mikrokrmilnika U3. Mikrokrmilnikova naloga je periodično proženje senzorja U2 z zgoraj omenjenim 0,32 ms dolgim impulzom, še pred njegovim iztekom (padcem) sprožiti analogno-digitalni pretvornik ter rezultat prikazati na preprostem prikazovalniku s tremi LED-diodami.

Za prikaz stanja prašnih delcev sem uporabil naslednji prikaz: Če sveti samo zelena LED-dioda, je zrak čist in brez prašnih delcev. Rahlo onesnaženje se prikaže z utripanjem te iste zelene svetleče diode. Pri še večjem onesnaženju se prižge rumena LED-dioda, zelena pa še naprej utripa. Pri naslednji stopnji onesnaženja se prižge rumena svetleča dioda, ki lahko tudi utripa. Maksimum, ki ga senzor še lahko zazna, prikažemo z utripanjem vseh treh svetlečih diod.

Vseh šest stopenj onesnaženja odgovarja napetostnemu razponu od 0,60 do 3,60 V. Velikost prašnih delcev ponazorimo z dolžino utripov svetlečih diod. Večji kot so prašni delci, širši so svetlobni bliski.

Vezje ne potrebuje posebnega vzdrževanja (justiranja). O tem pa več v nadaljevanju, kot tudi o napotkih za sestavljanje ter izvedbo projekta.

### Razlikovanje drobnih (npr. cigaretnega dima) od grobih prašnih delcev

*Cigaretni dim vsebuje izredno gostoto koncentracijo precej mobilnih majhnih delcev (Ø ~0,1 µm), medtem ko so običajni (grobi) prašni delci lahko tudi za več razredov večji in v primerjavi s cigaretnim dimom razmeroma redki. Slednje tu uporabljeni senzor zazna kot sporadične impulze, bližnjo prižgano cigareto pa kot strnjen niz impulzov s periodo 10 ms.*

***Senzor tako loči naslednje vrste onesnaženja:***

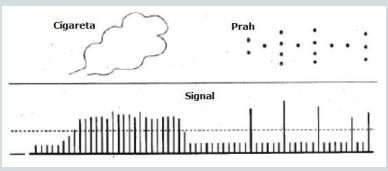
***a)*** *cigaretni dim in podobne drobne prašne materiale,*

***b)*** *grob prah,*

***c)*** *kombinacijo zgornjih dveh*

***d)*** *in seveda čist zrak brez prisotnosti prahu.*

*Amplitudo zaznanih impulzov lahko povežemo z velikostjo (gostoto) onesnaženja zraka s prašnimi delci.*




*Karakterističen izhodni signal glede na vrsto prahu*



# PRVI KORAKI V ARDUINO – SPOROČILO ZA STISKO SOS

**Milan Gaberšek in Slavko Kocijančič**

Računalniki in pametni telefoni so postali del našega vsakdanjika. Čeprav jih ves čas uporabljamo, se večinoma ne zavedamo, kaj se dogaja v ozadju. Kaj pa, če nas zanima, kaj se skriva pod pokrovom, kako deluje in ali lahko sami kaj spremenimo? Na ta vprašanja lahko deloma odgovorimo z uporabo krmilnika Arduino (v nadaljevanju Arduino). Ne gre za pravi računalnik, saj nima operacijskega sistema in razen programa, ki ga namestimo, ne omogoča nalaganja dodatnih aplikacij. Ker pa deluje zelo podobno kot pravi računalniki, je odlično orodje za učenje. V nadaljevanju si bomo ogledali primer s svetlečo diodo in vsem, kar je potrebno, da svetleča dioda prek Arduina začne utripati. Nazadnje pa bomo predstavili še primer sporočila za stisko SOS s pomočjo Morsejevih svetlobnih znakov.

## Material

- krmilnik Arduino Nano ali podoben,
- USB-kabel za povezavo … (mini USB) z računalnikom,
- prototipna ploščica (angl. breadboard),
- dve vezni žici različnih barv (v našem primeru rdeča in modra),
- upor vrednosti 1 kΩ (rjava, črna, rdeča, zlata),
- svetleča dioda,
- baterija 9 V.

