ELEKTRONIKA

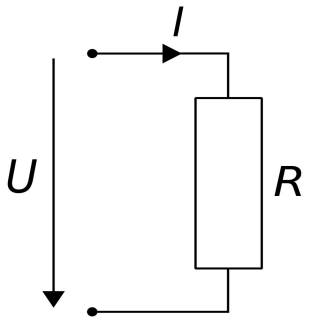
Osnovni fizikalni zakon, ki določa obnašanje električnega toka v ovisnosti od napetosti je Ohmov zakon:

$$U = I * R$$

(padec napetosti na porabniku / uporu je sorazmeren električnemu toku)

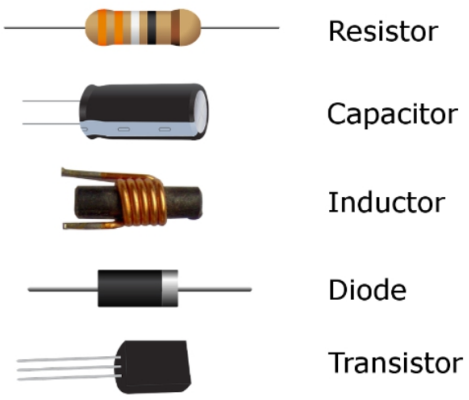
$$P = U * I$$

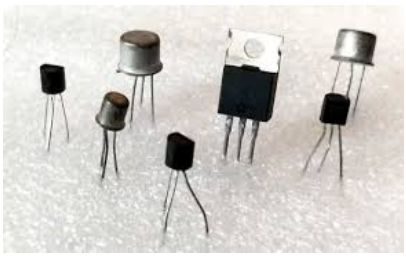
(električna moč je sorazmerna napetosti in toku)



/slika: osnovna shema električnega vezja; R je lahko upora (grelec), motor ali karkoli /

Elektronska vezja so posebni spoji osnovnih elektronskih elementov, ki povezani opravljajo neko funkcijo. Pasivni elementi so: upor, kondenzator, diode, svetilne diode, navitja, kondenzatorji, releji, zvočniki, itd. Aktivni elementi pa so tisti, ki ojačujejo ali predrugačijo energetska stanja. To so: različni transistorji in različna integrirana vezja.





/slika: pasivni in aktivni elementi; tranzistorji; integrirana vezja /

Nas zanimajo vezja, ki omogočajo raznovrstne uporabe dokaj univerzalnih mikrokontrolerjev, kot so tisti na ploščici Arduino. Pri mikrokontrolerjih (imenujemo jih tudi fizični vmesniki, saj komunicirajo s fizičnim svetom) imamo na voljo kar veliko število univerzalnih nožic (pinov). Precej jih je zelo univerzalnih in so lahko vhodi (za senzorje, potenciometre,…) ali izhodi (za svetilke, motorje, releje, …). Na te nožice po želji priključimo elektronska vezja, ki opravljajo predvideno opravilo.

