Agenzia nazionale per le nuove tecnologie,
l'energia e lo sviluppo economico sostenibile

MINISTERO DELLO SVILUPPO ECONOMICO

Ricerca di Sistema elettrico

# Integrazione di sensori acustici in sistemi smart home ed implementazione di algoritmi per l'individuazione e la localizzazione di segnali acustici

A. Laudani, G. M. Lozito, F. Riganti Fulginei, A. Salvini

ROMA TRE
UNIVERSITÀ DEGLI STUDI

Report RdS/PAR2017/044

# INTEGRAZIONE DI SENSORI ACUSTICI IN SISTEMI SMART HOME ED IMPLEMENTAZIONE DI ALGORITMI PER L'INDIVIDUAZIONE E LA LOCALIZZAZIONE DI SEGNALI ACUSTICI

A. Laudani, G.M. Lozito, F. Riganti, A. Salvini (Dipartimento di Ingegneria – Università degli Studi Roma Tre)

Settembre 2018

Report Ricerca di Sistema Elettrico

Accordo di Programma Ministero dello Sviluppo Economico - ENEA
Piano Annuale di Realizzazione 2017
Area: Efficienza energetica e risparmio di energia negli usi finali elettrici e interazione con altri vettori energetici
Progetto: D.6 Sviluppo di un modello integrato di smart district urbano
Obiettivo: b Sistemi e servizi smart per edifici
Responsabile del Progetto: Claudia Meloni, ENEA

Il presente documento descrive le attività di ricerca svolte all'interno dell'Accordo di collaborazione "Integrazione di sensori acustici in sistemi smart home ed implementazione di algoritmi per l'individuazione e la localizzazione di segnali acustici"
Responsabile scientifico ENEA: Ing. Francesco Romanello
Responsabile scientifico: prof. Ing. Antonino Laudani

# Indice

# Sommario

Questo documento descrive l'attività di ricerca svolta al fine di sviluppare sensori acustici a basso costo, da integrare in un ecosistema *smart-home*, per applicazioni di individuazione e localizzazione di segnali acustici. Tali sensori sono pensati per essere impiegati, una volta ingegnerizzati, per la segnalazione di allarmi o comandi, oppure per l'individuazione di rumori o di segnali acustici specifici al fine di migliorare la qualità dell'ambiente domestico, per la localizzazione delle sorgenti acustiche, anche al fine della valutazione della presenza di individui all'interno della smart home. Un tale sistema potrebbe in futuro dare anche suggerimenti sulla riduzione del rumore dovuto alle apparecchiature domestiche. Nel documento è inizialmente descritta sia la problematica approcciata, sia le fasi di lavoro previste e i relativi obiettivi preposti.
La prima fase di lavoro si è concentrata sullo studio del sensore acustico, investigando eventuali soluzioni già presenti sul mercato dal punto di vista dei loro limiti e delle loro prestazioni. Sulla base di questa analisi è poi descritto nel dettaglio il progetto ed il dimensionamento di un sensore microfonico dedicato, rispondente alle specifiche richieste dalla applicazione.
La seconda fase di lavoro è volta all'integrazione di questo sensore all'interno di un ecosistema smart-home già esistente. È necessario identificare un sistema programmabile (e.g. un microcontrollore) in grado di gestire la acquisizione del segnale analogico proveniente dal sensore e l'interfacciamento digitale con altri sensori e/o centraline. Sono quindi presentate diverse soluzioni a microcontrollore con le loro principali caratteristiche funzionali, sulla base delle quali è stata scelta la piattaforma di sviluppo per questo progetto.
La terza ed ultima fase del lavoro riguarda l'implementazione di un algoritmo numerico per l'identificazione e la localizzazione di una sorgente acustica sulla base del segnale raccolto da sensori acustici spazialmente distribuiti. L'algoritmo viene brevemente introdotto in un caso di test semplificato, e viene valutato sia in termini di prestazioni sia di precisione. Viene infine discussa sia l'implementazione dell'algoritmo in MATLAB con i relativi test numerici, sia le prove effettuate acquisendo direttamente il segnale audio dai sensori microfonici messi a punto, sia le problematiche relative alla profondità del campionamento effettuato dal microcontrollore. L'implementazione dell'algoritmo di acquisizione e localizzazione su microcontrollore ha messo inoltre in risalto le problematiche realizzative di un sistema smart di questo tipo, permettendo di individuare i corretti compromessi tra costo computazionale e costo di trasmissione del segnale acquisito, per uno sviluppo ottimale dell'intero sistema e della integrazione in esso di un certo numero di sensori acustici.

# 1 Introduzione

L'oggetto del contratto ENEA–UNIROMA3 sul tema "Integrazione di sensori acustici in sistemi smart home ed implementazione di algoritmi per l'individuazione e la localizzazione di segnali acustici" è un'attività di ricerca, che si propone lo studio di soluzioni innovative dal punto di vista tecnologico per lo sviluppo di sensori low-cost adatti per l'integrazione in sistemi smart home. Tali sensori possono essere impiegati per la segnalazione di allarmi o comandi, oppure per l'individuazione di rumori o di segnali acustici specifici al fine di migliorare la qualità dell'ambiente domestico (un tale sistema potrà in futuro dare anche suggerimenti sulla riduzione del rumore dovuto alle apparecchiature domestiche). A tale scopo in questa relazione vengono riportati i risultati relativi allo sviluppo di sensori acustici per smart home e allo studio ed ed implementazione di algoritmi per la localizzazione delle sorgenti acustiche. Nel corso dell'attività si è valutata inoltre la possibilità del loro eventuale riconoscimento e la loro distinzione in termine di tipologia, nonché la possibilità di utilizzo di tali algoritmi per il conteggio delle persone in un ambiente chiuso.

L'attività svolta e la presente relazione è strutturato nelle seguenti fasi:

a) Studio dei sensori acustici low-cost per applicazioni smart home.

b) Sviluppo ed integrazione in un sistema smart home di sensori acustici

c) Studio e sviluppo di algoritmi per la localizzazione di sorgenti sonore

Queste fasi risultano strettamente interconnesse tra di loro, in quanto la soluzione hardware e/o software erano legate sia in termini di realizzazione sia in termini di scelte di progetto. Nei paragrafi successivi verranno discusse le attività svolte ed i risultati ottenuti per i punti precedentemente elencati, tenendo comunque presente che le attività, sebbene distinte in fasi diverse, rappresentano un unico percorso per la realizzazione del progetto.

# 2 Descrizione delle attività svolte e risultati

## *2.1 Studio dei sensori acustici low-cost per applicazioni smart home.*

Lo scopo di questa attività è stato uno studio di fattibilità su sensori acustici a basso costo per applicazioni di smart home con una eventuale configurazione in ambienti confinati. L'impiego di tali sensori deve avere caratteristiche smart, ossia i segnali acquisiti devono essere elaborati per consentire la segnalazione di allarmi o comandi, oppure per la localizzazione e il conteggio delle persone in un ambiente chiuso (per una discussione generale sui segnali acustici e sul rumore ambientale si consiglia la lettura di [1]). Altre funzionalità eventualmente utili sono il rilevamento di rumori noti, ad esempio legati a specifici elettrodomestici posizionati in punti noti, ecc.
In questa fase, a partire dai dispositivi microfonici [2-3] ed acustici presenti sul mercato, si sono valutate le diverse soluzioni per la realizzazione di sensori acustici low-cost. Il primo passo è stato quindi la valutazione dei prodotti già presenti sul mercato sia in termini di prestazioni acustiche, sia di possibilità di integrazione con gli altri sistemi di una smart home. In questa fase, e nelle successive, si è tenuto in considerazione nella valutazione anche la tipologia di applicazione per tali sensori, ossia il conteggio delle persone presenti in un ambiente confinato e l'individuazione/localizzazione di sorgenti di suoni (ad esempio elettrodomestici accesi in assenza di individui).

A tal fine è necessario disporre in uno stesso ambiente confinato di più sensori dello stesso tipo, che quindi devono risultare:
- piccoli e poco invadenti in termini di impatto visivo sulle persone (a tal fine ne deve essere anche considerata l'integrabilità all'interno di soprammobili o strutture esistenti, quali prese elettriche, lampade, ecc.);

- a basso costo;
- facilmente collegabili con hardware di semplice realizzazione (ad esempio microcontrollori con ADC) ed eventualmente con moduli wireless dedicati.
- omnidirezionali, secondo specifiche esigenze di localizzazione e/o di monitoraggio di alcuni elettrodomestici.

La prima fase dello sviluppo di questa attività ha quindi riguardato un'analisi di mercato sui dispositivi esistenti, considerando le loro caratteristiche elettriche e le loro prestazioni acustiche. Da questa prima fase di analisi è risultata una assenza, sul mercato, di sensori acustici in grado di soddisfare le esigenze di una smart-home con costi congrui agli scopi proposti.
Infatti la maggior parte dei sensori acustici disponibili sul mercato o sono pensati per applicazioni industriali (come ad esempio il monitoraggio dei sistemi meccanici) e quindi presentano bande non utili per i segnali tipicamente presenti nelle nostre abitazioni, o costi elevati, o sono pensati per applicazioni all'aperto (come ad esempio il monitoraggio del traffico) e quindi presentano caratteristiche geometriche e costi non adeguati per la nostra applicazione. Esistono poi i microfoni ambientali per indoor, che presentano caratteristiche che appaiono molto simili a quelle da noi desiderate, ma con prezzi spesso proibitivi (oltre la decina di euro per il solo microfono). Oltretutto tali dispositivi nascono per applicazioni audio vere e proprie e, quindi, necessitano un hardware dedicato per il trattamento del segnale audio vero e proprio, presentando di conseguenza anche una flessibilità minima per l'uso in applicazioni diverse rispetto a quelle per cui sono state pensate (accoppiamento con telecamere o sistemi di videosorveglianza). Un'ultima categoria spesso incontrata sul mercato è quella dei sensori acustici con scheda elettronica di interfaccia. Questa categoria, che apparentemente si potrebbe ben prestare per applicazioni di smart home, consiste in dei sensori basati su capsula microfonica con soglia regolabile. Tali sensori sono diffusi nel mondo dell'elettronica amatoriale. L'uscita di questi sensori non consiste in un segnale acustico, ma piuttosto un segnale binario (vero/falso) che indica una presenza di "rumore" nell'ambiente al di sopra di una soglia regolabile agendo su un potenziometro.
Di conseguenza nessuno dei prodotti presente sul mercato è pensato per il genere di applicazioni di smart home, che sono di interesse per il presente progetto. Inoltre presentano dei costi non compatibili con l'uso di una molteplicità di sensori. Infine nessuno dei prodotti individuati si presenta accoppiato ad un sistema di elaborazione di costo congruo per l'estrazione di features interessanti per l'analisi del segnale acustico in ambienti confinati, con occupanti umani. Per questo motivo si è giunti alla conclusione che si renda necessaria la progettazione specifica di un sensore acustico intelligente per questo tipo di applicazioni. Dall'alto canto questa soluzione viene anche suggerita dal costo di realizzazione pratica di un microfono a partire dall'hardware che lo costituisce, ossia la capsula microfonica e il circuito di condizionamento ed amplificazione.
Nella successiva fase di questa attività si è quindi passati ad affrontare la stesura di uno schema di massima per la creazione di sensori acustici intelligenti, in grado di soddisfare le specifiche richieste.
Questi sensori smart devono essere costituiti da una parte dal sensore acustico vero e proprio, e dall'altra dal sistema di pre-elaborazione e trasmissione del segnale acquisito. Tale soluzione, sebbene più complicata dal punto di vista realizzativo rispetto a quella di un dispositivo già esistente sul mercato, è allo stesso tempo innovativa e flessibile, e meglio si adatta ad essere integrata in un ambiente smart home. Si presterebbe inoltre allo sviluppo di nuovi dispositivi standard, che dopo la fase di studio ed ingegnerizzazione potrebbero risultare sia direttamente inseribili sul mercato, sia brevettabili.
In questa sezione vengono discussi gli aspetti relativi al sensore acustico vero e proprio, costituito da capsula microfonica e circuito di amplificazione, mentre nel successivo paragrafo 2.2 verranno discussi gli aspetti relativi all'hardware di acquisizione.

### 2.1.1 Capsule microfoniche

Questa fase dello sviluppo dei sensori acustici low-cost si è concentrata sulla possibilità di utilizzo di una capsula microfonica a condensatore, magnetica o piezoelettrica, e sulla analisi della eventuale necessità di

pre-amplificazione del segnale. Si premettono alcune note generali sui microfoni che hanno portato alla scelta finale. La capsula microfonica è la "testa" del microfono stesso [2-3]: essa è ricoperta dalla griglia microfonica (parte passiva) e dal trasduttore (parte attiva); il trasduttore ha il compito di prelevare la pressione sonora incidente e trasformarla in impulsi elettrici corrispondenti: ciò avviene mediante l'uso di un diaframma mobile (materiale in lega metallica adibito al movimento corrispondente alla pressione sonora incidente su di esso). La griglia microfonica ha lo scopo di isolare la capsula da urti, correggendo la risposta in frequenza, regolando il diagramma polare cancellando o attenuando suoni provenienti da angoli indesiderati. La griglia è costruita con leghe metalliche: acciaio, alluminio, carbonio; in modo tale da ottenere una maggiore affidabilità con meno usure o rischi di deformazione nel tempo. Infine la forma a ragnatela della griglia consente, sfruttando il principio della gabbia di Faraday, di avere una grande protezione contro le interferenze elettromagnetiche che andrebbero ad alterare le alte frequenze. Chiaramente la griglia spesso si riduce notevolmente fino ad diventare un sottile velo o un sottilissimo velcro che permette di proteggere la capsula dalla polvere, o addirittura la griglia non risulta presente.

La *capsula microfonica a condensatore* [2-3], detta anche microfono a condensatore o microfono elettrostatico, è considerata il microfono per eccellenza e può essere utilizzata sia in ambito musicale sia in ambito ambientale. Un condensatore è un apparato elettrico costituito da due armature (chiamate anche lamine, elettrodi o piastre) di materiale conduttore, separate da un isolante. Se alle due piastre viene applicata una tensione, sulle lamine si accumula una carica Q, proporzionale alla tensione, tramite la quantità C, chiamata capacità, che dipende dalla geometria della capsula e soprattutto dalla distanza tra le armature. Il microfono a condensatore è composto da due lamine. Quella esterna è mobile e rappresenta il diaframma del microfono. Quella interna, invece, è fissa e stabile. Il diaframma mobile si muoverà per via delle onde sonore che lo colpiscono, avvicinandosi alla lamina fissa e creando di conseguenza una variazione del campo elettrostatico, ossia della capacità C. La variazione della capacità porta quindi ad una variazione della tensione ai suoi capi. La presenza di un circuito di amplificazione è fondamentale in quanto il livello di sensibilità generato dal processo di trasduzione è molto basso, dell'ordine dei µV, al quale si aggiunge un valore di impedenza inutilizzabile per applicazioni audio. Il processo di amplificazione porta i valori di tensione a qualche mV e l'impedenza di carico da circa 50 Ω a 250 Ω. Un altro aspetto da considerare è che il sistema di trasduzione elettrostatica necessita di un'ulteriore componente: affinché avvenga una variazione di campo statico tra le due armature è necessario che la lamina mobile (diaframma mobile) sia alimentata da una corrente continua e stabile, cosicché il suo movimento produrrà variazioni del campo elettrico per differenza di potenziale con l' armatura fissa; se non fosse alimentata non creerebbe variazioni di tensioni e di conseguenza nessun suono. Per far ciò il microfono a condensatore deve essere in qualche modo alimentato dall'esterno. Questa alimentazione è ottenibile tramite un circuito di polarizzazione.

