Arduino: la scheda, i sensori e gli attuatori

Sembra che Arduino sia nato dall'idea di un gruppo di professori, assistenti e studenti dell'Interaction Design Institute di Ivrea che, chiacchierando intorno ad un aperitivo nel caffe' ***Arduino***, hanno ipotizzato una piattaforma hardware e software capace di avvicinare gli studenti o semplicemente le persone animate da un poco di curiosita', al mondo dello sviluppo dei prototipi.

Riducendo il tutto ai minimi termini e' possibile affermare che Arduino e' un computer in grado di interpretare i segnali provenienti da sensori (termometri, interruttori, fotocellule, sensori di movimento, sensori di infrarossi, rilevatori di ultrasuoni, eccetera) e di azionare degli attuatori (luci, avvisatori acustici, motorini elettrici, rele', display, ecc.) sulla base di un programma che interpreta i segnali dei sensori e li elabora, decidendo se, come e quali attuatori attivare.

Arduino unisce quindi due mondi: quello hardware, rappresentato dalla scheda e dai componenti ad essa collegabili e quello software, rappresentato dal programma scritto dall'utente (o copiato, se trova qualcosa di pronto che soddisfa le sue esigenze).

Cosa serve per iniziare

Per iniziare e senz'altro consigliabile procurarsi uno degli innumerevoli kit per principianti, contenenti sia la scheda che un certo numero di componenti. In questo modo si disporra' del materiale minimo necessario per riprodurre i primi esercizi proposti in queste note ed impadronirsi velocemente del linguaggio di programmazione.

Dove trovare Arduino ed i componenti elettronici

L'Arduino originale, oltre ad un kit per una quindicina di esercizi ed un po' di manualistica, anche in italiano, si puo' trovare qui:
http://www.arduino.cc/

I cloni di Arduino (perfettamente legali e funzionanti), cosi' come parecchie combinazioni di starter kit, si possono trovare nei siti che vendono componentistica elettronica. Basta digitare su google ***arduino kit*** e poi c'e' solo l'imbarazzo della scelta.

In linea di massima sembra conveniente iniziare con un kit di basso prezzo (con poco piu' di trenta di euro si prende un kit abbastanza completo, arduino compreso), in modo da impratichirsi un poco nell'utilizzo della scheda e dei componenti e poi acquistare via via solo i componenti che servono per i progetti che si intendono realizzare.

Il tempo di consegna di un kit o anche di un singolo componente, da Hong Kong, con un corriere espresso, e' di circa una settimana (o un mese, se si usa il normale servizio postale) mentre il tempo di consegna da un sito italiano, tramite poste italiane, e' di circa 3 giorni lavorativi

Un minimo di approfondimento sulla scheda

I componenti collegabili ad Arduino

I sensori

- **sensori di luminosita' o infrarossi**
  - fotoresistenza
  - ricevitore di raggi infrarossi
  - modulo "avoidance" – sensore di ostacoli a raggi infrarossi
  - sensore di traccia
  - Sensore di fiamma

- **sensori di temperatura**
  - modulo LM35
  - modulo DS18B20
  - DHT11 umidita' e temperatura

- **sensori di suono**
  - microfono
  - microfono amplificato
  - modulo HC-SR04 - rilevatore di distanza ad ultrasuoni

- **sensori di movimento**
  - sensore di tilt
  - sensore di battito
  - luci magiche
  - rilevatore di rotazione (rotary encoder)
  - joystick PS2

- **sensori di campo magnetico**
  - sensore magnetico 44E938
  - sensore reed

- **sensori di altro tipo**
  - sensore tattile
  - sensore di gas
  - HC06 connessione bluetooth

Gli attuatori

- **sorgenti luminose, infrarosse e laser**
  - led
  - led RGB
  - generatore di infrarossi
  - cifra digitale
  - display led a 4 cifre
  - matrice led 8x8
  - display a cristalli liquidi 1602
  - display lcd 1602 con driver LCMI 602
  - illuminatore laser

- **generatori di suono**
  - buzzer attivo e passivo

- **generatori di movimento**
  - motore passo passo
  - servomotore microservo 9g
  - modulo L298 per la gestione di due motori

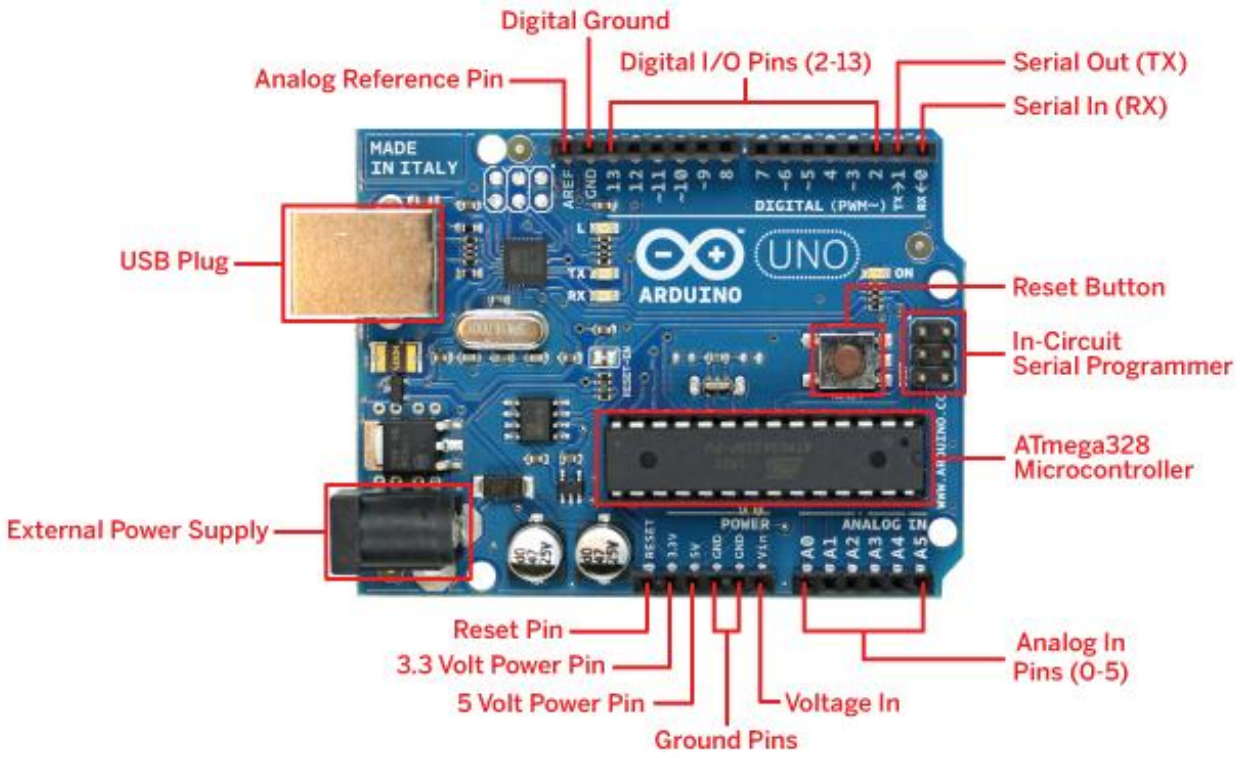
I componenti di supporto

- banco di lavoro (breadboard)
- resistenza
- resistenza variabile
- pulsante

Un minimo di approfondimento sulla scheda

In questa nota viene presa in considerazione la scheda Arduino Uno che, pur con alcune peculiarita', non si differenzia (almeno nei principi) dalle altre schede della famiglia.

Su internet e' presente questa bella rappresentazione:



La scheda e' dotata di una memoria flash da 32 k bytes (non tantissimi ma sufficienti per gestire programmi anche abbastanza complessi), 2 kbyte di sdram ed 1 kbyte di eeprom. La scheda e' priva di sistema operativo ed e' in grado di gestire un solo programma per volta. Quando si costruisce o si ricostruisce un prototipo bisogna quindi caricare, da pc, il relativo programma. Una caratteristica fondamentale della memoria flash e' la capacita' di mantenere il suo stato anche in assenza di alimentazione per cui un programma, una volta caricato sulla scheda, resta memorizzato e disponibile sino a quando non ne viene caricato un altro.

Nel disegno, oltre al microcontrollore (atmega328), sono riconoscibili i connettori (altrimenti chiamati porte o pin) ai quali collegare i sensori o gli attuatori (le due file orizzontali situate sul lato superiore e inferiore della scheda), la presa USB, per collegare la scheda ad un pc e la presa di alimentazione (external power supply) indispensabile per garantire il funzionamento anche quando non e' attivo il collegamento USB con un computer.

In dettaglio, partendo dall'angolo superiore sinistro dell'immagine e procedendo in senso orario:

- **AREF**, analog reference pin, porta gestita dall'istruzione *analogReference,* e' in grado di fornire una tensione specifica ad eventuali sensori analogici che richiedono tensioni diverse da 5 o 3,3 volt;
- **GND** digital ground: collegamento di terra per i componenti (digitali o analogici) collegati alla scheda;
- **digital i/o pins**: porte utilizzabili per l'ingresso o l'uscita di segnali digitali. Sono numerate da 2 a 13 e sono singolarmente indirizzabili dal microcontrollore. Il loro utilizzo non e' predefinito per cui in un programma si puo' ad esempio utilizzare la porta 13 come porta di

output alla quale, sempre per esempio, collegare un led, e la porta 12 come porta di input alla quale collegare un sensore (ad esempio un pulsante) ed in un altro programma fare esattamente l'opposto. Tipicamente, su ogni porta presa in considerazione dal programma, Arduino assegnera' o rilevera' (a seconda che la porta sia stata definita di output o di input) il valore 1 (HIGH) in caso di "porta attivata" ed il valore 0 (LOW) in caso di "porta disattivata". Il valore 1 implica che attraverso la porta circoli una tensione di 5 volt mentre il valore 0 implica che attraverso la porta non circoli alcuna tensione. In realta' le cose non sono cosi' nette e se alla porta sono collegati dei sensori e' possibile che circoli comunque una tensione, anche quando dovrebbe essere rilevato uno stato "LOW". Arduino considera LOW una tensione in entrata minore di 1,5 volt, considera HIGH una tensione superiore a 3 volt e non fornisce alcun responso (mantiene il preesistente valore HIGH o LOW) per le tensioni comprese tra 1,5 e 3 volt. Alcune porte digitali e piu' precisamente la 3, 5, 6, 9, 10 e 11 possono essere utilizzate anche in modalita' PWM (Pulse Width Modulation) e cioe' possono emettere un segnale di tipo analogico e quindi, ad esempio, aumentare o diminuire il voltaggio in uscita per, ad esempio, aumentare o diminuire l'intensita' luminosa di un led

