e il display OLED. Normalmente il chip lavora a +3.3 V, ma c'è un regolatore di tensione sul chip che consente di operare anche a +5 V.



BLOKDOTS: PROGRAMMARE ARDUINO SENZA SCRIVERE UNA RIGA DI CODICE

di **Andrea Garrapa**

Blokdots è un software di facile utilizzo per realizzare prototipi hardware interattivi senza scrivere una riga di codice. L'idea nasce da una tesi di laurea presso l'Università di Scienze Applicate Schwäbisch Gmünd. In questo articolo andremo a fare un'analisi di questo strumento che consente di realizzare dei prototipi hardware anche per chi sa poco o nulla di programmazione. Nella parte iniziale descriveremo il software nella sua interfaccia e nelle sue opzioni, mentre nella parte finale faremo un esempio pratico per dimostrarne la semplicità d'uso e l'immediatezza.

INTRODUZIONE

Olivier Brückner, studente di design industriale, aveva fatto alcune esperienze extracurriculari nella progettazione dell'interazione. Durante questo periodo molti studenti di progettazione dell'interazione e studenti di design industriale si erano avvicinati chiedendogli aiuto. Avevano bisogno di una mano per collegare componenti come pulsanti o scrivere un paio di righe di codice, ad esempio, per far pulsare un LED. Roba piuttosto semplice per persone esperte, ma difficile per principianti assoluti. Nel 2017 Olivier ha focalizzato il lavoro della sua tesi di laurea sul colmare il divario tra le due discipline. Sulla base delle sue esperienze con gli studenti in cerca di aiuto, ha progettato uno strumento per **creare facilmente prototipi hardware**, ed è nato **Blokdots**. *Christoph Labacher*, invece, che era stato un caro amico e cassa di risonanza per idee e concetti durante la tesi, si è unito dopo la laurea e si è concentrato sulla creazione di logica e funzionalità. I due stanno ora costantemente sviluppando ulteriormente il progetto e migliorando il software, guidati dalle idee stimolanti e affascinanti che le persone stanno creando. La loro filosofia, che è poi diventata quella di Blokdots, è rendere in grado tutti di creare le proprie idee il più rapidamente possibile, senza essere limitati dai vincoli delle lacune di competenze.

REQUISITI

Attualmente **Blokdots** **può essere installato solo su sistemi operativi MacOS**, mentre una versione Windows è in fase di sviluppo. Il software **supporta solo schede Arduino**. Le schede supportate sono:

- Arduino Uno
- Arduino Leonardo
- Arduino Micro

Per poter comunicare con Blokdots, la scheda Arduino che si utilizza deve essere aggiornata con *Firmata*. *Firmata* è un protocollo speciale che consente alla scheda Arduino di comunicare con un computer. Avviando Blokdots e collegando una scheda Arduino, Blokdots flasherà automaticamente la scheda con il *Firmata*. Ciò è indicato da una piccola casella verde nella parte superiore della Live View. **Blokdots utilizza i componenti Grove creati da Seeed Studio**, poiché sono molto semplici da usare e non richiedono alcun cablaggio. Basta mettere semplicemente lo shield sulla scheda Arduino e collegare i componenti agli slot. **Blokdots funziona comunque anche senza lo shield Grove**. Ovviamente, si possono anche

