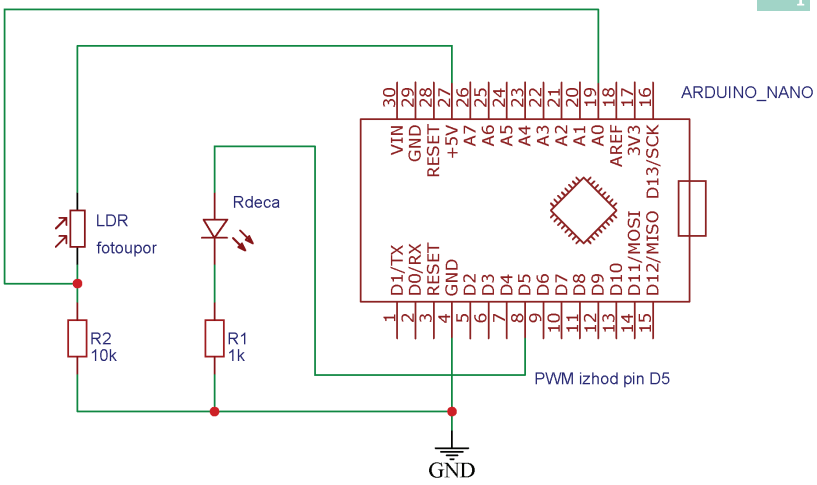
Prvi koraki v Arduino

Pulzno širinsko krmiljenje osvetljenosti

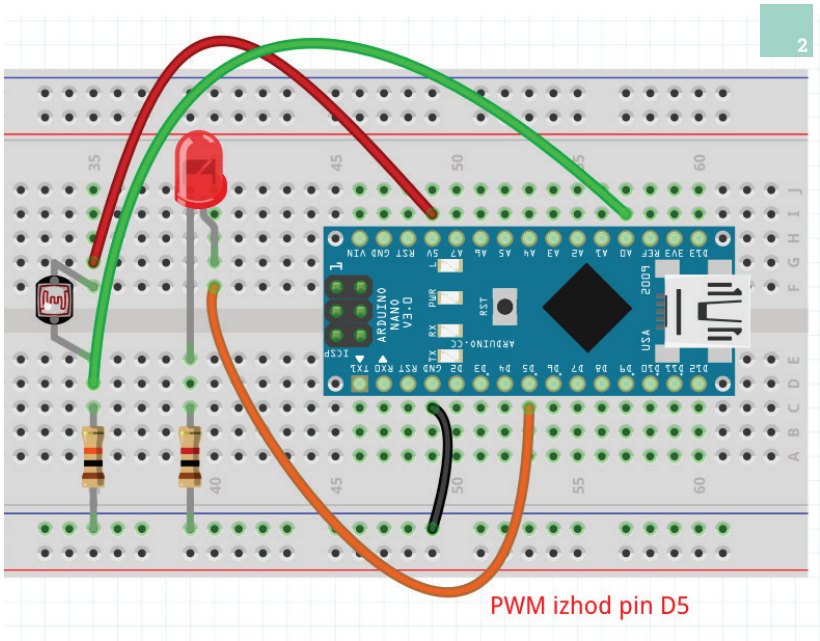
Milan Gaberšek in Slavko Kocijančič

V štirinajstem zaporednem prispevku na temo Prvi koraki v Arduino se bomo avtomobilčku posvetili le še posredno. Spoznali bomo namreč, kako s preprostim vezjem lahko spremljamo jakost vpadle svetlobe in glede na to počasi povečujemo moč svetlobe svetila, kar bomo dosegli s pomočjo pulzno širinske modulacije. Za svetilo bomo uporabili najprej svetlečo diodo in nato še žarnico. Slednja za delovanje zahteva večjo moč, kot jo lahko zagotovi krmilnik, zato si bomo pomagali s t. i. H-mostičem, ki je prvenstveno namenjen krmiljenju motorčkov. Podobno lahko avtomobilčku spreminjamo tudi hitrost vrtenja koles.

1



2



Material in orodje

- krmilnik Arduino Nano ali podoben,
- kabel USB za povezavo krmilnika z računalnikom,
- prototipna ploščica (angl. breadboard),
- fotoupor, npr. GL5539,
- vezne žičke (najbolje rdeča, črna, oranžna in zelena),
- upor 1 kΩ (rjava, črna, rdeča, zlata),
- upor 10 kΩ (rjava, črna, oranžna, zlata) ali večji,
- svetleča dioda (rdeča ali kakšne drugačne barve),

Dodatno:

- modul L9110 za dva motorja (ali primerljiv H-mostič) z veznimi žičkami,
- halogenska žarnica 12 V – 5 W ali podobna (opcijsko motorček).