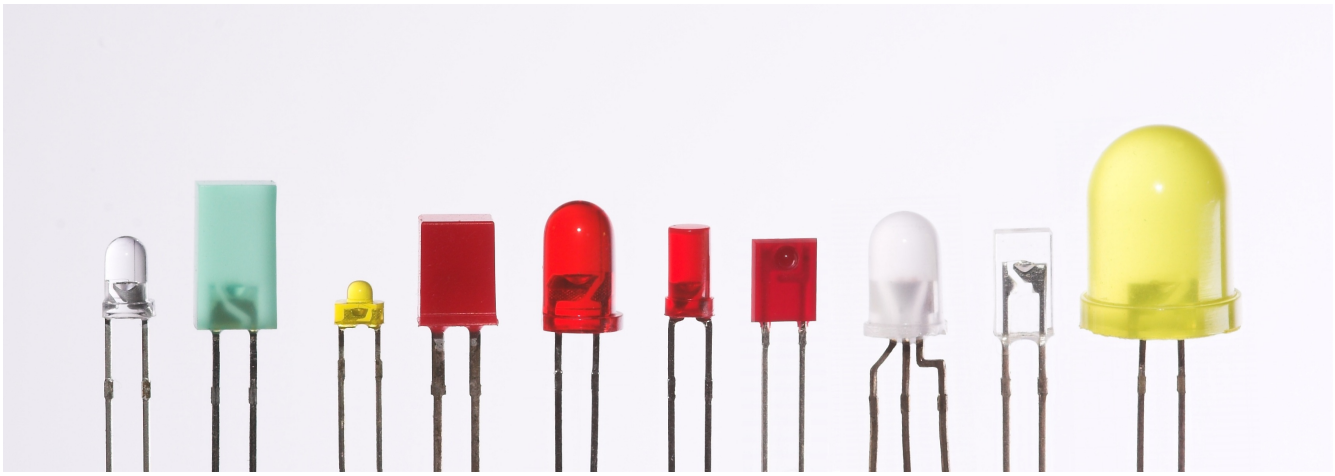
Pomemben koncept je **masa** (**zemlja**, **skupna točka**). Elektronska vezja sestavljamo iz sestavnih delov, ki so lahko prav tako že sestavljena elektronska vezja. Pri tem je potrebno ohraniti neko referenčno (skupno) točko glede na katero se vsa ta različna vezja "pripnejo". To je zemlja (ali masa) – točka z najnižjim potencialom (minus). Običajno je pritrjena na negativni pol baterije ali napajalnika. V nasprotnem primeru se medsebojni odnosi napetosti in tokov porušijo – govorimo o lebdečem / nedefiniranem stanju.

Včasih pa želimo električna vezja med seboj električno ločiti, informacijo (ali signal) pa vseeno prenesti. To storimo običajno z optokoplerjem, ki je kombinacija svetilne diode in fototranzistorja. Infomacijo torej prenesemo s pomočjo svetlobe.

Začnimo z izhodnimi opravili. V tem primeru je potrebno nožici mikrokontrolerja določiti osnovni namen kot izhod. To storimo z ukazom:

pinMode(pin, OUTPUT); // set the pin as output

SVETLOBA



/slika: različni LED elementi

Kot svetlobni vir najbolj pogosto uporabimo LED (light emitting diode), ki je polprevodniški element dioda. Diodo sestavlja t.i. PN spoj, kar sta dva materiala z različno vsebnostjo elektronov. Dioda ima en PN spoj, tranzistor pa ima dva PN spoja.

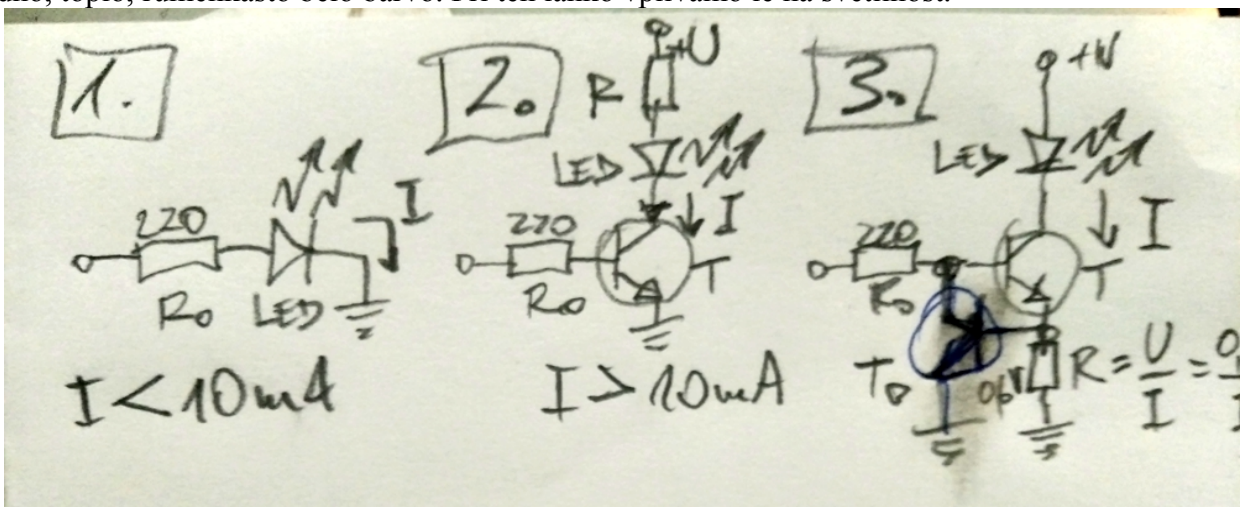
Običajna silicijeva dioda se uporablja za usmerjanje izmeničnega toka v enosmernega. LED pa je narejena iz snovi, ki del energije električnega toka pretvori v svetlobo (pojav elektroluminiscenca). Različne snovi/ materiali omogočijo različne barve svetlobe.