- **TX** serial output: porta di output seriale utilizzabile per inviare dati ad un'apparecchiatura esterna. Puo' anche essere utilizzata come porta digitale ma, almeno nella mia esperienza, e' soggetta ad interferenze dovute forse alla sua duplice funzione
- **RX** serial input: porta di input seriale utilizzabile per ricevere dati seriali da un'apparecchiatura esterna. Puo' anche essere utilizzata come porta digitale ma, sempre per esperienza personale, puo' essere soggetta ad interferenze dovute probabilmente alla sua duplice funzione
- **reset button**: pulsante di reset. Da utilizzare per riavviare il programma memorizzato nel microcontrollore.
- **in circuit serial programmer:** connettori utilizzati per caricare il bootloader, e cioe' un piccolo programma stabilmente residente sulla scheda, che al momento dell'accensione avvia il microprocessore e lancia il programma utente. Sono connettori normalmente non utilizzati da coloro che si avvicinano per la prima volta ad Arduino e di fatto non tratti in queste note.
- **AT328mega microcontroller:** e' il cuore della scheda, le cui specifiche sono state in parte gia' dichiarate. E' un minicomputer che incorpora un microprocessore da 16 mhz, la memoria da 32 k byte, una eeprom, una sdram, i relativi bus, il software di boostrap (il bootloader, il software di inizializzazione del microprocessore) ed il programma di gestione della connessione usb.
- **Analog in**: porte analogiche di input, numerate da 0 a 5, alle quali possono essere collegati dei sensori di tipo analogico (es. sensore di temperatura oppure sensore di luce). Le porte analogiche sono porte di input e generano un valore che va da 0 a 1023, a seconda della tensione rilevata sul sensore (il valore 0 corrisponde a 0 volt mentre il valore 1023 corrisponde a 5 volt). Attraverso una porta analogica e' quindi possibile ricevere segnali come il livello di una temperatura o il livello di luminosita', che non potrebbero essere rappresentati da un semplice 0 oppure 1. Al di la del nome, inoltre, queste porte possono anche essere utilizzate anche come porte digitali di output ed in questo caso sono indirizzabili con i numeri da 14 a 19 (la porta 14 e' la A0 mentre la porta 19 e' la A5).
- **Vin** (voltage input): porta attraverso la quale puo' essere all'occorrenza fornita energia alla scheda tramite un'alimentazione esterna (da 7 a 12 volt cc).
- **GND** ground pin: porte di terra alle quali collegare il polo negativo dei sensori analogici o digitali
- **5v**: 5 volt power pin, porta che fornisce corrente continua stabilizzata a 5 volt ai componenti collegati alla scheda
- **3.3v**: 3,3 volt power pin, porta che fornisce corrente continua stabilizzata a 3,3 volt
- **Reset**: porta attraverso la quale e' possibile effettuare il reset (il riavvio) del programma memorizzato nel microprocessore
- **External power supply**: porta di alimentazione esterna da 9 volt. L'alimentazione esterna diviene indispensabile nel momento in cui si vuole "sganciare" la scheda dal pc e farla

vivere di vita propria. L'energia puo' essere fornita da un alimentatore esterno o, visto il basso consumo del dispositivo, anche da una comune batteria da 9 volt. La scheda e' dotata di dispositivo che esclude automaticamente l'alimentazione esterna nel momento in cui viene collegata ad un pc. Attenzione: quando si utilizza l'alimentazione esterna in assenza di connessione usb, bisogna essere certi che il programma non emetta alcun segnale seriale poiche' in caso contrario, in mancanza di un monitor sul quale scaricare detti segnali, le aree di memorizzazione e transito dei segnali potrebbero venire rapidamente saturate, con conseguente blocco del programma.

- **USB plug**: porta di connessione ad un computer. La connessione non solo fornisce energia alla scheda (se non e' gia' presente una fonte di alimentazione collegata alla external power supply), ma e' anche indispensabile per ricevere il programma, per inviare segnali seriali al monitor di sistema, residente sul pc e per ricevere segnali dalla tastiera del pc.

La scheda e' dotata anche di alcuni led che lampeggiano quando transitano informazioni attraverso la porta usb o quando si usa la porta digitale 13.

Qualche nota sui componenti collegabili ad Arduino

I componenti collegabili ad Arduino possono essere classificati in tre macrocategorie

- i sensori
- gli attuatori
- i componenti di supporto, come i fili, resistenze, condensatori, fusibili, breadboard, pulsanti ed altri, utilizzati nella costruzione dei circuiti

I sensori

I sensori sono componenti elettronici in grado di percepire e misurare le caratteristiche fisiche dell'ambiente circostante e quindi, principalmente, luminosita', temperatura, umidita', suono, movimento, campo magnetico ed elettromagnetico.

Gli attuatori

Gli attuatori sono componenti in grado di modificare le caratteristiche fisiche dell'ambiente circostante e quindi essenzialmente, sorgenti di luce, calore, umidita', suono, movimento, campo magnetico ed elettromagnetico.

I componenti di supporto

Sono componenti che supportano l'operativita' di sensori ed attuatori. Tra i componenti di supporto possiamo annoverare le resistenze, i condensatori, i fusibili ed altri ancora come la breadboard, gli shield, i pulsanti ed i cavi di collegamento

I sensori

Sensori di luminosita’ (sensori di luce e di segnali infrarossi)

Fotoresistenza

Una fotoresistenza e' una resistenza la cui impedenza (e cioe’ la cui capacita’ di far circolare elettricita’) varia al variare della luce che la colpisce. All'aumentare della luce diminuisce la resistenza, e viceversa. E’ tipicamente un sensore di tipo analogico. Per utilizzarlo si collega una gamba ad una porta analogica e, in parallelo, ad una resistenza da 10k ohm collegata a terra mentre si collega l’altra gamba all’alimentazione da 5 volt. La porta analogica restituisce un valore da 0 a 1023 che varia al variare della luce che colpisce la fotoresistenza. Piu’ la luce e’ forte, piu’ il valore si avvicina a 1023 (esercizio 9 – luce crepuscolare).

Ricevitore di raggi infrarossi

I raggi infrarossi sono delle radiazioni elettromagnetiche la cui lunghezza d’onda varia da 0,7 micron a 0,4 millimetri. Si tratta in pratica di una luce non visibile all’occhio umano ma percepita da particolari fotoresistenze. Volendo sintetizzare al massimo i tecnicismi relativi alla trasmissione infrarossa e’ possibile affermare che il telecomando altro non e’ che un led che emette luce infrarossa mentre il ricevitore e’ una fotoresistenza sensibile agli infrarossi. Alla pressione di un tasto il telecomando emette una serie di brevissimi flash di luce infrarossa che vengono letti ed interpretati dal ricevitore. Gli impulsi infrarossi ricordano, a tratti, il vecchio alfabeto morse. Alla pressione di un qualunque tasto il trasmettitore invia al ricevitore 4 bytes da 8 bit componendo gli “0” e gli “1” modulando opportunamente gli intervalli di tempo tra un flash e l’altro. Se il telecomando emettesse una luce visibile e se i tempi fossero dilatati potremmo distinguere lo zero dall’uno semplicemente misurando il tempo trascorso tra un’illuminazione e la successiva. Un intervallo lungo corrisponde ad un 1 mentre un intervallo breve corrisponde ad uno zero.
In realta’ il protocollo di codifica e’ un po’ piu’ complicato di quanto detto poiche’ ogni comando e’ composto da una sequenza di 32 bit (in totale 4 byte) preceduta da un segnale di sincronismo composto da un intervallo luminoso da 9 microsecondi seguito da un intervallo non luminoso da 4,5 microsecondi. Ogni bit e’ preceduto da flash da 0,56 microsecondi e da un intervallo non luminoso che, se corto (0,56 microsecondi), indica uno zero mentre se lungo (1,7 microsecondi) indica un 1.

I quattro bytes rappresentano, nell’ordine, un indirizzo, la ripetizione dell’indirizzo, il comando e la ripetizione del comando (quest’ultimo con la sequenza di bit invertita).

Sia il ricevitore che il trasmettitore devono essere in connessione “visiva” non devono cioe’ frapporsi ostacoli tra l’apparecchio trasmittente ed il modulo ricevente. Il ricevente inoltre interpreta al meglio il segnale quando il trasmettitore e’ posto esattamente davanti al bulbo (ad una distanza massima di 8 metri).

In questa tabella sono riportati i segnali trasmessi dai 21 tasti del telecomando illustrato in figura (valore HEX = valore esadecimale e valore DEC = valore decimale):

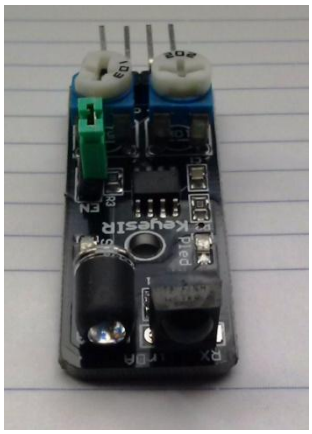
pulsante	valore HEX	Valore DEC
Bottone rosso	0xff00	0
VOL +	0xfe01	1
FUNC/STOP	0xfd02	2
Indietro veloce	0xfb04	4
Pausa/avvio	0xfa05	5
Avanti veloce	0xf906	6
Punta bassa	0xf708	8

pulsante	Valore HEX	Valore DEC
VOL -	0xf609	9
Punta alta	0xf50a	10
0	0xf30c	12
EQ	0xf20d	13
ST/REPT	0xf10e	14
1	0xef10	16
2	0xee11	17

pulsante	Valore HEX	Valore DEC
3	0xed12	18
4	0xeb14	20
5	0xea15	21
6	0xe916	22
7	0xe718	24
8	0xe619	25
9	0xe51a	26

Il ricevitore, marchiato 1057 D21B ha la piedinatura illustrata in figura. E’ ovviamente possibile utilizzare altri tipi di ricevitori, purche’ vengano rispettate le funzionalita’ dei piedini: il positivo (VCC) all’alimentazione da 5 volt, il negativo (GND) a terra ed il sensore (VOUT) alla porta di Arduino utilizzata per rilevare il segnale (vedi anche esercizio 16 ed esercizio 17).

Modulo “avoidance” - sensore di ostacoli a raggi infrarossi



Il modulo keyes riportato in figura e’ una basetta del costo di un paio di euro, sulla quale sono montati un generatore e un rilevatore di infrarossi, un paio di resistenze variabili, due led, un timer (usato per modulare la sequenza dei segnali infrarossi), svariate resistenze e condensatori.

Le basi sulle quali opera il sensore sono abbastanza semplici. Il generatore lancia un fascio di segnali infrarossi che vengono riflessi da un eventuale ostacolo e poi intercettati dal modulo di ricezione.

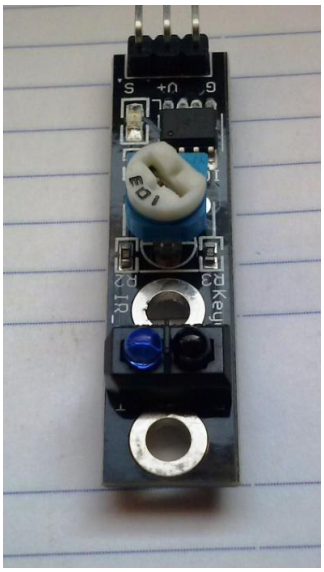
I segnali riflessi perdono gran parte della loro forza per cui questo tipo di sensore puo’ rilevare ostacoli posti ad una distanza massima di 20 centimetri (alcune specifiche parlano di 40 centimetri, ma prove pratiche hanno dimostrato di non poter andare oltre i 20 centimetri).

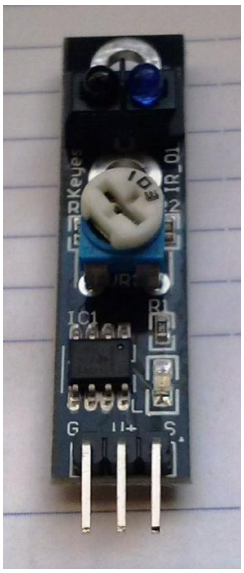
Agendo sulle resistenze variabili e’ possibile diminuire la distanza massima di rilevamento fino a portarla ad un minimo di due centimetri.

La basetta fornisce, sul pin “out”, un segnale di tipo digitale ed il sensore non sembra in grado di poter comunicare la distanza dell’ostacolo modulando opportunamente il segnale.

L’utilizzo e’ quindi ridotto alla semplice individuazione di ostacoli a distanza ravvicinata e puo’ probabilmente trovare applicazione pratica come contatore in impianti di media velocita’ (vedi anche esercizio 29).

Sensore di traccia

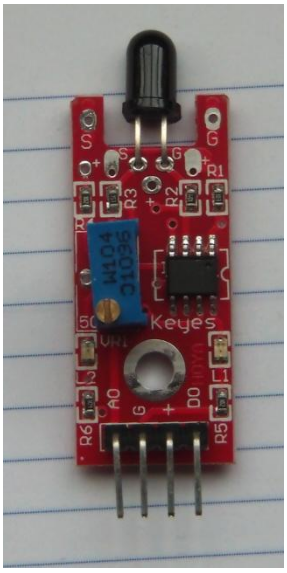




Il sensore di traccia e' in grado di rilevare la presenza o meno di una traccia nera su di un fondo bianco. Utilizza il medesimo principio del modulo avoidance, solo che il generatore ed il ricevitore di infrarossi sono un po' meno sensibili e sono in grado di rilevare solo i segnali di ritorno da una superficie bianca.