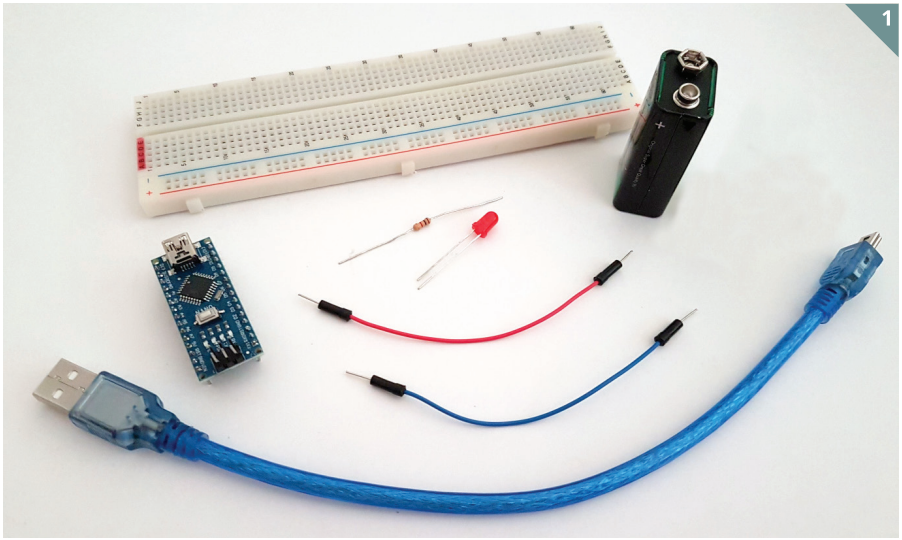
## Orodja in pripomočki

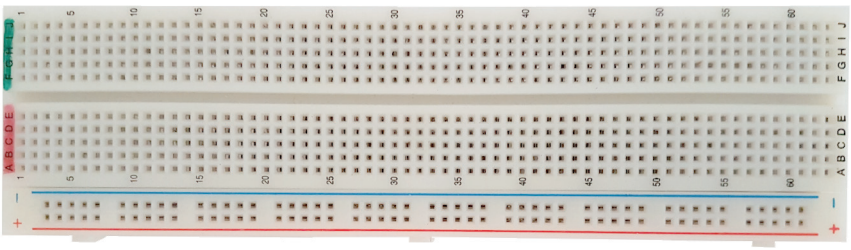
- osebni računalnik z nameščenim operacijskim sistemom Windows, Linux ali macOS,
- Arduino IDE, integrirano programsko razvojno okolje, ki je brezplačno dostopno na spletni strani *www.arduino.cc*.

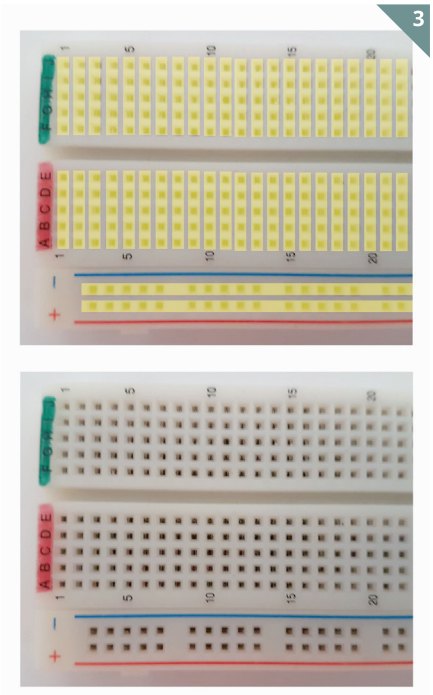
## Izvedba

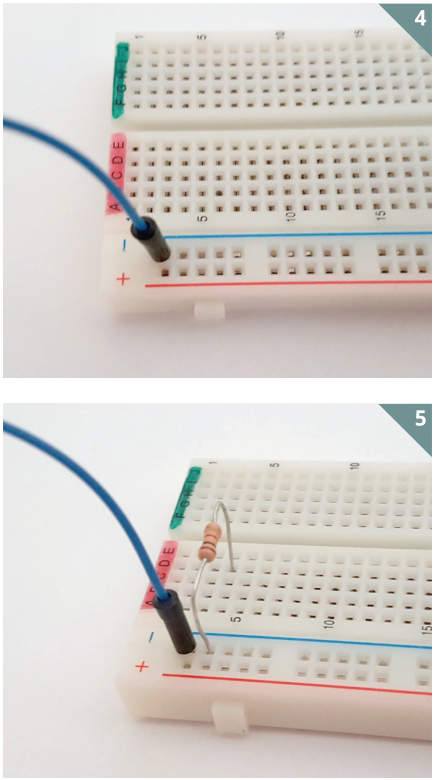
Na sliki 1 so prikazani vsi sestavni elementi. Ker so bralci revije različne starosti in stopnje predznanja, se bomo najprej podrobneje seznanili s priklopom svetleče diode. Bolj vešči bralci lahko ta del le preletijo. Svetlečo diodo poznamo tudi pod kratico LED. Poleg svetleče diode moramo obvezno dodati še upor, ki omeji tok. Če ga ne dodamo, lahko svetlečo diodo ali krmilnik celo uničimo. Pozneje bo omenjeni upor zaščitil tudi sam Arduino.