Računalniški pripomočki

- osebni računalnik z nameščenim operacijskim sistemom Windows, Linux ali Mac OS,
- Arduino IDE, integrirano programsko razvojno okolje, ki je brezplačno dostopno na spletni strani *www.arduino.cc*.

Izvedba

Posamezne elemente povežemo, kot kažeta električna shema (**slika 1**) oziroma slika, narejena s pomočjo programa Fritzing (**slika 2**). Shema spominja na vezje, ki smo ga spoznali v okviru prispevka Prvi koraki v Arduino – Stik s svetom in računalnikom (januar 2020), kjer je bilo tudi podrobno razloženo njegovo delovanje. Bistvena razlika je v tem, da smo tokrat svetlečo diodo priključili na nožico 5, kar bo razloženo pozneje.

Ko je vse povezano, se lahko lotimo programiranja krmilnika Arduino. S spletne strani *www.drti.si/tim.html* presnamemo program VklopIzklopLED.ino in ga prenesemo na krmilnik Arduino. Program preskusimo tako, da pri običajni osvetljenosti, kakršno imamo pri branju, s prstom ali z dlanjo pokrijemo fotoupor. Ob tem mora svetleča dioda ugasniti. Naslednji korak je, da si zapišemo najmanjšo in največjo opaženo vrednost spremenljivke vrednost pri najmanjši in največji osvetljenosti. V razvojnem okolju Arduino IDE odpremo okno za prikaz vrednosti z izbiro Orodja, Serijski vmesnik oziroma krajše s hkratnim pritiskom tipk Ctrl, Shift in M.

Delovanje samega vezja je razloženo čisto na kratko, saj so podrobnosti objavljene v prej ome-

```
//
// Program Arduino - VklopIzklopLED.ino
// Vklop/izklop svetleče diode
//

// Pozor – LED priklopimo
// na pin 5
const int led = 5;
const int svetlobniSenzor = A0;
// Vrednost iz senznorja bomo
// shranili v spremenljivko in ga
// primerjali z referenčno vrednostjo
int vrednost = 0;
int refVrednost = 100;

void setup() {
  // Branje podatkov preko
  // serijskega vmesnika USB
  Serial.begin(9600);
  pinMode(led, OUTPUT);
}

void loop() {
  // Najprej preberemo vrednost s senzorja
  vrednost = analogRead(svetlobniSenzor);
  // Dobljeni podatek izpišemo na zaslon
  Serial.println(vrednost);
  // Ob vrednost < refVrednost vklopimo LED
  if (vrednost <= refVrednost) {
    digitalWrite(led, HIGH);
  } else {
    digitalWrite(led, LOW);
  }
}
```

njenem članku. Ko prst položimo na fotoupor, se na njem in na njemu zaporedno vezanem uporu, ki z njim sestavlja delilnik napajalne napetosti +5 V, spremeni napetost. To prek nožice 5 zazna krmilnik Arduino, ki s pomočjo programa ob zagotovljenem pogoju vključi svetlečo diodo.

Ko smo si zapisali obe vrednosti osvetljenosti, s spleta prenesemo program pwmLED.ino. Glede na zapisani vrednosti popravimo spremenljivki minSvetlo in maxSvetlo, ki imata v našem primeru vrednosti 10 in 500. Če želimo nekoliko večje vrednosti, lahko namesto upora 10 kΩ uporabimo upor 100 kΩ (rjava, črna, rumena in običajno zlata). Na podlagi vnesenih vrednosti se bo izračunala vrednost spremenljivke koeficient, ki je tipa float, saj gre za decimalno število.

Vezja nam ni treba spreminjati. Program samo prenesemo na krmilnik Arduino in mora takoj delovati. Če prst počasi približamo fotouporu in ga zatemnimo, mora svetleča dioda postopoma zasvetiti do največje moči. Pri prejšnjem programu smo imeli na voljo samo dve možnosti, da je namreč svetleča dioda svetila ali pa je bila ugasnjena.

Delovanje

S pomočjo ukaza analogWrite smo uporabili sposobnost krmilnika Arduino, da lahko nekatere nožice krmili s pulzno širinsko modulacijo (angl. PWM – Pulse Width Modulation). Žal te možnosti nimamo pri vseh nožicah (glej preglednico!), kar je tudi razlog, da smo diodo priključili na nožico 5.