Una *capsula microfonica piezoelettrica* [2-3] o microfono piezoelettrico è anche chiamato microfono a contatto in quanto per un suo miglior rendimento è necessario porre la capsula a contatto con la sorgente. Il suo utilizzo principale è negli strumenti musicali o nei sistemi meccanici in cui è necessario trasdurre le vibrazioni meccaniche in suoni acustici. Attualmente è usato nei moderni telefoni e cellulari ed ha una risposta in frequenza limitata. Data la sua natura, può considerarsi un filtro naturale che produce esclusivamente frequenze alte e medio - alte. Anche in questo tipo di microfono è presente un diaframma mobile (membrana) a cui è collegata una lamina di cristallo, generalmente quarzo; quando il diaframma si muove andrà a puntellare la lamina di cristallo che comincerà a muoversi, generando un segnale elettrico corrispondente. Ponendo una sola lamina di cristallo, la quale è molto leggera e cedevole, è molto facile arrivare al punto di rottura anche con una minima pressione applicata; per ovviare a ciò sono implementate delle lamine di cristallo doppie e triple al fine di aumentare la rigidità del dispositivo. In uscita, un microfono piezoelettrico presenta un'impedenza molto elevata, e per poterlo interfacciare correttamente con un dispositivo pre-amplificatore anche esso dovrà avere una impedenza molto elevata.

Un *microfono magnetico o dinamico* [2-3] è strutturalmente molto simile ad un piccolissimo altoparlante: sfrutta il fenomeno dell'induzione elettromagnetica per convertire il movimento della membrana (normalmente costituita da una pellicola molto sottile) in una forza elettromotrice, grazie ad un avvolgimento di filo conduttore sottilissimo meccanicamente fissato alla membrana stessa, chiamato bobina mobile,

immerso nel campo magnetico generato da un magnete permanente. Il movimento della bobina mobile nel campo magnetico genera una corrente indotta. Questa corrente è proporzionale all'ampiezza dei movimenti dell'avvolgimento, che sono a loro volta proporzionali all'intensità della pressione sonora, ossia del segnale acustico. Questa corrente costituisce il segnale elettrico audio. Sebbene sostanzialmente simile alla capsula a condensatore (la quale è basta su una interazione di campo elettrico, a differenza della presente che è basata su una interazione di campo magnetico), rispetto ad essa risulta più ingombrante e pesante a causa del nucleo ferromagnetico, il che rappresenta una grossa limitazione per una applicazione in smart home.

### 2.1.2 Realizzazione del microfono dedicato

Partendo dalla condizione che la banda udibile dall'orecchio umano si attesta in un range da 20 Hz a 20000 Hz è necessario lavorare su una capsula che operi all'interno di questo intervallo. Il compromesso migliore in termini di costi e prestazioni è dato da una capsula a condensatore. È inoltre più sensibile ai transitori (rapida variazione dell'ampiezza del segnale) e alle sollecitazioni meccaniche. Tuttavia, la capsula, non ha un elevato guadagno e per questo motivo è necessario aggiungere un preamplificatore atto ad aumentare l'ampiezza del segnale generato. Questi preamplificatori possono essere di tipo già integrato nel dispositivo completo o realizzati come moduli indipendenti. Purtroppo, sebbene siano disponibili sul mercato diverse soluzioni a costi variabili, la loro adozione non è sempre funzionale, soprattutto visti gli scopi finali di realizzazione di un sensore per smart home, ossia flessibile nella implementazione ed interfacciamento con sistemi a microcontrollore e/o DSP. Dall'altro canto la realizzazione di questi sistemi di amplificazione non è critica né dal punto di vista progettuale né in termini di costi di realizzazione. Di seguito le scelte progettuali effettuate per la realizzazione del sistema completo. Il prototipo è stato costruito a partire da diverse capsule microfoniche a basso costo, ma con buone prestazioni acustiche. Ad esempio è stata utilizzata la capsula CMC-4015-40P [4] (del costo unitario inferiore ai 2 euro), con le caratteristiche riportate nella tabella 2.1.1.

**Tabella 2.1.1 Caratteristiche della capsula microfonica CMC-4015-10P**

| Parameter | Condition | Typical Value |
|---|---|---|
| Directivity | | Omnidirectional |
| Sensivity | f=1K[Hz], 1Pa, 0dB=1V/Pa | -40dB |
| Zout | f=1K[Hz], 1Pa | 2.2kΩ |
| Frequency | | 100 –20000 [Hz] |
| SNR | f=1K[Hz], 1Pa | 58dBA |
| Dimensions | | 0.40x1.5 mm |

La prima fase dello sviluppo hardware ha riguardato la realizzazione del circuito di test e polarizzazione della capsula microfonica considerata. Come è possibile vedere dalla figura 2.1.1 la capsula microfonica (nella schematizzazione è rappresentata dal circuito interno al rettangolo tratteggiato, nel circuito realizzato è il cilindretto con il velcro nero) richiede per il funzionamento il collegamento ad una tensione di alimentazione Vs positiva rispetto al *ground*. Il morsetto di segnale è quello indicato con output. Il circuito di polarizzazione/test è necessario se vogliamo usare la capsula in un sistema hardware non dedicato all'audio, mentre non è necessario se, ad esempio colleghiamo la capsula al jack microfonico di una scheda audio (ad esempio di un PC o notebook).

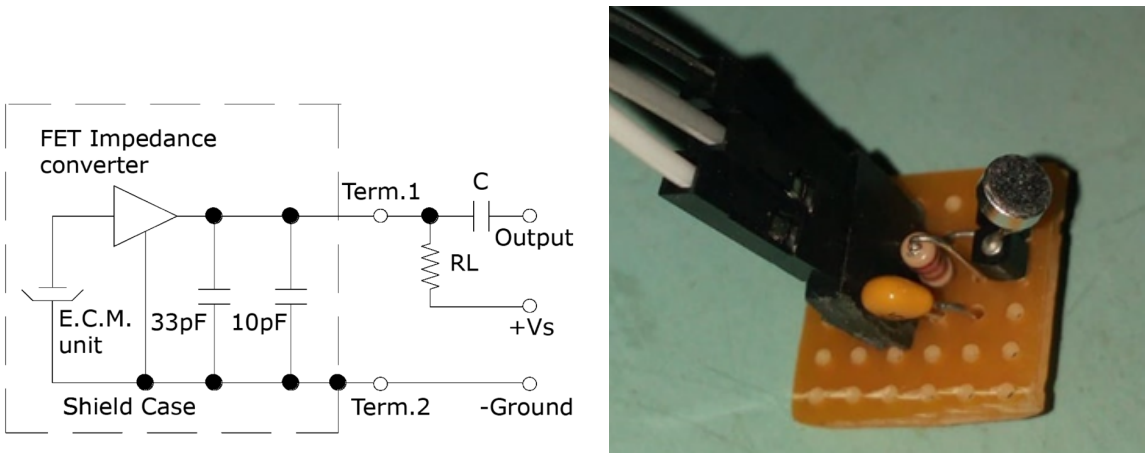


**Figura 2.1.1 Circuito di test/polarizzazione (sinistra) della capsula microfonica a condensatore CMC-4015-40P e relativa realizzazione (destra).**

Nell'ipotesi di usare un dispositivo a microcontrollore dotato di ADC per l'acquisizione del segnale di questa capsula, sono stati realizzati alcuni circuiti di polarizzazione per il suo corretto funzionamento. Nei test di acquisizione del segnale con l'uso del calcolatore, effettuati e descritti nel successivo paragrafo 2.3, la capsuale è stata direttamente collegata all'ingresso audio del PC, avendo cura di distinguere il morsetto di segnale (ed alimentazione positiva, indicato con Term.1) dal morsetto di ground (Term.2). Come sarà descritto in seguito, i test di acquisizione effettuati con il calcolatore hanno confermato il perfetto funzionamento della capsula. Si è tra l'altro verificato che la dinamica picco-picco della capsula fosse limitata a qualche mV. Questo ha reso necessario, per mantenere una intellegibilità della registrazione, l'utilizzo nella fase di campionamento/quantizzazione tramite calcolatore (con algoritmi implementati in Matlab) di una conversione analogico-digitale a 16 bit.
Va sottolineato che la maggior parte dei microcontrollori a basso costo presentano ADC con numero di bit raramente superiore a 10. Per ovviare a questa problematica, si è pensato di utilizzare un circuito di amplificazione e traslazione di livello. La necessità di un traslatore di livello deriva dalla polarizzazione dell'ADC del microcontrollore, per il quale il segnale di ingresso deve essere strettamente positivo. Nativamente, il segnale audio oscilla intorno ad un valor medio nullo. Per ottenere un segnale convertibile dall'ADC è necessario traslare il riferimento del segnale audio a metà della dinamica (alimentazione) dell'ADC. Per realizzare l'amplificatore sarebbe stato possibile sfruttare delle semplici soluzioni (ampiamente adottate in ambito audio) basate su integrati quali LM386 [5], con un po' di circuiteria annessa per la polarizzazione (costo complessivo sotto i 2 euro). Alternativamente, un circuito di amplificazione e traslazione può essere facilmente dimensionato e realizzato con un singolo transistor bipolare. Si è scelta questa seconda strada perché fornisce più gradi di libertà, fermo restando il bassissimo costo del tutto comparabile, se non addirittura inferiore a parità di prestazioni. La configurazione scelta per la realizzazione è un emettitore comune (impiegante il BJT 2N4401 di tipo NPN), realizzato a partire da una rete di polarizzazione a quattro resistenze, in modo da poter fissare l'uscita in assenza di segnale ad una tensione di circa 2.5 V (metà dei 5 V del microcontrollore utilizzato). Il dimensionamento è abbastanza semplice, così come la teoria relativa al guadagno e rimandiamo alla bibliografia [6-7] per i relativi dettagli. Lo schema circuitale è comunque presente in figura 2.1.2 insieme alla sua realizzazione pratica.
Il guadagno complessivo in tensione è stato circa 20 V/V, e la dinamica della capsula microfonica è risultata tale da non portare mai il transistor a lavorare in saturazione. La banda passante dell'amplificatore è risultata perfettamente compatibile con la capsula microfonica, anche tenendo presente che le maggiori limitazioni nascono dal campionamento del segnale.
Infatti la velocità di acquisizione tramite l'ADC di un microcontrollore non è solitamente molto alta, a meno di non investire in sistemi con un certo costo o aggiungere ulteriore hardware di conversione al sistema, il che si è pensato non fosse utile alla luce dell'economicità che il sistema pensato debba mantenere. Infine, i

test effettuati (e descritti nei successivi paragrafi) con il microcontrollore hanno dimostrato che il microfono così realizzato e il sistema di acquisizione a 10 bit fossero compatibili per un buon risultato in termini di interpretazione del segnale audio acquisito.

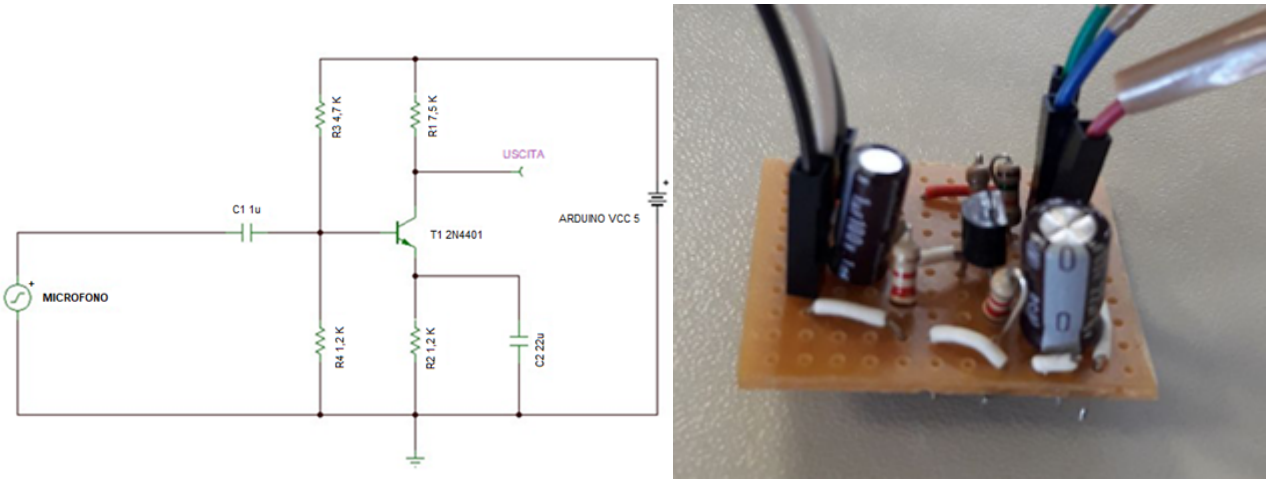


**Figura 2.1.2 Schematizzazione del circuito amplificatore e sua realizzazione.**

In definitiva il microfono a capsula a condensatore così costruito si è dimostrato adatto all'applicazione in termini di requisiti acustici, a basso costo complessivo e poco ingombrante.

