Prototipazione con sensore prossimità Sonar HC-SR04 e collegato a thingspeak on line

<https://it.mathworks.com/help/thingspeak/rapid-prototyping-with-thingspeak.html>

Questo esempio mostra come prototipare con ThingSpeak ™ usando una scheda Arduino ® connessa alla rete wireless . Leggi e scrivi i dati del canale con la libreria di comunicazione ThingSpeak. In questo esempio, una soglia regolabile viene letta da ThingSpeak per creare un rilevatore di prossimità. L'hardware è costituito da un Arduino MKR100 con un sensore sonar e un LED di indicazione per mostrare quando un oggetto è presente.

La prototipazione di un progetto di sensore può richiedere il reflash multiplo del codice per mettere a punto i parametri di calibrazione, come quelli per un sensore di temperatura. Una regolazione hardware aggiunge complessità, riduce l'affidabilità e può essere difficile da raggiungere quando il progetto viene distribuito. Invece, puoi leggere i parametri di calibrazione da ThingSpeak e modificare i parametri in tempo reale. È possibile migliorare la qualità delle misurazioni in tempo reale quando si regola un parametro di sintonizzazione memorizzato su ThingSpeak.

**Hardware supportato**

* Arduino MKR1000
* Arduino Uno, Mega, Due, Leonardo con rete wireless o connessione Ethernet
* Particle Photon (con alcuni codici e regolazioni schematiche)

In questo esempio, un sensore sonar monitora una cucitrice e registra la misurazione sul tuo canale. Il sensore sonar funziona inviando un impulso sonoro e misurando il tempo necessario all'impulso per tornare da un oggetto. Il sistema utilizza la soglia di distanza letta da ThingSpeak e confronta la soglia con la distanza misurata. Il LED è acceso, a indicare la presenza della cucitrice.



Dopo aver completato questo esempio, vedrai i dati di prossimità e le impostazioni nel tuo canale. La vista del canale ThingSpeak mostra:

* La soglia è impostata su 15 cm
* Dieci misurazioni sono prese per ogni media
* Il flag di lettura è impostato per forzare la lettura dei parametri di calibrazione
* I dati mostrano che qualcuno ha "preso in prestito" la spillatrice alle 11 del mattino, riportandola successivamente in una nuova posizione



**Prerequisiti**

1. [Crea un canale ThingSpeak](https://it.mathworks.com/help/thingspeak/collect-data-in-a-new-channel.html) . È possibile memorizzare i dati nel primo campo e memorizzare le impostazioni del dispositivo in campi successivi. Le impostazioni del dispositivo includono una soglia di distanza, il numero di misurazioni da misurare e un indicatore per controllare se il dispositivo aggiorna le impostazioni su ciascun ciclo.
2. Abilita i campi 1, 2, 3 e 4 nella vista **Impostazioni canale**. Puoi anche fornire un titolo informativo per ogni campo.

Il browser restituisce il numero di voci nel canale. Se hai appena creato il canale, vedi 1, 2 e 3 quando compila i tre campi di calibrazione richiesti. Questa immagine mostra un singolo canale configurato per raccogliere i dati nel primo campo e memorizzare i parametri di calibrazione nei campi 2, 3 e 4.



1. Notare le chiavi API di lettura e scrittura appropriate dalla scheda **Chiavi API**nella vista **Impostazioni canale** (cerchiate nell'immagine).
2. Compila i tuoi campi di calibrazione utilizzando una finestra del browser. Puoi copiare il testo dalla scheda Chiavi API o modificare questo testo con la tua chiave API di scrittura. Inserisci ogni URL direttamente nella barra degli indirizzi del tuo browser, modificando il testo in **grassetto** :
	1. Imposta la soglia per rilevare un oggetto, 50 cm è un buon punto di partenza:
	2. https://api.thingspeak.com/update?api\_key= **LA TUA API WRITE Key** & field2 = **THRESHOLD**
	3. Imposta il numero di misurazioni in media:
	4. https://api.thingspeak.com/update?api\_key= **LA TUA API WRITE** Key & field3 = **NUMTOAVERAGE**
	5. Imposta il flag di controllo in modo che il dispositivo verifichi periodicamente nuove impostazioni:

https://api.thingspeak.com/update?api\_key= **LA TUA API WRITE** Key & field4 = 1

**Hardware richiesto**

* Arduino MKR1000 o altro Arduino con connessione di rete Ethernet o wireless
* Sensore sonar HC-SR04
* Fili dei ponticelli (almeno 4 o 5)
* GUIDATO
* Resistore piccolo (100 Ω - 1 kΩ)
* cavo USB

**Schematico e connessioni**

1. Collega Vcc a 5 V sull'MKR-1000.
2. Collegare la terra del sensore alla terra Arduino.
3. Collegare il pin Trig per il pin 8 di Arduino.
4. Collega il pin "Echo" al pin 12 di Arduino.
5. Collegare un resistore al pin 5 di Arduino e collegare il resistore a un LED indicatore. Collegare l'anodo del LED a terra.

|  |  |
| --- | --- |
| https://it.mathworks.com/help/thingspeak/sonar_connections_pic.png | https://it.mathworks.com/help/thingspeak/sonar_sch.png |

**Programmare il tuo Arduino**

1. Scarica l'ultimo IDE di Arduino.
2. Aggiungere la [libreria ThingSpeak per Arduino e ESP8266](https://github.com/mathworks/thingspeak-arduino) al gestore librerie.