Za kaj pravzaprav gre? Sam pojem modulacije pomeni, da nekemu periodičnemu nihanju oziroma nosilnemu signalu vgradimo informacijo. Imamo

```
//
// Program Arduino - pwmLED.ino
// Pulzno širinsko krmiljenje (PWM)
// svetleče diode

const int led = 5;
const int svetlobniSenzor = A0;
// Vrednost iz senznorja bomo
// shranili v spremenljivko
int vrednost = 0;
// Ker so vrednosti za
// analogWrite med 0 in 255,
// moramo izračunati koeficient
int minSvetlo = 80;
int maxSvetlo = 280;
// Izračun koeficienta
float koeficient = (float) (255.0 / (maxSvetlo - minSvetlo));
int vrednostPWM = 0;

void setup() {
  // Branje podatkov preko
  // serijskega vmesnika USB
  Serial.begin(9600);
  pinMode(led, OUTPUT);
}

void loop() {
  // Vrednost senzorja preberemo
  // in izračunamo vrednost za PWM
  vrednost = analogRead(svetlobniSenzor);
  // vrednost pošljemo preko USB
  Serial.print(vrednost); Serial.print("  ");
  // Izračunamo vrednostPWM
  vrednostPWM = (int) (255 - koeficient * (vrednost - minSvetlo));
  // Zagotovimo, da je vrednostPWM v mejah
  if (vrednostPWM < 0) {vrednostPWM = 0;}
  if (vrednostPWM > 255) {vrednostPWM = 255;}
  // vrednostPWM pošljemo preko USB
  Serial.print(koeficient); Serial.print("  ");
  Serial.println(vrednostPWM);
  // In nastavimo vrednost na pinu led
  analogWrite(led, vrednostPWM);
}
```

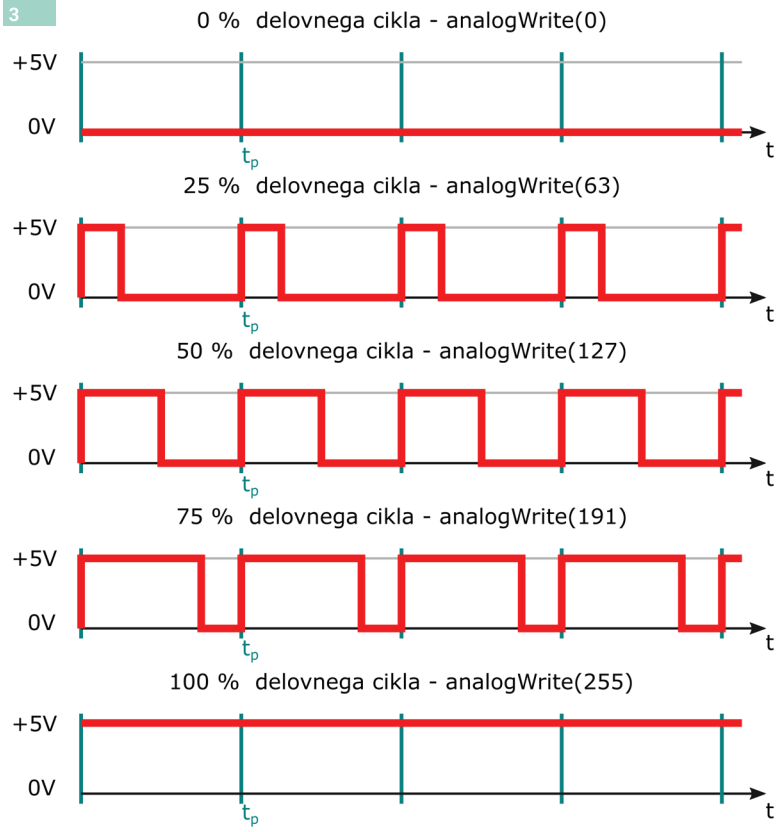
več vrst modulacije, pač glede na to, ali spreminjamo amplitudo, frekvenco oziroma fazni kot nosilnega signala. Tako se prenašajo radijski valovi, optične komunikacije idr.