## *2.2 Sviluppo ed integrazione in un sistema smart home di sensori acustici*

In questa seconda fase, che come detto in precedenza si è sovrapposta con le altre data l'importante connessione esistente, sono stati studiati e sviluppati i sistemi di acquisizione, interconnessione ed interfacciamento per la gestione e lo scambio dati dei sensori acustici. In particolare, sono state esaminate le soluzioni hardware meno invadenti per una rapida prototipazione ed installazione, nonché la possibilità di sfruttare connessioni wireless e le prestazioni di sistemi diversi anche in termini della loro messa a punto. Inoltre sono state considerate tutte le caratteristiche hardware dei sensori acustici e dei sistemi di gestione di tali sensori.
Per quanto riguarda la parte relativa al sistema di elaborazione/acquisizione si è considerata la possibilità di utilizzo di un scheda contenente un microcontrollore. La possibilità di utilizzare schede con wireless integrato permette di risparmiare i costi di messa a punto. Infatti, tale soluzione semplifica lo sviluppo e l'integrazione nel caso il sistema comprenda numerosi sensori. D'altro canto è bene valutare il posizionamento dei sensori all'interno dei singoli locali. La possibilità di comunicare usando protocolli *z-wave* o *zigbee* permette di connettere, tramite una rete a maglia, più sensori tra loro. L'utilizzo di diverse tipologie di protocolli di trasmissione, tuttavia, è uno svantaggio che può essere mitigato affidando a due sistemi diversi l'acquisizione/elaborazione, da una parte, e la trasmissione dall'altra. Questa soluzione si presta meglio all'integrazione ed è quindi impossibile determinare a priori quale sia la soluzione migliore per la realizzazione finale. Nel seguito di questa relazione verranno analizzate e proposte le diverse soluzioni considerate, tenendo conto sia dei costi realizzativi del sistema sia della possibilità di integrazione in sistemi smart home più complessi.
Oltre alle caratteristiche del sistema di connessione, è stato valutato anche l'aspetto "cognitivo" che il sistema di sensore-acustico smart debba avere. Si è esaminata la possibilità di affidare ad un sistema a microcontrollore l'estrazione delle features del segnale acustico o meno. Infatti non è detto che l'elaborazione debba essere affidata necessariamente ad un microcontrollore per ogni microfono, quanto piuttosto ad un Digital Signal Processor (DSP) dedicato. Questi infatti rappresentano una classe di dispositivi

integrati di particolare interesse per questa applicazione, essendo a tutti gli effetti dei microcontrollori specificamente ottimizzati per l'elaborazione di segnali.
La scelta del dispositivo a microcontrollore o DSP può ricadere su board sviluppate con dispositivi AVR/Atmel come Arduino, o su sistemi della Texas Instruments, che è ben nota per lo sviluppo di sistemi a DSP adatti per l'elaborazione dei suoni, ma anche su sistemi della Microchip, che sviluppa sia microcontrollori che DSP, nonché schede che contengono entrambi i dispositivi o per finire su schede recentemente sviluppate dalla ST-Microelectronics. Una fase preliminare dell'attività ha quindi riguardato l'individuazione di sistemi di sviluppo che integrassero microcontrollori e DSP, presenti sul mercato.
Un esempio di un sistema di sviluppo valutabile per la successiva fase di ricerca è MPLAB Starter Kit for dsPIC DSC, costituito da *DSC dsPIC33FJ256GP506* a 40 MIPS (la scheda è mostrata in figura 2.2.1), con memoria flash da 256 KB, SRAM da 16 KB, Memoria flash seriale da 4 Mbit a bordo scheda, CODEC audio da 16/24/32 bit con una frequenza di campionamento massima di 48 kHz, circuiti di riproduzione e rilevamento audio dai costi contenuti grazie all'uso di un ADC a 12 bit e segnali PWM, ingressi di linea e microfono con guadagno di ingresso regolabile, amplificatore per cuffie da 100 mW con controllo digitale del volume, interruttori utente, 3 LED utente e sensore di temperatura. Il costo di tale scheda si aggira intorno ai 60/70€, ma il problema principale è la sua scalabilità, ossia l'impossibilità di utilizzare il processore dsPIC33FJ256GP506 in sistemi più semplici senza ricorrere all'intera progettazione hardware di una scheda dedicata.


**Figura 2.2.1 dsPIC DSC starter kit**

Un'altra possibilità è data dal Development Kit *Texas Instruments eZdsp.* La famiglia C55x della Texas Instruments rappresenta la classe entry-level dei DSP programmabili. Le principali caratteristiche della famiglia C55x sono il costo contenuto e i consumi ridotti pur mantenendo le elevate capacità computazionali richieste da un DSP (0.3 GMACS, ovvero $0.3\times10^9$ operazioni di Moltiplicazione ed Accumulazione per Secondo). Pur essendo un dispositivo di gamma economica (un dispositivo serie C66x ha prestazioni superiori a 300GMACS) la capacità computazionale è estremamente elevata se confrontata con quella di un microcontrollore general purpose nel quale non è implementato un hardware dedicato per operazioni di moltiplicazione ed accumulo (MAC). Il consumo energetico è scalabile con la frequenza di clock, e passa da 10mW a 60MHz fino a 150mW a 150MHz. Va sottolineato comunque che, sebbene questi consumi siano bassi nell'ambito di un confronto con altri DSP, sono molto elevati se paragonati a quelli di un microcontrollore general purpose. Come periferiche, la famiglia C55x presenta una ampia gamma di interfacce digitali standard (I2C, SPI, UART), hardware dedicato per le operazioni in virgola fissa e timer a 64bit per garantire un riferimento temporale preciso. La memoria on-chip è 512kB di RAM. Non presenta tuttavia un ADC dedicato, che deve essere quindi aggiunto esternamente. Una scheda di sviluppo basata su questo tipo di DSP è il kit *TMS320C5502 / C5509A eZdsp* (vedi figura 2.2.2)

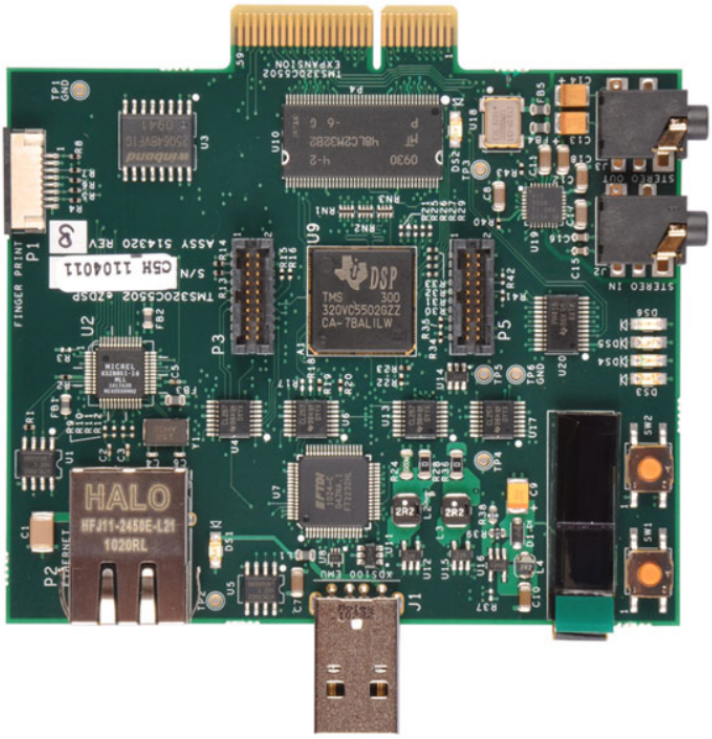


**Figura 2.2.2. Scheda di sviluppo della Texas Instruments basata su DSP della famiglia C5000**

Questo kit è specifico per l'utilizzo dei DSP C5502 o C5509A in applicazioni audio. Il kit è composto da una scheda di sviluppo su cui è montato il DSP, una interfaccia USB per il collegamento al PC, una porta ethernet e due jack audio di ingresso/uscita. I due jack audio sono accessibili dal DSP (che è sprovvisto di ADC) tramite un integrato dedicato, TI AIC3204, che opera la conversione ADC sulla linea di ingresso e DAC sulla linea di uscita. La comunicazione tra DSP e AIC3204 avviene con protocollo I2C. La scheda di sviluppo può essere programmata usando la versione gratuita di Code Composer Studio. Il problema principale legato a questo tipo di soluzione è, oltre al costo che si aggira sugli 80€, la scarsa disponibilità sul mercato del sistema, per cui l'ordine ed il test richiederebbe un'attesa superiore ai 6 mesi.

Un'altra soluzione possibile è quella proposta da Analog Devices, con il kit *Analog Devices ADSP-BF706 Ez-Kit Mini* (figura 2.2.3), che adotta come DSP mid-range dispositivi basati su core Blackfin+. In particolare, il dispositivo ADSP-BF706 opera a 400MHz e garantisce una operazione MAC a 32 bit per ciclo, ottenendo quindi 0.4GMACS. Nel caso sia preferibile aumentare la velocità al costo di una ridotta precisione, il MAC può essere diviso in due unità da 16 bit raddoppiando le prestazioni. La stessa strategia può essere usata per operazioni MAC su numeri complessi. La memoria on-chip è 1Mbyte di RAM più 136KB con controllo di parità di bit. Analogamente alla soluzione TI, anche questo DSP presenta una ampia gamma di periferiche per interfacciamento, quali USB, I2C, UART, SD, CAN, ePPI Video I/O. Il consumo dichiarato a frequenza di clock operativa è più basso rispetto alla soluzione precedente, assorbendo 100mW a 400MHz. Anche questo DSP non presenta un modulo ADC dedicato, che viene sostituito da un codec esterno nelle schede di sviluppo.

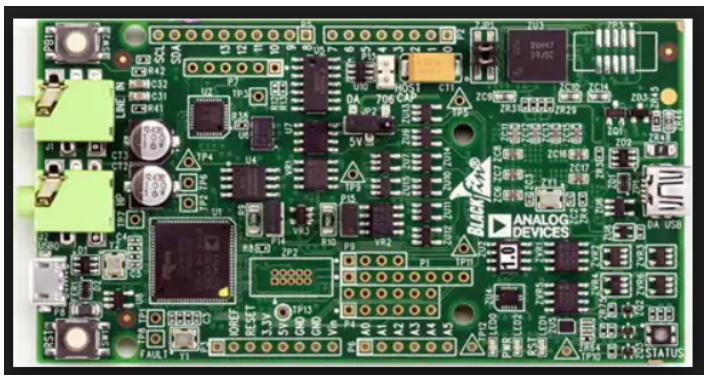


**Figura 2.2.3 Scheda DSP dell'Analog Devices**

In particolare, il kit ADSP-BF706 Ez-Kit Mini, mostrato in Fig. 2.2.3, è mirato alle applicazioni audio di questo dispositivo. Oltre alla interfaccia USB per la programmazione sono presenti due jack audio in/out che comunicano con il DSP, su protocollo I2C, attraverso l'encoder stereo a 24-bit ADAU1761. Il kit fornisce inoltre una memoria esterna flash da 32Mb da usare come program-memory. Il costo è ancora sugli 80 euro ed ancora una volta la scalabilità risulta estremamente bassa.
Sebbene la ST non abbia tra i suoi prodotti un DSP vero e proprio, è presente comunque una scheda di sviluppo specializzata in applicazioni audio.

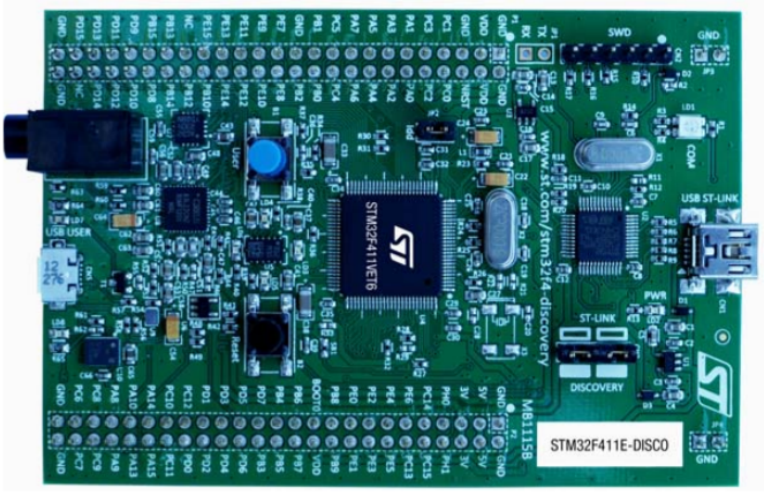


**Figura 2.2.4. Scheda di sviluppo Discovery dell'STMicroelectronics**

La scheda *32F411EDISCOVERY,* mostrata in Fig. 2.2.4, monta un microcontrollore STM32F411VET6, ovvero un microcontrollore con architettura ARM Cortex M4 a 32 bit con unità di calcolo in virgola mobile (Floating-Point Unit), frequenza operativa di 100MHz, 128Kb di RAM e 512Kb di Flash. Analogamente ad un DSP, questo microprocessore supporta istruzioni per il signal-processing, tuttavia non è fornito dell'hardware dedicato (MAC) che permette di eseguire queste istruzioni in pochi cicli di clock come è invece in grado di fare un DSP (il vantaggio risiede principalmente nella semplicità di programmazione di queste istruzioni). Rispetto ai DSP precedentemente presentati, il microcontrollore STM32F411VET6 è fornito di una gamma più vasta di periferiche ed interfacce, tra le quali è notevole il convertitore ADC a 12-Bit 2.4MSPS. La scheda di sviluppo 32F411EDISCOVERY è fornita di interfaccia USB per la programmazione, un jack audio OUT connesso ad un DAC dedicato CS43L22 a 24-bit, e un microfono MEMS omini-direzionale integrato MP45DT02. Il costo di tale scheda è abbastanza contenuto (circa 20 euro), ma il sistema di sviluppo e la poca disponibilità di codice "pronto all'uso", ossia di librerie ben testate, la rende estremamente debole dal punto di vista dell'usabilità. Nonostante ciò è bene tenere presente che questa scheda rappresenta probabilmente la seconda scelta, per i nostri scopi.