Selezionare Sketch> Include Library> Manage Libraries. Seleziona ThingSpeakper aggiungerlo al tuo schizzo.

1. Aggiungi la libreria WiFi101 nel gestore librerie.

**Nota**

Assicurati di installare la versione 0.13 della libreria. C'è un bug nella versione 0.14 che fa fallire i post.

Selezionare Sketch> Include Library> Manage Libraries. Seleziona WiFi101per aggiungerlo al tuo schizzo.

1. Crea l'applicazione:

Apri una nuova finestra nell'IDE di Arduino e salva il file. Aggiungi il codice fornito qui. Assicurati di regolare le informazioni sulla rete wireless, l'ID del canale e le chiavi API.

1. Dopo aver caricato correttamente il programma, è possibile monitorare l'output utilizzando il monitor seriale. Se un oggetto è presente a una distanza inferiore rispetto alla soglia impostata ThingSpeak, il dispositivo registra la distanza dal tuo canale. Prova a fare la media con il numero di misurazioni per vedere come cambiano le tue fluttuazioni misurate. Una volta impostati i parametri, è possibile modificare il valore della flag di lettura da 1a 0. Una volta che il tuo dispositivo ha letto questo flag come 0, smette di controllare ThingSpeak per i parametri, risparmiando energia e larghezza di banda.

 [Codice completo per copia e incolla](https://it.mathworks.com/help/thingspeak/rapid-prototyping-with-thingspeak.html)

* 1. Inizia includendo le librerie e inizializzando i pin di controllo.
	2. #include <ThingSpeak.h>
	3. #include <WiFi101.h>
	4. #define triggerPin 8
	5. #define echoPin 12

#define LEDpin 6

* 1. Definire e inizializzare le variabili. Assicurati di regolare le informazioni sulla rete wireless, l'ID del canale e le chiavi API. Puoi trovare il tuo ID canale nella parte superiore della pagina principale del tuo canale.
	2. // Informazioni sulla rete
	3. const char \* ssid = "SSID";
	4. const char \* password = "xxxxxxxxxx";
	5. // thingSpeak informazioni
	6. char \* writeAPIKey = "XXXXXXXXXXXXXXXX";
	7. char \* readAPIKey = "YYYYYYYYYYYYYYYY";
	8. const long channelID = 000000;
	9. const unsigned int firstReadFieldNumber = 2;
	10. const unsigned int secondReadFieldNumber = 3;
	11. const unsigned int switchField = 4; // Numero del campo (1-8) da utilizzare per modificare lo stato del dispositivo. Determina se i dati vengono letti da ThingSpeak.
	12. // Altre costanti
	13. const unsigned long postInterval = 60L \* 1000L; // Posta i dati ogni 60 secondi
	14. // Variabili globali
	15. unsigned long lastConnectionTime = 0;
	16. long lastUpdateTime = 0;
	17. float distanceThreshold = 0;
	18. bool getInfo = 1; // Imposta questo a zero se non vuoi più leggere i dati da ThingSpeak (cioè calibrazione completa)
	19. int points = 7;

Client WiFiClient;

* 1. Inizializza i pin per input e output e leggi i parametri di calibrazione per la prima volta nella setup routine.
	2. void setup () {
	3.
	4. Serial.begin (9600);
	5. pinMode (triggerPin, OUTPUT);
	6. pinMode (LEDpin, OUTPUT);
	7. pinMode (echoPin, INPUT);
	8. connectWifi ();
	9. Serial.println ( "Start");
	10. // Ottieni i parametri iniziali da ThingSpeak.
	11. distanceThreshold = readTSData (channelID, firstReadFieldNumber, readAPIKey);
	12. punti = readTSData (channelID, secondReadFieldNumber, readAPIKey);