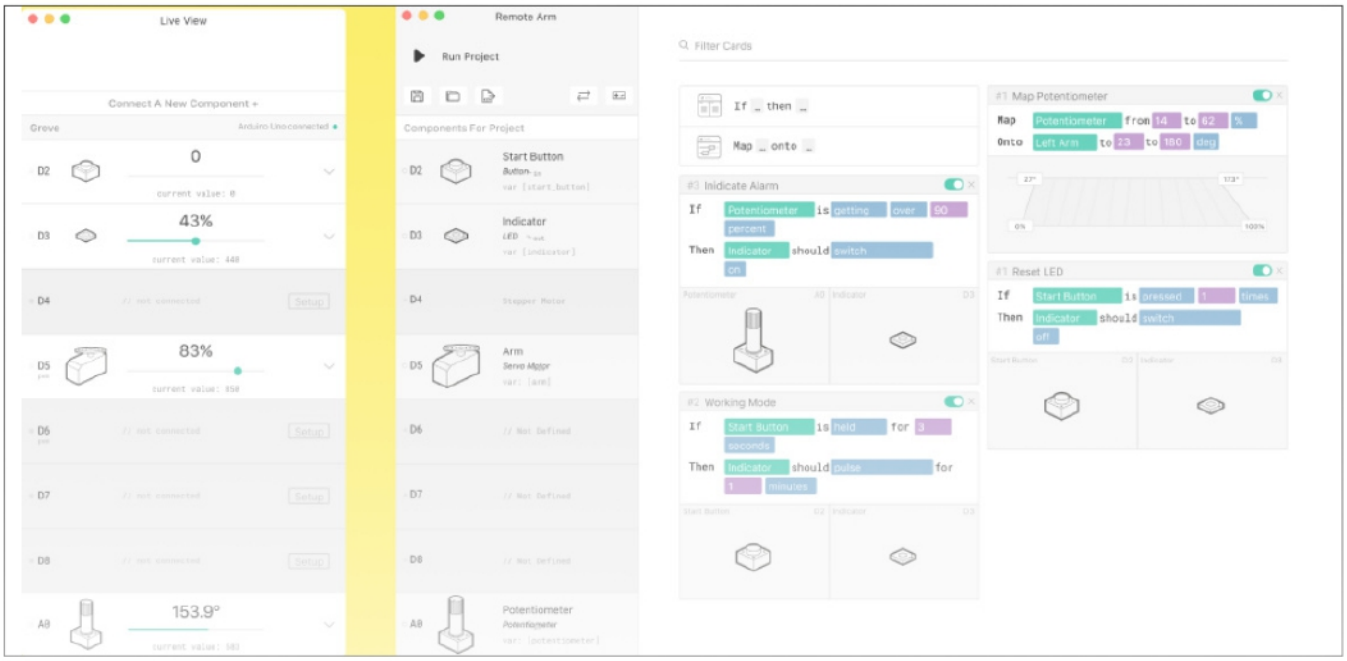


Figura 1: L'interfaccia grafica di Blokdots presenta due finestre in cui operare, sulla sinistra la Live View e sulla destra la Project View

usare i normali componenti. Il nome dei pin Arduino assomiglia ai nomi degli slot Grove. Per esempio, quello che sarebbe stato lo slot D4 è il pin 4 sulla scheda Arduino. L'interfaccia di Blokdots presenta due finestre (**Figura 1**): la **Live View** e la **Project View**. Andiamo ad analizzare nel dettaglio le due finestre.

LIVE VIEW

La Live View rappresenta il centro di controllo per la scheda Arduino collegata. Essa dovrebbe sempre riflettere lo stato hardware della scheda. Inoltre, è possibile

si trova quello slot su Arduino. Blokdots risolve questo problema per l'utente, con l'installazione guidata (Wizard). Facendo clic sul pulsante "Connect A New Component" nella parte superiore della Live View si aprirà la procedura guidata (**Figura 2**). Sul lato sinistro della procedura guidata è presente un elenco scorrevole dei componenti disponibili tra cui è possibile scegliere quello da connettere. Blokdots consiglierà uno slot libero in cui collegare il componente e mostrerà dove si trova quello slot sulla scheda nella piccola mappa sul lato destro. Se si è soddisfatti