Una striscia nera e' quindi invisibile e per questo motivo il sensore puo' essere utilizzato per "seguire" una traccia nera dipinta su di un fondo bianco. Quando esce dalla traccia infatti il sensore genera un segnale digitale e Arduino puo' quindi azionare dei dispositivi atti a ritrovare e seguire la traccia nera. Vedi esercizio 30

Sensore di fiamma



Una fiamma non solo produce effetti percepibili dai sensi umani quali calore e luce, ma produce anche onde elettromagnetiche, non percepibili dai nostri sensi, la cui lunghezza d'onda varia dai 700 ai 1000 nanometri.

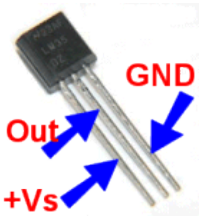
Si tratta di segnali di tipo infrarosso che possono essere individuati da un sensore opportunamente tarato.

In figura e' rappresentato un sensore di questo tipo montato su di una basetta sulla quale sono presenti anche sei resistenze, due led, una resistenza variabile ed un comparatore.

Il rilevatore di fiamma trasforma una improvvisa ed importante variazione di raggi infrarossi in un segnale elettrico. Se l'intensita' del segnale supera una soglia (definita tramite la resistenza variabile) la basetta produce un impulso digitale rilevabile da Arduino. Vedi anche esercizio 32

Sensori di temperatura

Modulo LM35

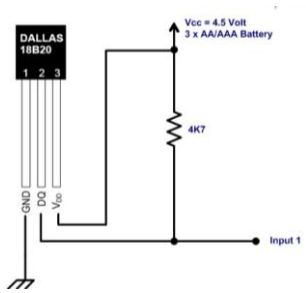


il sensore di temperatura LM35 e' caratterizzato da tre pin e da un corpo semicilindrico. Se, guardando il lato piatto del componente, si alimenta il pin di sinistra con una tensione continua di 5 volt e si collega il pin di destra a terra, sul pin centrale e' presente una tensione che, letta da una porta analogica di Arduino, assume un valore (da 0 a 1023) proporzionale alla tensione stessa: 0 corrisponde ad una tensione di 0 volt e 1023 corrisponde ad una tensione da 5 volt (vedi esercizio 10 – sensore di temperatura LM35). La tensione rilevata sul

pin centrale e' proporzionale alla temperatura dell'ambiente che circonda il sensore: ogni grado sopra lo zero provoca un aumento di tensione di 10 millivolt. La temperatura e' quindi pari al valore fornito dalla porta analogica moltiplicato per 5000 e diviso per 10230 e di conseguenza:

temperatura = (valore fornito dalla porta analogica * 0,488758)

modulo DS18B20



Il modulo DS18B20 ha la forma di un transistor. E' composto da un sensore in grado di misurare temperature da – 55 a +125 gradi centigradi, con un'approssimazione di 0,5 gradi, e da un chip in grado di convertire il segnale analogico in un segnale digitale e svolgere altre interessanti funzioni. E' alimentato da una fonte da 3 a 5,5 volt ed ha un suo numero seriale (ogni modulo DS18B20 ha un suo univoco numero seriale). Questa particolare caratteristica consente di avere piu' moduli collegati alla medesima porta di Arduino poiche' il software presente nelle librerie e' in grado di riconoscere il numero seriale e quindi fornire la temperatura di ogni singolo ambiente in cui e' collocato un modulo.

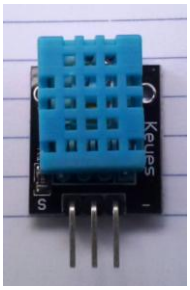
Il modulo puo' anche essere alimentato dall'energia (dalle correnti parassite) presenti sulla linea in cui scorrono i dati per cui e' possibile limitare i collegamenti alla sola terra ed alla linea dati.

Ultima, ma non ultima caratteristica, il chip dispone di una memoria non volatile nella quale possono essere memorizzate due temperature (tipicamente una minima ed una massima) oltrepassate le quali lancia un segnale di allarme, sempre sottoforma di impulsi digitali, che puo' essere letto ed interpretato da Arduino.

In foto vediamo un DS18B20 premontato su di una basetta nella quale sono gia' presenti una resistenza da 4,7k ohm ed un led. Qualora si disponesse di un DS18B20 non premontato, bisognera' collegare la linea dati alla porta di rilevamento di arduino e in parallelo, all'alimentazione da 5 volt limitata da una resistenza da 4,7k ohm. I piedini della basetta non rispettano la configurazione dei piedini del modulo per cui se si dispone del solo modulo non premontato bisogna fare attenzione ai collegamenti (vedi anche schema a fianco della foto).

Prima di procedere alla compilazione del programma che utilizza il modulo (esercizio 10bis) e' necessario installare la versione 2.2 della libreria "OneWire.h" e la versione 3.7.2 della libreria "DallasTemperature.h", entrambe reperibili qui:
http://milesburton.com/Dallas_Temperature_Control_Library Le librerie devono essere scaricate e poi installate tramite la funzione sketch->importa libreria, dell'IDE. Una volta installate bisognera' chiudere e riaprire la IDE per renderle operative.

DHT11 - umidita' e temperatura



Il modulo DHT11 e' un sensore composito che combina, in un sol corpo, un termometro ed un igrometro.

Esistono parecchi moduli della serie DHT ed il modulo 11, trattato in questo esercizio, e' il piu' semplice ed il piu' economico.

Il modulo misura i valori di umidita' e temperatura e, attraverso un microcontrollore ad 8 bit in esso racchiuso, li trasforma in segnali digitali.

Si tratta di quindi di un componente evoluto, composto da un sensore di umidita' di tipo resistivo, un sensore di temperatura di tipo NTC (Negative Temperature Coefficient – un sensore analogico che diminuisce l'impedenza di una resistenza all'aumentare della temperatura) e da un microcontrollore.

Questo sensore, su sollecitazione di Arduino, trasmette sul pin S (signal) un treno di quaranta bit:

- 8 bit per indicare la parte intera del valore di umidita'
- 8 bit per indicare la parte decimale dell'umidita'
- 8 bit per indicare la parte intera della temperatura
- 8 bit per indicare la parte decimale della temperatura
- 8 bit per indicare il numero di controllo (per validare il valore dei precedenti 32 bit)

La gestione del segnale e' ovviamente a carico di una libreria (la DHT.h) che deve essere scaricata ed installata prima della compilazione del programma.

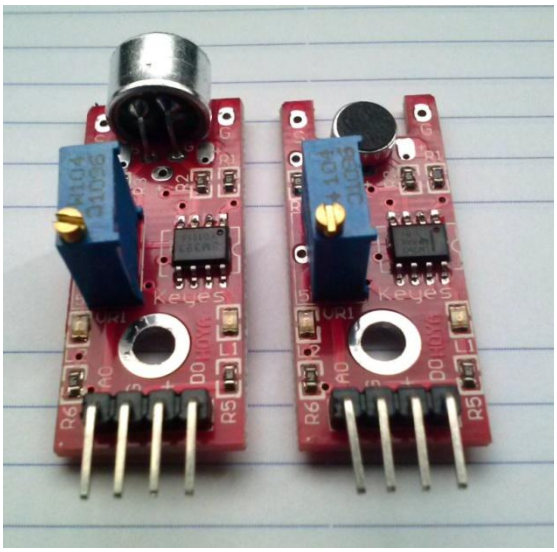
Per scaricare ed installare la libreria:

- Aprire questo link (github e' un sito sicuro) : https://github.com/RobTillaart/Arduino
- Scaricare tutte le librerie premendo il bottone "download ZIP" (sulla destra dello schermo)
- Disimpaccare il file e ottenere una cartella con decine di librerie, tra cui la libreria DHTlib che a sua volta contiene una cartella example, e tre files: dht.cpp e dht.h ed un readme
- Rinominare in "DHT" la libreria "DHTlib"
- Installare la libreria DHT nella libreria di arduino (IDE->sketch->importa libreria->add library ->indirizzo della nuova libreria DHT->apri)
- Chiudere e riaprire l'IDE per rendere operativa la libreria
- Compilare il programma presente in IDE->file->esempi_>DHT->DHT11_test per verificare l'avvenuta installazione della libreria

Se tutto e' andato bene si puo' procedere alla misurazione di umidita' e temperatura (vedi anche esercizio 28 – umidita' e temperatura).

Sensori di suono

Microfono



In figura sono riportati due moduli keyes dotati di microfono. Uno e' un normale microfono mentre l'altro (quello con la capsula microfonica di dimensioni maggiori) dovrebbe essere un modello ad alta sensibilita'.

Ogni basetta keys rappresentata in figura contiene:

- un microfono,
- una resistenza variabile,
- un modulo LM393
- sei resistenze
- due led

Entrambi i microfoni sono dotati di un'uscita analogica e di un'uscita digitale. Quest'ultima fornisce un segnale positivo (HIGH) quando il livello del suono raggiunge un limite fissato dalla resistenza variabile. Il microfono converte i segnali audio in segnali elettrici mentre il modulo

LM393 viene utilizzato come comparatore di voltaggio. Quando il voltaggio del segnale proveniente dal microfono e' maggiore del voltaggio regolato dal potenziometro, il modulo LM393 produce sul pin DO in uscita un segnale HIGH mentre quando il voltaggio e' inferiore provoca un segnale LOW.

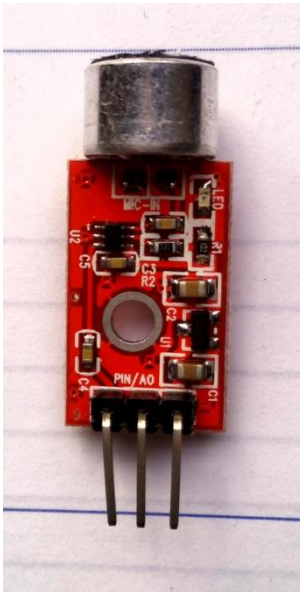
Per ottenere un segnale digitale bisogna quindi preventivamente agire sul potenziometro in modo da ottenere il segnale HIGH solo quando il rumore supera la soglia che desideriamo monitorare (nel nostro caso il battito delle mani). La taratura del microfono e' abbastanza semplice: si agisce in senso orario sulla vite del potenziometro fino a quando il led posizionato a sinistra si spegne. Si prova poi a battere le mani e se il led a sinistra si accende significa che il sensore e' in grado di riconoscere il battito e quindi di produrre un segnale HIGH in uscita. Se invece non si accende provare a regolare lentamente la vite sino a quando non si ottiene l'effetto desiderato

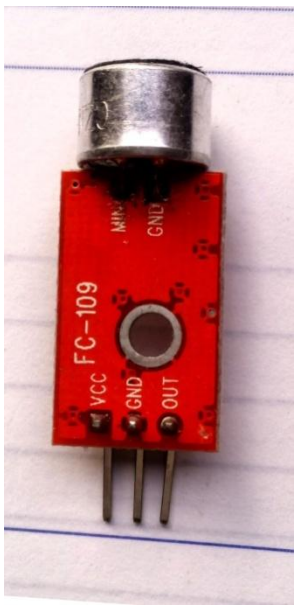
Il segnale analogico invece varia al variare dell'intensita' o meglio, del volume del suono. Purtroppo il modulo LM393 presente sulla basetta non e' utilizzato nella sua usuale veste di amplificatore ma solo come comparatore per cui sulla porta analogica sono rilevabili solo le tensioni prodotte direttamente dal microfono e quindi molto basse e con modestissime escursioni.

Per questo motivo l'uscita analogica e' di fatto inutilizzabile a meno che non le si affianchi un modulo di amplificazione.

Alla prova dei fatti (esercizio 25 – microfono) l'uscita analogica si e' dimostrata praticamente inutilizzabile e si e' quindi fatto ricorso all'uscita digitale, la cui fase di regolazione si e' dimostrata abbastanza semplice.