}

* 1. Ogni volta che si esegue il ciclo principale, eseguire più misurazioni. Se i dati non sono stati scritti su ThingSpeak per un po 'di tempo, scrivere i dati sul canale di uscita. Inoltre, se il flag di lettura è stato letto l'ultima volta come vero, controllare i dati di calibrazione e i flag in ogni ciclo.
	2. void loop () {
	3.
	4. distanza di galleggiamento = 0;
	5. // Assicurati che ci sia una connessione internet.
	6. if (WiFi.status ()! = WL\_CONNECTED) {
	7.
	8. connectWifi ();
	9. }
	10. for (uint16\_t loops = 0; loop <points; loop ++) {
	11. distance + = getDistance (triggerPin, echoPin); // effettua una misura, memorizza la somma di tutte le misure
	12. ritardo (5);
	13. }
	14.
	15. distanza = distanza / punti;
	16.
	17. if (distance <distanceThreshold) {
	18. digitalWrite (LEDpin, HIGH);
	19. }
	20.
	21. altro{
	22.
	23. digitalWrite (LEDpin, LOW);
	24. }
	25.
	26.
	27. Serial.println ("Ave:" + Stringa (distanza) + "cm");
	28.
	29. if (millis () - lastUpdateTime> = postingInterval) {
	30.
	31. lastUpdateTime = millis ();
	32.
	33. if (! (getInfo == 0)) {
	34.
	35. distanceThreshold = readTSData (channelID, firstReadFieldNumber, readAPIKey);
	36. punti = readTSData (channelID, secondReadFieldNumber, readAPIKey);
	37. getInfo = (bool) readTSData (channelID, switchField, readAPIKey);
	38. }
	39.
	40. if (distance <distanceThreshold) {
	41.
	42. // Scrivi dati sul canale ThingSpeak.
	43. writeTSData (channelID, 1, distance, writeAPIKey);
	44. }
	45.
	46. }
	47.
	48. ritardo (500); // Fornire un ritardo tra le misurazioni.
	49.
	50. }
	51. Utilizzare queste funzioni per leggere e scrivere dati da e verso ThingSpeak.
	52. int writeTSData (long TSChannel, unsigned int TSField, float data, char \* ReadAPIKey) {
	53. int writeSuccess = ThingSpeak.writeField (TSChannel, TSField, data, writeAPIKey); // scrive i dati sul canale
	54. return writeSuccess;
	55. }
	56. // Usa questa funzione se vuoi più campi contemporaneamente.
	57. / \*
	58. int writeTDData (long TSChannel, unsigned int TSField1, float data1, unsigned int TSField2, data2, char \* ReadAPIKey) {
	59. ThingSpeak.setField (TSField1, data1);
	60. ThingSpeak.setField (TSField1, data2);
	61.
	62. writeSuccess = ThingSpeak.writeFields (TSChannel, writeAPIKey);
	63. return writeSuccess;
	64. }
	65. \* /
	66.
	67. float readTSData (long TSChannel, unsigned int TSField, char \* ReadAPIKey) {
	68.
	69. dati float = 0;
	70.
	71. data = ThingSpeak.readFloatField (TSChannel, TSField, ReadAPIKey);
	72. Serial.println ("Dati letti da ThingSpeak" + String (dati));
	73. restituire i dati;

}

* 1. Usa *getDistance* per ottenere una singola misura dal sensore.
	2. float getDistance (int tPin, int ePin) {
	3.
	4. lunga durata, distanza;
	5.
	6. digitalWrite (tPin, LOW); // Ripristina il grilletto.
	7. delayMicroseconds (2);
	8. digitalWrite (tPin, HIGH); // Inizia una misurazione.
	9. delayMicroseconds (10); //
	10. digitalWrite (tPin, LOW); // Completa l'impulso.
	11. duration = pulseIn (ePin, HIGH); // Attendi un impulso di riflessione.
	12. distanza = (durata / 2) / 29,1; // Calcola la distanza appropriata usando la velocità stimata del suono.
	13. // Questa sezione è utile quando si esegue il debug del sensore.
	14. / \*
	15. if (distance> = 200 || distance <= 0) {
	16. Serial.println ("Fuori intervallo");
	17. }
	18. altro {
	19. Serial.print (distanza);
	20. Serial.println ("cm");
	21. }
	22. \* /
	23. distanza di ritorno;

}

* 1. Collegare il dispositivo a una rete wireless utilizzando la connectWiFifunzione.
	2. int connectWifi () {
	3.
	4. while (WiFi.status ()! = WL\_CONNECTED) {
	5. WiFi.begin (ssid, password);
	6. ritardo (2500);
	7. Serial.println ("Connessione al WiFi");
	8. }
	9.
	10. Serial.println ( "Connected");
	11. ThingSpeak.begin (client);

 }

**Guarda anche**

[Read Data](https://it.mathworks.com/help/thingspeak/readdata.html) | [Write Data](https://it.mathworks.com/help/thingspeak/writedata.html)

**Guarda anche**

[Analisi e visualizzazione MATLAB](https://it.mathworks.com/help/thingspeak/matlab-analysis-and-visualization.html) | [ThingSpeak Apps](https://it.mathworks.com/help/thingspeak/thingspeak-apps.html)