



Raspberry Pi

La rivista ufficiale Raspberry Pi tradotta in italiano per RaspberryItaly

PARTI CON L'ELETTRONICA

Impara come
costruire e
programmare circuiti





PARTI CON L'ELETTRONICA

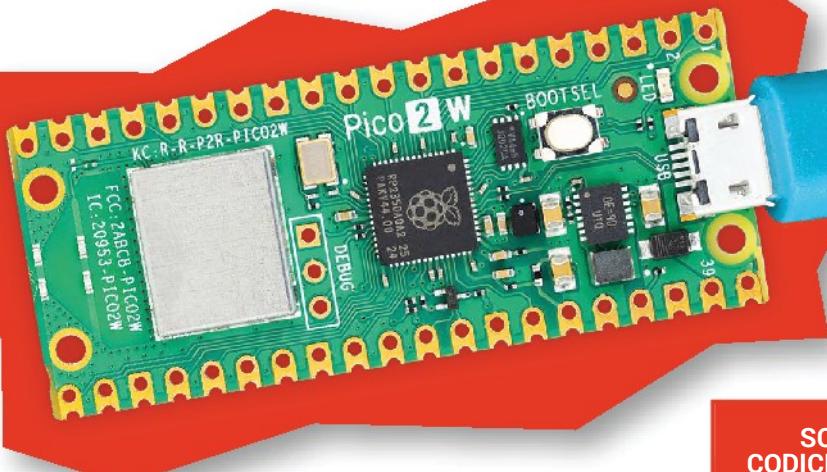
Inizia con l'elettronica con questo semplice gioco monodimensionale per Raspberry Pi o Raspberry Pi Pico



Maker
Ben Everard

La missione di vita di Ben è quella di aggiungere più LED al mondo. Deve ancora trovare un progetto che non venga da LED in più.

glowingart.co.uk



SCARICA IL
CODICE COMPLETO:



rpimag.co/github

Linguaggio: CircuitPython

computer e i microcontrollori Raspberry Pi sono strumenti eccellenti per interagire con gli oggetti nel mondo reale.

Possiamo aggiungere il nostro hardware o interfacciare con altri dispositivi elettronici utilizzando i pin GPIO (General-Purpose Input/Output). Questi pin possono rilevare tensione (in ingresso), inviare tensione (in uscita) e persino trasferire dati più complessi utilizzando protocolli di basso livello. In questo articolo, useremo queste tre diverse capacità per creare un gioco.

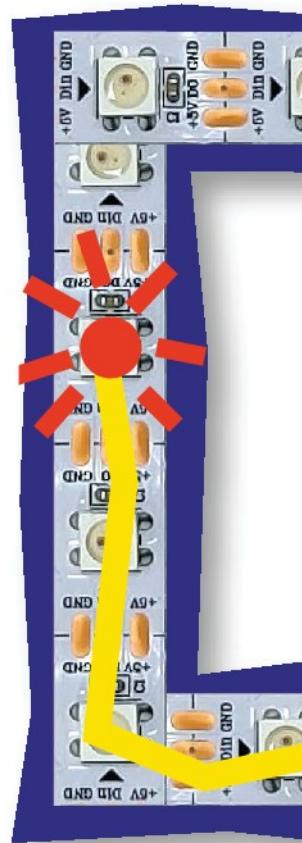
Useremo CircuitPython, perché ci permette di usare lo stesso codice sia su Pico che su Raspberry Pi

Useremo CircuitPython, perché ci permette di usare lo stesso codice sia su Pico che su Raspberry Pi. Questo ti consente di usare qualsiasi hardware hai già e ti aiuta anche a comprendere meglio le capacità di ciascun dispositivo. Se non hai mai usato CircuitPython prima, dai un'occhiata ai riquadri di questa guida per iniziare.

Nozioni di base sui pulsanti

Iniziamo con uno degli usi più semplici di un pin GPIO: la lettura di una tensione. La maggior parte dei pin GPIO sono digitali, il che significa che non misurano il livello di tensione esatto. Invece, rilevano solo se la tensione è alta (3,3 V su Pico e Raspberry Pi) o bassa (0 V). (Alcuni pin su Pico possono misurare diversi livelli di tensione – questi sono chiamati pin analogici – ma non li useremo in questo progetto.)

Per leggere i dati utilizzando un pin GPIO, dobbiamo costruire un semplice circuito che ci fornisca 3,3 V quando qualcosa è "acceso" e 0 V quando è "spento".



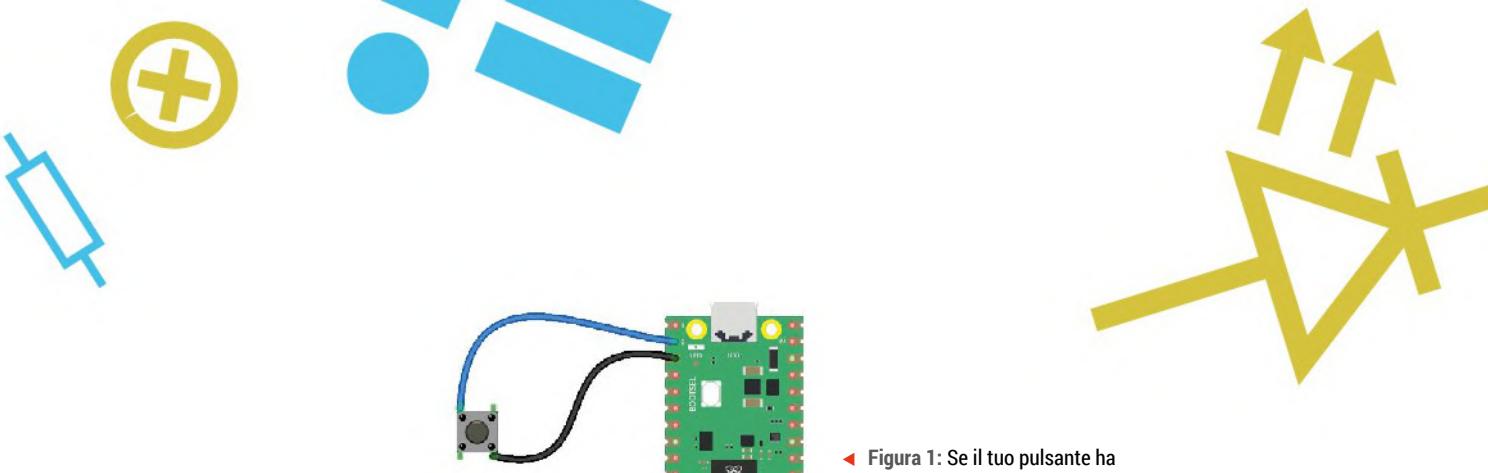
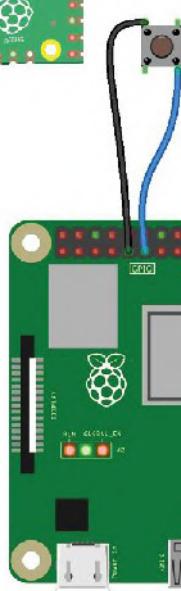


Figura 1: Se il tuo pulsante ha quattro connessioni, usa quelle diagonalmente opposte



I pulsanti sono chiamati anche switch a pressione momentanea. Di solito sono "aperti" e quando li si preme, "chiudono" la connessione e completano il circuito.

È importante capire che 0 V non è la stessa cosa di un pin non collegato. 0 V è un livello di tensione definito, mentre un pin non collegato (chiamato "floating") non ha una tensione definita e può comportarsi in modo imprevedibile. In questo stato, il pin può essere letto casualmente come ALTO o BASSO a causa di rumore elettrico o interferenze. Per questo motivo utilizziamo una resistenza di pull-up o pull-down: per dare al pin uno stato predefinito e affidabile quando non è collegato nient'altro.

La **figura 1** mostra un semplice circuito per collegare un pulsante ad un pin GPIO su Pico. Quando il pulsante non viene premuto, il pin GPIO è collegato a 3,3 V tramite la resistenza di pull-up, quindi legge ALTO. Quando il pulsante viene premuto, collega il pin GPIO direttamente a massa (il pin GND). Questo crea un percorso libero per la corrente che scorre verso massa e il GPIO legge un segnale BASSO (0 V).

Poiché le resistenze di pull-up e pull-down sono molto spesso necessarie con i pin GPIO, sia Pico che Raspberry Pi dispongono di resistenze interne che è possibile utilizzare. È sufficiente inizializzarle nel programma.

Ora, con questa teoria alle spalle, collegiamo un pulsante a Pico. Ci sono molti pulsanti diversi che si possono usare per questo. Se lo si desidera, si può anche semplicemente usare un filo per collegare GND e un pin GPIO e attaccare e staccare il filo per simulare la pressione del pulsante.

Usare Raspberry Pi

È possibile utilizzare il codice CircuitPython su un Raspberry Pi installando Blinka. Blinka funge da livello di compatibilità, consentendo alle funzionalità di CircuitPython di funzionare con Python standard. Per installarlo, dobbiamo prima creare un ambiente virtuale.

```
$ python3 -m venv venv --system-site-packages
$ source venv/bin/activate
```

Successivamente, configura il Raspberry Pi e installa Blinka con questi comandi:

```
$ pip3 install --upgrade adafruit-python-shell
$ wget https://raw.githubusercontent.com/adafruit/Raspberry-Pi-Installer-Scripts/master/raspi-blinka.py
$ sudo -E env PATH=$PATH python3 raspi-blinka.py
```

Il primo comando installa un pacchetto richiesto, il secondo scarica lo script di installazione di Blinka e il terzo esegue il programma di installazione.

Durante l'esecuzione di questa operazione, ti potrebbe essere richiesto di riavviare il computer.

Una volta completata l'installazione, puoi usare Blinka per eseguire il codice CircuitPython con il normale interprete Python. Assicurati solo che l'ambiente virtuale sia attivato. Puoi attivarlo in qualsiasi momento eseguendo il seguente comando da terminale.

```
$ source venv/bin/activate
```

Per eseguire il codice presente in questo articolo, salvalo in un file e scrivi:

```
$ python <filename.py>
```



Il codice per leggere questo in CircuitPython è mostrato di seguito. La differenza principale tra Pico e un computer Raspberry Pi è che la numerazione dei pin è leggermente diversa. I pin di Pico sono chiamati da GPIO a GP22, mentre i pin di Raspberry Pi sono da D0 a D26. Non tutti i pin possono essere utilizzati per ogni cosa, poiché alcuni sono riservati a funzioni specifiche. In generale, i pin di Pico sono più flessibili rispetto ai pin di un computer Raspberry Pi.

Abbiamo fornito entrambe le versioni dei pin nel codice seguente e dovete commentare (aggiungendo un # all'inizio della riga) quello che non state utilizzando e de-commentare (rimuovere il # all'inizio della riga) quello che state usando.

I pulsanti di solito hanno due o quattro connessioni. Se il pulsante ha due connessioni, potete collegarlo in entrambi i modi. Se ha quattro connessioni, allora ci sono in realtà due serie da due. Il modo più semplice per farlo è utilizzare connessioni diagonalmente opposte: entrambe le diagonali funzionano.

Mostriremo i collegamenti diretti al circuito con dei fili, ma puoi anche utilizzare una breadboard.

```
import board
import digitalio
import time

#Pico
BUTTON_PIN = board.GP1

#Raspberry Pi
#BUTTON_PIN = board.D17

button = digitalio.DigitalInOut(board.GP14)
# Usa il pin GPIO appropriato
button.direction = digitalio.Direction.INPUT
button.pull = digitalio.Pull.UP # Abilita le Resistenze di pull-up interne

while True:
    if not button.value: # Pulsante premuto
        (il pin viene letto in stato basso)
        print("Il pulsante è premuto")
    else:
        print("Il pulsante non è premuto")
    time.sleep(0.1)
```

Una cosa un po' strana di questo codice è la riga:

```
if not button.value
```

Ci si potrebbe aspettare che venga visualizzato **if button.value**, ma stiamo utilizzando un pull-up che riporta HIGH (che viene valutato come **True**) quando il pulsante viene premuto e LOW (ovvero **False**) quando viene premuto. Potrebbe sembrare un po' al contrario a prima vista, ma aiuta se si pensa a cosa sta succedendo a livello elettrico.