Microfono amplificato





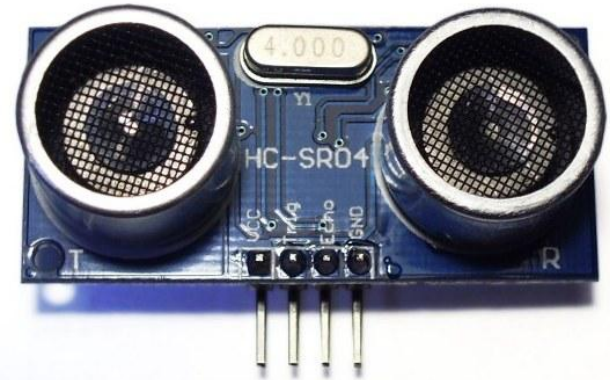
Questo e' un microfono sostanzialmente diverso dal precedente. La circuiteria presente sulla basetta non si riduce ad un semplice comparatore ma propone un vero amplificatore, in grado di produrre in output una tensione proporzionale al volume del rumore percepito.

Questo tipo di microfono produce, in uscita, un segnale che non copre l'intera gamma dei segnali rilevabili da arduino (e cioe' da 0 a 5 volt) ma segnali che oscillano di qualche decimo di volt intorno ad un valore base (normalmente compreso tra 1,2 e 2 volt).

Per una ragione al momento non chiara (sembra comunque un difetto della scheda Arduino utilizzata per provare il micorfono), il valore base cambia ogni volta che si fornisce energia al circuito e pertanto deve essere manualmente rilevato tramite una traccia di debug e manualmente inserito nel programma di utilizzo.

Il segnale ha un'escursione non molto ampia ma comunque sufficiente a distinguere i suoni (o meglio i rumori) di volume maggiore da quelli di media o bassa intensita' (vedi anche esercizio 25bis- microfono amplificato).

Modulo HC-SR04 - rilevatore di distanza ad ultrasuoni



Il modulo ad ultrasuoni HC-SR04 viene normalmente utilizzato per rilevare eventuali ostacoli e misurarne la distanza (da 2 a 400 cm). Il modulo opera usando la medesima tecnica di rilevamento utilizzata, in natura, dai pipistrelli.

E' formato da un generatore di ultrasuoni, da un ricevitore e da un circuito di controllo. Il modulo si avvia quando riceve un impulso di almeno 10 microsecondi attraverso il "trig pin" (la sua porta di attivazione). A questo punto lancia una serie di otto onde sonore da 40 kHz e si mette in attesa di un segnale di ritorno. Appena lo riceve attiva la porta di uscita (echo pin) e la mantiene attiva per un tempo proporzionale al tempo intercorso tra l'invio del segnale acustico ed il suo ritorno (vedi anche esercizio 21, sensore di parcheggio e poliphemus).

Conoscendo la velocita' del suono e sapendo che il "viaggio" dell'onda sonora e' stato il doppio della distanza tra il modulo e l'ostacolo (l'onda e' andata dal generatore all'ostacolo e da qui' e' tornata al sensore) la distanza e' derivata dalla seguente formula:

distanza = tempo * 340 / 2 dove:

- *distanza* e' la distanza, in metri tra il modulo HC-SR04 e l'ostacolo,
- *tempo* e' il tempo in secondi di attivazione della porta di uscita
- *340* e' la velocita' del suono in metri al secondo.

e quindi

distanza = valore di echo / 58 dove:

- *distanz*a = distanza in centimetri dall'ostacolo
- *valore di echo* = valore fornito dall'istruzione *pulseIn*
- *58* = valore, senza decimali di (2/340)*10000 (10000 e' un moltiplicatore inserito per trasformare i metri in centimetri ed i secondi in millisecondi

Attenzione: il sensore intercetta i segnali di ritorno con un "angolo di visuale" di circa 15 gradi per cui intercetta anche l'eventuale segnale di ritorno proveniente dal pavimento, dal soffitto o da eventuali pareti laterali. Per ottenere una valida misurazione bisogna quindi tenere il modulo ad una altezza da terra e ad una distanza da ostacoli laterali sufficiente ad evitare interferenze. In linea di massima puo' essere utilizzata la seguente formula per calcolare la distanza massima misurabile in funzione della posizione (in termini di distanza minima da suolo e pareti) del sensore :

distanza massima misurabile = posizione del sensore * 3,73 (e' la cotangente di 15 gradi)

oppure

posizione del sensore = distanza massima misurabile / 3,73

Questo significa che se vogliamo "vedere" ostacoli ad una distanza massima di un metro dobbiamo tenere il sensore ad una distanza di almeno 26 centimetri da terra o da eventuali pareti.

Sensori di movimento

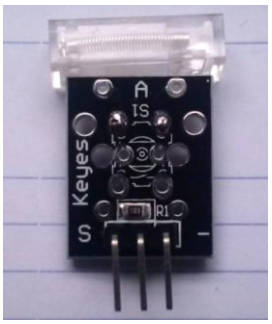
Sensore di tilt

Il sensore di tilt puo' essere paragonato ad un tubicino all'interno del quale scorre una sfera. Nel momento in cui il tubicino viene inclinato, la sfera raggiunge un capo del tubo chiudendo un circuito. E' come un pulsante automatico, che si apre o si chiude nel momento in cui lo si inclina (vedi anche esercizio 7 – sensore di tilt).

E' possibile collegare il sensore di tilt sia ad una porta digitale che ad una porta analogica. Se collegato ad una porta analogica non e' necessario frapporre alcuna resistenza mentre se lo si collega ad una porta digitale e' necessario frapporre una resistenza da 10k

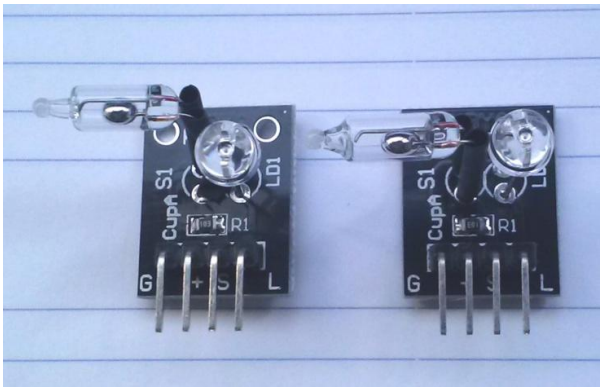
Sensore di battito

Il sensore di battito e' simile al sensore di tilt, ma e' un po' piu' sensibile e reagisce anche a vibrazioni di moderata entita'. Il sensore, alimentato da una tensione di 5 volt, produce un segnale digitale ogni volta che percepisce una vibrazione (vedi anche esercizio 7 – sensore di battito).



Il sensore rappresentato in foto e' montato su di una basetta sulla quale e' presente anche una resistenza.

Luci magiche



Le "luci magiche" sono due moduli, ognuno composto da un led rosso, un sensore di tilt al mercurio ed una resistenza.

Non si tratta di veri sensori, ma di un mix di componenti che, se utilizzati con un adeguato programma, consentono di ottenere effetti abbastanza inusuali.

Nell' esercizio 24 il programma reagisce all'inclinazione contemporanea dei due moduli diminuendo gradualmente l'intensita' luminosa del led di un modulo ed aumentando, nel contempo, la luminosita' di quello dell'altro modulo. Una specie di travaso di luce, ottenuto semplicemente inclinando i moduli in un senso e nell'altro.

Niente di magico, ovviamente, ma solo un programma che, pilotato dai due sensori di tilt, gestisce un effetto fade su ognuno dei due led. Non si tratta di un vero esercizio ma poco piu' di un gadget, la cui parte interessante sembra essere la gestione PWM dei due led.

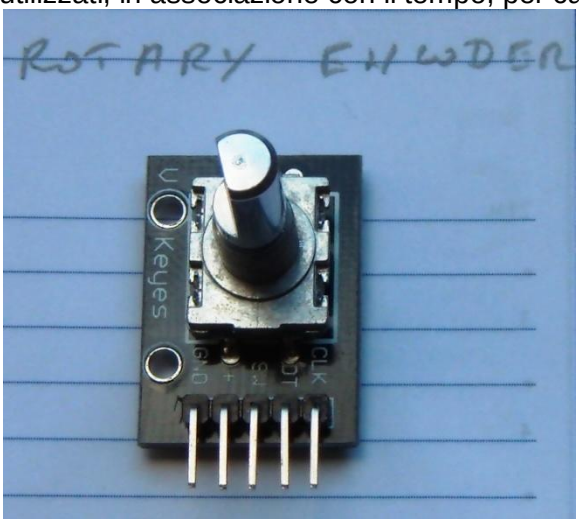
Rilevatore di rotazione (rotary encoder)

Un rilevatore di rotazione è un dispositivo elettromeccanico che converte un movimento di rotazione in un codice analogico o digitale. I rilevatori di rotazione agiscono come sensori per rilevare angolo, velocità di rotazione e accelerazione sull'asse.
Esistono due tipi di rilevatori di rotazione: i rilevatori assoluti e quelli incrementali.

Un rilevatore assoluto indica la posizione corrente dell'albero (in termini di gradi angolari) mentre un rilevatore incrementale fornisce informazioni sul moto dell'albero, che in genere viene ulteriormente trasformato in informazioni quali angolazione, direzione, velocità di rotazione ed accelerazione.

Il rilevatore angolare in figura e' un rilevatore di tipo incrementale.

E' un sensore rotativo che trasforma una rotazione in impulsi digitali che possono poi essere utilizzati, in associazione con il tempo, per calcolare le informazioni sopraccitate.



Il rilevatore fornisce un impulso digitale ogni volta che l'asse ruota di 18 gradi (e quindi 20 impulsi per ogni rotazione completa). Insieme ad ogni impulso, fornisce anche indicazioni sul senso di rotazione.

In questo particolare componete la manopola del rilevatore, oltre ad essere ruotata, puo' anche essere premuta, come succede per la manopola principale delle autoradio che, se premuta, accende o spenge l'apparecchio e se ruotata alza o abbassa il volume.

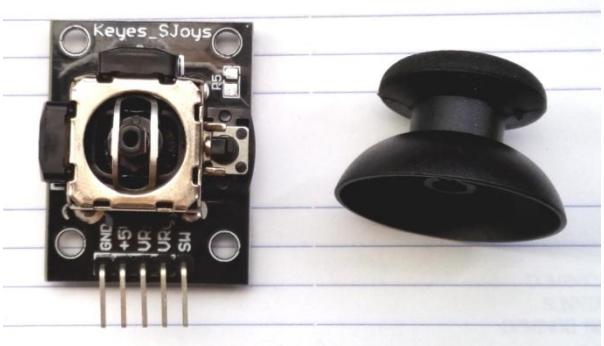
Il rilevatore e' dotato di cinque pin contrassegnati, partendo da sinistra, dalle scritte GND, +, SW, DT, CLK.

I pin GND e + sono utilizzati per alimentare il rilevatore, il pin SW fornisce indicazioni sullo stato del pulsante (premuto o non premuto) il pin DT indica il senso di rotazione mentre il pin CLK indica la presenza di un movimento.

Piu' in dettaglio, il passaggio da HIGH a LOW del pin CLK segnala la presenza di un movimento mentre il pin DT indica il senso di rotazione. Se in presenza di un movimento il pin DT e' LOW significa che la rotazione e' stata oraria mentre se e' HIGH significa che la rotazione e' stata antioraria.

La pressione della manopola, infine, provoca l'emissione di un segnale LOW sul pin SW (vedi anche esercizio 27 – rilevatore di rotazione e led a due colori).

Joystick PS2



Il joystick in figura e' un interessante device che, al variare della posizione del pomello, propone sulle porte vrX e vrY dei segnali analogici che possono essere interpretati come le coordinate X e Y di un piano cartesiano.

Il joystick e' anche dotato di un interruttore (che si attiva premendo il pomello) in grado di fornire un segnale digitale.