Un'altra azienda che produce sistemi DSP a costi relativamente bassi è la *VLSI*. Questa azienda propone una serie di prodotti, pensati soprattutto per progetti di elettronica low-cost (giocattoli o piccoli sistemi che fanno uso di periferiche audio), che permettono l'acquisizione, la compressione ed il trattamento di segnali audio tramite microfoni. Sebbene le schede presentino un costo contenuto (tra i 10 ed i 20 euro), necessitano l'interfacciamento con altri sistemi di controllo e gestione, e quindi risultano utilizzabili solo per la parte di trattamento del segnale audio.
Dopo la valutazione dei diversi sistemi sopraelencati, la scelta è caduta nel più semplice sistema a microcontrollore basato su Arduino, sia per la facilità di utilizzo sia per l'ampia disponibilità di sistemi hardware già sviluppati per l'interconnessione, nonché per la flessibilità nella scrittura del codice di programmazione, anche grazie all'abbondanza di librerie.
Questo sistema, nonostante presenti limitazioni non indifferenti in termini di capacità computazionali, permette una migliore ingegnerizzazione del sistema a sensori acustici smart. Questo grazie appunto alla velocità di messa a punto del sistema e alla possibilità di ricorrere a moduli facilmente collegabili tra loro. Di seguito vengono presentate, con una breve nota sulle caratteristiche principali, le principali soluzioni di schede di sviluppo Arduino, ed in particolare Arduino Uno, Due, Leonardo, Nano e Mega 2560, mostrate in figura 2.2.5.
*Arduino Uno*: è un microcontrollore basato su ATmega328P, pre-programmato con bootloader, il quale permette di caricare un nuovo codice senza l'uso di un programmatore hardware esterno. Dispone di 14 pin di input/output digitali, di cui 6 utilizzabili come uscite PWM, 6 ingressi analogici, un cristallo di quarzo a 16MHz, una connessione USB, un jack di alimentazione, una interfaccia ICPS e un pulsante di reset. La scheda contiene tutti i componenti necessari per supportare funzionamento del microcontrollore, che può essere utilizzato sia collegandolo al computer tramite un cavo USB, sia con alimentazione esterna (non USB) con un adattatore AC-to-DC, sia da batteria. Le caratteristiche sono riassunte nella tabella 2.2.1

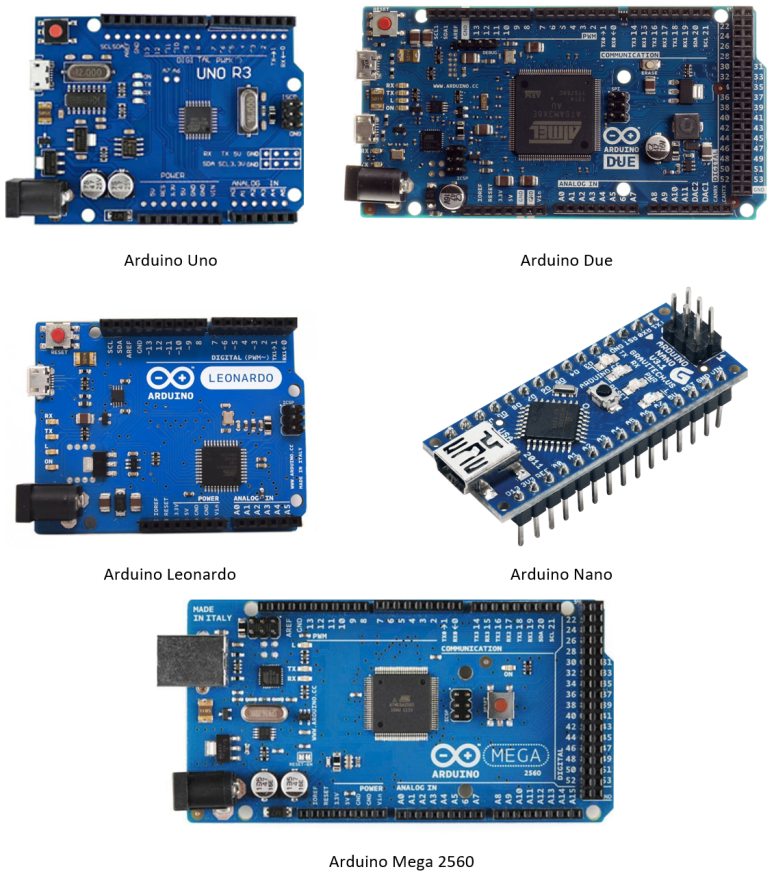


**Figura 2.2.5 Schede Arduino compatibili con lo sviluppo di sensoristica per smart home.**

*Arduino Due*: è basato su un microcontrollore ARM a 32 bit ed è la piattaforma più utilizzata per i progetti Arduino. Dispone di 54 pin di input/output digitali, 12 dei quali utilizzati come uscite PWM, 12 ingressi analogici, 4 porte seriali hardware (UART), un clock a 84 MHz, una connessione USB, 2 convertitori digitale-analogico, un jack di alimentazione, una interfaccia SPI, una interfaccia JTAG e un pulsante di reset sulla scheda. Arduino Due può essere alimentato tramite il connettore USB o con un alimentatore esterno; la fonte di alimentazione viene selezionata automaticamente. L'alimentazione esterna può provenire da un adattatore AC-to-DC o da una batteria. A differenza della maggior parte delle altre schede, Arduino Due funziona a 3.3 V. La tensione massima tollerabile dai pin input /output è 3.3 V. Arduino Due è anche compatibile con tutti gli Shield Arduino (schede di espansione per applicazioni specifiche) che lavorano a 3.3 V ed è conforme all'architettura di Arduino Uno. Le caratteristiche principali di Arduino Due sono elencate nella tabella 2.2.2.

**Tabella 2.2.1 Caratteristiche principali della scheda Arduino Uno**

| | |
|---|---|
| Microcontrollore | ATmega328P |
| Tensione di esercizio | 5V |
| Tensione di ingresso | 7-12V |
| Pin I/O digitali | 14 |
| Pin I/O digitali PWM | 6 |
| Pin di ingresso analogico | 6 |
| Corrente continua per pin I/O | 20mA |
| Corrente continua per pin 3,3V | 50mA |
| Memoria flash | 32KB di cui 0.5KB utilizzati come bootloader |
| SRAM | 2KB |
| EEPROM | 1KB |
| Velocità di clock | 16M[Hz] |

**Tabella 2.2.2 Caratteristiche principali della scheda Arduino Due**

| | |
|---|---|
| Microcontrollore | AT91SAM3X8E |
| Tensione di esercizio | 3,3V |
| Tensione di ingresso | 7-12V |
| Tensione di ingresso (limite) | 6-16V |
| Pin I/O digitali | 54 |
| Pin di uscita analogici | 2 (DAC) |
| Pin di ingresso analogico | 12 |
| Corrente di uscita continua per pin I/O | 130mA |
| Corrente continua per pin 3,3V | 800mA |
| Corrente continua per pin 5V | 800mA |
| Memoria flash | 512kB tutti disponibili per applicazioni utente |
| SRAM | 96kB |
| Velocità di clock | 84MHz |

*Arduino Leonardo*: è una scheda a microcontrollore basata su ATmega32u4. Dispone di 20 pin di input/output digitali (di cui 7 possono essere utilizzati come uscite PWM e 12 come ingressi analogici), un oscillatore a cristalli 16 MHz, una connessione micro USB, un jack di alimentazione, interfaccia ICSP e un pulsante di reset. Contiene tutto il necessario per supportare il microcontrollore, è sufficiente collegarlo a un computer con un cavo USB o alimentarlo con un adattatore AC-to-DC o una batteria. Arduino Leonardo differisce da tutte le

schede precedenti in quanto l'ATmega32u4 ha una comunicazione USB integrata, che permette di non utilizzare un processore secondario. Ciò consente a Leonardo di apparire su un computer collegato come un mouse e una tastiera, oltre a possedere una porta seriale/COM virtuale (CDC). Le principali caratteristiche sono elencate nella tabella 2.2.3.

**Tabella 2.2.3 Caratteristiche principali di Arduino Leonardo**

| | |
|---|---|
| Microcontrollore | ATMEGA32U4 |
| Tensione di esercizio | 5V |
| Tensione di ingresso | 7-12V |
| Tensione di ingresso (limite) | 6-20V |
| Pin I/O digitali | 20 |
| Pin I/O digitali PWM | 7 |
| Pin di ingresso analogico | 12 |
| Corrente di uscita continua per pin I/O | 40mA |
| Corrente continua per pin 3,3V | 50mA |
| Memoria flash | 32kB di cui 4kB come bootloader |
| SRAM | 2,5kB |
| EEPROM | 1KB |
| Velocità di clock | 16MHz |

*Arduino Nano*: è una scheda di dimensioni ridotte adatta al montaggio su breadboard basata su ATmega328. Ha più o meno la stessa funzionalità dell'Arduino Duemila nove, ma in un pacchetto diverso; manca solo un jack di alimentazione di corrente continua e funziona con un cavo USB Mini-B invece di uno standard. Arduino Nano può essere alimentato tramite connessione USB Mini-B, un alimentatore esterno non regolato 6-20 V (pin 30) o un alimentatore esterno regolato a 5 V (pin 27). La fonte di alimentazione viene automaticamente selezionata sulla sorgente di tensione più alta. Le caratteristiche principali sono riportate nella tabella 2.2.4.

**Tabella 2.2.4 Caratteristiche della scheda Arduino Nano**

| | |
|---|---|
| Microcontrollore | ATmega328 |
| Tensione di esercizio | 5V |
| Architettura | AVR |
| Tensione di ingresso | 7-12V |
| Pin I/O digitali | 22 |
| Pin I/O digitali PWM | 6 |
| Pin di ingresso analogico | 8 |
| Corrente di uscita continua per pin I/O | 40mA |
| Memoria flash | 32KB di cui 2KB come bootloader |
| SRAM | 2KB |
| EEPROM | 1KB |
| Velocità di clock | 16M[Hz] |

**Tabella 2.2.5. Caratteristiche principali della scheda Arduino Mega**

| | |
|---|---|
| Microcontrollore | ATmega2560 |
| Tensione di esercizio | 5V |
| Tensione di ingresso (limite) | 6-20V |
| Tensione di ingresso consigliata | 7-12V |
| Pin I/O digitali | 54 |
| Pin I/O digitali PWM | 15 |
| Pin di ingresso analogico | 16 |
| Corrente di uscita continua per pin I/O | 20mA |
| Corrente continua per pin 3,3V | 50mA |
| Memoria flash | 256kB di cui 8kB come bootloader |
| SRAM | 8kB |
| EEPROM | 4kB |
| Velocità di clock | 16MHz |

*Arduino Mega 2560*: E' la scheda pensata per progetti più complessi, ha uno spazio più ampio per gli sketch ed è consigliata per stampanti 3D e progetti di robotica. Basata su ATmega2560, la scheda è dotata di 54 pin digitali di input/output, di cui 15 per PWM, 16 ingressi analogici, 4 porte seriali hardware, un cristallo oscillatore 16 MHz, una connessione USB, un jack di alimentazione, un'interfaccia ICSP, e un pulsante di reset. Contiene tutto il necessario per supportare il microcontrollore; è sufficiente collegarlo a un computer con un cavo USB o alimentarlo con un adattatore AC-to-DC o una batteria. Inoltre, è dotato di un poli-fusibile ripristinabile che protegge le porte USB del computer nel caso si verifichino corto-circuiti sulla scheda. Sebbene la maggior parte dei computer forniscano la propria protezione interna, il fusibile fornisce un ulteriore livello di protezione: se si applicano più di 500 mA alla porta USB, il fusibile interromperà automaticamente la connessione. Le caratteristiche principali sono riportate nella tabella 2.2.5.
Le diverse schede hanno potenzialità e costi diversi. Dal punto di vista della potenza di calcolo, quantità di memoria, disponibilità di input/output, i sistemi più complessi (e anche i più costosi), Arduino Due e Mega, presentano le caratteristiche più adatte al nostro progetto. D'altro canto l'idea che ogni sistema controlli un numero limitato di sensori acustici (quelli presenti in una stanza), ci ha fatto muovere verso una scelta diversa, minimale ed economica, ossia l'uso della piattaforma base Arduino Uno. Di conseguenza questa è stata utilizzata per lo sviluppo del sensore smart, sia in termini di realizzazione hardware a partire dalla capsula microfonica, sia in termini di algoritmi per la localizzazione delle sorgenti.
In fase di test sono risultate evidenti sia le limitazioni relative alla memoria (che impediva di acquisire un numero elevato di campioni) sia quelle relative al tempo di acquisizione del convertitore analogico digitale (ADC). In ogni caso, il compromesso tra semplicità d'uso, robustezza e costi ridotti si è dimostrato effettivo. Inoltre è risultato utilissimo, in fase di sviluppo, il poter usare Arduino Uno come periferica di acquisizione comandata direttamente da PC, tramite codice Matlab. Questo ha infatti permesso di velocizzare la scrittura delle routine di acquisizione e localizzazione delle sorgenti, attraverso parte del codice scritto in linguaggio C per Arduino e parte del codice scritto in Matlab.
Nel prossimo paragrafo 2.3, la descrizione dell'intero sistema in azione verrà illustrata al fine di renderne più chiara la comprensione, assieme alla presentazione dei test. Qui di seguito continuiamo a descrivere gli aspetti hardware più interessanti e la schematizzazione complessiva del sistema, tenendo conto che alcuni aspetti relativi alle soluzioni implementative sono comunque rimandate al prossimo paragrafo. Infine va ricordata l'importanza della disponibilità di manuali [8-10] e di un'elevatissima quantità di codice ed applicazioni e risorse di vario genere disponibili su appositi forum (https://forum.arduino.cc/index.php).

### 2.2.1 Schematizzazione del sistema del sensore acustico con capsula microfonica e Arduino Uno.

Per comprendere meglio come è strutturato il sistema basato su capsula microfonica e Arduino Uno, è necessario fornire qualche dettaglio in più sulla scheda Arduino Uno in termini di input/output e capacità di programmazione. Arduino Uno è sicuramente una delle schede più diffuse e utilizzate in tutto il mondo, anche grazie all'eredità di codice inizialmente sviluppato per i processori ATmel a partire dalla fine degli anni 90. Essa è infatti basata sul microcontrollore ATmega328, un microcontrollore a 8 bit ad architettura RISC sviluppato da ATmel (in seguito acquisita dal diretto rivale Microchip). Dispone di 14 pin digitali di ingresso/uscita, sei dei quali possono essere usati come output PWM (Pulse-width modulation), sei ingressi analogici, un risonatore ceramico da 16 MHz, un connettore USB, un jack di alimentazione, un header ICSP, ed un pulsante di reset. Una volta collegato ad un computer, ed installato l'ambiente di sviluppo, è possibile direttamente programmare la scheda. Di seguito vengono descritti con maggior dettaglio alcune parti della scheda e nella figura 2.2.6 è descritta la piedinatura della scheda Arduino Uno.