Sbatti le palpebre e te lo perdi

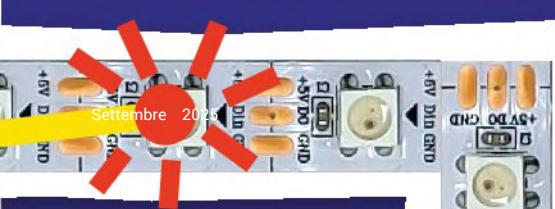
Abbiamo creato un semplice dispositivo di input e ora diamo un'occhiata al dispositivo di output più semplice: un diodo a emissione luminosa, o LED. I LED si accendono quando vengono sottoposti a tensione. Tuttavia, i LED sono sensibili a correnti troppo alte, che possono danneggiarli. È possibile calcolare il valore esatto della resistenza necessaria utilizzando formule basate su tensioni e limiti di corrente, ma per semplificare le cose, è sufficiente utilizzare una resistenza da 330Ω in serie al LED. La **figura 2** (sul retro) mostra come collegare una resistenza e un LED a un pin GPIO. A differenza delle resistenze di pull-up o pull-down, Pico non fornisce resistenze interne di limitazione della corrente per i LED, quindi è necessario aggiungerne una esterna.

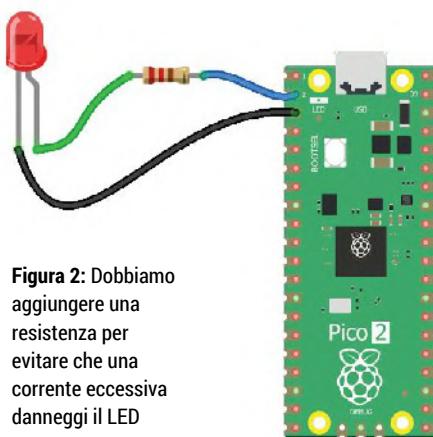
CircuitPython su Pico

Questo tutorial funziona su qualsiasi versione di Pico. Per installare CircuitPython su Pico, devi prima scaricare il file UF2 appropriato da rpimagic.co/circuitpython. Quindi, tieni premuto il pulsante BOOTSEL su Pico mentre lo colleghi al computer tramite USB. Continua a tenere premuto il pulsante fino a quando il Pico non è completamente connesso.

Una volta connesso, dovresti vedere apparire una nuova unità chiamata "RP2". Trascina e rilascia il file UF2 sull'unità RP2. Questo installerà CircuitPython su Pico. Dopo alcuni secondi, l'unità RP2 scomparirà e apparirà una nuova unità chiamata "CIRCUITPY".

Ora scarica l'editor di codice Mu da codewith.mu e installalo sul computer. Puoi usarlo per inserire, modificare e salvare il codice che creeremo in questo articolo. Fai clic sul pulsante Serial in Mu per visualizzare l'output di CircuitPython in esecuzione su Pico. Può essere particolarmente utile per individuare eventuali errori.





► Figura 2: Dobbiamo aggiungere una resistenza per evitare che una corrente eccessiva danneggi il LED

A differenza dei pulsanti, i LED hanno una polarità. Significa che devono essere collegati nel modo giusto. Di solito, c'è un lato piatto che indica il lato che deve essere collegato a terra (GND). Collegandolo nel modo sbagliato, non lo danneggerai, quindi puoi semplicemente inserirlo e, se non funziona, girarlo.

Ecco del codice CircuitPython per far lampeggiare un LED:

```
import time
import board
import digitalio
#Pico
#LED_PIN = board.GP1

#Raspberry Pi
LED_PIN = board.D17

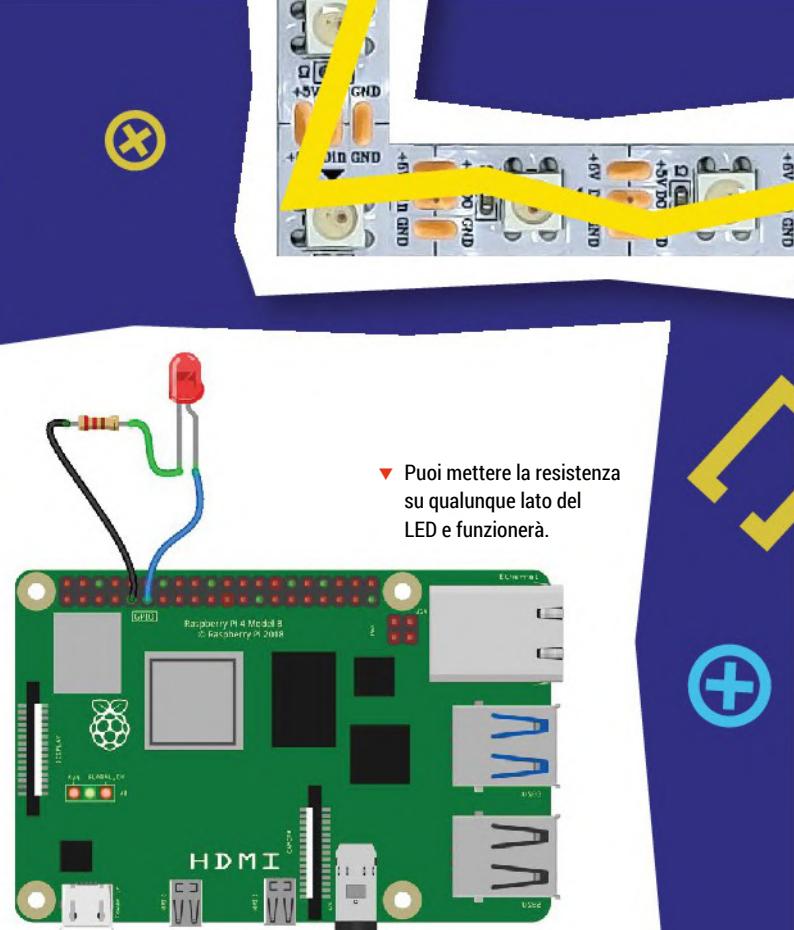
led = digitalio.DigitalInOut(LED_PIN)
led.direction = digitalio.Direction.OUTPUT

while True:
    led.value = True    # Accendi il LED
    time.sleep(0.5)    # Attendi 0.5 secondi
    led.value = False   # Spegni il LED
    time.sleep(0.5)    # Attendi 0.5 secondi
```

Che i LED facciano luce (ancora di più)

Sia i pulsanti che i LED funzionano con un singolo bit di dati. Significa che possono trovarsi in uno solo di due stati. Un pulsante è premuto o non premuto, e un LED è acceso o spento. Molti altri dispositivi sono più complessi e devono inviare e ricevere informazioni più dettagliate, ad esempio per controllare colori, posizioni o movimenti.

Utilizzeremo i LED WS2812B, noti anche come NeoPixel. Questi consentono di controllare molti LED RGB da un singolo pin GPIO utilizzando uno speciale protocollo di comunicazione. Il colore di ciascun LED può essere cambiato regolando la quantità di luce rossa, verde e blu che emette.



▼ Puoi mettere la resistenza su qualunque lato del LED e funzionerà.

Il cablaggio è semplice. Basta collegare il pin da 5 V della striscia LED all'uscita da 5 V del Pico, GND a GND e il pin di ingresso dati della striscia a un pin GPIO. Tieni presente che queste strisce LED sono direzionali. Significa che devi collegare il pin GPIO all'estremità corretta, solitamente contrassegnata con "DIN" o una freccia. Se colleghi l'estremità sbagliata, i LED non risponderanno.

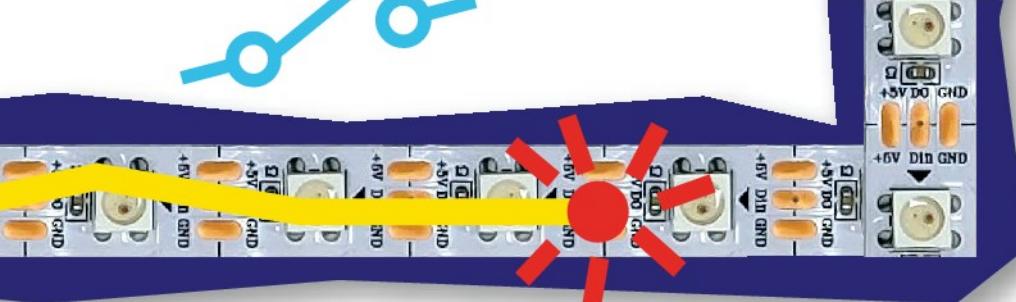
I LED WS2812B sono disponibili in molte forme diverse, ma quando li acquisti, in genere hanno un filo con un connettore a tre pin all'estremità. Di solito, il connettore si trova sul lato di ingresso dati, ma questo non è standard. Per testare il tutto, preferiamo utilizzare connettori pin-to-socket e possiamo inserire il pin nella presa della striscia LED, quindi collegare la presa a un GPIO su Pico o Raspberry Pi.

Poiché questi hanno un protocollo di comunicazione leggermente più complesso, dobbiamo aggiungere un modulo che lo implementi. Il processo per aggiungere moduli è leggermente diverso su ogni piattaforma.

Su Pico, vai su rpimagic.co/circpylibraries e scarica il bundle per la versione di CircuitPython che possiedi. Sarà un file zip. Estrailo e copia il file **neopixel.mpy** dalla cartella **lib** alla cartella **lib** sull'unità CIRCUITPY.

Su Raspberry Pi, assicurati di essere nell'ambiente virtuale creato durante la configurazione di Blinka, quindi esegui:

```
pip install adafruit-circuitpython-neopixel-spi
```



Ecco un po' di codice per testare il tutto, facendo lampeggiare alcuni colori.

L'inizializzazione dei LED WS2812B è leggermente diversa su ogni piattaforma, quindi ci sono due blocchi di codice: uno per Raspberry Pi e uno per Pico.

```
import board
import busio
import time

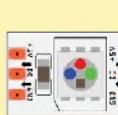
NUM_PIXELS = 10
BRIGHTNESS = 0.5

#Per Pico
#import neopixel
#PIXEL_PIN = board.GP0
#BUTTON_PIN = board.GP1
#pixels = neopixel.NeoPixel(PIXEL_PIN,
#    NUM_PIXELS, brightness=BRIGHTNESS,
#    auto_write=True)

#Per Raspberry Pi
#import neopixel_spi
#PIXEL_PIN = board.D10
#CLOCK_PIN = board.D11
#pixels = neopixel_spi.NeoPixel_SPI(busio.
#SPI(clock=CLOCK_PIN, MOSI=PIXEL_PIN), 10)

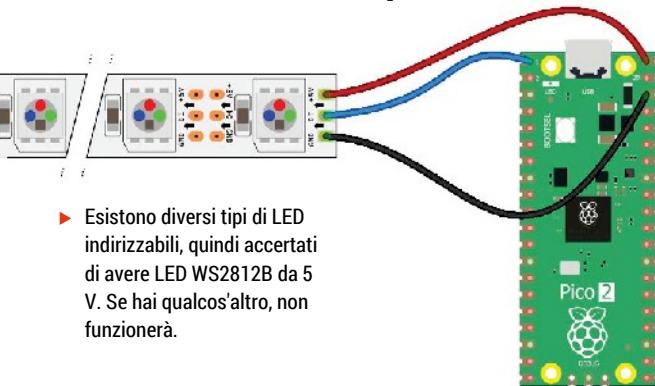
colors = [
    (255, 0, 0),      # Red
    (0, 255, 0),      # Green
    (0, 0, 255),      # Blue
    (100, 100, 100) # White
]

while True:
    for color in colors:
        pixels.fill(color)
        time.sleep(0.5)
```



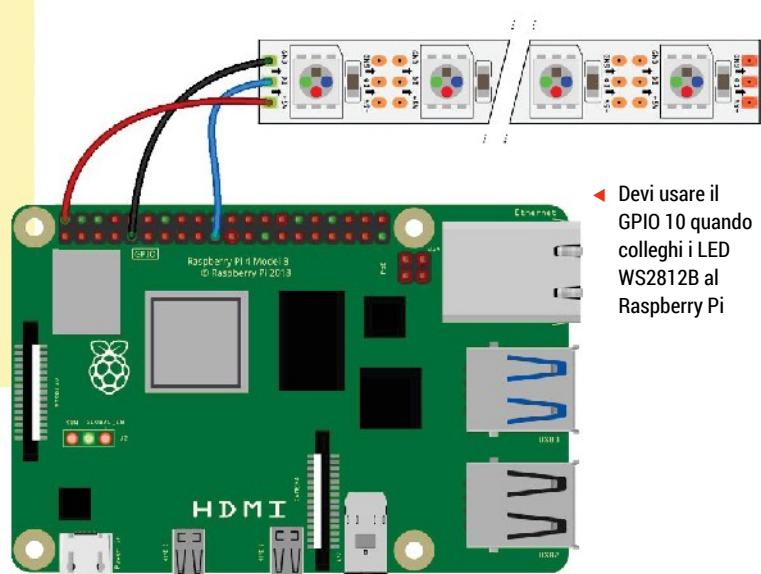
Su Pico, possiamo usare qualsiasi pin per i LED; tuttavia, su Raspberry Pi siamo più limitati. Dobbiamo usare il GPIO 10 per il pin pixel perché è collegato alla periferica SPI, ed è quella che usiamo per inviare i dati ai LED.