Kako je z modulacijo v našem primeru? Digitalni signal ima na voljo samo vrednosti 0 in 1. Gledano s stališča napetosti imamo na izhodu krmilnika Arduino samo 0 V ali okoli +5 V; drugih vrednosti ni. To pri svetleči diodi pomeni, da ne sveti oziroma sveti z največjo močjo. Z uporabo pulzno širinske modulacije pa dosežemo, da svetleči diodi lahko spreminjamo svetilnost. Namesto da bi bil izhod ves čas v stanju +5 V, ga znotraj časovnega intervala (angl. Duty Cycle) periodično vklapljamo in izklapljamo (**slika 3**). Časovni interval predstavlja periodo signala tp, ki

Krmilnik	Priključki, ki podpirajo PWM	frekvenca PWM
Uno, Nano, Mini	3, 5, 6, 9, 10, 11	490 Hz (nožici 5 in 6: 980 Hz)
Mega	2–13, 44–46	490 Hz (nožici 4 in 13: 980 Hz)
Leonardo, Micro, Yún	3, 5, 6, 9, 10, 11, 13	490 Hz (nožici 3 in 11: 980 Hz)

Preglednica
(vir: www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/)

3



je označena na sliki. Iz periode izračunamo frekvenco nosilnega signala f = 1 / tp, ki je v primeru funkcije analogWrite enaka 490 Hz (preglednica). Kolikor dlje v enaki časovni enoti pustimo stanje +5 V, toliko dlje sveti dioda. Na sliki je to prikazano z rdečim signalom, in sicer za deleže 0, 25, 50 ter 100 % celotne časovne enote. V času ene periode tako dioda sveti manj kot 100 % možne moči. Ker se izklopi in vklopi dogajajo zelo hitro, jih naše oko ne zazna, ampak imamo občutek, da svetleča dioda sveti z manjšo jakostjo (**slike 4a, b in c**). Podobno je pri motorčku, kjer ta za trenutek nima napetosti, vendar se rotor zaradi vztrajnosti ne ustavi. Naslednji napetostni pulz ga znova požene naprej, zaradi česar se rotor motorčka vrti počasneje od največje možne hitrosti.

Pulzno širinsko krmiljenje prek H-mostiča

Kadar želimo s krmilnikom Arduino pulzno širinsko krmiliti kak močnejši porabnik, na primer žarnico ali motorček, moramo (podobno kot pri avtomobilčku) uporabiti H-mostič (**slika 5**), ki poskrbi za primerno napetost in dovolj velik električni tok. Programska koda se spremeni le minimalno, zato so prikazane samo spremenjene vrstice. Celoten program je dostopen na prej omenjeni spletni strani.

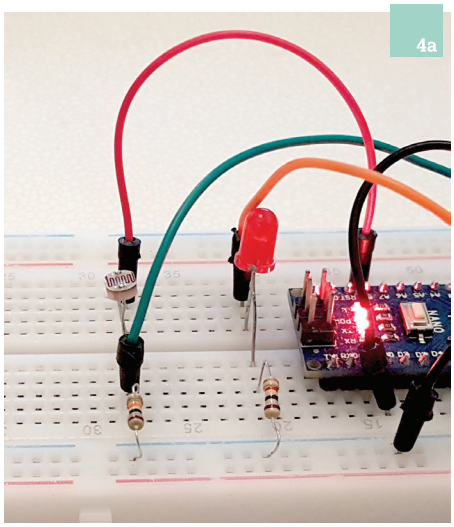
Kot je razvidno iz programa, moramo dodati spremenljivki pinA1 in pinA2, ki določata nožici 9 in 10 na izhodu krmilnika Arduino. Če se odločimo drugače in uporabimo drugi nožici, moramo paziti, da ti podpirata krmiljenje PWM. V setup() napovemo, da bosta nožici delovali v izhodnem (OUTPUT) načinu. V okviru loop() prek ukaza analogWrite postavimo enega od izhodov na 0, na drugega pa vplivamo prek izračunane vrednosti vrednostPWM. Podobno kot pri svetleči diodi žarnica sveti glede na spremembe osvetljenosti fotoupora. Če namesto žarnice uporabimo motorček, se ta vrti hitreje oziroma počasneje. Tudi tu lahko H-mostič (podobno kot pri avtomobilčku) priključimo na višjo napajalno napetost, s čimer se bo motorček vrtel hitreje, lučka pa bo svetila z večjo jakostjo.