+ 130.000
REGISTERED USERS

6.138 AVERAGE DAILY PAGEVIEWS (DEC2019)

824.057 2019 ANNUAL VISITORS

THE BIGGEST **EMBEDDED COMMUNITY** IN ITALY

SOCIAL CONNECTIONS

+ 83.000

+ 23.000

CATEGORIES

COMPANIES/CONSULTANTS
53 %

ACADEMICS/STUDENTS
25 %

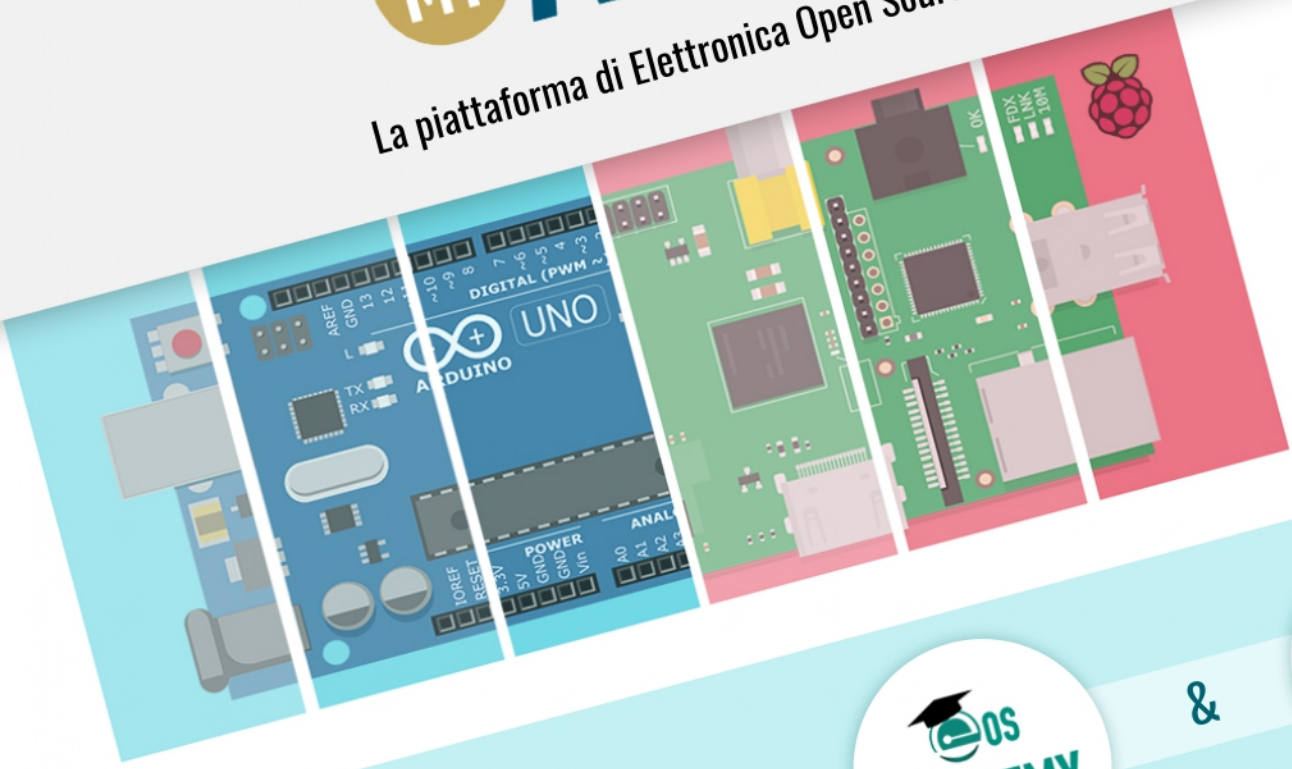
MAKERS/HOBBYISTS
22 %

Elettronica Open Source

I NOSTRI CORSI DI ELETTRONICA
PER I PROFESSIONISTI
E I MAKERS

MY ACADEMY

La piattaforma di Elettronica Open Source dedicata ai corsi



PUOI AVERE TUTTI I CORSI DI

EOS ACADEMY

&

MAKERS ACADEMY

A PORTATA DI CLICK