Fizikalna zgradba polprevodniške diode določa nekaj, kar se kaže kot pragovna napetost. Na vsaki diodi moramo to napetost preseči, da lahko steče tok. Pri slicijevi diodi (in tranzistorju) je pragovna napetost 0.6V. Pri LED je ta napetost okoli 2,2V (rdeča, zelena, rumena), ali okoli 3,2V (modra, vijoličasta, bela). Te razlike so zaradi različnih uporabljenih snovi.

LED elementi potrebujejo stalen električni tok, ki ne sme biti prevelik – zato ga moramo omejiti. Temu običajno služi upor, s katerim na podlagi Ohmovega zakona določimo tok. Za večje tokove (in večjo svetilnost LED) kot stikalo in pretvornik moči uporabimo tranzistor. Za večji nadzor toka lahko uporabimo tokovni vir, ki aktivno nadzira količino električnega toka.

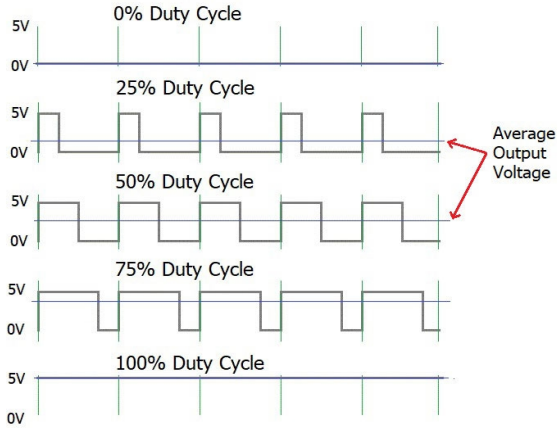
RGB (red, green, blue) LED je kombinacija treh LED elementov, s katerimi lahko z mešanjem treh osnovnih barv dobimo poljubne barve celotnega vidnega spektra človekovega vida. Vse tri barve skupaj se sestavijo v belo svetlobo.

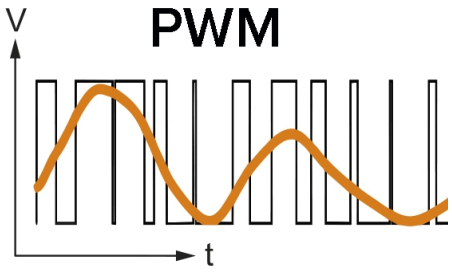
Obstajajo tudi LED bele svetlobe, ki niso sestavljene iz RGB in pri katerih so barve že zmešane v hladno, toplo, rumenkasto belo barvo. Pri teh lahko vplivamo le na svetilnost.



/slika: osnovne povezave LED; 1: brez ojačanja toka, 2: z ojačanjem in omejitvenim uporom, 3: z ojačanjem in aktivnim omejevanjem toka z lokalno povratno zanko /

Svetilnost / intenzivnost svetlobe posamezne LED lahko določimo analogno - s spreminjanjem toka, ki teče skozi LED element. Z mikrokontrolerji uporabimo drugačno tehniko: s krajšanjem in daljšanjem trajanja tokovnih impulzov.