Su Raspberry Pi abbiamo usato due pin per i LED, ma ne lasceremo uno scollegato. Questo perché stiamo usando un hardware progettato per fare qualcosa di diverso per controllare i NeoPixel. Abbiamo maggiore flessibilità con il pin del pulsante, ma non possiamo usarne uno qualsiasi perché alcuni sono riservati a funzioni specifiche.

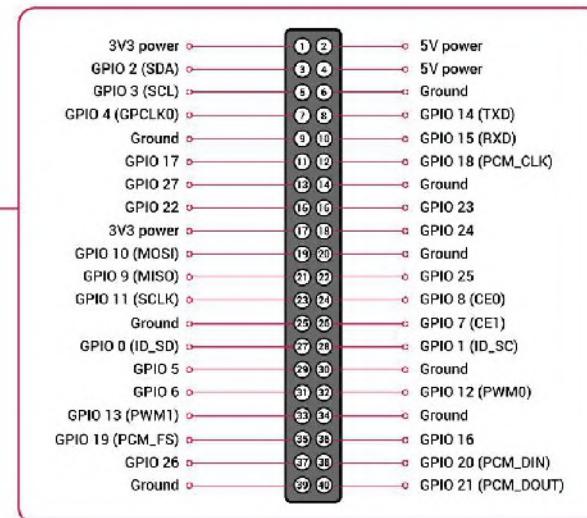
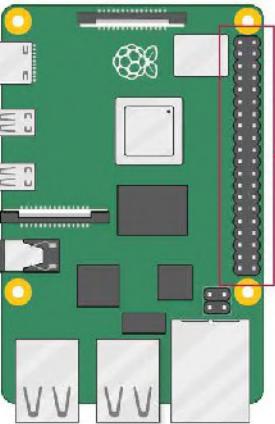


► Esistono diversi tipi di LED indirizzabili, quindi accertati di avere LED WS2812B da 5 V. Se hai qualcos'altro, non funzionerà.

I LED WS2812B esistono in molte forme diverse



► Devi usare il GPIO 10 quando colleghi i LED WS2812B al Raspberry Pi



► I pin del Pico sono etichettati sul lato inferiore, ma per il Raspberry Pi, è necessario fare riferimento a uno schema di pinout

Pronti, partenza, luce!

Ora aggiungiamo un pulsante e creiamo un gioco. Questo è un incrocio tra il gioco del dinosauro di Chrome e una serie di luci natalizie. L'idea è semplice. Le luci si illumineranno di verde e un pixel rosso apparirà casualmente e si muoverà verso la fine. Bisogna premere il pulsante in tempo per "saltare" il pixel rosso quando raggiunge la fine della striscia. Se ci si riesce, la striscia si illuminerà di bianco.

Il codice è:

```
import board
import busio
import time
import random
import digitalio

NUM_PIXELS = 10
BRIGHTNESS = 0.5

#Per Pico
#import neopixel
#PIXEL_PIN = board.GP0
#BUTTON_PIN = board.GP1
#pixels = neopixel.NeoPixel(PIXEL_PIN,
#NUM_PIXELS, brightness=BRIGHTNESS,
#auto_write=False)

#Per Raspberry Pi
#import neopixel_spi
#PIXEL_PIN = board.D10
#CLOCK_PIN = board.D11
```

```
#BUTTON_PIN = board.D11
pixels = neopixel_spi.NeoPixel_SPI(busio.SPI(clock=CLOCK_PIN, MOSI=PIXEL_PIN), 10)
```

```
DELAY = 0.3
```

```
GREEN = (0, 255, 0)
RED = (255, 0, 0)
WHITE = (100, 100, 100)
FLASH_DURATION = 0.6
```

```
# Impostazione pulsante
button = digitalio.DigitalInOut(BUTTON_PIN)
button.direction = digitalio.Direction.INPUT
button.pull = digitalio.Pull.UP
```

```
def flash_color(color, duration=FLASH_DURATION):
    pixels.fill(color)
    pixels.show()
    time.sleep(duration)
```

```
while True:
    red_pos = 0
```

```
# Sposta il pixel rosso in avanti
while red_pos < NUM_PIXELS:
    # Disegna la striscia
    for i in range(NUM_PIXELS):
        if i == red_pos:
            pixels[i] = RED
```

```

        else:
            pixels[i] = GREEN
        pixels.show()

        # Attendi il DELAY, controlla la
        # pressione del pulsante
        t0 = time.monotonic()
        pressed = False
        while time.monotonic() - t0 < DELAY:
            if not button.value():
                pressed = True
                break

        if pressed:
            if red_pos == NUM_PIXELS - 1:
                # SUCCESSO!
                flash_color(WHITE)
            else:
                # FALLITO!
                flash_color(RED)
                break

            red_pos += 1
        else:
            # Se raggiunge la fine e non hai
            # premuto, lampeggia in rosso per "mancato"
            flash_color(RED)

        # Piccola pausa prima di ripartire
        time.sleep(0.7)
    
```

Ordine dei colori

Il protocollo WS2812B descrive come inviare dati ai LED, ma non quale bit di dati corrisponde a quale colore, quindi potresti scoprire che i colori non sono corretti sui tuoi LED. Puoi configurare quale bit di dati corrisponde a quale colore, modificando il codice.

Qui termina il nostro vorticoso tour dell'elettronica su Pico e Raspberry Pi. Abbiamo saltato alcuni dettagli per aiutarti a iniziare subito, ma speriamo che ora tu conosca le basi per collegare altro hardware a questi due dispositivi.

Probabilmente hai anche un'idea più precisa delle differenze tra Pico e Raspberry Pi. Il primo offre opzioni di I/O più flessibili e una configurazione software molto più semplice, e il secondo ha diversi ordini di grandezza in più di potenza di elaborazione. Sebbene ci siano alcune sovrapposizioni tra i progetti che funzionano su ciascun dispositivo, si tratta in realtà di piattaforme molto diverse.

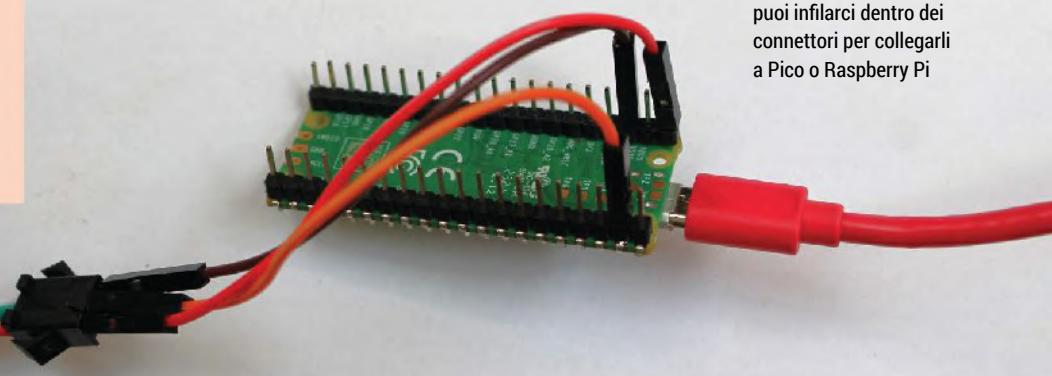
Se metti in pratica le competenze apprese in questo articolo, ci piacerebbe avere tue notizie. Ogni lunedì andiamo sui social media per trovare progetti interessanti, quindi se pubblicherai foto all'inizio della settimana, tagga #MakerMonday.

Problemi di tensione

I LED WS2812B richiedono che la tensione del segnale digitale sia almeno il 70% della tensione di alimentazione. Ciò significa che dovrebbe essere almeno 3,5 V. Pico e Raspberry Pi funzionano però a 3,3 V. Questo valore è così vicino che quasi sempre non dovete preoccuparvi di collegare la linea di ingresso dati al pin GPIO. Solo ogni tanto, otterrete una striscia che si comporta in modo anomalo.

Si può fare una di queste due cose. Si può incrociare le dita e sperare per il meglio (ed è quello che fa la maggior parte delle persone) oppure si può correggere il problema, il che in genere comporta l'uso di variatori di livello. Noi quasi sempre utilizziamo un segnale dati a 3,3 V e speriamo per il meglio, a meno che il progetto che stiamo creando debba funzionare in modo affidabile, soprattutto in condizioni avverse (come all'aperto sotto la pioggia).

- ▼ Se i tuoi LED sono dotati di un collegamento a tre pin, puoi infilarci dentro dei connettori per collegarli a Pico o Raspberry Pi



Elettronica semplice con GPIO Zero: In principio

Collega componenti elettronici al tuo Raspberry Pi e scrivi codice per interagire con il mondo reale



Maker

Phil King

Utente e appassionato di Raspberry Pi da molto tempo, Phil è uno scrittore e redattore freelance con focus sulla tecnologia.

philkingeditor.com

Raspberry Pi è un'ottima piattaforma per imparare l'informatica. Che si tratti di scrivere i propri programmi o di costruire un server multimediale, Raspberry Pi offre gli strumenti, le risorse e il supporto della community per aiutarti a imparare a costruire ciò che desideri. Raspberry Pi è anche ottimo per il physical computing, ovvero la programmazione e l'interazione con il mondo reale attraverso l'elettronica. Come suggerisce il nome, il physical computing consiste nel controllare gli oggetti nel mondo fisico tramite i programmi: utilizzando l'hardware insieme al software. Quando imposti il programma sulla lavatrice, modifichi la temperatura sul termostato programmabile o premi un pulsante al semaforo per attraversare la strada in sicurezza, stai utilizzando il physical computing.

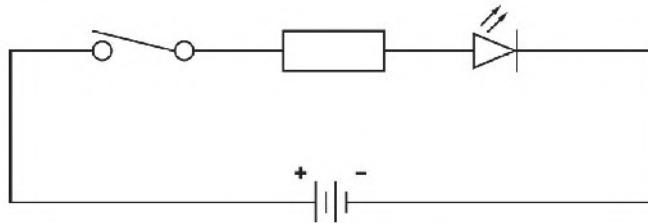
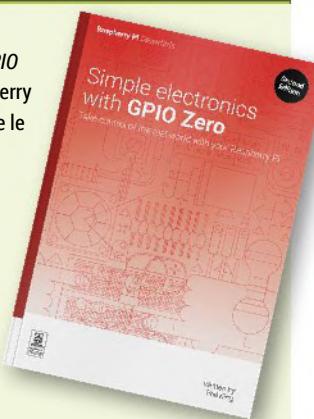
I circuiti elettronici sono la parte fisica di un progetto di physical computing. Collegherai questi circuiti al Raspberry Pi, che, insieme al codice che scriverai, costituisce la parte informatica del progetto. Questi circuiti possono essere semplici o molto complessi e sono costituiti da componenti elettronici come LED, cicalini, pulsanti, resistenze, condensatori e persino chip di circuiti integrati (IC).