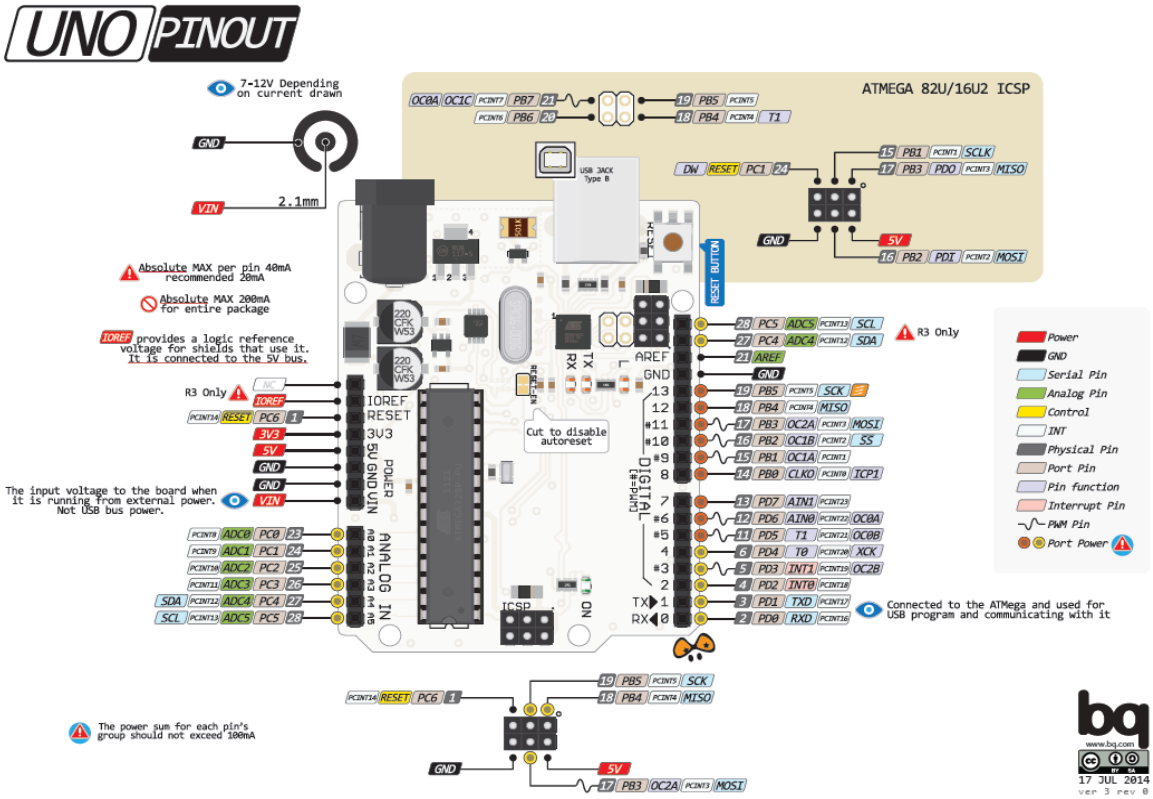


**Figura 2.2.6 - Piedinatura funzionale della scheda Arduino Uno**

La memoria di questo microcontrollore è limitata (32KB ISP flash memory, 1kB EEPROM, 2kB SRAM, 32 registri general purpose) e questo ne rende necessario un utilizzo oculato. In particolare, l'attenzione va posta nei 2kB di SRAM, essendo questa usata per la memorizzazione dei dati acquisiti. L'utilizzo alternativo della EEPROM infatti presenta problematiche di lentezza di accesso che limiterebbe nel complesso le prestazioni della applicazione. Le porte general purpose possono essere utilizzate come porte digitali o analogiche, che a loro volta potrebbero servire come logica di controllo per diverse funzionalità, come quelle aggiunte da uno shield (i.e. una scheda di espansione) specifico. A proposito di shield, è bene ricordare che diverse aziende (ad esempio la Sparkfun, https://www.sparkfun.com/categories/103) hanno sviluppato molteplici shield per Arduino Uno, in grado di aggiungere numerose funzionalità. I timer, anche detti contatori, sono componenti hardware preinstallati sulla scheda, e possono essere utilizzati per misurare i tempi di un evento. Arduino Uno ha tre timer interni chiamati timer0, timer1, e timer2. Il timer1 ha una risoluzione di 16 bit, il che significa che è possibile contare un numero di valori pari a 65536, mentre gli altri due timer hanno una risoluzione di 8 bit e quindi 256 possibili valori. La possibilità di interrupt permette di spezzare il normale flusso di istruzioni sequenziali a seguito di un evento prestabilito. La procedura si chiama ISR, acronimo per Interrupt Service

Routine. Una volta che le istruzioni dedicate all'interrupt sono terminate, il programma continua nel punto esatto in cui era stato interrotto. La porta seriale SPI, acronimo per Serial Peripheral Interface è utile per la comunicazione tra il microcontrollore e altre periferiche vicine ad Arduino, ed è spesso utilizzata per far comunicare due microcontrollori. Il timer watchdog è utilizzato per controllare che Arduino non cada in blocco o in cicli infiniti, dove in questi casi è utile, se non indispensabile, resettare la scheda. Arduino UNO può essere alimentata via USB, o da un'alimentazione esterna. La sorgente di alimentazione viene selezionata automaticamente dalla scheda. L'alimentazione esterna può provenire sia da un adattatore AC/DC che da una batteria. L'adattatore può essere collegato ad Arduino tramite un jack da 2.1mm. La batteria può essere collegata ai pin GND e VIN sul connettore di alimentazione della scheda. La tensione di alimentazione esterna deve essere compresa tra i 6 ed i 20 V, anche se le tensioni raccomandate sono dai 7 ai 12 V. È importante notare che i piedini dell'alimentazione sono disponibili tramite il connettore power (vedi figura 2.2.7):
− VIN: restituisce la tensione applicata dall'alimentatore al jack e può essere usato per alimentare altri circuiti che dispongano già di un regolatore di tensione (ad esempio gli Shield applicati al modulo).
− 5V: fornisce i 5 volt prelevati dall'uscita del regolatore interno ed è utile per alimentare altri circuiti compatibili con i 5 volt.
− 3.3V: fornisce i 3,3 volt ricavati dal regolatore corrispondente e consente di alimentare altri circuiti compatibili con tensioni di 3,3 volt (la massima corrente prelevabile è di 150 mA).
− GND: è il contatto di massa (Ground).
− RESET: portando questa linea a livello basso permette di resettare il microcontrollore.
− IOREF: consente agli Shield di adattarsi alla tensione fornita dalla scheda.

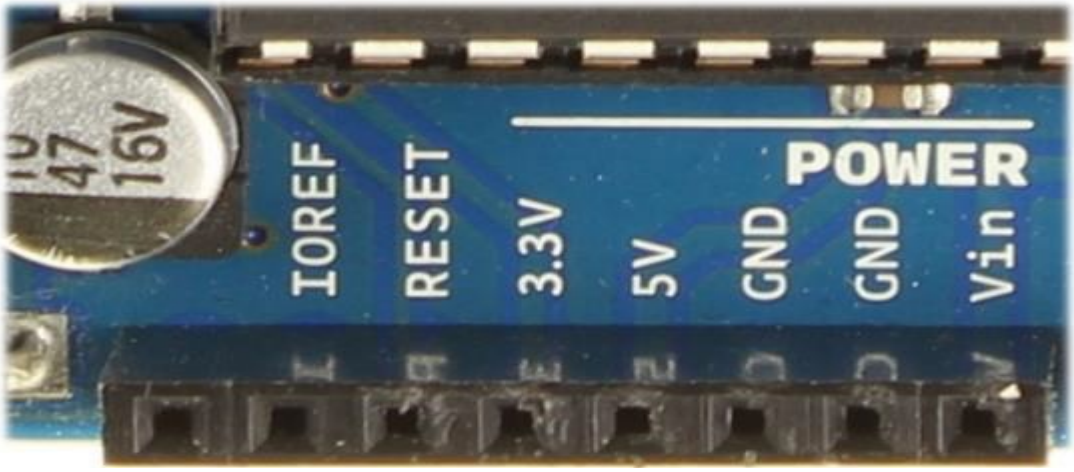


**Figura 2.2.7 – Connettore power**

Inoltre ciascuno dei 14 pin digitali presenti sulla Arduino Uno, e mostrati in figura 2.2.8, possono essere configurati in modalità input oppure output. La tensione di uscita di ogni output è di 5 volt, ed ogni pin è in grado di lavorare con un massimo di 40 mA. Ogni pin è dotato di una resistenza pull-up del valore di 20-50 kΩ. Alcuni pin hanno anche comportamenti specifici:
− Pin 0 (RX) e 1 (TX): possono essere utilizzati per la comunicazione seriale.
− Pin 2 e 3: possono essere configurati come trigger per eventi esterni, come ad esempio il rilevamento di un fronte di salita o di discesa di un segnale in ingresso.
− Pin 3, 5, 6, 9, 10 e 11: possono essere configurati via software per generare segnali PWM con risoluzione di 8bit.
− Pin 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK): possono essere programmati per realizzare una comunicazione SPI.
− Pin 13: è connesso a un LED interno alla scheda, risulta utile in situazioni di debug.
− GND: è il contatto di massa (Ground).
− AREF: Tensione di riferimento per gli ingressi analogici. Utilizzato con la funzione analogReference().


**Figura 2.2.8– Connettore digitale**

Inoltre sono presenti sei ingressi analogici etichettati da A0 ad A5 (vedi figura 2.2.9), ognuno dei quali può essere usato come ADC con 10 bit di risoluzione e quindi 1024 valori differenti. Per impostazione predefinita possono misurare una tensione di 5 volt riferita a massa, anche se è possibile cambiare l'estremità superiore del loro intervallo utilizzando il pin AREF e la funzione analogReference(). Alcuni piedini hanno comportamenti specifici:

− Pin A4 (SDA) e A5 (SCL): permettono di realizzare una comunicazione nello standard I2C a due fili.

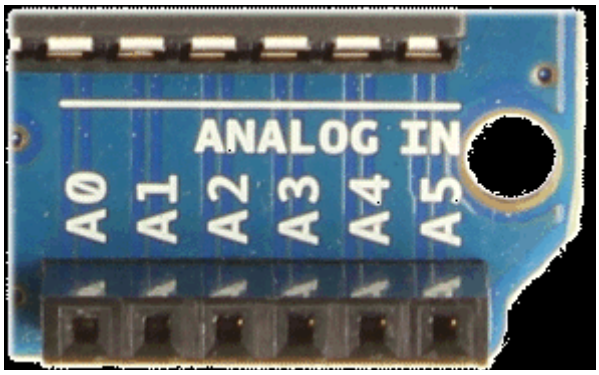


**Figura 2.2.9 – Connettore analogico**

Una caratteristica estremamente importante è costituita dall'ambiente di sviluppo integrato (IDE) per Arduino. Esso è scaricabile gratuitamente dal sito ufficiale. L'editor si presenta in veste minimale, infatti abbiamo a disposizione solamente un editor di testo in cui scrivere il codice, un'area in cui saranno visualizzati gli eventuali errori in fase di compilazione, la possibilità di aprire un monitor seriale con l'Arduino connesso via USB al computer, e poche altre finestre.

Per scrivere un programma, che in gergo si chiama sketch, si utilizza l'editor di testo messo a disposizione dall'IDE. L'estensione di un file di codice Arduino è ".ino". Gli unici controlli di testo che abbiamo a diposizione sono i classici copia/incolla, cerca/sostituisci. L'area dei messaggi sarà utile solo a tempo di compilazione in quanto, se commettiamo un errore di sintassi o semantica mentre scriviamo il codice, non saremo in nessun modo avvisati fintanto che il codice non viene compilato. Arduino mette a disposizione una serie di librerie di base con le quali è possibile sviluppare dei programmi completi. Inoltre sono disponibili anche diversi esempi di progetto. Tutti i costrutti, le strutture, le variabili, e le funzioni di base e le librerie standard, sono consultabili nel sito ufficiale, nell'apposita pagina di riferimento. Se il nostro Arduino è correttamente collegato al nostro pc, nella parte bassa dell'editor sarà notificato in quale porta USB è stato collegato. Tra le varie opzioni dell'editor è possibile importare le librerie di terze parti che risultano molto spesso utili. Come in molti altri linguaggi di programmazione sarà necessario includere l'header di libreria all'interno del codice.

Ogni sketch di Arduino segue sempre le stesse regole. Uno sketch può essere diviso in due parti. Queste due parti sono i punti di ingresso di qualsiasi programma Arduino. Il primo punto è chiamato fase di setup. Nel setup vengono inizializzate tutte le risorse di Arduino, per esempio vengono scelti i pin da considerare come ingressi e quali come uscite, i servizi SD ed Ethernet, ecc. Il secondo punto è chiamato fase di loop, e come suggerisce il nome, Arduino ripeterà senza fine tutto il codice che sarà scritto all'interno di

questo blocco, a meno di un evento esterno, come ad esempio un utente che preme il tasto reset. Come quasi tutti i linguaggi di questa tipologia, le zone esterne (prima e dopo) a queste due fasi possono essere utilizzate per l'inclusione di librerie, la definizione di variabili globali e di funzioni. Il diagramma di flusso in figura 2.2.10 mostra il ciclo di esecuzione di uno sketch. La fase di setup avviene una sola volta, all'accensione di Arduino, dopodiché, a meno di fattori esterni, Arduino eseguirà la fase di loop all'infinito.

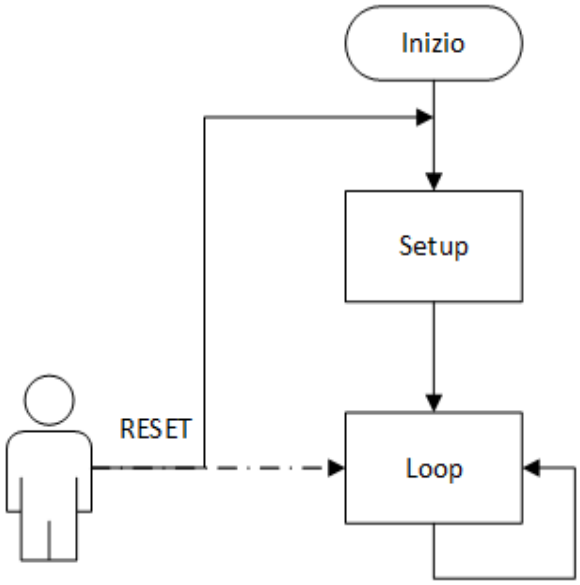


**Figura 2.2.10 - Ciclo di esecuzione Arduino.**

Nel nostro progetto specifico, la scheda di Arduino Uno è stata utilizzata come sistema di acquisizione del segnale audio della capsula microfonica a condensatore. Come è stato indicato nel paragrafo 2.1 la dinamica della capsula microfonica era troppo bassa per garantire prestazioni adeguate per l'acquisizione con pochi bit di ADC di un segnale audio decente. Questo come è stato detto è stato verificato collegando direttamente l'uscita della capsula microfonica al jack del microfono di un notebook e registrando il segnale tramite Matlab. Ulteriori dettagli su questa fase di test saranno indicati nel successivo paragrafo 2.3.

Per migliorare la dinamica è stato realizzato un amplificatore ad emettitore comune, già descritto nel paragrafo 2.1. Questo amplificatore è stato alimentato direttamente tramite Arduino Uno con una tensione di 5V (questa tensione ha permesso anche una corretta polarizzazione della capsula a condensatore, che, come detto in precedenza nel paragrafo 2.1, necessita di un'alimentazione in continua per poter funzionare correttamente. Il nostro amplificatore è stato usato anche come traslatore di livello, impostando il "ground" effettivo del segnale a circa 2.5 V e realizzando un guadagno complessivo di circa 20/25 volte. Avendo a disposizione ben 6 ingressi analogici, sarebbe teoricamente possibile con un'unica scheda Arduino Uno acquisire i segnali di 6 microfoni costruiti come in precedenza. In realtà è ben noto che in ogni microcontrollore tutte le operazioni, ivi comprese quelle di conversione da analogico a digitale, avvengono in maniera seriale, per cui non è praticamente possibile acquisire più di 3 o 4 segnali acustici (il tempo minimo di conversione dell' ADC è 13us). Inoltre, anche la necessità di memorizzare i dati in una memoria di dimensioni ridotte è un altro fattore limitante.

Nei test effettuati, abbiamo sempre considerato contemporaneamente attivi almeno due microfoni e quindi ci siamo preoccupati di verificare la correttezza del processo in questa condizione, scalando poi i risultati ai casi con tre e quattro microfoni. Lo schema del sistema è quindi quello rappresentato in figura 2.2.11. La discussione sui test è rimandata al successivo paragrafo 2.3, dove sono descritti oltre che gli algoritmi di acquisizione, quelli implementati per la localizzazione delle sorgenti.