/slika: PWM (pulse width modulation) – spreminjanje širine impulza in analognemu podobni rezultat /

Ker je zaznavanje svetlobe pri človeku kemični proces ima kar veliko inercijo. To se imenuje tudi persistenca vida (ohranjanje vidne informacije). Na ta način vidimo zaporedne slikice pri filmu kot eno gibljivo sliko. Kratki impulzi svetlobe se podobno ohranijo kot stalno šibkejša svetloba, dolgotrajnejši pa kot stalna močnejša svetloba. Najmanjša hitrost impulzov (ali posameznih slikic pri filmu) je 25 impulzov (slikic) na sekundo. Temu rečemo frekvenca = 25 Hz. Višja frekvenca ne škodi (na primer: 50 ali 60 sličic na sekundo).

Najprej definiramo način delovanja pina kot izhod:

pinMode(led, OUTPUT); // set the led pin as output

Pri mikrokontrolerju (arduino) temu načinu impulznega delovanja rečemo PWM (pulse width modulation). Ker lahko sorazmerno analogno spreminjamo količine (električne energije) so predvideli naslednji ukaz (funkcijo):

analogWrite(led, value); // set the value (range from 0 to 255) on led pin

Če bi hoteli LED polno prižgati, bi uporabili funkcijo:

digitalWrite(led, 1); // set the maximum value on led pin

Če bi hoteli LED popolnoma ugasniti, bi uporabili funkcijo:

digitalWrite(led, 0); // set the zero value on led pin

To so tri osnovne funkcije/ ukazi, ki se tičejo določanja izhodnega stanja na posameznih nožicah (pinih) mikrokontrolerja.

Te ukaze lahko uporabimo tudi za določanje hitrosti vrtenja motorja, ki v osnovi pretvori električno energijo v gibalno energijo.