Nella sua forma più semplice, un circuito elettronico consente di instradare l'elettricità a determinati componenti in un ordine specifico, dal polo positivo di un circuito al polo di massa (o zero volt). Pensa a una luce in casa tua: l'elettricità la attraversa, quindi si accende. Puoi aggiungere un interruttore che interrompe il circuito, in modo che si accenda solo quando lo premi. Ora è un circuito elettronico interattivo.

Elettronica semplice con GPIO Zero

Questo articolo è un estratto dal libro di Raspberry Pi "Simple electronics with GPIO Zero". Aggiornato per i dispositivi Raspberry Pi più recenti, questo libro contiene tutte le informazioni necessarie per iniziare a creare progetti elettronici utilizzando i pin GPIO di Raspberry Pi. Codificati in Python con la libreria GPIO Zero, i progetti includono luci a LED, un allarme con sensore di movimento, un telemetro, un filo a scatto alimentato a laser e un robot Raspberry Pi.

rpimag.co/gpiozerobook



Leggere gli schemi circuituali

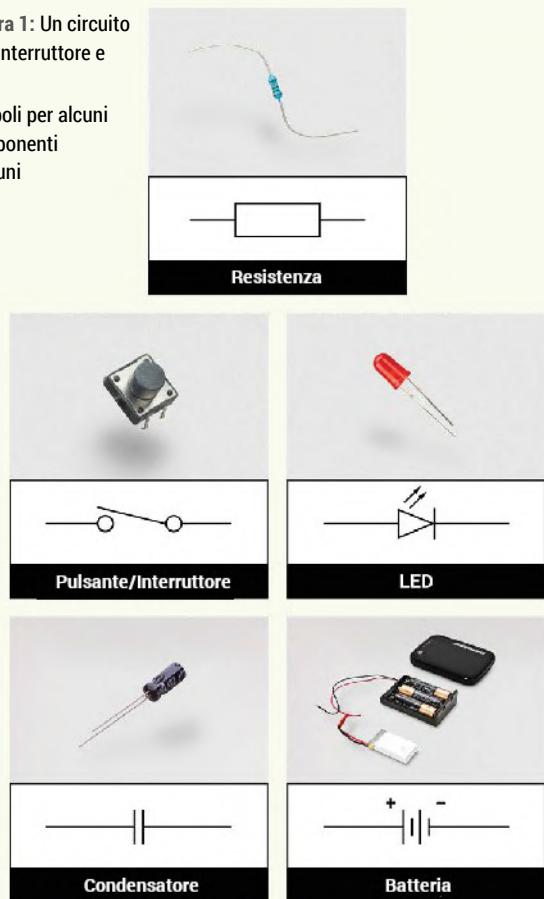
Costruire un circuito può essere facile se sai cosa stai facendo, ma se stai realizzando un nuovo circuito o sei alle prime armi con l'elettronica in generale, molto probabilmente dovrai fare riferimento a uno schema elettrico. Questo è un modo comune per rappresentare un circuito, e questi schemi sono molto più facili da leggere e comprendere rispetto a una foto di un circuito. Tuttavia, i componenti sono schematizzati con simboli che dovrete imparare o cercare.

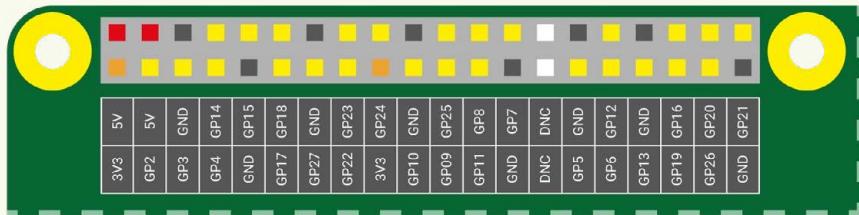
Se stai realizzando un nuovo circuito, dovrà fare riferimento a uno schema elettrico

La **Figura 1** è un esempio di circuito luminoso. Qui abbiamo una fonte di alimentazione (una batteria in questo circuito), un interruttore, una resistenza e un LED. Le linee rappresentano il modo in cui i circuiti sono collegati, tramite filo o altri mezzi. Alcuni componenti possono essere inseriti con qualsiasi orientamento, come la resistenza o l'interruttore. Tuttavia, altri hanno una polarità specifica, come il LED. LED sta per diodo a emissione luminosa (Light-Emitting Diode) e i diodi lasciano fluire liberamente l'elettricità in una sola direzione; fortunatamente, i LED hanno dei marcatori come una gamba più lunga o un bordo piatto per indicare quale lato è positivo, rendendoli più facili da collegare.

▲ Figura 1: Un circuito con interruttore e LED

► Simboli per alcuni componenti comuni





► Figura 2: Piedinatura del GPIO Raspberry Pi

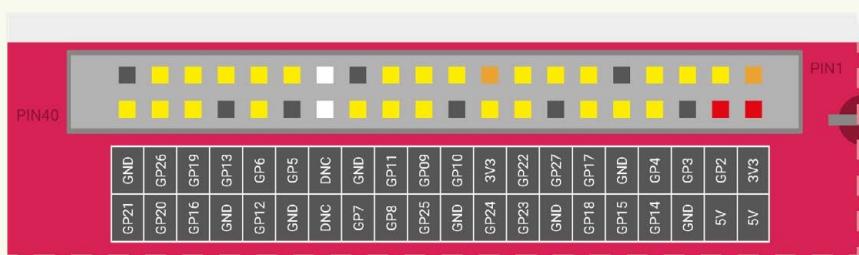


Figura 3:
Piedinatura
del GPIO
Raspberry Pi
400 e 500

Introduzione al connettore GPIO

Nella parte superiore della scheda del Raspberry Pi, o sul retro di un Raspberry Pi 400 o 500 o 500+, troverete due file di pin metallici. Questo è il connettore GPIO (general-purpose input/output) e serve per collegare componenti elettronici al Raspberry Pi. Come suggerisce il nome, questi pin possono essere utilizzati sia per l'input che per l'output.

Il connettore GPIO di Raspberry Pi è composto da 40 pin, come mostrato nella **Figura 2**. Alcuni pin sono disponibili per l'uso nei tuoi progetti di physical computing, altri forniscono alimentazione e altri pin vengono utilizzati per comunicare con hardware aggiuntivo.

I computer con tastiera compatta Raspberry Pi 400, 500 e 500+ hanno lo stesso connettore GPIO con tutti gli stessi pin, ma è capovolto rispetto ad altri modelli di Raspberry Pi. La **Figura 3** presuppone che stiate guardando il connettore GPIO dal retro del Raspberry Pi 400 o 500. Controllate sempre il cablaggio quando collegate qualcosa al connettore GPIO su uno dei modelli di computer compatti: è facile dimenticarsene, nonostante le etichette Pin 40 e Pin 1 sul case!

Anche il Raspberry Pi Zero 2 W ha un connettore GPIO, ma non necessariamente ha i pin del connettore saldati. Se desideri eseguire operazioni di physical computing con il Raspberry Pi Zero 2 W o un altro modello della famiglia Raspberry Pi Zero, devi saldare i pin in posizione utilizzando un saldatore. Se per ora ti sembra un po' azzardato, rivolgiti a un rivenditore Raspberry Pi autorizzato per un Raspberry Pi Zero 2 WH che ha i pin del connettore già saldati in posizione.

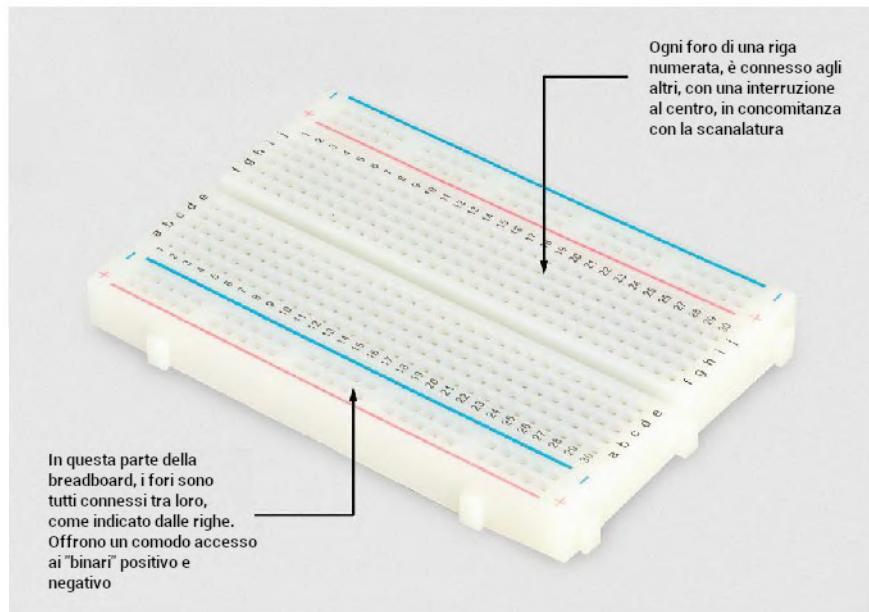
Raspberry Pi e i circuiti elettronici

Coinvolgere un computer Raspberry Pi in un circuito è piuttosto semplice. Nella sua forma più elementare, può fornire alimentazione a un circuito, così come la massa (abbreviata in GND) attraverso i pin GPIO. Alcuni pin sono sempre alimentati, principalmente a 3,3 V, e diversi pin offrono una connessione a massa. La maggior parte dei pin può essere programmata per creare o riconoscere un segnale ALTO o BASSO; nel caso di Raspberry Pi, un segnale ALTO è a 3,3 V e un segnale BASSO è a massa o 0 V.

In un circuito LED, è possibile collegare un LED a un pin da 3,3 V e a un pin di massa e si accenderà, ma sarà necessaria una resistenza di basso valore (330 Ω è un'ottima scelta) da qualche parte per evitare di bruciare il LED. Se invece colleghi il polo positivo del LED a un pin GPIO programmabile, puoi accenderlo solo eseguendo un codice che porti quel pin ad un valore ALTO. Nella prossima parte di questa serie, parleremo del controllo dei LED.

La massa

A volte la massa viene indicata come negativa, in particolare nelle descrizioni dei terminali positivo e negativo di una batteria, ma anche come simbolo meno (-) su una breadboard e su alcuni componenti. Nel tipo di circuiti che vedrete in questa serie di tutorial, lavorerete con 5 volt, 3,3 volt e 0 volt (massa).



Collegare un circuito a un computer Raspberry Pi è semplice. Per creare i circuiti presenti nelle guide di questa serie, utilizziamo delle breadboard per prototipazione senza saldatura, come mostrato in **Figura 4**. Queste consentono di inserire componenti e fili per collegarli tra loro, senza doverli fissare in modo permanente. Grazie a questo, è possibile modificare i circuiti e riutilizzare completamente i componenti.

Usare GPIO Zero

Una volta che tutti i componenti sono collegati al tuo Raspberry Pi, devi essere in grado di controllarli. Raspberry Pi è configurato per consentirti di programmarlo con il linguaggio Python. GPIO Zero semplifica la programmazione dei componenti in Python. È preinstallato nell'ultima immagine desktop del sistema operativo Raspberry Pi. Se non lo hai, tuttavia, puoi installare GPIO Zero manualmente: dopo aver eseguito un aggiornamento dell'elenco dei pacchetti immettendo **`sudo apt update`** in un terminale, esegui **`sudo apt install python3-gpiozero`**.