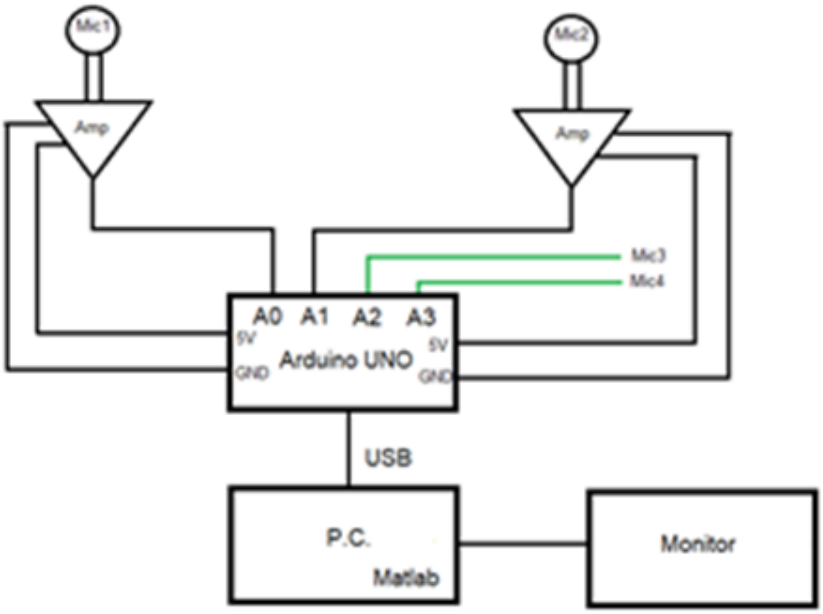


**Figura 2.2.11 Schematizzazione del sistema di acquisizione e trattamento del segnale con connessione di Arduino ad un PC.**

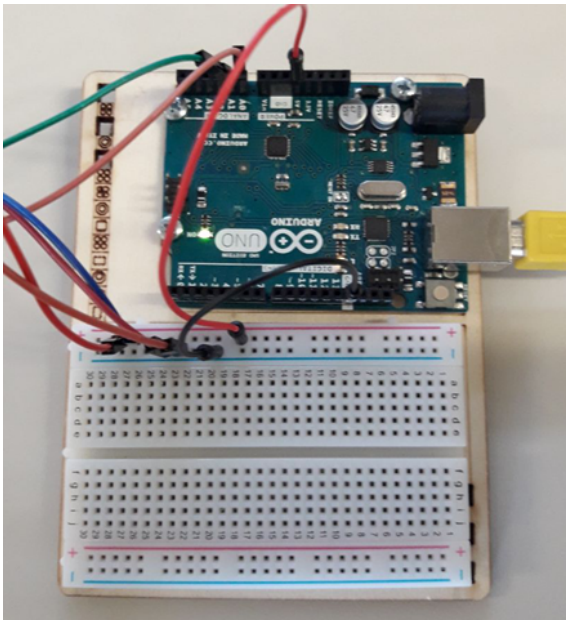

**Figura 2.2.12 Scheda Arduino Uno e collegamento alla breadboard per la connessione dei due microfoni**

**Figura 2.2.13 Schema complessivo del sistema di acquisizione. In primo piano il microfono connesso con lo stadio di amplificazione. Sul tavolo vicino al notebook la scheda Arduino connessa alla breadboard di figura 2.2.12**

### 2.2.2 Possibilità di interconnessione ed altre soluzioni hardware di interesse.

Come detto in precedenza tali schede presentano il vantaggio dell'alta interfacciabilità tramite schede shield disponibili sul mercato. Ad esempio la Sparkfun realizza schede shield per aggiungere la possibilità di connettersi WiFi o tramite XBee, ma anche la casa madre produce diverse schede per la connessione WiFi. Chiaramente il costo del dispositivo nel momento in cui si debba pensare di dover aggiungere funzionalità addizionali e shield per l'interconnessione aumenta. Del resto, nel sistema smart home, potrebbe essere già presente una scheda arduino per il controllo di altri sensori, ed in questo caso è sufficiente considerare solo l'uso della capsula microfonica con amplificatore e modificare il codice di Arduino per sfruttare una funzionalità aggiuntiva.

Si tenga conto che tra l'altro molto spesso le operazioni di monitoraggio dei sensori (ad esempio di temperatura, pressione dell'aria, e via dicendo) sono molto lente e richiedono poche operazioni in intervalli di tempo dell'ordine dei secondi, ossia perfettamente compatibili con acquisizioni del segnale acustico, che invece avvengono continuamente (a parte cioè le pause per l'interrogazione degli altri sensori). Chiaramente altre soluzioni non basate su Arduino sono possibili. L'uso ad esempio della scheda DISCOVERY della STMicroelectronics, risulterebbe essere altamente promettente, anche per le migliori caratteristiche del microcontrollore presente, paragonabile se non superiore a quello di Arduino Due. Purtroppo però mentre la politica di Arduino è stata quella dello sviluppo e della condivisione del codice, lo stesso non può dirsi per le schede delle altre case. Di conseguenza, la facilità di messa a punto e di integrazione del sistema Arduino (in caso di necessità di maggiori risorse si può passare ad usare il Due o il Mega) lo rendono in ogni caso la nostra prima scelta. Per completare il panorama di possibilità è doveroso ricordare la possibilità d'uso di schede a microcontrollore con WiFi integrato, che recentemente si stanno affacciando con successo nel mercato non cinese dell'elettronica. In particolare, il modulo ESP-12E è la variante più comune dei moduli basati sul microcontrollore ESP8266, un chip Wi-Fi a basso costo con supporto completo a TCP/IP e funzionalità da microcontrollore prodotto dall'azienda cinese di Shanghai Espressif Systems. Le dimensioni del modulo sono 24x16mm e può essere acquistato sia da solo, sia in combinazione con una scheda di sviluppo che ne facilita la programmazione e la prototipazione (vedi figura 2.2.13). Le caratteristiche tecniche del microcontrollore. CPU: Processore RISC a 80MHz modello L106 produttore Tensilica; ADC: 10 bit mappato su singolo pin; Memoria: 32kB RAM + 4MB Flash ; Interfacce: UART, I2C (software, più lenta), SPI, I2S; WiFi: stack completo, anche a livello fisico, compatibile con WPA-WPA2, IEEE 802.11 b/g/n; Consumi: In standby, dichiarati inferiori a 1mW. Costo della scheda circa 10 euro.

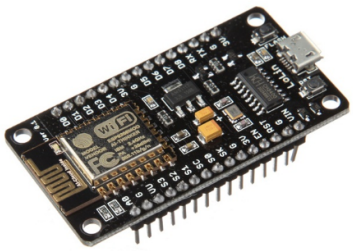

**Figura 2.2.13 Modulo ESP8266**

Purtroppo, le scarse informazioni sul prodotto e la necessità di fornitura sul mercato cinese hanno reso allo stato attuale impossibile testare questa tipologia di hardware, che, però, potrebbe rivelarsi estremamente utile per lo sviluppo effettivo di un sensore acustico smart ed allo stesso tempo estremamente low cost. In questo caso, inoltre, si potrebbe valutare l'uso di sensori distribuiti con poca intelligenza, ma grande capacità di acquisizione e comunicazione.

## 2.3 Studio e sviluppo di algoritmi per la localizzazioni di sorgenti sonore

In questa fase, sono stati studiati e messi a punto algoritmi per la rilevazioni di sorgenti sonore e la loro localizzazione. In particolare sono stati presi in considerazione sia algoritmi di acquisizione in grado di funzionare con un solo sensore, e che quindi possono essere implementati eventualmente anche in hardware locale, sia algoritmi che incrociano i dati di più sensori, e che quindi richiedono la loro implementazione in un sistema di aggregazione o in un hardware locale con caratteristiche opportune (come abbiamo visto in precedenza nel paragrafo 2.2 Arduino Uno era in teoria in grado di gestire fino a 6 sensori grazie ai suoi sei canali ADC). Viene tenuta in considerazione anche la versatilità delle diverse soluzioni in applicazioni di natura diversa, quale la segnalazione eventi, la localizzazione e l'eventuale conteggio delle persone che occupano un locale.

### 2.3.1 Test sui sistemi di acquisizione da capsula microfonica

Questa parte del lavoro è iniziata con lo studio dei segnali acquisiti dalla capsula microfonica selezionata, mediante l'uso di script MATLAB per l'elaborazione del segnale acquisito. In particolare, si è provato a riprodurre, con precisione variabile in termini di campionamento e quantizzazione del segnale, l'acquisizione degli avvenimenti acustici in una stanza, tramite l'istruzione di "audiorecorder" di MATLAB.
La sintassi di questa istruzione è:

```
recorder = audiorecorder(Fs,bit,nchannels,ID)
recordblocking(recorder,seconds)
```

Tramite questi comandi viene creato nella memoria di MATLAB un l'oggetto "recorder", all'interno del quale sarà presente una registrazione con una frequenza di campionamento Fs, un numero di bit per campione (8,16 o 24), un numero per il canale mono (1) o stereo (2), e il Device Identifier, che restituisce l'ID del "device" che si sta utilizzando. L'istruzione "recordblocking" invece, specifica che l'oggetto "recorder" avrà una durata di alcuni secondi (la variabile "seconds").
Per verificare le caratteristiche dei sensori utilizzati si sono svolte diverse acquisizioni, in vari giorni. Le acquisizioni del sensore utilizzato sono state confrontate con quelle del microfono interno di un notebook per verificarne la qualità. Si è lavorato alternativamente con 4 kHz o 8 kHz come frequenza di campionamento e con 8 o 16 bit per la quantizzazione del segnale. Le registrazioni sono durate fino a 5 o 10 minuti per verificare anche la quantità di dati gestiti dal processo di acquisizione, nell'ottica di un loro utilizzo in una post elaborazione. Il confronto tra i due microfoni (capsula microfonica connessa tramite jack al notebook e microfono interno del notebook) ha messo in evidenza la necessità di amplificazione della capsula microfonica (è possibile settare una amplificazione software dal pannello di controllo di gestione, ma questo dovrà poi corrispondere ad un guadagno hardware) in maniera tale da garantire un buona qualità con un numero contenuto di bit di quantizzazione e frequenza di campionamento (questa problematica non era comunque esclusiva della capsula microfonica, ma riguardava anche il microfono interno, sebbene in modalità leggermente inferiore, grazie al migliore adattamento di impedenza realizzato internamente).

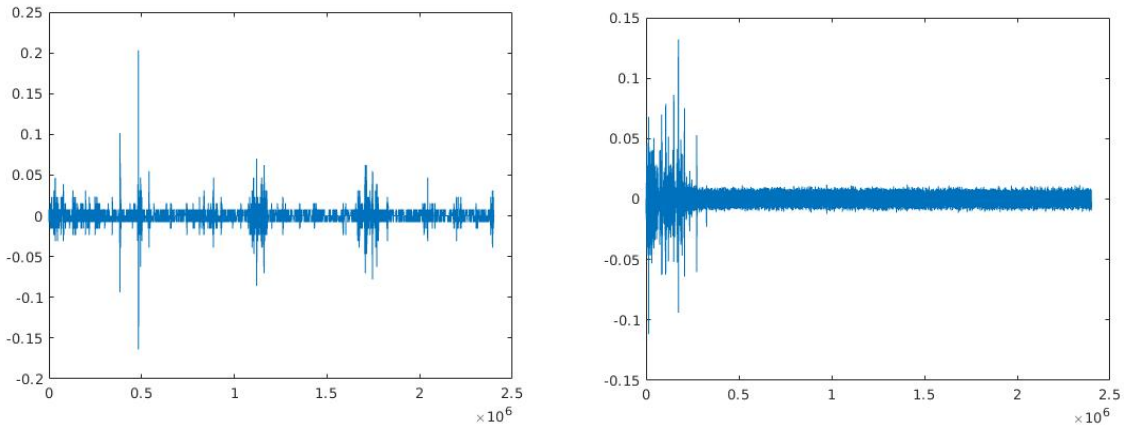


**Fig. 2.3.1 Confronto tra due casi con frequenze di campionamento di 8 kHz e 16 bit (destra) e 4 kHz e 8 bit (sinistra)**

Dalle varie prove effettuate si è ricavata anche la quantità di dati da gestire per un dato intervallo di analisi, che vanno da circa 30 MB per frequenze di campionamento di 8 kHz e 16 bit (2^16 livelli di quantizzazione) per 10 minuti, a meno di 10 MB per frequenze di campionamento di 4 kHz e 8 bit (256 livelli di quantizzazione) per 5 minuti. Un confronto in termini grafici è visibile nella fig. 2.3.1. Come è evidente dalla figura a sinistra i livelli di quantizzazione appaiono netti, e la dinamica del microfono è comunque bassa (verificata anche tramite oscilloscopio in un range del mV). Le prove hanno invece mostrato dal punto di vista dell'ascolto che una frequenza di 4kHz garantiva una sufficiente intellegibilità della voce umana e rendeva possibile ascoltare i dialoghi senza particolari sforzi.
Alla luce del fatto che la maggior parte dei convertitori analogico digitali presenti nei microcontrollori ha una risoluzione di 10-12 bit si è ritenuto necessario amplificare il segnale in modo da garantire una adeguata quantizzazione con un numero ridotto di bit. Un buon compromesso utilizzato nelle successive implementazioni su Arduino è stato 4 kHz con 10 bit (questo è quanto riesce a fare la scheda Arduino Uno), previa amplificazione del segnale della capsula microfonica di un fattore circa 20 e traslazione del livello di tensione intorno a 2.0/2.5V, in modo da permettere un accoppiamento corretto tra la capsula e l'ADC di Arduino in DC senza richiedere l'aggiunta di ulteriori componenti esterni. A tal proposito è importante ricordare il ruolo di traslatore di livello del sistema ad emettitore comune, che si rende necessario per il corretto funzionamento del sistema di acquisizione.