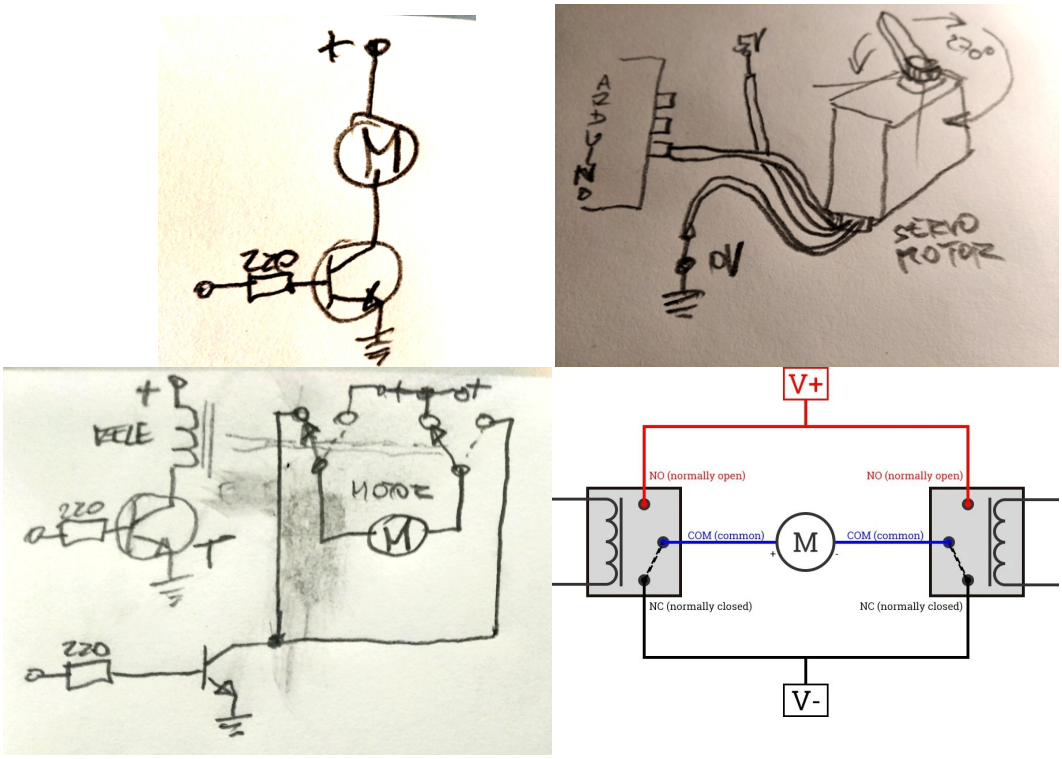
GIBANJE, VRTENJE, PREMIKANJE

/slika: poceni dc motor, dc motor z reduktorjem. servo motor; *dc: enosmerna napetost* /

Pri motorjih imamo naslednje možnosti: začetek vrtenja, zaustavitev, sprememba smeri vrtenja – in spreminjanje hitrosti vrtenja. Običajno je v malih motorjih nevrtljivi del (stator) stalen magnet, vrtljivi del (rotor) pa je tripolno navitje, ki omogoči sile, ki motor zavrtijo. Z električnim tokom določamo smer in jakost magnetnega polja v motorju – s tem pa tudi sile.

Servo motor je zanimiv, ker ima reduktor in vsebuje vso elektroniko za precizno merjenje kota obrata, ki je običajno omejen na okoli 270 stopinj. Za smer in hitrost potrebujemo le eno nožico. Vendar tu že potrebujemo dodatno kodo, ki se jo vključi kot knjižnico (library). O tem kasneje.

Motor potrebuje za vrtenje večje količine električne energije in ne potrebuje omejevanja električnega toka. S pomočjo mikrokontrolerja ga zavrtimo preko stikalnega tranzistorja. S pomočjo PWM impulzov mu lahko tudi spreminjamo hitrost vrtenja. Če želimo še spremeniti smer vrtenja, pa mu moramo spremeniti smer toka, kar naredimo s preklopom smeri napetosti na priključkih Za spreminjanje hitrosti in smeri vrtenja tako potrebujemo dve nožici mikrokontrolerja.



/slika: priklop motorja prek tranzistorja za regulacijo hitrosti vrtenja v eno smer; desno zgoraj: enostaven priklop servo motorja; levo spodaj: priklop z dodanim DPDT relejem prek tranzistorja z obojesmerno regulacijo hitrosti vrtenja (za smer je zgornji tranzistor); desno spodaj: princip H-mostiček /

Rele za obojesmerno vrtenje lahko vidimo kot črko H, zato se temu reče H-mostiček (H-bridge). Pogosti so tranzistorski H-mostički, ki imajo stikala izvedena s tranzistorji in imajo tudi dodatne uporabnosti.

Motorji za vzpostavitev magnetnih sil potrebujejo neko spodnjo mero električne energije, potem pa se vrtijo zelo hitro in na hitrost vrtenja tako lahko vplivamo le v nekih mejah. Zato v primeru želje po počasnem (in močnejšem) vrtenju uporabimo motor s prenosi (zobatimi kolesci) – reduktorjem. Ta način sorazmerno zmanjša hitrost vrtenja in poveča moč na osi motorja (navor).