GPIO Zero è stato creato per semplificare il processo di physical computing, aiutando i nuovi programmati a imparare. È una libreria Python che si basa sulle librerie GPIO RPi.GPIO e pigpio esistenti. Tuttavia, mentre queste librerie forniscono un'interfaccia ai pin GPIO stessi, GPIO Zero si trova sopra di essi e fornisce un modo per interfacciarsi con i dispositivi che colleghi a quei pin.

Questo semplifica il concetto di physical computing. Consideriamo il collegamento di un semplice pulsante al GPIO 4 e ai pin di massa. Per reagire a questo pulsante, dobbiamo sapere che il pin deve essere configurato con una resistenza di pull-up e che lo stato del pin quando il pulsante viene premuto sarà BASSO.

▲ Figura 4: Una breadboard

Ecco come apparirebbe nella classica libreria RPi.GPIO:

```
from RPi import GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(4, GPIO.IN, GPIO.PUD_UP)
GPIO.wait_for_edge(4, GPIO.FALLING)
print("Pulsante premuto")
```

Per i principianti assoluti, ci sono parecchie cose da fare, che ostacolano la sperimentazione e persino l'apprendimento della semplice logica richiesta. Ecco il codice equivalente in GPIO Zero:

```
from gpiozero import Button

btn = Button(4)
btn.wait_for_press()
print("Pulsante premuto")
```

Il *boilerplate*, il codice di configurazione che bisogna scrivere senza necessariamente comprenderne il funzionamento interno, è ridotto al minimo indispensabile. Il nome "GPIO Zero" deriva da questa filosofia del "zero boilerplate", adottata per la prima volta dalla libreria Pygame Zero di Daniel Pope.

La logica è semplice, senza alcuna curiosa inversione del valore di input.

Quindi, ora che hai imparato a conoscere GPIO Zero e come semplifica molto la programmazione, è il momento di iniziare a fare un po' di physical computing. La prossima volta ti mostreremo come collegare alcuni LED su una breadboard e controllarli utilizzando la classe LED di GPIO Zero.



IMMERGITI NELLA TECNOLOGIA **MARINA**

Scopri i progetti marini con Raspberry Pi
e costruisci un robot sottomarino.

Di Lucy Hattersley



L'oceano è un argomento vasto: misterioso per gli esseri umani, sia fisicamente che tecnicamente. È sorprendente quanto sappiamo più dello spazio rispetto al nostro fondale oceanico. Abbiamo mappato la Luna, Marte e Venere più dei nostri oceani.

L'oceano è fisicamente impegnativo, con alta pressione, oscurità totale e freddo estremo

Non è una sorpresa trovare dispositivi Raspberry Pi che funzionano sotto l'oceano

aggiungono considerazioni meccaniche ai soliti requisiti di costruzione del progetto. L'acqua salata è ostile per le attrezzature e opaca alla luce. Le onde radio non viaggiano bene sott'acqua, soprattutto quelle a frequenze più alte come Bluetooth, Wi-Fi o comunicazioni radio standard. Gli oceani sono enormi!

Un hardware Raspberry Pi ben progettato è in prima linea in questa sfida. Non è una sorpresa trovare dispositivi Raspberry Pi che funzionano sott'acqua nell'oceano, misurando di tutto, dalle popolazioni marine, alla mappatura dei fondali oceanici.

In questo speciale, daremo un'occhiata ai professionisti. Poi Jo Hinchliffe ci illustrerà TOUV (un piccolo veicolo sottomarino open source). Dall'offrire ispirazione alla vostra guida pratica introduttiva alla robotica subacquea, questo articolo vi farà immergere nei dettagli.

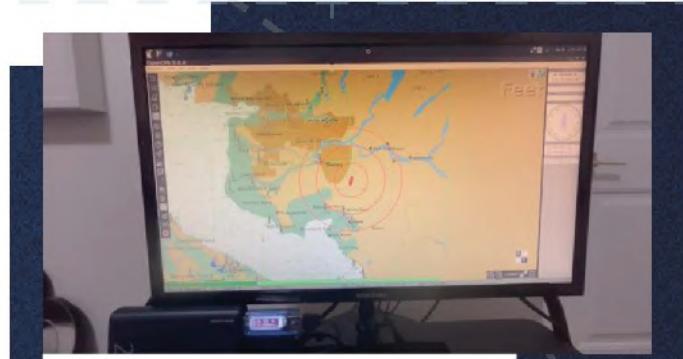
Tuffiamoci...



Nemo-Pi

Save Nemo crea stazioni meteorologiche subacquee per il monitoraggio delle barriere coralline. Queste boe alimentate a energia solare misurano le condizioni dell'acqua, come temperatura, pH, visibilità, forza della corrente e concentrazioni chimiche. I sistemi caricano dati in tempo reale su server pubblici, aiutando subacquei e ricercatori a studiare le condizioni della barriera corallina.

○ nemopi.com



Dave the MMP

Dave sta riadattando una barca a vela Cal 2-30 utilizzando un sistema di navigazione marittima basato su Raspberry Pi 5. Questo fornisce una potente alternativa ai costosi chartplotter commerciali, che utilizzano il software OpenCPN (opencpn.org) con piena integrazione NMEA (National Marine Electronics Association). Il suo canale YouTube mostra una completa integrazione del Raspberry Pi 5 con la sua barca.

○ rpimag.co/davethemmp

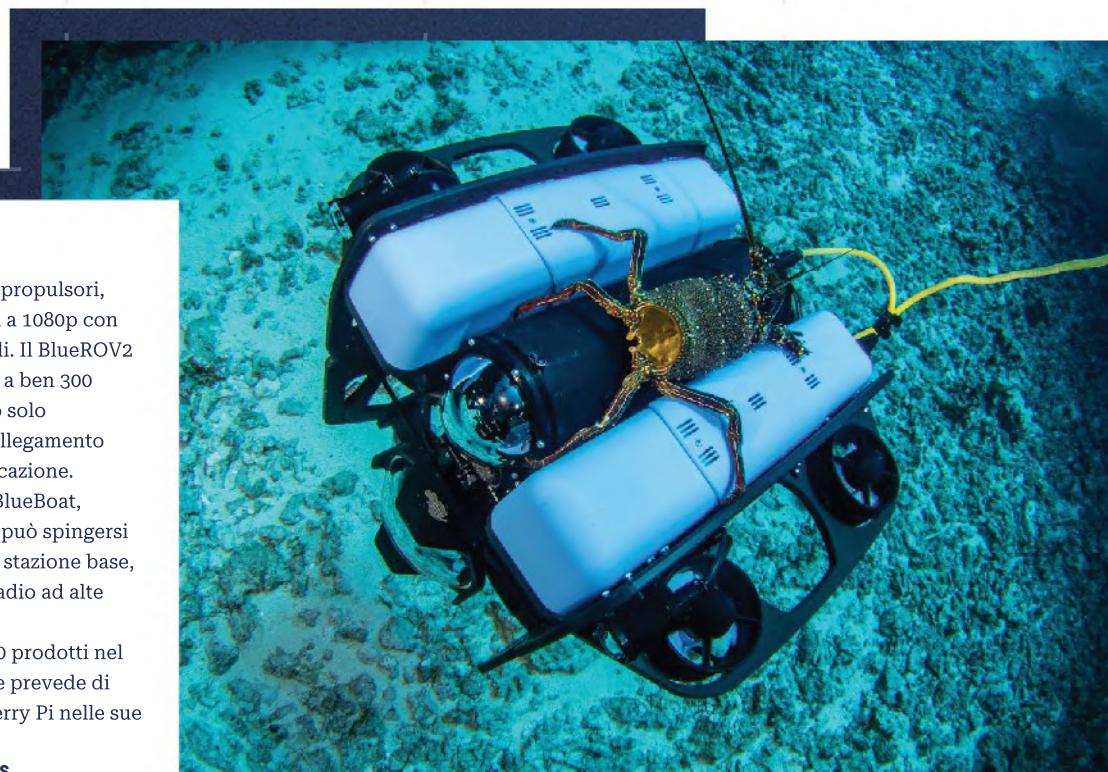


Arribada Initiative

Questo progetto utilizza Raspberry Pi Zero per seguire il viaggio delle tartarughe marine verdi. L'elevato costo delle attrezzature fotografiche rappresentava un enorme ostacolo per gli ambientalisti che monitoravano le popolazioni e i comportamenti delle specie. I kit Arribada, economici e robusti, basati su Raspberry Pi si sono rivelati una svolta. Il kit per tartarughe comprende un Raspberry Pi Zero W dotato di un modulo fotocamera Raspberry Pi, una scheda di gestione dell'alimentazione PiRA, due celle agli ioni di litio e un involucro. Le riprese riprese dal dorso di queste magnifiche creature sono incredibili.

○ arribada.org



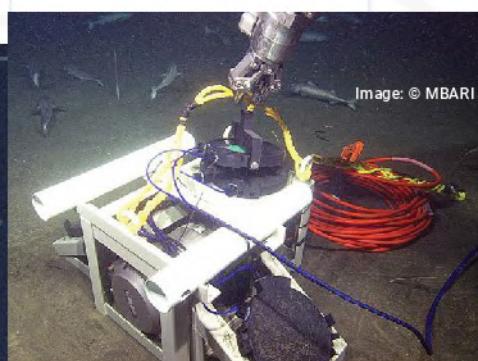


Blue Robotics

Il BlueROV2, un velivolo a sei propulsori, cattura filmati video in diretta a 1080p con una latenza di 200 millisecondi. Il BlueROV2 può operare a profondità fino a ben 300 metri, e oltre questo è limitato solo principalmente dal cavo di collegamento fisico utilizzato per la comunicazione. L'imbarcazione di superficie BlueBoat, invece, è collegata via radio e può spingersi fino a un chilometro dalla sua stazione base, o molto oltre se si utilizzano radio ad alte prestazioni o 4G.

Blue Robotics offre oltre 350 prodotti nel campo della robotica marina e prevede di continuare a utilizzare Raspberry Pi nelle sue produzioni professionali.

- rpimag.co/bluerobotics



Raspberry Shake

Siamo grandi fan di Raspberry Shake. Normalmente questa apparecchiatura viene utilizzata per misurare l'attività sismica sulla superficie del globo, ma nel 2020 il Monterey Bay Aquarium Research Institute ne ha posizionato una a 900 metri sotto il livello del mare. L'unità finale è costituita da un piccolo telaio in fibra di vetro con un alloggiamento a pressione in titanio che contiene Raspberry Shake. Ai lati e sulla parte superiore del telaio si trovano delle "palette" in plastica nera che utilizzano le onde radio per trasmettere energia e dati in modalità wireless sott'acqua.

- rpimag.co/shakeocean



OpenPlotter

Sulla superficie dell'oceano, Raspberry Pi è diventato la base per i sistemi di elaborazione marini. Un tempo dominato da costosi sistemi closed-source, il mondo dell'elettronica marina è stato stravolto da OpenPlotter. Progettato su misura per Raspberry Pi, questo sistema operativo basato su Linux trasforma il computer a scheda singola in un completo hub di navigazione marina. Riunisce info su meteo, tracciamento cartografico, coordinate GPS e bersagli AIS (sistema di identificazione automatica) nelle vicinanze. Fornisce persino un hotspot wireless a bordo.