Tistim z več znanja je treba pojasniti, da smo dva izhoda PWM uporabili zaradi morebitnega dvosmernega krmiljenja motorčka. Lahko pa izhod, ki smo ga postavili na 0, zamenjamo s katerim od digitalnih izhodov, ki ne podpira PWM. Obstaja tudi možnost dvosmernega krmiljenja motorčka, pri čemer prvega postavimo na 1, na izhodu PWM pa z analogWrite povečujemo vrednost od 0 do 255, s čimer se motorček vrti počasneje. Kot dodatno možnost velja omeniti, da obstajajo tudi taki H-mostiči, kjer namesto enega od izhodov uporabimo kar GND.

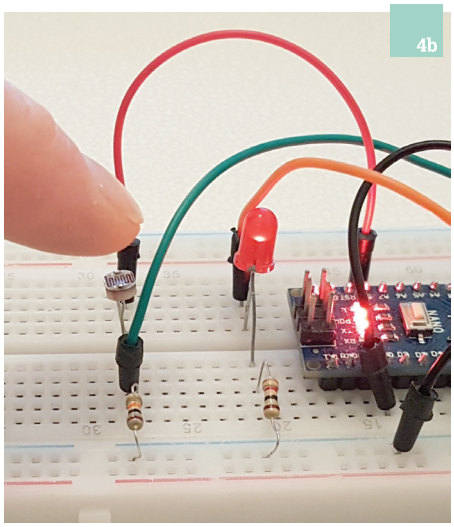
Za konec

Pulzno širinsko krmiljenje je zelo uporabno, saj lahko tudi prek digitalnega izhoda, ki ima le dve vrednosti napetosti, dosežemo zmanjšanje izhodne moči v časovni enoti. To pomeni, da žarnica navidezno sveti šibkeje, rotor motorčka se vrti po-