Pri mikrokontrolerju (arduino) se trije ukazi za vrtenje motorja razširijo tako:

Najprej seveda definiramo način delovanja pina kot izhod:

pinMode(motor, OUTPUT); // set the motor pin as output

Če bi hoteli motor polno zavrteti, bi uporabili funkcijo:

digitalWrite(motor, 1); // set the maximum value on motor pin

Če bi hoteli motor popolnoma ustaviti, bi uporabili funkcijo:

digitalWrite(motor, 0); // set the zero value on motor pin

Če bi hoteli motorju spremeniti hitrost, bi uporabili ukaz:

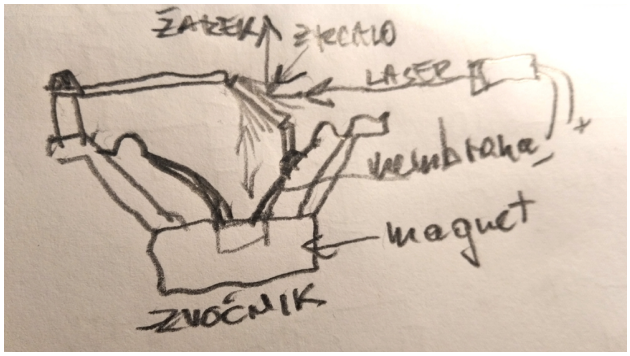
analogWrite(motor, value); // set the value (range from 0 to 255) on motor pin

Če bi hoteli motorju spremeniti smer, bi uporabili ukaz / funkcijo:
digitalWrite(direction, 1); // set the reverse rotation on direction pin

ali

digitalWrite(direction, 0); // set the forward rotation on direction pin

Za majhne kontrolirane premike lahko uporabimo zvočnik. Premike lahko s prenosi ali vzvodi spremenimo v usmerjanje svetlobe – na primer LED svetilke. Ker je zvočnik namenjen hitremu nihanju so ti premiki lahko zelo hitri – ustvarimo lahko svetlobno liso. Če premikamo ogledalce v katerega seva usmerjen svetlobni vir (led ali laser) dobimo liso ali kar črto. Če na poseben način sestavimo dva zvočnika z ogledalci lahko dobimo celo krog, osmico in še mariskaj krožnega!



/slika – zvočnik in zrcalo /

Tudi tu najprej definicija pina kot izhod:

pinMode(laser, OUTPUT); // set the laser pin as output

Poceni polprevodniški laser vsebuje LED. Svetilnost laserja tako določimo z ukazom:

analogWrite(laser, value); // set the value (range from 0 to 255) on laser pin

Tudi za zvočnik definiramo uporabo kot izhod:

pinMode(speaker, OUTPUT); // set the speaker pin as output

Premik zvočnikove mebrane pa določimo z ukazom:

analogWrite(speaker, value); // set the value (range from 0 to 255) on speaker pin

Premik membrane na ta način poteka le v eni smer – naprej (odmik), vendar zelo precizno.

V primeru, da ne želimo slišati neprestanega zvoka PWM impusov (okoli 500 Hz), lahko to na zvočniku filtriramo s kondenzatorjem. V primeru izmeničnega toka uporabimo enačbo:

$$f = 1 / (2*pi * R * C)$$

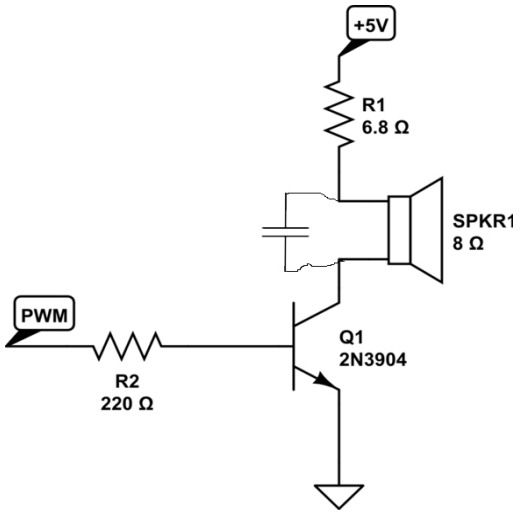
(R je pri zvočniku 4 ali 8 ohmov, f frekvenca je vse kar je višje od 50 Hz)

$$C = 1 / (6{,}28 * 8 * 50)$$

$$C = 390 \text{ uF}$$

(odločimo se za 100 uF)

Kondenzator v tem primeru uporabljamo kot hranilnik kratkoročne energije. Na ta način namesto impulzov na zvočniku dobimo spremenljivo analogno napetost – in posledično analogni električni tok. V fizičnem svetu je tako ali tako vse analogno.