Piu' in dettaglio, se teniamo il joystick con i pin a sinistra ed immaginiamo che il quadrante di una bussola rappresenti un piano cartesiano, i segnali rilevati sulle porte analogiche quando il pomello e' posizionato agli estremi (nord, est sud ed ovest) sono:

Tabella valori:

Posizione pomello	Segnale vrX	Segnale vrY
nord	512	0
est	1023	512
sud	512	1023
ovest	0	512

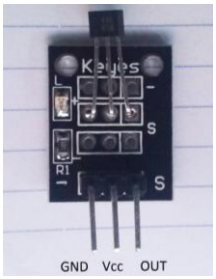
Tabella quadranti

quadrante	Segno x	Segno y
Nord/Est	+	+
Sud/Est	+	-
Sud/Ovest	-	-
Nord/Ovest	-	+

Se il pomello non e' posizionato agli estremi ma e' in una posizione intermedia tra la posizione centrale e gli estremi, i valori analogici cambiano i maniera proporzionale alla posizione del pomello (vedi anche esercizio 33). Togliendo 512 ad ognuno dei due segnali analogici ed invertendo il segno del solo segnale vrY, otteniamo due numeri il cui segno rappresenta il quadrante in cui e' posizionato il pomello (vedi tabella quadranti) ed il cui valore assoluto (da 0 a 512) rappresenta la distanza dall'intersezione degli assi.

Sensori di campo magnetico

Sensore magnetico 44E/938 montato su basetta

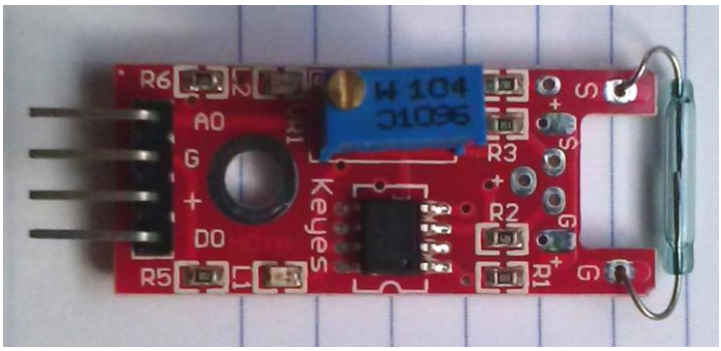


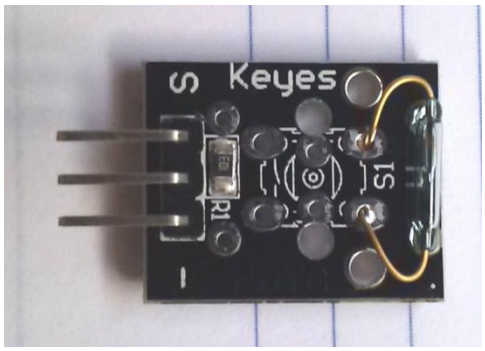
Il modulo 44E/938 e' un sensore in grado di rilevare la presenza di un magnete.

Il sensore utilizza un principio fisico (l'effetto Hall) per variare, sotto l'effetto di un magnete, la tensione presente sul piedino di uscita. Il sensore, alimentato con una tensione da 5 volt, propone sulla porta di uscita una tensione di 3,5 volt se non ci sono magneti nelle vicinanze mentre la fa scendere quasi a zero all'avvicinarsi di un magnete (vedi anche esercizio 23 – sensore magnetico).
Queste due tensioni (3,5 e zero), sono interpretate da Arduino come presenza o assenza di un segnale e pertanto possono essere utilizzate per programmare l'attivazione di eventuali attuatori. I campi di utilizzo di un sensore di questo tipo sono molteplici: dall'attivazione di un allarme quando si apre una finestra o una porta, alla verifica della posizione di oggetto all'interno di un impianto chiuso, che non puo' ad esempio essere attraversato da onde luminose o ultrasuoni.

Il sensore in figura e' montato su di una basetta che contiene al suo interno un led (che si illumina all'avvicinarsi di un campo magnetico) ed una resistenza da circa 700 ohm. I piedini di alimentazione della basetta (piedino i sinistra e piedino di centro) hanno la polarita' invertita rispetto ai piedini del sensore.

Sensore magnetico di tipo Reed, montato su basetta





I sensori Reed sono componenti elettromeccanici che lavorano usando la tecnologia dei contatti Reed. Il contatto Reed è un interruttore a lamina (normalmente aperto) che si chiude in presenza di un campo magnetico.

Nella forma più semplice è costituito da due lamine, realizzate con materiale ferromagnetico, parzialmente sovrapposte e separate tra loro di qualche decimo di millimetro. Le lamine vengono sigillate all'interno di un piccolo contenitore di vetro riempito di gas inerte (azoto o argon). Le estremità delle lamine (opposte ai contatti) fuoriescono dal contenitore e costituiscono i terminali del contatto.

In presenza di un campo magnetico le lamine diventano sede di flusso magnetico e sulle estremità si formano poli di segno opposto che tendono ad attrarsi. I contatti Reed presentano significativi vantaggi:

- i contatti sono protetti in un ambiente stagno con atmosfera inerte; questo permette di avere un'affidabilità molto elevata (fino a 100 milioni di commutazioni);
- la forza d'attrazione, una volta che le lamine si sono toccate, è molto alta, e questo riduce la generazione rimbalzi e quindi la produzione di falsi segnali.

Di fatto un modulo reed e' un interruttore che si chiude in presenza di un campo magnetico e si apre in sua assenza.

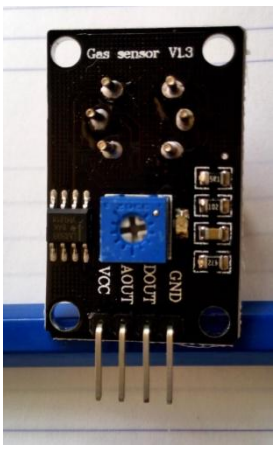
Sensori di altro tipo

Sensore tattile

Il sensore tattile e' un transistor (o almeno e' un sensore che ha la forma di un transistor) la cui gamba centrale e' ripiegata su di esso. La gamba centrale e' in realta' un'antenna in grado di percepire, al semplice contatto, le onde elettromagnetiche emesse dal corpo umano.

Le onde rilevate dal contatto vengono amplificate e, se superiori ad soglia definita da una resistenza variabile, producono un segnale digitale interpretabile da Arduino. Il sensore tattile in figura e' montato su di una basetta keyes sulla quale sono anche presenti un comparatore, una resistenza variabile, sei resistenze fisse e due led (vedi anche esercizio 31).

Sensore di gas



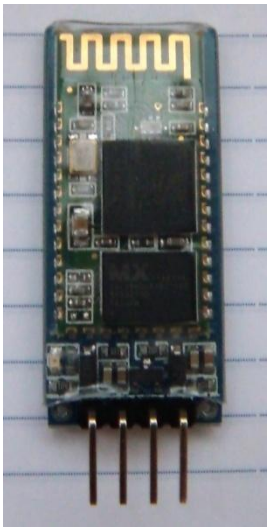
Il sensore di gas rappresentato in figura rileva la presenza di gas combustibili e fumo. Genera un segnale digitale quando la concentrazione di detti gas supera una soglia prestabilita.

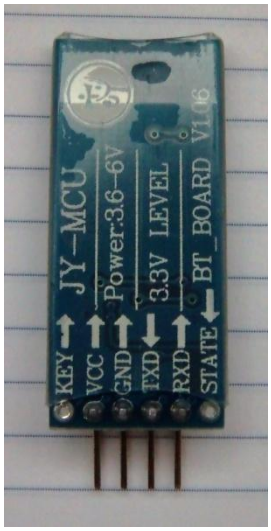
Il sensore genera anche un segnale analogico il cui valore e' proporzionale alla concentrazione dei gas. Si tratta di un sensore a semiconduttore (SnO2), la cui conducibilita' viene modificata dall'assorbimento dei gas a contatto con la sua superficie, riscaldata elettricamente a una prestabilita temperatura. Alcuni sensori sono in grado, a seconda della temperatura della superficie porosa del semiconduttore, di individuare il tipo di gas presente nell'aria.

Il sensore in figura non e' in grado di distinguere tra i vari tipi di gas ed e' montato su di una basetta sulla quale trovano posto anche un led, un potenziometro, alcune resistenze ed un comparatore. Il potenziometro consente di "tarare" il sensore in modo che l'uscita digitale venga attivata solo quando il comparatore rileva che la concentrazione dei gas (e quindi il segnale generato dal sensore) supera la soglia predefinita dal potenziometro (vedi anche: esercizio 33).

Attenzione: il segnale digitale e' HIGH quando non viene rilevata la presenza di idrocarburi e viceversa e' LOW quando la loro presenza supera la soglia regolata dal potenziometro.

HC-06 scheda di connessione bluetooth





Una connessione bluetooth e' un mezzo semplice, economico e potente per far interagire apparecchiature diverse. Un'applicazione classica e' un sensore che, via bluetooth, trasmette i segnali ad un'apparecchiatura (tipicamente un attuatore) collocata in un'altra stanza o addirittura a qualche decina di metri di distanza.

La scheda in figura e' del tipo HC06 ed e' abbastanza semplice da utilizzare.

Questi i passi per stabilire una connessione tra arduino ed un telefono android

Si connette la scheda HC06 ad arduino, la si alimenta ed il led di cui e' dotata lampeggera' velocemente, segno che e' attiva ma che non e' ancora associata ad un dispositivo bluetooth.

Si attiva la connessione bluetooth sul telefono, (impostazioni -> bluetooth). Toccando *bluetooth* si apre, sempre sul telefono, la schermata con l'elenco dei dispositivi a distanza di rilevamento. Si seleziona *HC-06* e inserisce il pin *1234*. Si attiva quindi, sempre sul telefono, l'app *bluetooth terminal* (gratuita, scaricabile da google paly) e si seleziona dal menu' in alto a destra la "connessione sicura" (o anche la connessione insicura, che funziona benissimo). Si stabilira' immediatamente la connessione, il led sulla scheda bluetooth diventera' fisso ed il sistema sara' pronto ad operare. (vedi anche esercizio 35)

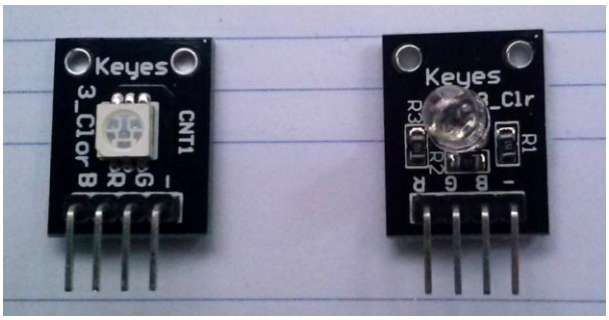
Gli attuatori

Sorgenti luminose, infrarosse e laser

Led

I Led (rossi verdi gialli o di altro colore) sono diodi luminosi, e cioe' componenti elettrici monodirezionali che si illuminano nel momento in cui viene fatta circolare al loro interno una corrente da 5 volt.
La gamba lunga (anodo) viene normalmente collegata ad una porta digitale mentre quella corta (catodo) viene collegata a terra. Sono normalmente utilizzati in un circuito da 5 volt limitato da una resistenza da 220 ohm (vedi anche esercizio 2 – led lampeggiante ed esercizio 3 effetto fade).

Led RGB



Un modulo RGB racchiude suo interno tre minuscoli led (uno rosso, uno verde ed uno blu) singolarmente pilotabili.

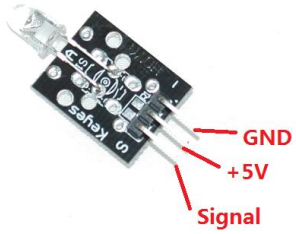
Ogni modulo RGB ha un catodo (il negativo) in comune e tre anodi, uno per ogni led.

Nella foto sono raffigurati due differenti tipi di led rgb: uno a superficie piatta ed un secondo avente la tradizionale forma di un diodo luminoso. I due componenti, seppur diversi nell'aspetto, offrono le medesime prestazioni.

Sono cioe' capaci di assumere colori differenti a seconda della tensione di alimentazione di ognuno dei tre anodi (vedi anche esercizio 3bis led RGB).

Ogni anodo infatti, alimentato da una tensione che varia da 0 a 5 volt, illumina il relativo led con una saturazione piu' o meno elevata a seconda della tensione ricevuta. E' quindi abbastanza facile riprodurre i vari colori bilanciando opportunamente la tensione su ogni singolo anodo.