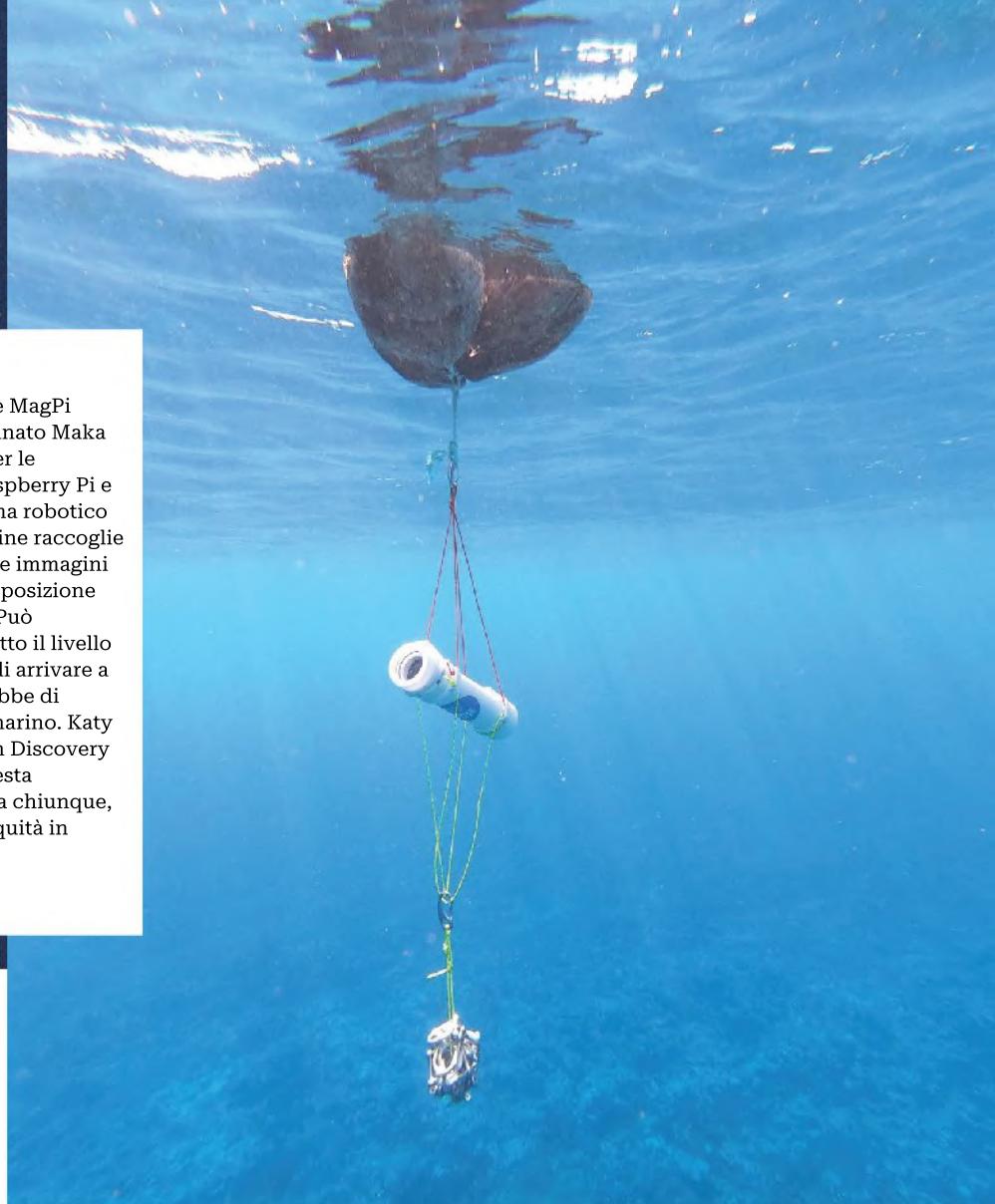
- openmarine.net/openplotter

Maka Niu

Nel numero 125 della rivista The MagPi (rpimag.co/125) abbiamo esaminato Maka Niu, un esploratore modulare per le profondità marine basato su Raspberry Pi e Camera Module 2. Questo sistema robotico di imaging delle profondità marine raccoglie diversi tipi di dati, tra cui video e immagini fisse, profondità, temperatura e posizione GPS quando è fuori dall'acqua. Può raggiungere fino a 1500 metri sotto il livello del mare: il team ha progettato di arrivare a 6000 metri, il che gli permetterebbe di raggiungere il 99% del fondale marino. Katy Croff Bell, presidente dell'Ocean Discovery League, afferma: "Rendendo questa tecnologia aperta e disponibile a chiunque, speriamo di portare maggiore equità in questo importante settore".

o rpimag.co/makaniu

Può arrivare fino a 1500 metri sotto il livello del mare



Monitoraggio popolazione Pinguini

L'iniziativa Arribada utilizza anche Raspberry Pi nell'ambito di Penguin Watch, un progetto in corso per monitorare i pinguini che sopravvivono a molteplici inverni antartici rigidi. Questo progetto è stato trattato insieme al monitoraggio delle tartarughe nel numero 123 della rivista The MagPi (bit.ly/MagPi123It). Il direttore dell'iniziativa Arribada, Alasdair Davies, spiega che il Raspberry Pi è sempre stato uno degli strumenti che utilizza "perché è così accessibile e alla portata di chiunque. Si collabora sempre con un ricercatore o un membro della comunità locale che lavora con una ONG e ha una sfida specifica. Veniamo chiamati a risolverla con la tecnologia. Dicono sempre che deve essere conveniente, riparabile e accessibile".

o rpimag.co/arribada





ROV sottomarino

Questo sottomarino telecomandato fatto in casa si integra con un'app Unity per un dispositivo Android che viene utilizzato per registrare video e pilotare il sottomarino.

Utilizza un Raspberry Pi come unità di controllo centrale, un Modulo Fotocamera 2 per la registrazione video, sensori e driver, come descritto nello schema (linkato di seguito). Un veicolo di prova è stato costruito con un tubo di scarico in PVC-U con parti in PLA stampate in 3D personalizzate e una cupola in acrilico, il tutto sigillato con colla epoxidica.

È un ottimo esempio di hacking fatto in casa e di cosa si può fare per iniziare a mettere un Raspberry Pi sott'acqua.

- rpimag.co/underwaterrov

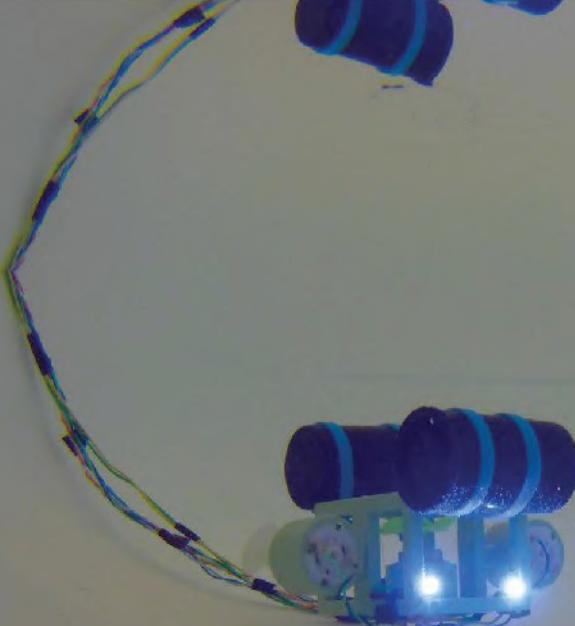
Registrazione autonoma per l'ecologia marina

Le specie di cetacei, tra cui balene, delfini e focine, sono considerate, in tutto il mondo, indicatori della salute degli ecosistemi marini. Sebbene alcune siano note per essere in pericolo, la mancanza di dati rende impossibile stimare la dimensione della popolazione e lo stato di conservazione di molte specie. Questi animali sono vulnerabili agli effetti delle attività umane e al rumore che causano.

Un team dell'Università di San Paolo in Brasile ha pubblicato un articolo sul registratore autonomo flessibile ed economico che hanno costruito, basato su un Raspberry Pi. L'idrofono – un microfono subacqueo – è protetto da una gabbia in acciaio inossidabile. Questa struttura resiste con successo a pressioni fino a 10 bar, equivalenti a quelle sperimentate a quasi 100 metri sott'acqua, nei test in camera a pressione.

- rpimag.co/underwaterrecording





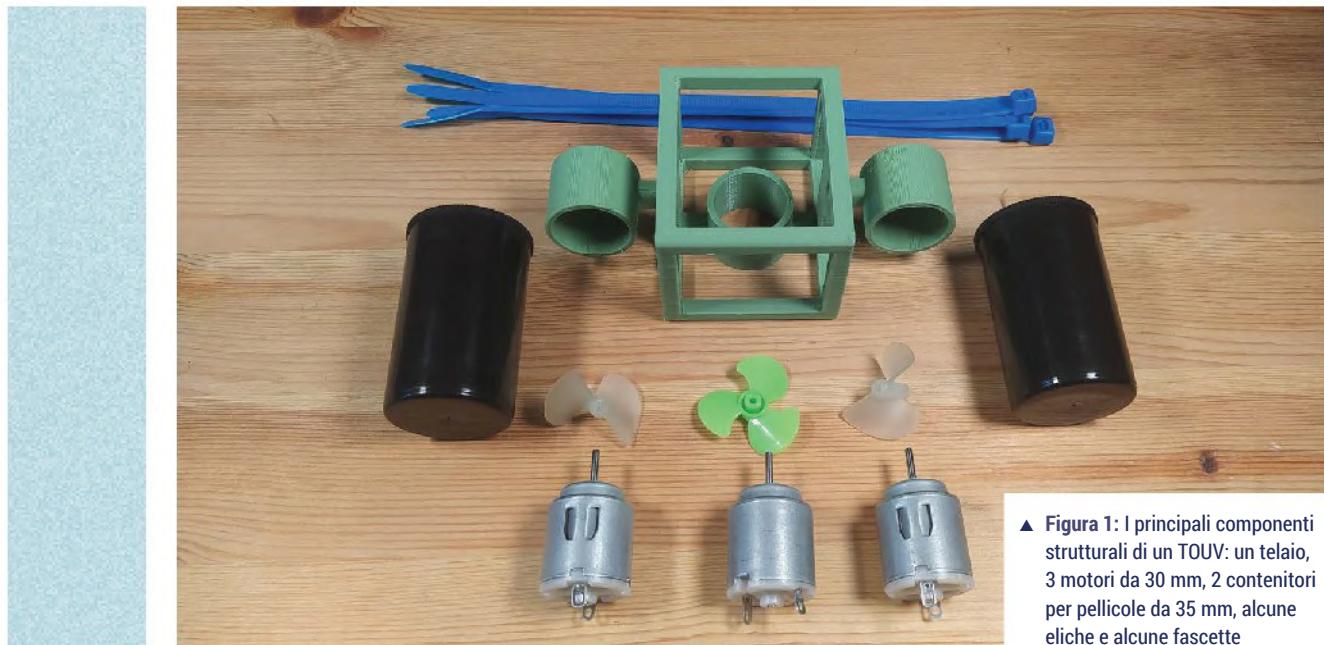
COSTRUisci UN PICCOLO VEICOLO SOTTOMARINO OPEN-SOURCE

Con un po' di stampa 3D, dei motori e un Raspberry Pi Pico, puoi iniziare a costruire un esploratore sottomarino!

Di Jo Hinchliffe

▲ Il piccolo veicolo sottomarino open source (TOUV) messo alla prova in acqua dolce (l'acqua salata corrode i componenti)





▲ Figura 1: I principali componenti strutturali di un TOUV: un telaio, 3 motori da 30 mm, 2 contenitori per pellicole da 35 mm, alcune eliche e alcune fascette

Circa dieci anni fa, siamo stati ispirati da un workshop al Liverpool Makefest in cui un'organizzazione chiamata Dark Water Foundation realizzava ogni sorta di dispositivi per il monitoraggio dell'acqua.