/slika: zvočnik prek stikalnega tranzistorja – med sponkama zvočnika filtrski kondenzator /

ZVOK

S pomočjo impulznega PWM delovanja lahko z zvočnikom predvajamo tudi zvok. Zvočnik je elektromehanski element, ki za nihanje uporablja pretvorbo električne energije (toka) v gibalno (mehansko) energijo – prek magnetnega primikanja in odmikanja v polju močnega stalnega magneta.

Tudi zvočnik zahteva precej električne energije, zato za prenos skrbi tranzistor kot stikalo. Zvok je posledica spremembe hitrosti (frekvence) gibanja (nihanja) membrane. Za človekovo uho je najnižja frekvenca 20 Hz (nihajev v sekundi), najvišja pa 20.000 Hz (20 KHz). V primerjavi z zaznavanjem svetlobe je to precej hitreje. Tu ni persistence (ohranjanja) slišanega, saj poteka pretvorba tresljajev zvoka povsem fizično - in ne prek kemijskih pretvorb. Če bodo spremembe prepočasne (pod 20 Hz) se bo zvočnik sicer premikal, vendar zvoka ne bomo slišali.

/slika: zvočnik prek tranzistorja in zvočnik prek kondenzatorja /

Arduino PWM način deluje s stalno frekvenco impulzov, (490 Hz) – pri katerih se spreminja le trajanje teh impulsov. To je povsem v redu za prejšnje uporabe. Pri zvoku pa je potrebna spremenljiva višja frekvenca impulzov.

Najprej seveda za zvočnik definiramo uporabo kot izhod:

pinMode(speaker, OUTPUT); // set the speaker pin as output

Za kreiranje zvoka obstaja naslednji ukaz:

tone(speaker, frequency); // set the frequency on speaker pin
ali
tone(speaker, frequency, duration); **// set the frequency on speaker pin with duration time (ms)**

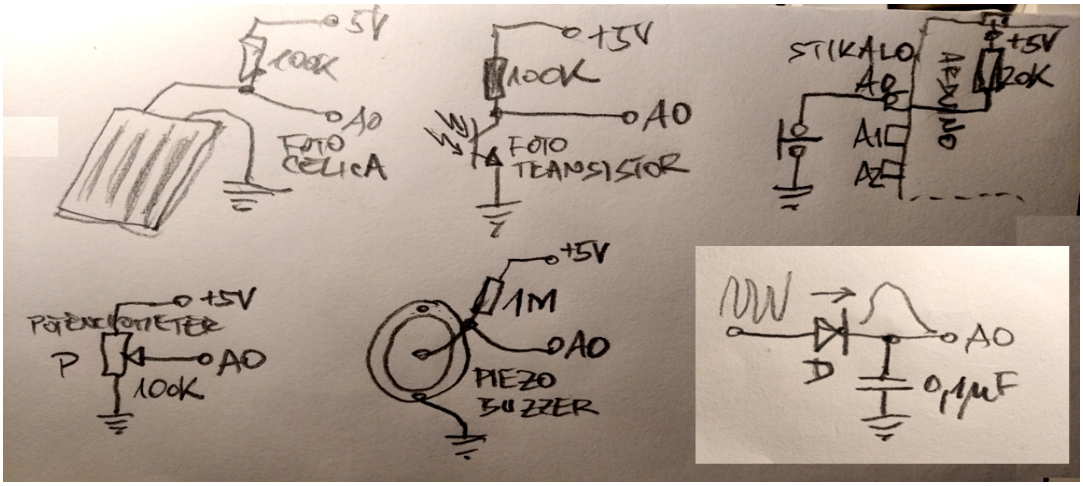
Zvok na nožici ustavimo z ukazom:

noTone(speaker) ; // stop the frequency tone on speaker pin

Najnižja frekvenca je zvoka je 31 Hz, najvišja pa celo 8.000.000 Hz (8 MHz). Arduino lahko s tem ukazom ustvari frekvenco tona le na eni nožici hkrati.