Generatore di raggi infrarossi



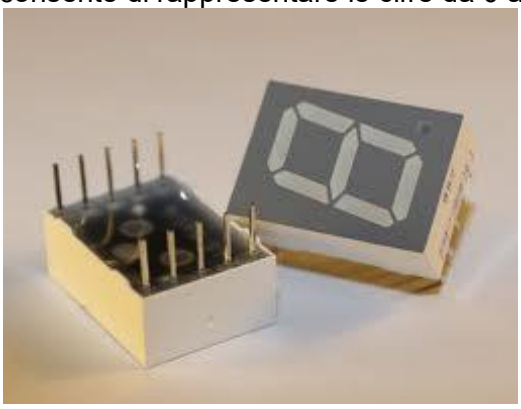
Un generatore di infrarossi non si differenzia, nell'aspetto, da un normale diodo bianco ma emette una luce non visibile all'occhio umano. In foto e' rappresentato un modulo keyes 005 sul quale e' premontato il generatore di infrarossi.

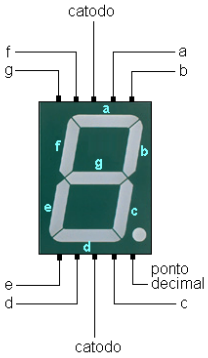
Al posto di un modulo keyes e' possibile usare anche un normale emettitore di infrarossi limitato da una resistenza da 100 ohm. Il modulo keyes KY 005 e' una basetta sulla quale e' stato saldato il generatore. La basetta non contiene solo il generatore, ma anche la relativa resistenza di limitazione per cui e' possibile collegarla direttamente ad Arduino.

Con un generatore di infrarossi e' possibile clonare la maggior parte dei telecomandi di uso comune (vedi esercizio 17 bis).

Cifra digitale

La cifra digitale e' un interessante composizione di segmenti led la cui illuminazione selettiva consente di rappresentare le cifre da 0 a 9. Con il modulo da una cifra e' possibile comporre i





i numeri illuminando o spegnendo i piccoli segmenti led di cui e' dotato. Il modulo e' composto da 7 segmenti led e da un ottavo led che rappresenta il punto decimale.

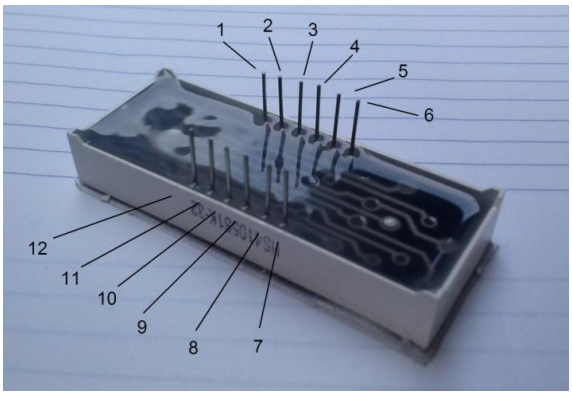
Dispone di 5 pin su ognuno dei due lati corti. Il pin centrale di ognuna delle due file e' il catodo mentre i restanti 8 sono anodi e pilotano, ognuno, un singolo segmento led ed il punto

decimale. La figura di destra illustra la relazione tra ogni singolo pin ed il segmento led pilotato.

Per utilizzare il modulo e' sufficiente collegare uno dei due catodi ad una resistenza da 220 ohm a sua volta collegata alla porta 3,3 volt e collegare gli altri 8 pin ad altrettante porte digitali. Sara' poi il programma a "scrivere" le cifre accendendo (o spegnendo) di volta in volta i vari segmenti luminosi.

Attenzione: poiche' il catodo e' collegato al polo positivo, il singolo segmento led si accende quando la relativa porta viene dichiarata "LOW", mentre si spegne quando viene dichiarata "HIGH". Si tratta, in pratica, di un pilotaggio "inverso" rispetto quello utilizzato per accendere o spegnere un normale led, ove il catodo e' collegato al polo negativo (gnd) e l'anodo e' collegato alla porta digitale che lo gestisce (vedi anche esercizio 11 – countdown).

Display led a 4 cifre



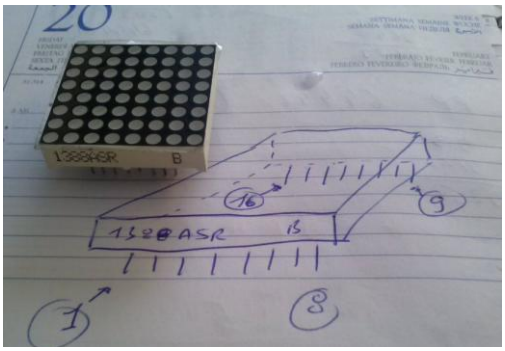
Il display led da 4 cifre HS410561K-32 e' caratterizzato da 4 anodi (uno per ogni cifra) e da 8 catodi, comuni a tutte le cifre, che pilotano i vari segmenti.

Tecnicamente si gestisce una cifra per volta e per effetto della velocita' del microcontrollore e della "latenza" nella luminosita' dei led, le cifre appaiono tutte contemporaneamente accese (vedi anche esercizio 12 - timer).

Tenendo il display con la parte luminosa rivolta verso l'alto ed i punti decimali in basso (figura di sinistra), la numerazione, da 1 a 12, parte dal primo piedino in alto a destra e procede in senso antiorario.

Questa la correlazioni tra i piedini e la loro funzione:
1 catodo del segmento verticale alto destro
2 anodo della terza cifra partendo da sinistra
3 anodo della seconda cifra partendo da sinistra
4 catodo del segmento verticale alto sinistro
5 catodo del segmento orizzontale alto
6 anodo della prima cifra da sinistra
7 catodo del segmento verticale basso sinistro
8 catodo del segmento orizzontale basso
9 catodo del punto decimale
10 catodo del segmento verticale basso destro
11 catodo del segmento orizzontale centrale
12 anodo della quarta ed ultima cifra partendo da sinistra
In commercio esistono altri moduli led da quattro cifre i cui piedini potrebbero avere una differente mappatura. Tale mappatura, qualora non fossero reperibili informazioni in proposito, puo' essere realizzata utilizzando una pila da 1,5 volt, un paio di cavetti, una breadboard e un po' di pazienza.

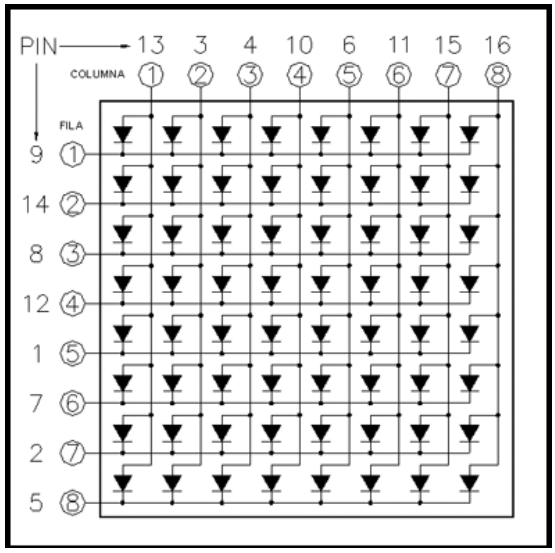
Matrice led 8X8



Attraverso la matrice led 8x8 possono essere visualizzate lettere e figure stilizzate. La matrice descritta in queste note e' la 1388ASR, caratterizzata da due file di 8 piedini. Ogni piedino e' un anodo che pilota una riga oppure un catodo che pilota una colonna. I singoli led si accendo solo quando la riga su cui il led risiede e' HIGH e la colonna e' LOW. Numerando i piedini come in figura (assegnando cioe' il numero 1 al piedino basso di sinistra e procedendo in senso antiorario) la matrice di correlazione tra piedini e righe/colonne pilotate e' la seguente:

Piedino della matrice 8x8	Riga pilotata (anodo)	Colonna pilotata (catodo)	Da connettere alla porta di Arduino (*)
1	5		4
2	7		11
3		2	12
4		3	7
5	8		14 (A0)
6		5	2
7	6		10
8	3		3
9	1		17 (A3)
10		4	13
11		6	6
12	4		8
13		1	9
14	2		16 (A2)
15		7	15 (A1)
16		8	5

(*) l'ultima colonna della tabella riporta il numero della porta di Arduino cui il pin deve essere collegato per eseguire gli esercizi 14 e 15.



Per limitare le tensioni circolanti nel circuito e per evitare di danneggiare i led e' opportuno frapporre, tra i catodi (i pin delle colonne) e le relative porte, una resistenza da 220 ohm.

Sotto il profilo tecnico la gestione di ogni singolo led e' resa possibile dal fatto che ogni piedino pilota un'intera riga o un'intera colonna del display.

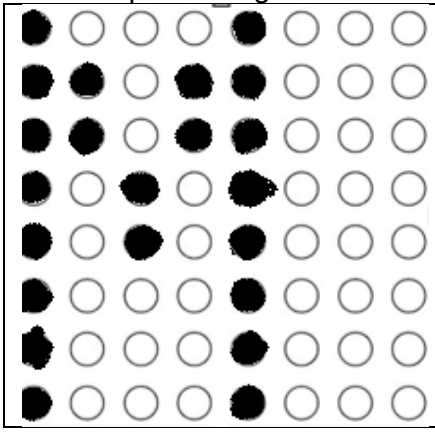
Come gia' detto, i pin che pilotano le righe sono degli anodi mentre quelli che pilotano le colonne sono catodi. Il singolo led, all'incrocio di ogni riga e colonna, si accende quindi solo quando la polarita' e' positiva sulla riga e negativa sulla colonna mentre resta spento in ogni altro caso.

Il programma gestisce ovviamente un led per volta ed il trucco adottato per gestire i 64 led in maniera apparentemente contemporanea si basa sulla persistenza della luminosita' di ogni singolo led.

La persistenza e' tale per cui Arduino puo' pilotare, in sequenza, ognuno dei led e giungere alla gestione del sessantaquattresimo led quando la luce del primo e' ancora visibile (vedi anche esercizio 14 ed esercizio 15).

Unendo quindi la persistenza della luminosita', la velocita' del microprocessore e la corretta polarizzazione delle righe e delle colonne, e' possibile gestire ogni singolo led facendolo illuminare per una frazione di tempo minima ma sufficiente a mantenerlo luminoso sino al successivo ciclo di refresh (fino alla successiva illuminazione).

Nell'esercizio 15 ogni carattere o disegno e' rappresentato in una tabella da 65 bytes disegnata come uno schema da 8 righe per 8 colonne (il primo byte non viene considerato). Ogni byte rappresenta un led e se nella tabella il byte viene posto a 1, il relativo led si illuminera' quando la tabella verra' presa in carico dalla routine di visualizzazione. In questa figura e' rappresentato lo schema per il disegno del carattere "M" e la relativa tabella di programma:



```
int carM [65] = {
  99, // il primo campo, con valore "99" e' un filler
  1,0,0,0,1,0,0,0,
  1,1,0,1,1,0,0,0,
  1,1,0,1,1,0,0,0,
  1,0,1,0,1,0,0,0,
  1,0,1,0,1,0,0,0,
  1,0,0,0,1,0,0,0,
  1,0,0,0,1,0,0,0,
  1,0,0,0,1,0,0,0,
};
```

Display a cristalli liquidi 1602

Il display 1602 e' una versatile interfaccia di output sulla quale possono essere visualizzati messaggi di due righe da 16 caratteri. Il modulo e' facilmente gestibile attraverso il controller Hitachi HD44780, emulato da Arduino tramite uno specifico software presente nelle sue librerie.