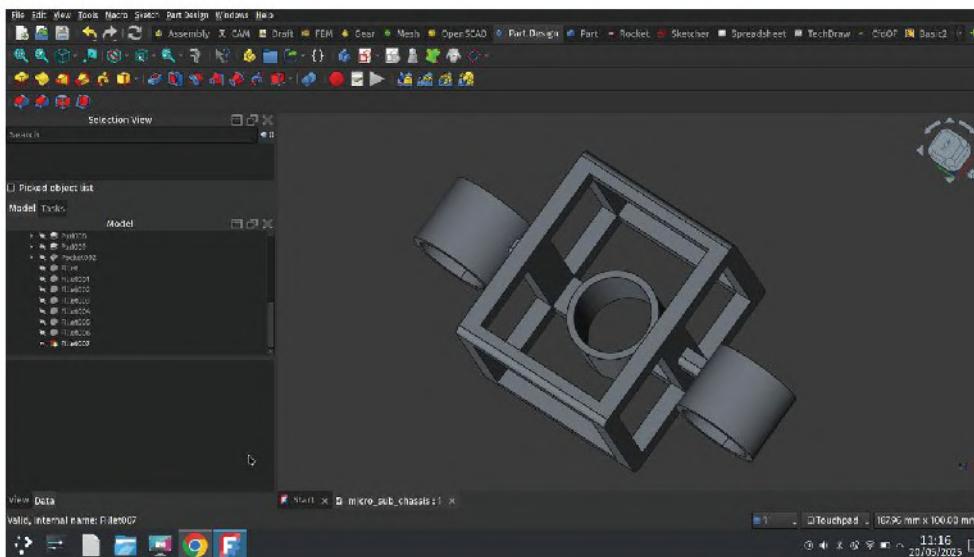
La Dark Water Foundation ha chiesto ai partecipanti di costruire rover subacquei in miniatura, grandi quanto un palmo di mano, utilizzando Lego, contenitori di pellicola da 35 mm e alcuni piccoli motori a corrente continua a spazzole. "Motori a corrente continua?", vi sento gridare. Ebbene sì, la rivelazione nel vedere questo workshop è stata che i piccoli motori a corrente continua da 3-6 V funzionano molto bene in acqua dolce senza bisogno di alcun tentativo di impermeabilizzarli. L'avvertenza è che probabilmente ne si sta riducendo un po' la durata e bisogna fare un piccolo lavoro per asciugarli dopo ogni missione.

Molti dei link originali a questo workshop ora sono andati perduti; quindi, con assoluto rispetto per il progetto originale, abbiamo deciso di fare un tentativo più moderno, influenzato da quel progetto originale. Esiste ancora una pagina Instructables con alcune immagini ed elenchi di componenti, ma purtroppo i link ai componenti dalla pagina al momento non funzionano, quindi abbiamo fatto alcune ipotesi plausibili e sperimentato per trovare una soluzione funzionante.

Per i nostri test più seri, ci siamo spostati su una vasca idromassaggio gonfiabile usata

Il progetto originale si basava su un cubo da 60-75 mm per il telaio, realizzato con i Lego. Abbiamo deciso di stampare in 3D un telaio di dimensioni simili utilizzando il pacchetto gratuito e open source FreeCAD per realizzare le nostre idee. Il progetto originale utilizzava motori CC a spazzole da 30 mm e abbiamo scoperto che all'epoca erano classificati per 5400 giri/min. Non siamo riusciti a trovare quella specifica esatta, ma ne abbiamo trovati alcuni abbastanza vicini indicati come 3-6 V e 6500 giri/min. Questi sono perfetti perché intendiamo realizzare un controller moderatamente più avanzato per il nostro progetto: possiamo utilizzare i driver dei motori per controllarne la velocità.

Una volta arrivati i motori (Figura 1), abbiamo potuto prenderne le misure e iniziare il nostro lavoro CAD. In FreeCAD, abbiamo utilizzato il workbench Part Design per creare il telaio,



◀ **Figura 2:**
Progettare il telaio
in FreeCAD è stato
un compito
piuttosto semplice

costruendolo tramite aggiunta di sketch alle superfici ed estrusione degli stessi (**Figura 2**). La struttura principale del nostro telaio è un cuboide, con tre propulsori: due montati ai lati del telaio per la propulsione in avanti e le manovre (e potenzialmente anche per la retromarcia, se necessario), e uno all'interno del telaio, posizionato sull'asse verticale per controllare la posizione verticale. I due supporti orizzontali dei propulsori sono leggermente decentrati rispetto al centro, nel tentativo di distribuire meglio il peso.

Una parte del progetto ci ha portato a riflettere sull'approccio da adottare per la galleggiabilità. Ci sono due, forse tre, opzioni. Possiamo puntare alla galleggiabilità neutra assoluta, in cui il veicolo non sale o scende in acqua finché non si attiva un motore; alla galleggiabilità positiva, in cui il veicolo tende sempre a galleggiare e il propulsore sull'asse Z lo spinge verso il basso; oppure alla galleggiabilità negativa, in cui il veicolo tende ad affondare finché il propulsore sull'asse Z non lo solleva. Ovviamente, la galleggiabilità neutra sarebbe ideale, ma è difficile da ottenere, soprattutto con i cavi di collegamento che modificano la massa del sistema durante il movimento; inoltre, per ottenerla, sarebbe necessario poter invertire la rotazione del propulsore sull'asse Z.

La galleggiabilità negativa è relativamente facile da ottenere. Abbiamo intenzione di aggiungere due contenitori per pellicole da 35 mm (si trovano ancora online) e possiamo semplicemente riempirli d'acqua per far affondere il veicolo. La galleggiabilità neutra presenta alcuni vantaggi e uno svantaggio. Un vantaggio è che, se si riempiono completamente i contenitori d'acqua, essi risultano uguali e bilanciati – se invece sono riempiti solo in parte, può essere difficile evitare che l'acqua si muova all'interno, alterando l'orientamento del veicolo.

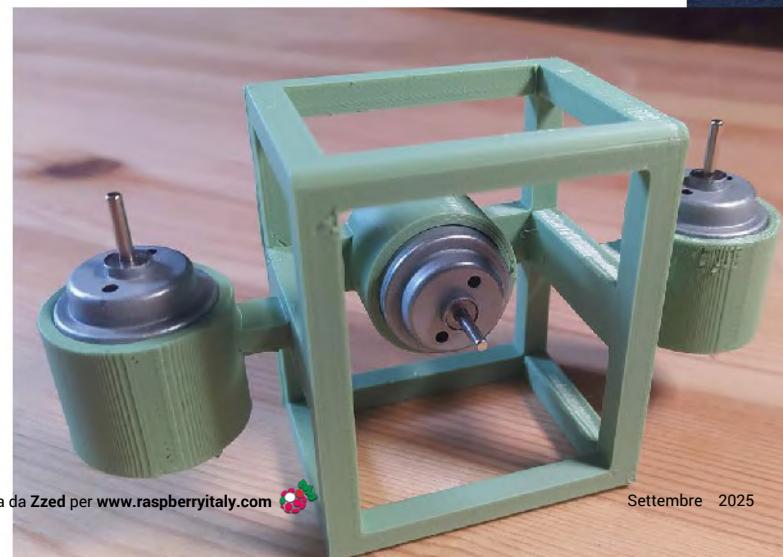
Lo svantaggio della spinta negativa è che consuma un po' più di potenza, poiché si utilizza molto di più il propulsore. E se per qualche motivo la potenza o i motori si guastano, il veicolo affonderà, il che non è un grosso problema a questa scala, dato che è possibile estrarlo con il cavo di sicurezza.

Dopo aver stampato un telaio di prova in filamento PETG, l'assemblaggio del veicolo è piuttosto semplice. Con una stampante 3D ragionevolmente ben calibrata, i supporti del propulsore sono abbastanza stretti da consentire l'inserimento a pressione dei motori (**Figura 3**). Abbiamo utilizzato eliche in nylon a due pale da 26 mm per i propulsori orizzontali e un'elica a tre pale da 30 mm più grande per l'asse Z, assicurandoci di ordinarne una con i fori corretti da 2 mm per il montaggio sui nostri motori CC a spazzole da 30 mm da 3-6 V.

È importante cercare di trovare un equilibrio tra fili lunghi, sottili e flessibili per creare le linee elettriche per i motori. Abbiamo trovato un vecchio cavo patch di rete da tre metri che



▶ **Figura 3:**
I motori sono
montati a
frizione nel
telaio, il che
significa che
è possibile
regolarne un
po' la
posizione,
e che sono
facili da
sostituire, se
necessario

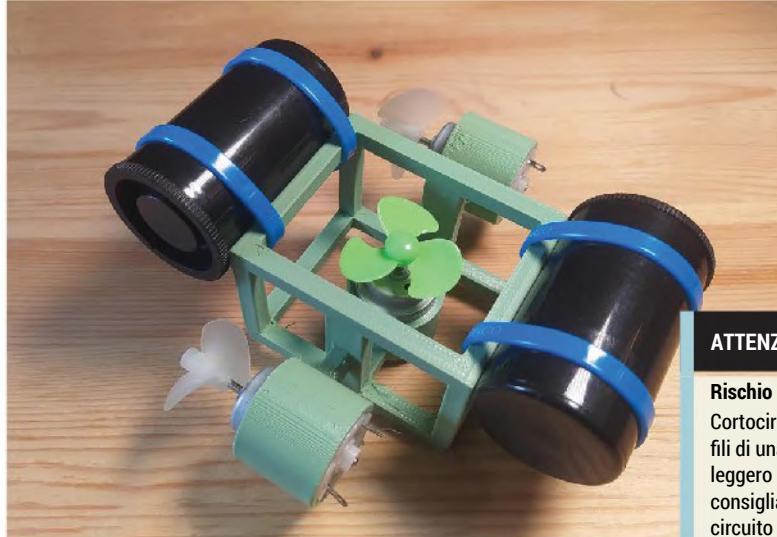


abbiamo aperto per ottenere lunghi spezzoni di cavo adatto. se voleste spendere piuttosto che trovare qualcosa tra i vostri cavi di scarto, probabilmente trovereste soluzioni più sottili e flessibili. Dato che il telaio si appoggia abbastanza bene su un piano di lavoro, abbiamo inserito i motori nel telaio e poi saldato i cavi, usando il telaio come aiuto.

Aggiungere i serbatoi di galleggiamento è semplice come avvolgere due fascette attorno al telaio e ai contenitori per pellicole da 35 mm (**Figura 4**). Potete sistemare leggermente la posizione dei serbatoi dopo averli montati. Se li posizionate più verso le estremità, quando il veicolo sta galleggiando, potrete scoprire che il propulsore dell'asse Z si trova leggermente fuori dall'acqua. Sollevando i serbatoi sopra il veicolo, l'elica dell'asse Z è sempre coperta d'acqua.

Una volta posizionati e cablati i motori e montati i serbatoi di galleggiamento, è il momento di testarli. Abbiamo eseguito i nostri primi test semplicemente tenendo per un attimo i cavi del motore su una batteria 18650 per capire la polarità. Una volta fatto, non abbiamo resistito a giocarci in una bacina piena d'acqua. Si è rivelato molto utile perché, anche se non si può fare troppo, si comprende se si desidera avere una galleggiabilità positiva o negativa e giocare aggiungendo con la quantità d'acqua nei serbatoi di galleggiamento.

A questo punto, si potrebbe certamente creare un semplice controller con alcuni pulsanti o interruttori per ciascun motore e delle batterie. Tuttavia, abbiamo optato per l'aggiunta di un Raspberry Pi Pico come controller, perché ci ha permesso di pilotare i motori utilizzando la modulazione di larghezza di impulso (PWM), che ci consente di controllarne la velocità e la direzione. Per farlo, abbiamo dovuto usare dei driver per motori e abbiamo utilizzato una scheda StoRPer, che è un PCB che abbiamo realizzato come parte del nostro libro *Design an RP2040 board with KiCad*.



▲ Figura 4: Aggiungi dei contenitori per pellicole da 35 mm per fungere da serbatoi di galleggiamento

ATTENZIONE!