### 2.3.2 Algoritmi per la localizzazione della sorgente acustica

Il problema di localizzazione di una sorgente di rumore in una stanza è stato approcciato e risolto in forma analitica. Per la descrizione dell'algoritmo supponiamo che il sistema sia costituito da due microfoni disposti su di un piano xOy con coordinate note e la sorgente S di coordinate (x ,y) incognite. Supponiamo di disporre i due microfoni su uno dei lati di una stanza, e consideriamo la distanza tra i due microfoni nota e pari a *d*. Per rendere più semplice la trattazione matematica si è supposto che il sistema di riferimento abbia origine nel punto medio del segmento che congiunge i due microfoni ed abbia asse x coincidente con la parete in cui sono posti i microfoni ed asse y ortogonale. Per rendere più semplice la trattazione per adesso trascuriamo la coordinata z che rappresenta l'altezza dal pavimento e quindi consideriamo un caso semplificato bidimensionale. In base a quanto detto i due microfoni si troveranno in corrispondenza dei punti M1 ed M2 di seguito indicati. La schematizzazione del sistema di riferimento è rappresentata in figura

M1 =$\left(-\frac{d}{2}, 0\right)$

M2=$\left(\frac{d}{2}, 0\right)$

S=$(x, y)$

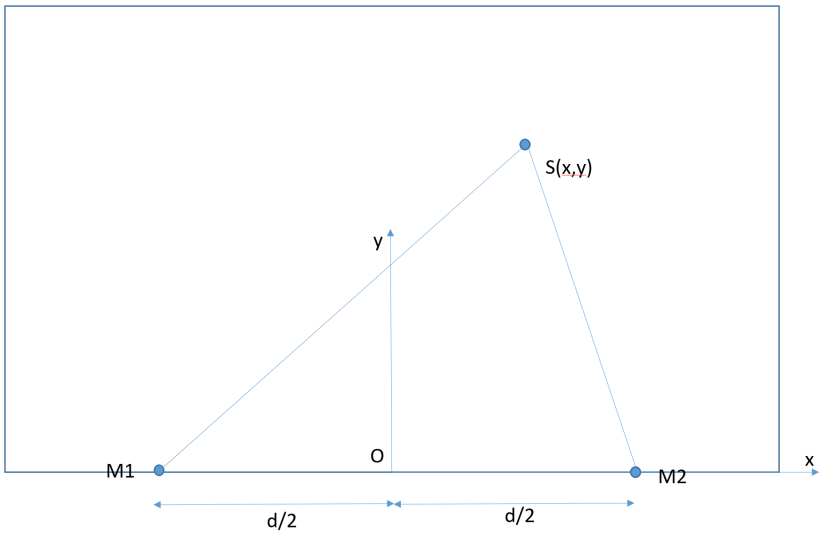


**Fig. 2.3.2 Rappresentazione dell'ambiente e del sistema di riferimento usato**

Per evidenziare la geometria coinvolta sono state tracciate le rette congiungenti il microfono M1 e il microfono M2 con la sorgente S. Nell'esempio considerato la sorgente S si trova più vicina al microfono M2 e quindi il segnale acustico emesso da essa arriverà prima al microfono M2 e dopo un certo ritardo $\Delta t$ giungerà anche al microfono M1. Il ritardo $\Delta t$ è legato alla velocità del suono e alla differenza tra la distanza della sorgente dai due microfoni (vedi figura 2.3.3)

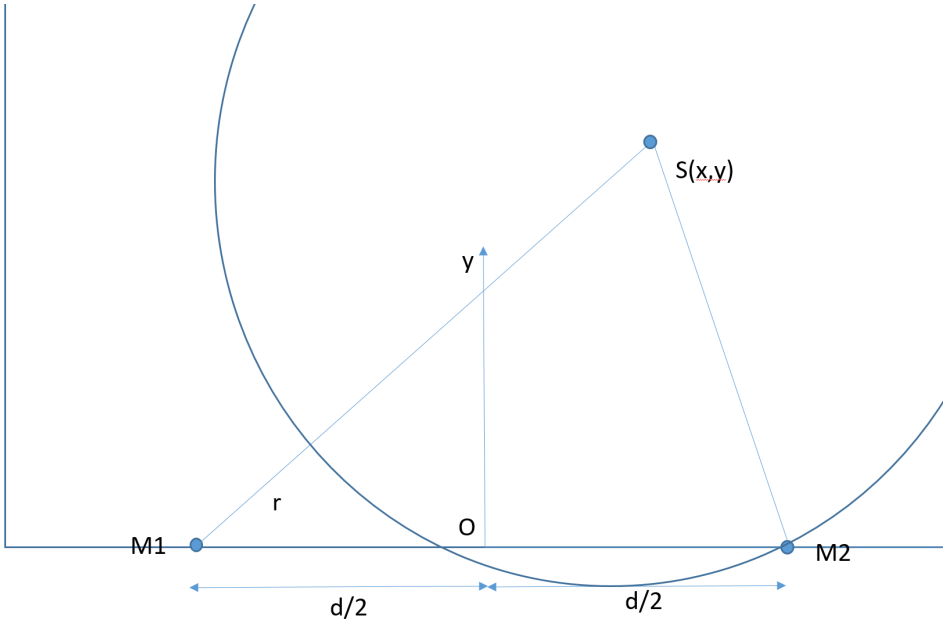


**Figura 2.3.3 Distanza r in termini di differenza della distanza tra la sorgente e i due microfoni.**

Considerando che la velocità di propagazione del suono nell'aria è $340\frac{m}{s}$, si può calcolare la distanza associata al tempo di ritardo che sarà necessaria per svolgere poi il problema analitico:

$$r = \Delta t * v_s$$

Quindi se si suppone noto il ritardo attraverso una qualche misura sul segnale acustico registrato è possibile valutare r e quindi le posizioni ammissibili con l'algoritmo indicato di seguito. Per il ritardo si devono

distinguere due casi: $\Delta t < 0$ *e* $\Delta t > 0$, ossia la sorgente è più vicina al microfono M1 o M2. Tratteremo solo il caso maggiore di zero, in quanto l'unica differenza riguarda la posizione rispetto l'asse y (ossia in un caso troveremo x > 0, nell'altro x < 0).
Ricordando la generica formula per calcolare la distanza tra due punti di cui si conoscono le coordinate troviamo:

$$M_1S = \sqrt{(x+\frac{d}{2})^2 + (y)^2}$$

$$M_2S = \sqrt{(x-\frac{d}{2})^2 + (y)^2}$$

E quindi considerando la differenza tra le due distanze pari ad r troviamo

$$\sqrt{(x+\frac{d}{2})^2 + (y)^2} - r = \sqrt{(x-\frac{d}{2})^2 + (y)^2}$$

$$(x+\frac{d}{2})^2 + (y)^2 + r^2 - 2r\sqrt{\left(x+\frac{d}{2}\right)^2 + (y)^2} = (x-\frac{d}{2})^2 + (y)^2$$

$$\frac{d^2}{4} + x^2 + dx + r^2 - 2r\sqrt{\left(x+\frac{d}{2}\right)^2 + (y)^2} = \frac{d^2}{4} + x^2 - dx$$

$$2dx + r^2 = 2r\sqrt{\left(x+\frac{d}{2}\right)^2 + (y)^2}$$

$$4d^2x^2 + 4dr^2x + r^4 = 4r^2(x+\frac{d}{2})^2 + 4r^2y^2$$

$$4d^2x^2 + 4dr^2x + r^4 = 4r^2x^2 + 4dr^2x + 4r^2\frac{d^2}{4} + 4r^2y^2$$

$$4(d^2 - r^2)x^2 - 4r^2y^2 = r^2(d^2 - r^2)$$

$$\frac{4x^2}{r^2} - \frac{4y^2}{(d^2 - r^2)} = 1$$

Come si evince dalla formula canonica

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$$

si otterrà un'iperbole con equazione del tipo:

$$\frac{x^2}{\frac{r^2}{4}} - \frac{y^2}{\frac{d^2 - r^2}{4}} = 1$$

Dove $a = \frac{r}{2}$ $b = \sqrt{\frac{d^2-r^2}{4}} = \frac{\sqrt{d^2-r^2}}{2}$

Gli asintoti hanno equazione: $y = \pm\frac{b}{a}x$, mentre il fuoco coinciderà con il microfono $M_2$. La sorgente quindi si potrà trovare in un punto qualsiasi dell'iperbole descritta, e per la localizzazione esatta sarà necessario incrociare l'informazione ricavata con il dato proveniente da un ulteriore microfono.
Inoltre notiamo che qualora il tempo di ritardo fosse nullo non sarà possibile descrivere alcuna curva in quanto la sorgente sarebbe posizionata (a meno di errori di misura del tempo di ritardo) nella retta passante per l'origine ed ortogonale all'asse x, ossia coinciderà con l'asse delle y (vedi figura 2.3.4). Anche in questo caso l'aggiunta di un terzo microfono permetterebbe di risolvere il problema della localizzazione. In ogni caso bisognerà sempre valutare la presenza di errori di misura dovuti alle limitazioni dell'intero smart system.

**Figura 2.3.4 Caso della sorgente posizionata nel punto medio tra i due sensori:**

L'ultima considerazione da fare riguarda la valutazione del tempo di ritardo. Questo può essere fatto in maniera relativamente semplice ricorrendo alla funzione di cross-correlazione tra i due segnali acquisiti dai due microfoni. Ricordando che la cross-correlazione, rappresenta la misura di similitudine di due segnali come funzione di uno spostamento o traslazione temporale applicata su uno di essi. Considerando due segnali a valori reali $x$ e $y$ che differiscono solamente per una traslazione temporale sull'asse $t$, si può calcolare la correlazione incrociata per mostrare di quanto $y$ debba essere anticipato per renderlo identico ad $x$. La formula essenzialmente trasla il segnale $y$ lungo l'asse $t$, calcolando l'integrale del prodotto per ogni possibile valore dello spostamento. Per due segnali ad energia finita $x$ e $y$ la cross-correlazione è definita come:

$$Rxy \stackrel{\text{def}}{=} \int_{\infty}^{-\infty} x^*(\tau)y(t+\tau)d\tau$$

Poiché il segnale è campionato, l'integrale diverrà una sommatoria che coinvolgerà solo i campioni disponibili. Il massimo della cross correlazione corrisponderà alla posizione in cui i due segnali sono al massimamente correlati, cioè corrisponderà allo shift temporale tra i due segnali.

### 2.3.3 Descrizione del funzionamento software implementato e dei test effettuati

Come già indicato nel paragrafo 2.2 il sistema costituito da Arduino e due sensori a capsula microfonica a condensatore sono stati testati in un ambiente per verificarne limitazioni e funzionamento. L'approccio ha previsto l'uso di Arduino come sensore a bassa intelligenza, che acquisisca i segnali e li trasmetta all'unità di elaborazione (in questo caso un PC), mentre l'implementazione degli algoritmi di localizzazione è stata fatta nell'ambiente di calcolo Matlab [11]. In una prima fase dei test, è stato verificato l'utilizzo di Matlab non solo come unità di processamento, ma anche come unità di comando per Arduino. Infatti è possibile utilizzare delle apposite librerie su Arduino e Matlab tramite le quali la scheda Arduino sia direttamente controllabile da linea di comando Matlab. Questa modalità è stata però abbandonata perché rendeva il processo di campionamento del segnale più lento dei 4 kHz considerati necessari per una buona qualità del segnale acquisito. Di seguito presentiamo in dettaglio sia lo sketch di Arduino Uno (in tabella 2.3.1) sia il codice MATLAB (tabella 2.3.2).

**Tabella 2.3.1.Codice Arduino**

```
const int analogInPin = A0;
//Analog input pin that the potentiometer is attached to
const int analogInPin1 = A1;
//Analog input pin that potentiometer is attached to

int sensorValue[100];  //value read from the pot
int sensorValue1[100];
const int PinOut = 12;
void setup() {
  // Initialize serial communications at 115200 bps:
  Serial.begin(115200);
  pinMode(PinOut,OUTPUT);
}
void loop() {
  if (Serial.available() > 0) {
    //get incoming byte:
    while(Serial.read() != 'S');
    for(int i=0; i<100;i++) {
      digitalWrite(PinOut, HIGH);
      //read the analog in value;
      //delayMicroseconds(125);
      sensorValue[i] = analogRead(analogInPin);
      sensorValue1[i] = analogRead(analogInPin1);
    }
    digitalWrite(PinOut, LOW);
    for(int i=0;i<100;i++) {
      Serial.println(sensorValue[i]);
    }
    delay(100);
    for(int i=0;i<100;i++){
      Serial.println(sensorValue1[i]);
    }}
  //print the results to the Serial Monitor (usato nella fase di verifica
iniziale):
  /*  Serial.print("sensor = ");
   *   Serial.print(sensorValue);
   *   Serial.print("/t  output = ");*/
   delay(1000);

}
```

Per prima cosa si dichiarano due variabili i pin analogici $A_0$ *e* $A_1$ ai quali sono connessi i due microfoni. Si dichiarano due vettori interi chiamati "sensorValue" e "sensorValue1" di lunghezza pari ai 100 campioni che serviranno per memorizzare i dati misurati dai sensori che si vogliono analizzare; ogni casella dell'array avrà un valore compreso tra 0 e 1023 (non faremo la conversione in variabile float per evitare di consumare ulteriormente memoria). Infine viene dichiarata la costante intera "PinOut" che rappresenta il pin numero 12 di Arduino.
Nel codice di "setup", che verrà eseguito una sola volta, si inizializza la comunicazione seriale a 115200 bit per secondo e viene dichiarato il pin di uscita sul quale si vuole erogare corrente, ovvero il pin 12 "PinOut" con modalità OUTPUT.

Nel codice di "loop", che verrà eseguito ciclicamente, si attenderà di ricevere sulla porta seriale il comando di start indicato dalla Stringa "S", che viene inviato dal sistema di gestione del sensore (in questo caso dal PC). Quando giunge questa stringa, Arduino si mette in modalità acquisizione, accende il led per la segnalazione dello stato ed inizia un ciclo "for" con una variabile contatore che rappresenta l'indice dell'array. A questo punto si iniziano a riempire le caselle dei vettori leggendo i valori analogici dei segnali dei microfoni tramite i convertitori ADC. Dopo aver acquisito 100 campioni per ogni microfono ad una frequenza di campionamento di 4 kHz, si spegne il LED e si inviano i dati tramite la porta seriale al sistema centrale (in questo caso il PC dove il codice Matlab si preoccuperà di effettuare gli altri calcoli). Prima di procedere ad una nuova acquisizione si aspettano 100 ms (in maniera opzionale, e solo nella fase di test del codice, i vettori acquisiti vengono stampati sullo schermo seriale dell'IDE).