Le funzionalita' dei 16 pin del display sono illustrate nella seguente tabella:

1	VSS - collegamento a terra	9	D2 data bus - bit 2
2	VDD - alimentazione 5 volt	10	D3 data bus - bit 3
3	V0 - regolazione del contrasto	11	D4 data bus - bit 4
4	RS – register select: High=dati, Low=istruzioni	12	D5 data bus - bit 5
5	R/W - High=Read; Low=Write	13	D6 data bus - bit 6
6	Abilita (High) o disabilita (Low) l'ingresso di un byte	14	D7 data bus - bit 7
7	D0 data bus - bit 0	15	BL-A retroilluminazione (5 volt)
8	D1 data bus - bit 1	16	BL-K retroilluminazione (gnd)

In linea di massima e' possibile dire che la configurazione dei pin da 7 a 14 (che possono essere HIGH oppure LOW e cioe' 0 oppure 1) rappresenta il byte che si intende visualizzare. Il display legge, memorizza in un registro e visualizza il byte seguendo le istruzioni che gli vengono fornite attraverso i pin 4, 5 e 6.

I pin da 1 a 3 sono utilizzati per alimentare il display e per regolare (mediante un potenziometro) il contrasto dei caratteri.

Il pin 4 (register select) e' un selettore che, se dichiarato High predispone il registro del display a ricevere un byte mentre se dichiarato Low tiene il registro in attesa del prossimo evento.

Il pin 5 stabilisce la modalita' di utilizzo del registro. Se e' low il registro e' in ricezione (write) mentre se e' high il registro rende disponibile l'immagine del byte appena memorizzato (HIGH). Se non si ha necessita' di interrogare il byte presente nel registro conviene mantenere questo pin costantemente LOW e quindi collegato a terra.

Il pin 6 abilita o disabilita l'ingresso e la visualizzazione del byte

I pin da 7 a 14 rappresentano, come detto, la configurazione del byte da visualizzare

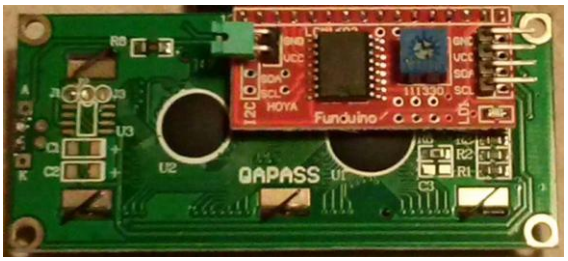
I pin 15 e 16 sono utilizzati per illuminare il fondo del display. Devono essere collegati all'alimentazione da 5 volt ed alla terra o, eventualmente, ad un interruttore. Il pin 15 puo' ovviamente, anche essere collegato ad una porta di Arduino e pilotato da programma (es. da una routine tipo "luce crepuscolare")

Il display puo' lavorare in modalita' 8 bit oppure in modalita' 4 bit; se lavora in modalita' 8 bit puo' visualizzare tutto il set di caratteri ascii (maiuscole, minuscole e caratteri speciali), ma richiede l'utilizzo di 11 porte di Arduino mentre se lavora in modalita' 4 bit utilizza solo 6 porte (4 per i dati, una per il selettore ed una per abilitare/disabilitare la visualizzazione).

Per mostrare un testo formato da numeri, caratteri maiuscoli o minuscoli e punteggiatura e' sufficiente utilizzare la modalita' a 4 bit (esercizio 18 – display lcd 1602).

Per utilizzare al meglio il display ed eliminare possibili falsi contatti, e' opportuno saldare al display una testata da 16 pin oppure 16 cavi facendo attenzione che siano, per dimensione e lunghezza, compatibili con le porte di arduino o con la breadboard.

Display a cristalli liquidi con driver LCMI 602 IIC



Come abbiamo visto, per pilotare un display lcd sono necessarie almeno sei porte digitali. Con queste premesse, vista la complessita' dei collegamenti e la relativa penuria di porte disponibili su arduino uno, il normale display lcd pur interessante in se e', di fatto, un oggetto quasi inutilizzabile.

E' stato pero' creato un driver, chiamato LCMI 602 IIC (o 2C) in grado di gestire il display utilizzando solo due porte **analogiche** (per eventuali altri tipi di driver e' possibile trovare qualche utile suggerimento qui (click)).

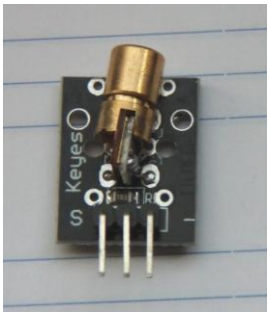
Con questo driver e con una nuova libreria (che sostituisce le preesistenti librerie per i display lcd) e' possibile gestire con semplicita' ogni problematica d'uso del display (esercizio 18bis – display lcd con driver LCMI 602).

Prima di compilare il programma che utilizza questo tipo di display bisogna:

- Scaricare l'ultima versione della nuova libreria di gestione del display a cristalli liquidi, reperibile qui (click).

- Installarla mantenendola compressa (IDE-> sketch-> importa libreria-> add library-> individuare la cartella di download-> fare doppio click sulla libreria liquidCrystal_Vx.x.x.zip)
- Cancellare ogni altra preesistente libreria di gestione dei display LCD (la nuova libreria ne incorpora le funzioni) andando in c:->programmi (x86) -> arduino->libraries, selezionando la vecchia libreria (dovrebbe chiamarsi liquidcrystal) e premendo il tasto “canc”.
- Riavviare l’IDE per rendere operativa la nuova libreria

Illuminatore laser montato su basetta



Da wikipedia: Il laser è un dispositivo in grado di emettere un fascio di luce coerente, monocromatica e, con alcune eccezioni, concentrata in un raggio rettilineo estremamente collimato attraverso il processo di emissione stimolata.

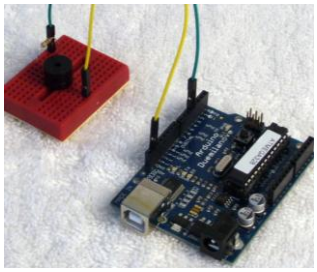
Inoltre la luminosità (brillanza) delle sorgenti laser è elevatissima a paragone di quella delle sorgenti luminose tradizionali.

Il laser trova impiego in applicazioni avanzate in numerosi campi (medico, militare, telecomunicazioni, media ed altri) ma tuttavia, per una strana legge del contrappasso, negli esercizi contenuti in questa raccolta il laser e’ stato utilizzato per dare vita ad un applicativo che trasforma i caratteri battuti sulla tastiera di un pc, negli ormai desueti segnali morse (vedi esercizio 26).

Sotto il profilo dell’impianto, l’utilizzo di un generatore laser non si differenzia molto dall’utilizzo di un led. Per pilotarlo e’ infatti sufficiente collegarlo ad una porta di Arduino e ad una fonte di alimentazione da 5 volt cc.

Sorgenti di suono

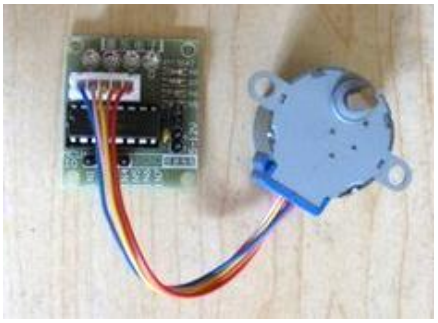
Buzzer attivo e passivo



Un buzzer attivo (marchiato HXD e con il fondo nero) e’ in grado di generare un suono senza particolari accorgimenti, Basta attivarlo o disattivarlo, esattamente come si attiva o si disattiva un led (vedi esercizio 6 - buzzer) . Un buzzer passivo (marchiato HX e con il fondo verde) e’ in grado di modulare un suono ma il programma che lo pilota deve generare il tono. Con sufficiente hardware e software e’ possibile fargli suonare musica. I buzzer sono monodirezionali, sono unita’ di output (sono cioe’ degli “attuatori”) e vengono utilizzati in circuiti da 5 volt limitati da una resistenza da 220 ohm.

Sorgenti di movimento

Motore passo passo



Un motore passo passo e' un'unita' elettromeccanica (composta da un motore a quattro poli ed un riduttore) in grado di convertire gli impulsi elettrici in movimenti meccanici.

Ogni impulso fa ruotare l'albero motore di un certo numero di gradi, predefiniti e sempre uguali.

Grazie a queste caratteristiche e' possibile utilizzare un motore passo passo per far compiere movimenti di grande precisione, in termini di angolo di rotazione, ad apparecchiature complesse, come ad esempio un braccio meccanico o un selettore. Agendo sulla frequenza degli impulsi e' anche possibile controllare la velocita' di rotazione.

Con un motore passo passo inoltre non ci si deve mai preoccupare di installare sistemi di rilevamento e controllo della posizione delle apparecchiature ad esso collegate (la posizione del braccio meccanico, ad esempio) poiche' e' sufficiente contare il numero degli impulsi inviati per poter calcolare l'esatta posizione di ogni elemento azionato dal motore.

In figura viene proposto il motore passo passo 28YBJ-48, con riduttore incorporato. Caratteristiche:

- motore a 4 fasi
- alimentazione 5-12 volt, da una fonte che puo' essere esterna ad Arduino
- assorbimento: 320 mA
- rapporto di riduzione 1/64

Un motore passo passo puo' operare in due modalita': a 64 oppure a 32 impulsi per rotazione completa dell'albero motore. Le librerie di Arduino gestiscono la modalita' a 32 impulsi per cui ad ogni impulso l'albero ruota di 11.25 gradi. Il motore e' pilotato da una scheda driver che, operando sotto il controllo di Arduino, lancia gli impulsi necessari al movimento. Poiche' il rapporto di riduzione e' 1/64, per una rotazione completa del perno in uscita sono necessarie 64 rotazioni dell'albero motore e quindi 32*64 = 2048 impulsi. La connessione tra il driver ed il motore e' assicurata da un cavo a 5 fili che termina in uno spinotto da inserire nell'alloggiamento bianco presente sulla scheda driver.

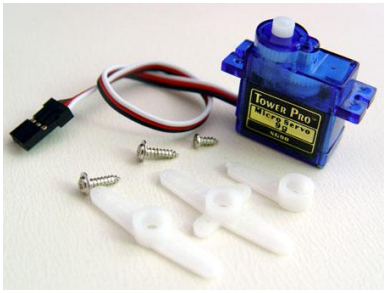
La connessione ad Arduino e' realizzata attraverso i 4 pin (In1, In2, In3, In4) di input. L'alimentazione (da 5 a 12 volt, eventualmente esterna ad Arduino), corre invece sui due spinotti posti nella parte bassa della scheda, contrassegnati da un – ed un + .

Per ottenere il corretto funzionamento del motore, bisogna attivare e disattivare le porte di ingresso (In1 – In4) rispettando la sequenza 1, 3, 2, 4. In realta' tutte le tematiche relative alla produzione ed al rilascio dei singoli impulsi sono gestite dal driver e dalle routine presenti nelle librerie di Arduino per cui la gestione pratica di un motore passo passo e' piuttosto semplice.

E' infatti sufficiente fornire alle routine presenti in libreria alcune informazioni iniziali quali il numero di impulsi necessari per un giro completo dell'albero motore ed i numeri delle porte di connessione ad Arduino con la relativa sequenza di attivazione.

Per ogni movimento bisogna poi fornire il numero di impulsi e la frequenza desiderata (in termini di impulsi al secondo). Il senso di rotazione (orario o antiorario) e' gestito dal segno associato al numero di impulsi. Se il numero di impulsi e' positivo l'albero motore gira in senso orario mentre se e' negativo gira in senso antiorario (vedi anche esercizio 19 – motore passo passo).

Servomotore



Il servomotore microservo 9g (sg90) puo' far ruotare il perno centrale di 180 gradi partendo dalla sua posizione zero. Viste le dimensioni e' dotato di un discreto momento torcente (1.2 kg/cm) e viene venduto insieme ad alcuni bracci forati mediante i quali e' possibile azionare il dispositivo che si intende pilotare.

Arduino gestisce il servomotore tramite una specifica libreria mediante la quale si possono impartire ordini di spostamento fornendo il numero dei gradi che si intende raggiungere mantenendo sempre come riferimento la posizione 0.