Rischio elettrico

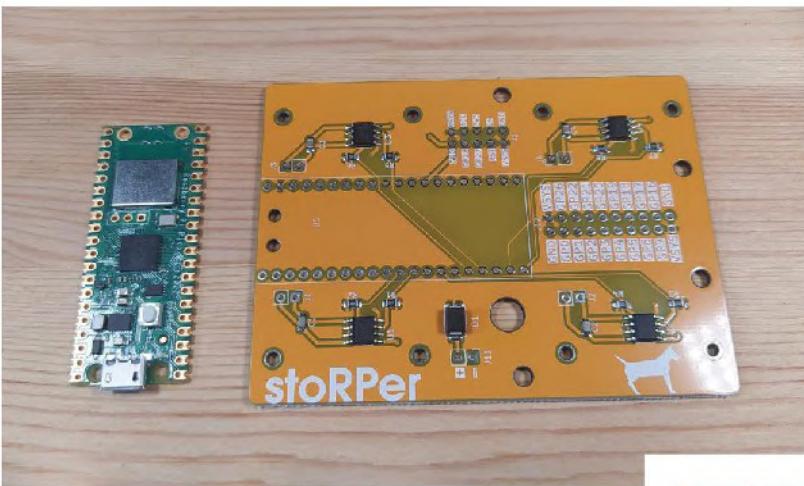
Cortocircuitare direttamente i fili di una batteria comporta un leggero rischio di scintille. Si consiglia di realizzare un circuito di prova. Anche le celle agli ioni di litio possono causare ustioni o incendi in caso di cortocircuito. Il progetto TOUV utilizza motori DC da 3-6 V. Si prega di non utilizzare tensioni superiori o motori con tensioni maggiori.

Motori e acqua

Uno dei motivi principali per cui questo progetto ci è rimasto impresso fin da quando abbiamo visto i robot Lego della Dark Water Foundation al Liverpool Makefest un decennio fa è che siamo rimasti scioccati nel vedere che i motori a corrente continua funzionavano sott'acqua senza alcun tentativo di impermeabilizzazione. Questa non è una pratica comune, ma funziona.

Non possiamo dire che sia l'ambiente più sano per i piccoli motori a spazzole, ma non è certo un'idea così sbagliata come sembra. I motori funzionano bene e abbiamo sempre prestato attenzione quando abbiamo terminato una sessione subacquea. Abbiamo fatto girare i motori fuori dall'acqua per almeno due minuti di funzionamento continuo per eliminare l'umidità residua. Abbiamo anche dato loro una bella soffiata di aria compressa, o un bel soffio quando l'aria compressa si è esaurita.

Siamo certi che far funzionare i motori a corrente continua in acqua dolce non faccia bene e ne stiamo sicuramente accorciando la durata; tuttavia, i nostri sono sopravvissuti a parecchio tempo sott'acqua e siamo ancora con la coppia originale. Abbiamo anche trovato i nostri motori online a meno di 70 cent l'uno, acquistandoli in confezione da dieci, quindi non è oneroso sostituirli se, e quando, si dovesse guastare.



▲ **Figura 5:**
Utilizzare
una scheda
StoRPer è
funzionalme
nte lo stesso
che
aggiungere
alcuni
moduli
DRV8833
a un Pico

(rpimag.co/kicadbook) e articoli pubblicati su HackSpace Magazine. La scheda StoRPer (**Figura 5**) è essenzialmente una scheda breakout che collega alcuni dei pin GPIO di Pico a quattro driver per motori DRV8833. Si può emulare questa scheda su una breadboard utilizzando lo schema e i componenti di StoRPer; tuttavia, i driver per motori DRV8833 sono comunemente venduti come piccoli moduli su schede breakout e questa rappresenterebbe un'opzione semplice.

Per il nostro primo sistema di controllo, abbiamo montato tre pulsanti su un pannello (**Figura 6**) e li abbiamo collegati al GPIO di Pico. Nel codice MicroPython (**Figura 7**), abbiamo identificato i pin GPIO per i pulsanti e li abbiamo definiti, configurando un sistema PWM per ogni motore. Scambiando la coppia di valori dei numeri di pin per ciascun motore, è possibile invertire la marcia e modificare anche i valori di velocità per ciascun motore.

Per i nostri test più seri, ci siamo spostati su una vasca idromassaggio gonfiabile usata. Queste vasche si trovano spesso di seconda mano, trascurate e sporche, a un prezzo molto basso e rappresentano un ottimo ambiente di prova per progetti in acqua. Abbiamo dato alla nostra una bella pulita e, quando l'abbiamo riempita, le abbiamo dato una dose di cloro per uccidere eventuali insetti un paio di giorni prima di usarla (per permettere al cloro di dissolversi). A parte questo, è molto divertente posizionare una action cam in una custodia impermeabile nella vasca e catturare qualche filmato/immagine delle vostre missioni di prova subacquee! Ecco un link a un minuto di test subacquei: rpimag.co/touvid.

Una cosa che cambieremo è che i nostri cavi sono cablati direttamente al controller e sarebbe utile, soprattutto in fase di test, poter rimuovere rapidamente i cavi in modo che solo il controller



Attenzione!

Rischio chimico

Si prega di maneggiare le sostanze chimiche come il cloro in modo responsabile.

rpimag.co/chemicals



◀ **Figura 6:**
Abbiamo
aggiunto un
piccolo
pannello
in cui
abbiamo
montato
i pulsanti
come primo
tentativo di
controller

Abbiamo montato tre pulsanti su un pannello

LED per l'aspetto

A volte si aggiungono delle funzionalità senza motivo. Questo è stato sicuramente il caso della coppia di LED che abbiamo montato. Ovviamente, se avessimo avuto un qualche tipo di telecamera a bordo, l'illuminazione avrebbe potuto avere un senso, ma li abbiamo aggiunti solo per estetica. È sufficiente saldare una resistenza limitatrice di corrente in serie all'anodo di un LED: abbiamo scelto 100 ohm, che proteggono per tutto intervallo di tensioni che intendevamo utilizzare sulla nostra build (3,7–5 V).

Una volta saldati alcuni fili, abbiamo aggiunto un po' di guaina restringente per evitare che l'anodo e il catodo del LED si toccassero, e per piccolo rinforzo strutturale. Abbiamo quindi utilizzato delle piccole fascette per fissare i LED al telaio.



può essere riportato al portatile (non vicino alla vasca idromassaggio) per apportare delle modifiche al codice. Abbiamo asciugato molto il veicolo e i cavi ogni volta che dovevamo apportare una modifica.

Abbiamo alimentato il nostro sistema in un paio di modi diversi: con una batteria 18650 o con un power bank USB. Con i 5 V del power bank, i motori girano abbastanza velocemente e il veicolo ha troppa spinta ed è difficile da controllare, ma questo problema si può risolvere facilmente modificando i valori di velocità nello script MicroPython. Al momento siamo impostati con una leggera spinta positiva, con il propulsore dell'asse Z settato per ruotare molto più lentamente rispetto ai propulsori orizzontali, perché è bello poter scendere in modo fluido e graduale. Abbiamo anche finito per praticare dei piccoli fori nel telaio per allargarlo quando è immerso.

È molto divertente sperimentare e abbiamo qualche idea per migliorarlo. Sarebbe fantastico rendere il cablaggio un po' più flessibile con fili più sottili. Vorremmo anche aggiungere più funzioni al controller. Un pulsante per invertire la direzione dei propulsori orizzontali sarebbe utile, così come alcuni interruttori che possano impostare i motori in modalità a bassa velocità, magari anche con potenziometri per variare la velocità. Ad esempio, potremmo impostare una bassa velocità per il motore dell'asse Z che, una volta attivata, contrasta la spinta idrostatica, consentendo al TOUV di avere una sorta di "mantenimento dell'altitudine". Potrebbe essere divertente aggiungere accessori come un gancio o un magnete per divertenti giochi e sfide di "salvataggio dell'oggetto". Naturalmente ci piacerebbe avere una telecamera a bordo, sensori e altre funzionalità, ma pensiamo che potrebbero essere più adatti a un progetto sommersibile fai da te più ampio. Teni d'occhio il progetto in futuro. Nel frattempo, se vuoi creare un TOUV, i file si trovano in questo repository GitHub: rpimag.co/touvgit.

```

1 #!/usr/bin/python3
2 # This script has been used with a stepper motor driver board but would equally work with a Pico connected via 2 universal motor driver modules.
3 #
4 # You can fine tune each motor's power with the GPIN Ms variables
5 # You can toggle the motor direction by either swapping the GPID pair or the PWM X/Y values
6 #
7 # As written it's set up for one motor to run a little slower as in the first TOUV prototype we found the Z axis thruster was too powerful
8 #
9 # M1: GPDO 9, G1: DIO7 attached to GP20 26 and M2
10 # M2: GPDO 10, G1: DIO11 attached to GP20 32 and M3
11 # M3: GPDO 11, G1: DIO10 attached to GP20 33 and M4
12 # M4: GPDO 12, G1: DIO9 attached to GP20 21 and M5
13 #
14 # From module import Pin, PWM
15 from machine import Pin, PWM
16 #
17 import time
18 #
19 # Set frequency for all motors
20 PWM = 1440
21 #
22 # A 1.5 ms pulse width is 100% duty
23 # SPEED_P1 = 40000 is right rear thruster from rear view on TOUV
24 # SPEED_P2 = 40000 is z axis tower
25 # SPEED_P3 = 40000 is left rear thruster from rear on TOUV
26 #
27 # Set pin config
28 import machine
29 # Motor Pins
30 (M1Pin1, M1Pin2), (M2Pin1, M2Pin2), (M3Pin1, M3Pin2), (M4Pin1, M4Pin2) = machine.Pin(9), machine.Pin(10), machine.Pin(11), machine.Pin(12)
31 #
32 # Buttons
33 button = Pin(14, Pin.IN)
34 #
35 # A timer for temp idle polling
36 buttons = []

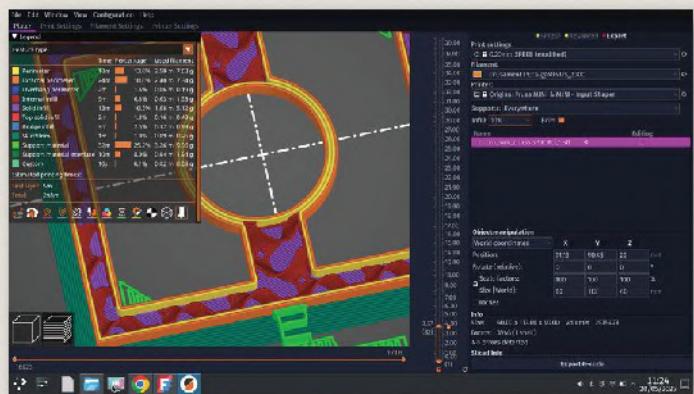
```



◀ Figura 7: Utilizzando MicroPython sul Pico abbiamo potuto apportare rapidamente modifiche alla velocità e alla direzione del motore

Veicoli più grandi

Facendo ricerche sui veicoli subacquei fai da te, si trova una gran serie di veicoli di grandi dimensioni con telai realizzati con tubi idraulici in PVC o ABS. La maggior parte di questi modelli presenta piccoli fori nei tubi per consentire al telaio di riempirsi d'acqua quando è immerso. Questo offre alcuni grandi vantaggi: la massa aggiuntiva stabilizza il veicolo in acqua e significa che si ha a che fare solo con i serbatoi o i tubolari di galleggiamento aggiuntivi attivamente durante la messa a punto dell'allestimento.



Dopo aver riflettuto sulla progettazione di fori nel piccolo telaio, ci siamo resi conto che si poteva ottenere in un modo diverso. All'interno di strutture stampate in 3D a filamento, di solito il software di slicer (che prepara gli oggetti 3D per la stampa) aggiunge automaticamente una percentuale di riempimento. Questo riempimento viene generato in pattern per creare una sorta di reticolato all'interno dell'oggetto, aumentando la resistenza della struttura. La maggior parte degli slicer offre una serie di pattern diversi che possono essere utilizzati per creare il riempimento. Spesso, una mesh quadrata o esagonale va bene, ma ci sono alcune opzioni interessanti. Una è un riempimento giroide curvo. Un aspetto interessante del riempimento giroide è che, sebbene crei forti legami tra le parti che riempie, non chiude mai le sezioni all'interno dell'oggetto. Questo significa che se si stampa con un riempimento giroide e poi si praticano dei piccoli fori negli angoli del telaio, il telaio può riempirsi completamente d'acqua e anche drenare completamente l'acqua dopo una sessione subacquea.