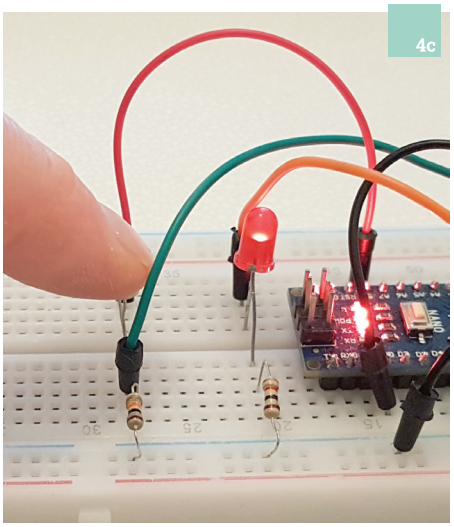
4a

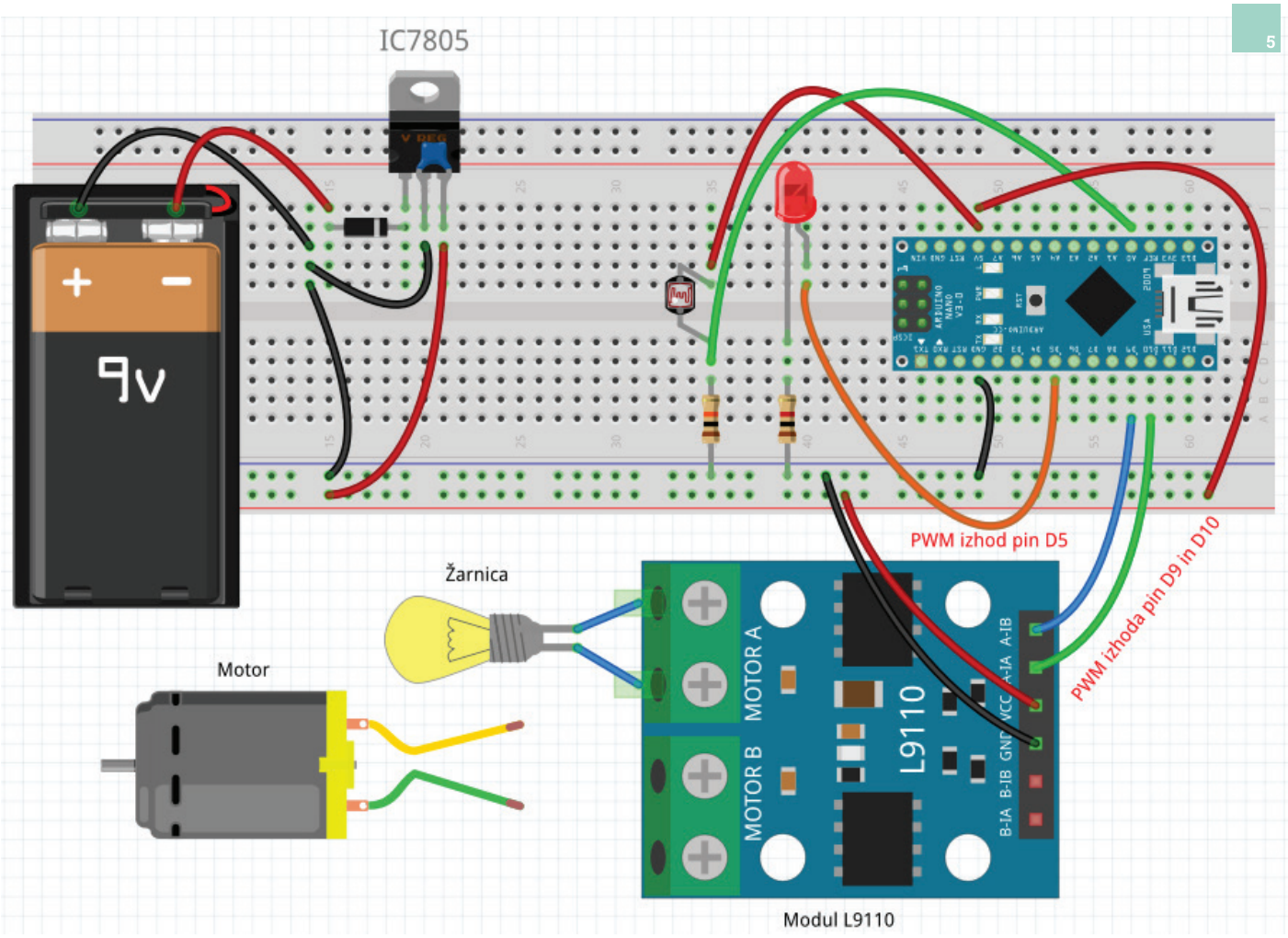


4b



4c





časneje, elektromagnet pritegne manj in še bi lahko naštevali. V okviru prispevka smo s spremembo osvetljenosti fotoupora počasi prižigali oziroma ugašali svetlečo diodo in pozneje žarnico. Podobno je mogoče spremeniti program za krmiljenje motorčka avtomobilčka iz prejšnjih prispevkov, s čimer ga lahko natančno usmerjamo pri vožnji po črti in drugače. Seveda si lahko zamislimo še celo kopico projektov, kjer nam bo usvojeno znanje prišlo prav.

Če boste izvedli kak zanimiv projekt, ga lahko posredujete uredništvu revije Tim.

```
//
// Program Arduino - pwmLUCKA.ino
// Pulzno širinsko krmiljenje (PWM)
// lučke preko H-mostiča

// PWM krmiljenje preko H-mostiča
const int pinA1 = 9;
const int pinA2 = 10;

const int led = 5;
const int svetlobniSenzor = A0;
// Vrednost iz senznorja bomo
// shranili v spremenljivko
int vrednost = 0;
// Ker so vrednosti za
// analogWrite med 0 in 255,
// moramo izračunati koeficient
int minSvetlo = 80;
int maxSvetlo = 280;

// Izračun koeficienta
float koeficient = (float) (255.0 / (maxSvetlo - minSvetlo));
int vrednostPWM = 0;

void setup() {
  // Branje podatkov preko
  // serijskega vmesnika USB
  Serial.begin(9600);
  pinMode(led, OUTPUT);

  // PWM krmiljenje preko H-mostiča
  pinMode(pinA1, OUTPUT);
  pinMode(pinA2, OUTPUT);
}

void loop() {
  // Vrednost senzorja preberemo
  // in izračunamo vrednost za PWM
  vrednost = analogRead(svetlobniSenzor);
  // vrednost pošljemo preko USB
  Serial.print(vrednost); Serial.print("  ");
  // Izračunamo vrednostPWM
  vrednostPWM = (int) (255 - koeficient * (vrednost - minSvetlo));
  // Zagotovimo, da je vrednostPWM v mejah
  if (vrednostPWM < 0) {vrednostPWM = 0;}
  if (vrednostPWM > 255) {vrednostPWM = 255;}
  // vrednostPWM pošljemo preko USB
  Serial.print(koeficient); Serial.print("  ");
  Serial.println(vrednostPWM);
  // In nastavimo vrednost na pinu led
  analogWrite(led, vrednostPWM);

  // PWM krmiljenje preko H-mostiča
  analogWrite(pinA1, 0);
  analogWrite(pinA2, vrednostPWM);
}
```