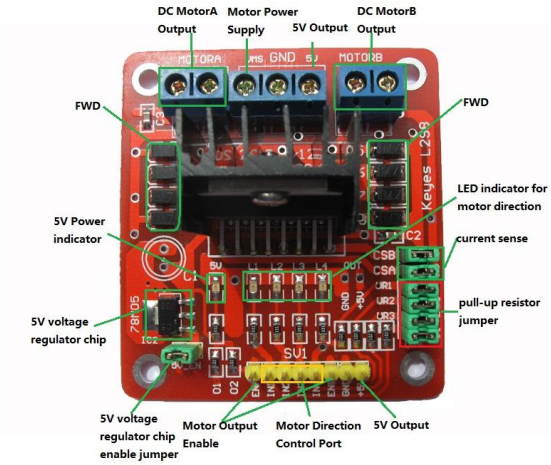
Non e' quindi importante conoscere la posizione del perno per impartire un ordine, ma solo sapere (in termini di gradi angolari) la posizione finale che deve essere assunta dal perno (vedi anche esercizio 20 - servomotore).

Modulo L298 per la gestione di due motori a spazzola

Il modulo L298 e' in grado di controllare due motori a corrente continua (i classici motori a spazzola) oppure un solo motore a 4 poli (un motore passo passo).

Caratteristiche principali del modulo in figura:

- Tensione: da 6 a 35 volt cc
- Assorbimento di picco: fino a 2 amp
- Potenza massima erogabile: 25 watt



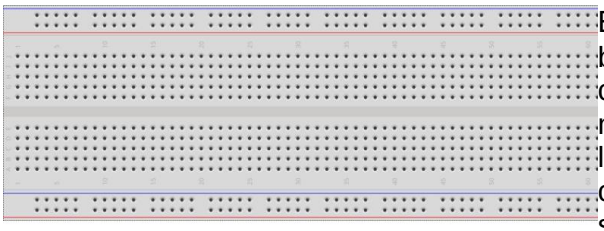
I motori a spazzola, ancorche' associati ad un importante riduttore, non sono in grado di offrire le medesime performance, in termini di controllo della rotazione del perno in uscita, offerti dai motori passo passo. Attraverso il modulo L298 e' comunque Possibile ottenere un certo controllo agendo sul tempo di attivazione e, attraverso le porte ENA ed ENB utilizzate con tecnica PWM, anche sul voltaggio erogato a ciascuno dei due motori. Fornendo tensioni diverse ai due motori e' possibile compensare eventuali differenze di efficienza ed ottenere quindi performance simili su entrambi i motori (vedi anche esercizio 22- modulo L298) .

Tabella utilizzo delle porte

VMS e GND		Da connettere ad un alimentatore esterno da 6 a 35 volt cc
ENA	input	attiva/disattiva (HIGH/LOW) il motore A e, se connesso ad una porta pwm, ne controlla anche la velocita'
IN1	input	Se IN1 e' HIGH ed IN2 e' LOW, il motore A gira in un senso
IN2	input	Se IN1 e' LOW ed IN2 e' HIGH, il motore A gira in senso contrario
ENB	input	attiva/disattiva (HIGH/LOW) il motore B e, se connesso ad una porta pwm, ne controlla anche la velocita'
IN3	input	Se IN3 e' HIGH ed IN4 e' LOW, il motore B gira in un senso
IN4	input	Se IN3 e' LOW ed IN4 e' HIGH, il motore B gira in senso contrario
MOTORA	output	Alimentazione del motore A. Non e' prevista alcuna differenziazione tra polo positivo e polo negativo poiche' la polarizzazione e' guidata dalla configurazione dei pin IN1 e IN2.
MOTORB	output	Alimentazione del motore B. Non e' prevista alcuna differenziazione tra polo positivo e polo negativo poiche' la polarizzazione e' guidata dalla configurazione dei pin IN3 e IN4.
5v e +5v	output	Pin dai quali e' possibile prelevare tensione a 5 volt per alimentare eventuali apparecchiature aggiuntive

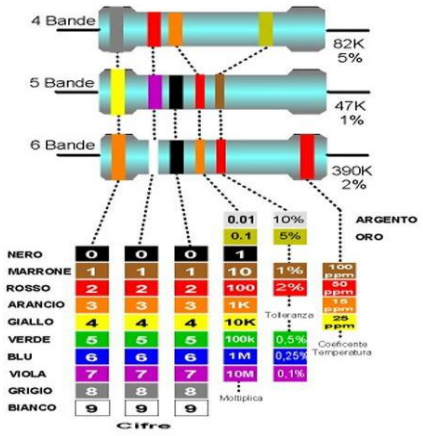
I componenti di supporto

Il banco di lavoro (breadboard)



E' indispensabile per gli esercizi e per i prototipi. Sul banco di lavoro possono essere posizionati i componenti e la cavetteria di collegamento. I fori nelle prime e nelle ultime due file sono collegati tra loro in senso orizzontale e possono essere utilizzati come linee di alimentazione mentre i restanti fori sono collegati in senso verticale (i cinque fori superiori sono comunque indipendenti dai cinque fori inferiori) e su di essi devono essere posizionati i componenti ed i cavi di collegamento del circuito.

Resistenza



Le resistenze sono utilizzate per limitare la *quantita'* (voltaggio e/o amperaggio) di corrente circolante nel circuito in cui sono inserite. Sono bidirezionali non hanno cioe' un polo positivo ed uno negativo e, nei circuiti a corrente continua, sono soggette alla seguente formula:

resistenza = voltaggio / intensita'

Nei circuiti gestiti da Arduino sono normalmente utilizzate resistenze da 220, 1000 e 10k ohm
Su ogni resistenza l'impedenza (l'unita' di misura della resistenza) e' definita da anelli colorati (vedi foto). L'utilizzo di un tester pero' rappresenta il piu' pratico metodo per la misurazione dell'impedenza.

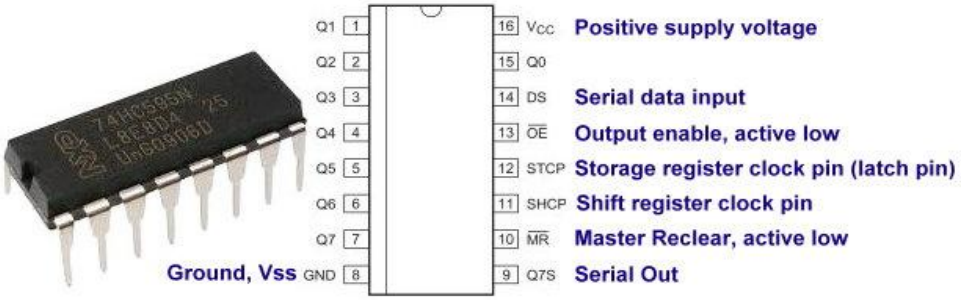
Resistenza variabile

La resistenza variabile, altrimenti detta potenziometro o trimmer, e' una resistenza dotata di una manopola attraverso la quale e' possibile modificarne l'impedenza. Collegando i pin estremi alla terra ed all'alimentazione da 5 volt si ottiene, sul pin centrale, un voltaggio che va da zero a 5 volt a seconda dell'impedenza incontrata. Quando l'impedenza e' al massimo il voltaggio e' pari a zero e viceversa. Puo' essere utilizzata in una circuito in cui si vuole poter regolare manualmente il flusso (o meglio il voltaggio) della corrente (vedi anche esercizio 8 – voltmetro su scala luminosa).

Pulsante a quattro piedini

In un pulsante a quattro piedini, I piedini opposti sono collegati tra loro mentre quelli in linea si collegano solo se si tiene premuto il pulsante (tipo campanello). Sono tipicamente delle unita' di input (sono in realta' dei "sensori") e possono essere utilizzati inserendoli in un circuito limitato da una resistenza da 10k ohm. Il circuito di base prevede un piedino collegato a terra (negativo) e l'altro piedino collegato ad una resistenza da 10k ohm a sua volta collegata ad una porta digitale (o anche analogica). La pressione del pulsante chiude il circuito e Arduino ne rileva la chiusura inserendo 1 (High) nella variabile utilizzata per rilevare lo stato della porta alla quale il pulsante e' collegato.

Il modulo 74HC595, il moltiplicatore di porte



Il chip 74HC595 consente di pilotare 8 porte di output utilizzando solo tre porte di input. Riducendo ai minimi termini le spiegazioni sul funzionamento di questo chip, e' possibile dire che il chip attiva o disattiva le 8 porte di uscita sulla base della configurazione binaria di un byte trasmesso da Arduino. Gli 8 bit, che compongono il byte rappresentano, per stato (0 oppure 1) e posizione (dal primo all'ottavo), la situazione che si intende creare nelle 8 porte di uscita. Di fatto Arduino invia il byte al chip che lo memorizza in un registro di entrata (*shift register*). Il byte viene poi trasferito in un registro di utilizzo detto *storage register* e da qui' utilizzato per attivare o disattivare, contemporaneamente, le 8 porte di uscita.

I 16 pin (i 16 piedini) del chip hanno la seguente funzione:

- 1-7 e 15 (da Q1 a Q7 e Q0): piedini di uscita, che vengono attivati o disattivati secondo le istruzioni ricevute da Arduino;
- 8 collegamento di terra;
- 9 (Q7S, *serial out*): piedino di uscita seriale che puo' essere collegato alla porta di entrata di un altro eventuale chip 74HC595 collegabile in cascata;
- 10 (MR, *master reclear, active low*): piedino di reset; se lo si pone in stato LOW cancella il byte memorizzato nello *shift register*. Per evitare problemi di solito viene tenuto HIGH e quindi alimentato con una tensione di 5 volt;
- 11(SHCP *shift register clock pin*): piedino per l'attivazione della fase di trasferimento del byte da Arduino allo *shif register;*

- 12 (STCP *storage register clock pin*, detto anche *latch pin*) piedino per l'attivazione della fase di trasferimento del byte dallo *shift register* allo *storage register*: quando viene dichiarato LOW viene consentito lo spostamento del byte mentre quando e' HIGH viene impedito. E' una specie di interruttore, utilizzato per decidere il momento di spostamento del byte dal registro di entrata al registro di utilizzo. Se poi il piedino 13 (il piedino OE,) e' attivo (e cioe' e' zero, LOW), il trasferimento dei dati nello shift register coincide con l'attivazione/disattivazione dei piedini di output.
- 13 (OE *output enable, active low*): piedino che consente l'utilizzo del byte per attivare o disattivare le porte di uscita. Quando e' LOW consente l'utilizzo del byte mentre quando e' HIGH non ne consente l'utilizzo. Per limitare il numero di porte utilizzate da Arduino, questo piedino viene normalmente lasciato attivo e cioe' viene mantenuto in stato LOW e quindi collegato direttamente a terra
- 14(DS, *Serial data input*) piedino sul quale viene fatto transitare (da Arduino al chip) il byte di configurazione
- 16 piedino di alimentazione, da collegare ad una tensione di 5 volt cc.

Come gia' detto, per pilotare il chip Arduino puo' limitarsi ad utilizzare tre sole porte:
- una per attivare (o meglio consentire) l'invio del byte al chip, e quindi da collegare al piedino numero 11 del chip (SHCP, *shift register clock pin*)
- una per inviare fisicamente al chip il byte di configurazione e quindi da collegare al piedino 14 del chip (DS, *Serial data input*)
- una per trasferire il byte dalla memoria di entrata alla memoria di utilizzo del chip. Da collegare alla *latch pin*, e cioe' al piedino 12 del chip.

Sotto l'aspetto della programmazione, il trasferimento del byte da Arduino al chip viene effettuato attraverso l'istruzione shiftOut, (vedi anche esercizio 13 – il moltiplicatore di porte) cosi' strutturata:

shiftOut (porta dati, porta latch, modalita' di utilizzo, byte di configurazione)

dove:
- porta dati: numero della porta di Arduino collegata al piedino 14 (DS) del chip;
- porta latch: numero della porta di Arduino collegata al piedino 11 (SHCP, oppure LATCH);
- modalita' di utilizzo: puo' assumere i valori: **MSBFIRST** or **LSBFIRST** (Most Significant Bit First (il bit piu' significativo per primo oppure Least Significant Bit First il bit meno significativo per primo) per scegliere l'ordine di assegnazione dei bit alle porte di output (da 0 a 7 oppure da 7 a 0);
- byte di configurazione: il byte che Arduino trasmette al chip.