Chiaramente il codice è facilmente modificabile. È possibile ad esempio impostare il sistema in ascolto, senza acquisire e trasmettere, finché da uno dei microfoni non sia rilevato un segnale che superi una certa soglia. Questo segnale avvia il processo di acquisizione dell'intero vettore necessario per la localizzazione, che poi viene elaborato e trasmesso.
È da notare che la dimensione del vettore necessario per la corretta localizzazione è legata alla dimensione della stanza stessa. Infatti N campioni a 4kHz corrispondono a N*250 µs, che per la velocità del suono ci dà una differenza massima tra le distanze tra la sorgente ed il microfono di 340*N*250e-6 m ossia N*0.085 m, quindi con N= 10 abbiamo 85 cm, con N = 100 abbiamo ben 8.5 m. È bene notare che 8.5 cm è la risoluzione massima che possiamo avere con 4 kHz, e quindi questo è l'errore di cui tener sempre conto in tutte le valutazioni.
In ogni caso, la scelta di N può farsi in ragione delle dimensioni del locale in esame. Per abitazioni civili N=100 è già più del necessario. Del resto, considerando che il vincolo principale di accuratezza è legato al tempo di campionamento, e che per l'acquisizione di 100 campioni, da due sensori, a 4kHz, servono 200*250e-6 secondi, ossia 50 ms, possiamo pensare che il nostro sistema controlli fino a 4 microfoni senza problemi, semplicemente alternando le acquisizioni di una coppia con quelle dell'altra, in modo tale che poi l'algoritmo di localizzazione individui la posizione con una certa precisione tramite l'intersezione.
È quindi possibile realizzare con tre o quattro capsule microfoniche (progettate secondo quanto specificato nel paragrafo 2.1) ed una scheda Arduino per la loro gestione e la trasmissione dei dati acquisiti ad un sistema centrale in cui l'algoritmo di localizzazione viene eseguito, il monitoraggio delle sorgenti acustiche in un ambiente indoor con costi contenuti (complessivamente si può stimare ben sotto i 50 euro). Tale sistema inoltre può verificare anche la presenza di più sorgenti acustiche attive contemporaneamente considerando diversi tempi di arrivo dei segnali ai microfoni (a meno che le due sorgenti non siano vicine a tal punto da non essere distinguibili, e dare più picchi nella funzione di cross correlazione).
Di seguito viene illustrato il codice Matlab implementato, tenendo conto che questo processo può facilmente essere implementato in un codice C su un microcontrollore più potente come ad esempio un Raspberry PI, o su una qualsiasi macchina dedicata alla gestione dei sensori. Non conviene invece usare direttamente Arduino, poiché durante i tempi di calcolo risulterebbe incapace di acquisire nuovi dati dai microfoni.

**Tabella 2.3.2 Codice Matlab**

```
Porta=serial('/dev/ttyACM0','BaudRate',115200,'Terminator','CR','InputBufferSize',4096)
fopen(Porta);
fprintf(Porta,'S');

for i=1:100
    mic1(i)=str2num(fscanf(Porta));
end
for j=1:100
    mic2(j)=str2num(fscanf(Porta));
end
y=xcorr(mic1,mic2);    %%si faccia la cross-correlazione tra i due segnali
[indy,valy]=max(y);
deltat=(indy-100)*2.5e-4; %%si calcoli il tempo di ritardo tra i due segnali
d=2.0; %%distanza tra i microfoni
```

```
r=deltat*340; %%si calcoli la distanza associata al tempo di ritardo
 A=(abs(r)/2);
B=sqrt(d^2-r^2)/2;

if deltat<0
    teta=linspace(0, pi/3, 100);
    Xc1=A./cos(teta);
    Yc1=B*tan(teta);

    plot(Xc1,Yc1,'b*');
    xlabel('asse Xc1');
    ylabel('asse Yc1');
    title('Localizzazione sorgente');
    box off
end

if deltat>0
    beta=linspace(0, pi/3, 100);
    Xc2=-A./cos(beta);
  Yc2=B*tan(beta);

    plot(Xc2,Yc2,'m*');
    xlabel('asse Xc2');
    ylabel('asse Yc2');
    title('Localizzazione sorgente');
    box off
end
```

Per quanto riguarda il codice di MATLAB si procede con i seguenti passi: prima di tutto viene creato l'oggetto seriale "Porta" nel quale sono specificati parametri come l'uscita della USB, la velocità di comunicazione con la piattaforma impostata a 115200ms, che dovranno essere uguali sia per Arduino sia per MATLAB, il Terminator (carattere che specifica di andare a capo una volta finita l'acquisizione di un array), e la dimensione della memoria (Buffer). Questa parte corrisponde alla creazione di un sistema di comunicazione. In questo caso viene usata la porta seriale USB, ma se fosse presente una qualsiasi interfaccia wireless potrebbe essere usata in maniera molto simile per la comunicazione.
Successivamente viene aperto l'oggetto Porta, e viene inviata la stringa "S" che inizializza l'acquisizione. I vettori poi, ritornano a MATLAB dove con due cicli *for* sono convertiti da stringhe a numeri. Da qui si procede direttamente chiamando la funzione di cross-correlazione e la funzione "max()" che indica il valore massimo di un vettore e l'indice della sua posizione. Tramite questo comando si riesce ad individuare la posizione del picco della funzione di cross-correlazione.
Il valore dell'indice verrà sottratto ai 100 campioni analizzati e il risultato sarà moltiplicato per 0.25 ms ottenuti dall'analisi di un singolo campione a 4kHz. Ottenuto il tempo di ritardo indicato nel codice come "deltat" si può calcolare la distanza associata al tempo di ritardo $r$ moltiplicandolo "deltat" per la velocità del suono che corrisponde a $340\,\frac{m}{s}$.
Successivamente, viene trattata la distinzione tra il caso in cui il ritardo è positivo, e il caso in cui il ritardo è negativo. Per la distinzione, viene effettuato un controllo con due istruzioni "if". Nel caso in cui il ritardo sia negativo, si crea una variabile "teta" con la funzione "linspace", che genera un vettore linearmente spaziato compreso tra $0$ e $\frac{\pi}{3}$ con risoluzione pari a 100, che sarà l'ampiezza dell'angolo che consente di tracciare il grafico soluzione: una iperbole. Con riferimento alle coordinate polari, si scrivono ascissa e ordinata della iperbole come:

$$\begin{cases} X_{c1} = \dfrac{a}{\cos\vartheta} \\ Y_{c1} = b\tan\vartheta \end{cases}$$

Con le funzioni "xlabel", "ylabel" e "title" rispettivamente si personalizzano gli assi con delle etichette e si inserisce il titolo del grafico. Con la funzione "plot" si traccia l'iperbole finale. Nei test effettuati è stata

sempre usata un'unica coppia alla volta, e quindi è stato effettuato il tracciamento dell'iperbole. Chiaramente, come detto in precedenza dai dati di due coppie è possibile ricavare due iperboli e quindi un punto di intersezione. Considerando l'errore dell'ordine della decina di cm dovuto al tempo di campionamento possiamo dire che la sorgente starà in un quadrato di 20 cm di lato e centro dato dal punto di intersezione.
Nella figura 2.3.5 di seguito riportata vediamo i grafici di output di Matlab che indica i punti ammissibili per la sorgente, ricavati dalla misurazione dei segnali con due soli microfoni e la sorgente posta in punti diversi della stanza

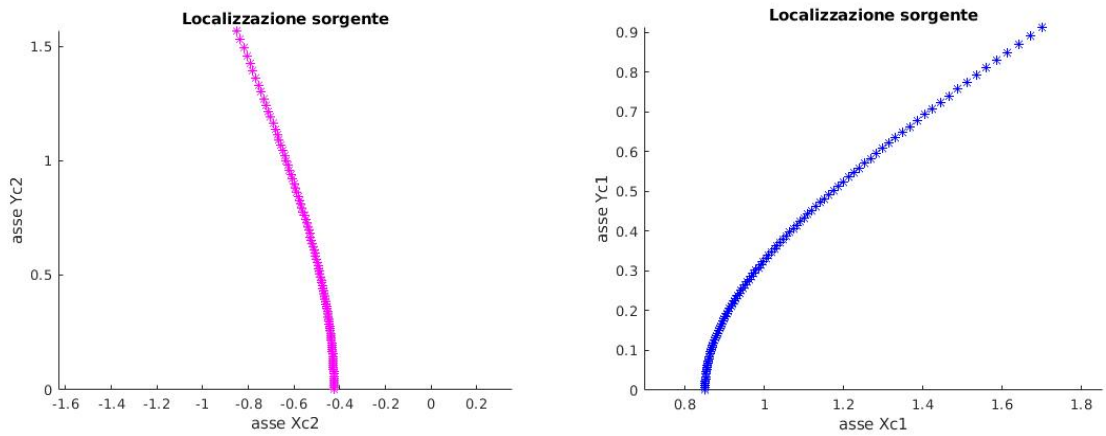


**Figura 2.3.5 Plot della soluzione della localizzazione di MATLAB per Δt<0 e per Δt>0**

# 3 Conclusioni

In questa relazione sono stati presentati i risultati ottenuti durante lo svolgimento dell'attività "Integrazione di sensori acustici in sistemi smart home ed implementazione di algoritmi per l'individuazione e la localizzazione di segnali acustici". Questi risultati si possono sintetizzare in:

a) studio e sviluppo dell'hardware per il sensore acustico;

b) studio e sviluppo dei sistemi di acquisizione e gestione dei segnali acustici;

c) studio e sviluppo degli algoritmi per la localizzazione di segnali acustici ambientali tramite l'hardware messo a punto.

Queste tre attività sono strettamente correlate tra di loro e le scelte fatte in ciascun punto sono state influenzate dallo sviluppo dalle altre due, quindi vanno viste come un'unica percorso di studio, ricerca e sviluppo.

Riguardo al punto a), si è potuto verificare la possibilità di realizzare ad un costo contenuto (qualche euro) un sensore acustico amplificato a partire da una capsula microfonica a condensatore ed ad un circuito di amplificazione a transistor bipolare in configurazione emettitore comune. A questa scelta si è arrivati dopo lo studio dei dispositivi presenti sul mercato, e considerando la necessità di interfacciarsi non necessariamente con un hardware di controllo nato per applicazioni audio, quanto piuttosto delle schede a basso costo a microcontrollore. Le scelte effettuate hanno portato alla costruzione ed al test di sensori perfettamente funzionanti, che sono stati poi utilizzati nelle successive fasi di test degli algoritmi. Gli sviluppi futuri di questa attività riguardano essenzialmente l'ingegnerizzazione del sensore, attraverso la fabbricazione di un'unica board contenente sia la capsuala microfonica che l'amplificatore. Tale board potrà tranquillamente risultare comunque di dimensioni ridotte (2x2 cm circa) per cui si è raggiunto anche l'obbiettivo di realizzare un sensore di dimensioni estremamente ridotte.

Riguardo al punto b), dopo un'analisi delle varie schede disponibili sul mercato si è deciso di proporre e sviluppare una soluzione basata sulle schede della famiglia Arduino. Questa scelta è legata alla ampia disponibilità di schede shield per queste schede che permettono una facile integrazione di sensori e sistemi di comunicazione wireless. In questi termini, essendo le problematiche di comunicazione facilmente risolvibili, ci si è potuti concentrare sugli altri aspetti legati al trattamento del segnale, ossia alla sua acquisizione tramite conversione analogico digitale e alla sua trasmissione al sistema centrale di elaborazione dati. Bisogna da questo punto di vista considerare che l'acquisizione dei segnali per la localizzazione deve essere necessariamente un processo sincronizzato, in quanto la localizzazione si basa sul ritardo di arrivo tra i segnali acquisiti in maniera sincrona dai microfoni. Pertanto i microfoni devono essere comandati tutti dalla stessa unità. In questo senso, l'uso di Arduino è ottimale per l'acquisizione dei segnali secondo questa modalità. E qualsiasi sistema si voglia utilizzare per la realizzazione della rete di sensori acustici deve prendere in considerazione questo aspetto. Bisogna comunque tenere conto della continua immissione sul mercato, soprattutto da parte di industrie cinesi, di schede a microcontrollore dotate di connessione wifi. Tali sistemi attualmente non si rivelano competitivi per la mancanza di forniture rapide e di verifiche sull'affidabilità, ma potrebbero comunque diventare utili per lo sviluppo di sensori acustici smart e per gli sviluppi futuri di questo genere di attività.

Riguardo al punto c), sono stati studiati, sviluppati ed implementati degli algoritmi per la localizzazione delle sorgenti in ambienti indoor che contengono due o più sensori. Le problematiche relative all'accuratezza di tali algoritmi in condizioni di funzionamento reale, ossia tenendo conto della loro effettiva implementazione su hardware sono state affrontate ed analizzate, evidenziando la possibilità di raggiungere una buona precisione (dell'ordine della ventina di cm) nella localizzazione di sorgenti acustiche. Questi algoritmi si prestano quindi all'analisi acustica di ambienti indoor, considerando che un unico sistema ad Arduino controlli almeno due o più sensori realizzati secondo quanto detto in precedenza.

Va ribadito inoltre che l'implementazione dell'algoritmo di acquisizione e localizzazione su microcontrollore ha messo sì in risalto le problematiche realizzative di un sistema smart di questo tipo, ma anche permesso di individuare i corretti compromessi tra costo computazionale e costo di trasmissione del segnale acquisito, per uno sviluppo ottimale dell'intero sistema e della integrazione in esso, di un certo numero di sensori acustici. Ad esempio, sono stati correttamente superati tutti i problemi di acquisizione e sono state individuate anche delle soluzioni di controllo che permettono di minimizzare la gestione della memoria (ad esempio realizzando l'acquisizione dei segnali dalle coppie di microfoni in maniera alternata). Gli sviluppi dell'attività potranno consistere nell'integrazione dell'intero sistema come da noi pensato in un ambiente domotico reale [12].

# 4 Riferimenti bibliografici


1. Giuliano Cammarata, "Acustica Applicata", Dispensa di Acustica, 2016, (reperibile su internet alla pagina dell'autore www.giulianocammarata.it).
2. John Eargle "The Microphone Book". Taylor & Francis, 2004
3. Ian Corbett, "Mic It!: Microphones, Microphone Techniques, and Their Impact on the Final Mix". CRC Press, 2014.
4. Datasheet della capsula microfonica CMC-4015-40P (https://www.cui.com/product/resource/cmc-4015-40p.pdf, retrieved il 13/11/2018)
5. Datasheet dell'integrato LM386 (amplificatore audio) (http://www.ti.com/lit/ds/symlink/lm386.pdf retrieved il 13/11/2018)
6. Sedra Smith, "Circuiti per la microelettronica", Edizione EdiSES anno 2013
7. Millman, Grabel, "Elettronica di Millman" (4th ed., Collana di istruzione scientifica Serie di elettronica). Milano, 2008, McGraw-Hill.
8. AA.VV. "Arduino Language Reference" disponinile su web alla pagina: http://www.ele.uri.edu/courses/ele205/ELE205Lab/ELE205_Lab_files/Arduino%20-%20Reference.pdf

9. Tiziana Marsella, Romano Lombardi, Elementi Base del linguaggio di programmazione di Arduino, disponibile su web su diversi siti, ad esempio http://www.alberti-porro.gov.it/wordpress/wp-content/uploads/2014/01/ProgrammareArduino.pdf
10. AA. VV. "Arduino notebook" , disponibile su web alla pagina: https://playground.arduino.cc/uploads/Main/arduino_notebook_v1-1.pdf
11. Delores M. Etter, David C. Kuncicky; Titolo: Introduzione a MATLAB, Copyright 1999 PRENTICE HALL, INC
12. P. Clerici Maestosi, L. Luccarini, S. Pizzuti, F. Romanello, S. Romano, A. Zanela; Titolo: Gestione energetica Smart Home e assisted living: sviluppo di sistemi e soluzioni integrate Ricerca di Sistema Elettrico 2017