To je bilo nekaj načinov preprostega ustvarjanja premikanja, gibanja, svetlobe in zvoka – kar so vse izhodna opravila . V nadaljevanju pa nekaj o nastavitvah, kontrolah, zaznavanju - oziroma o vhodnih opravilih.

VHODI – SENZORJI IN KONTROLE



/slika: merjenje svetlobe s fotocelico, s fototranzistorjem, stikalo za logični ukaz; potenciometer (spremenljivi upor) za zvezne vrednosti; piezo element za merjenje tresljajev; spodaqj desno: diodni integrator: če nas zanima le intenzivnost sprememb (vzporedno h kondenzatorju lahko dodamo še upor 100 Kohmov – za hitrejši odziv) /

Pri vseh teh je definirana uporaba nožic (pinov) kot izhod:

pinMode(pin, INPUT); // set the pin as input

Mikrokontrolerji imajo dva načina branja vhodnih sprememb (stanj). Analogni vhodi so označeni kot A0-A5 (pri Arduino UNO) – te lahko uporabimo za merjenje zveznih (analognih) sprememb. Za skokovite / logične spremembe (0V ali 5V) uporabimo digitalni način branja vhodnih sprememb.

Na to vhodne nožice lahko priključimo marsikaj:
- za merjenje svetlobe je primeren fototranzistor ali sončna celica,
- za nastavitev (na primer: hitrosti vrtenja motorja) uporabimo potenciometer
- za spremembe v programu, ki sprožijo nekatere ukaze, uporabimo tipko
- za merjenje tresljajev ali tapkanja s prsti uporabimo piezo-senzor,
- kot zvočni senzor lahko uporabimo mali kondenzatorski mikrofon, vendar pogosto potrebuje dodatno ojačanje napetosti

Osnovni namen nožice / pina definiramo kot vhod, kar storimo z ukazom:

pinMode(pin, INPUT); // set the pin as input

Posebna pogosto uporabljana oblika vhoda je “navzgor potegnjen” vhod, kar pomeni, da je na vhodu na pozitivni pol napetosti spojen upor – ki je vsebovan v integriranem vezju. To pomeni da je na tem vhodu vedno logična enka. Prednost je predvsem v tem, da je vhod manj občutljiv za okolico, saj sicer logični vhodi divje skačejo med logično ničlo in enko. Podoben učinek dobimo, če nožico “potegnemo navzdol” – med nožico in negativni pol (maso) spojimo upor (10K-100Kohmov. V tem primeru mora biti to fizični zunanji element.

Logično ničlo imenujemo tudi LOW (nizek nivo; 0V), logično enko pa HIGH (visok nivo; 5V).

Vhod potegnemo nazgor z ukazom:

pinMode(pin, INPUT_PULLUP); // set the pin as pulled-up input using internal resistor

Med mikrokontrolerjevimi vhodi jih je nekaj, ki so označeni kot analogni. Ti vsebujejo analogno-digitalni pretvornik (DAC), s katerim lahko v programu merimo spremembe v fizičnem svetu. Arduino UNO ima te vhode označene kot A0 do A5 Z merjenje zveznih / analognih sprememb uporabimo ukaz:

analogRead(pin); // read pin input as analog value

Za merjenje logičnih sprememb pa uporabimo ukaz:

digitalRead(pin); // read pin input as logic value 0 or 1

Zaključek:

Obstaja cel niz zelo kompleksnih vhodih in izhodnih vezij, ki so posebni moduli namenjeni povezavi z mikrokontrolerji. Ti imajo lahko različne priključke, saj mikrokontrolerji omogočajo tudi t.i. serijske povezave. To so že naprednejše uporabe.