Različna elektronska vezja najlažje testiramo s prototipno ploščico (slika 2). Pri tej elektronske elemente preprosto vstavimo v kontakte na njej. Prototipna ploščica je sestavljena tako, da sta v osrednjem delu dva velika med seboj ločena bloka, ob strani pa se čez celotno površino raztezata vrstici za napajanje. Stolpci so označeni s


1

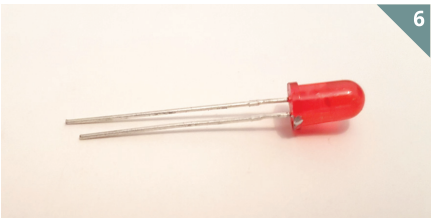

2

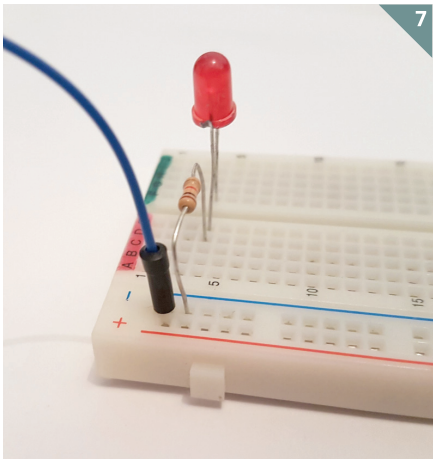


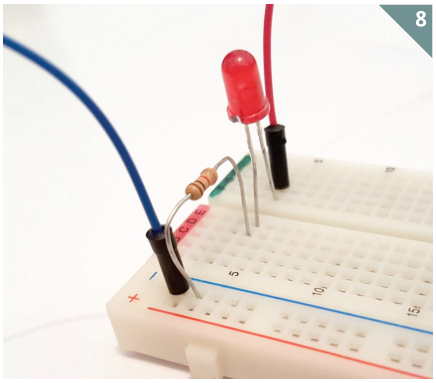



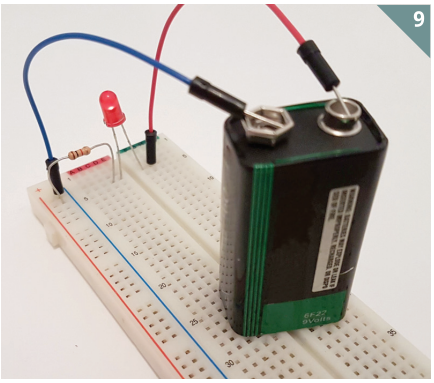







številkami od 1 do 60, vrstice pa s črkami od A do E v prvem in s črkami od F do J v drugem bloku, kar je na sliki 3 dodatno nakazano z oranžno in zeleno barvo. Z rumeno barvo je označeno, kako so med seboj povezani posamezni kontakti preizkusne ploščice (zgornji del slike 3). Če želimo na primer med seboj povezati nožici dveh različnih elementov ali žičk, ju moramo vstaviti tako, da sta obe v istem rumenem stolpcu. Drugo nožico posameznega elementa povežemo v kontakt enega od sosednjih stolpcev ali kamor koli v drugem bloku. Na ta način bomo pozneje povezali svetlečo diodo in upor.

Spodnji vrstici običajno uporabimo za priklop napajanja z žičkama; rdečo uporabimo za plus in modro, kot je to v našem primeru, za minus. Na tržišču najdemo povsem bele prototipne ploščice in take, ki imajo nad blokoma še dodatni »napajalni« vrstici. Pri nekaterih prototipnih ploščicah so napajalne vrstice na sredini prekinjene oziroma nepovezane, kar najlažje preverimo s pomočjo univerzalnega instrumenta (preverimo prevodnost sosednjih kontaktov na sredini vrstice).

Pa začnimo z delom. Najprej v modro napajalno vrstico vstavimo modro vezno žičko (slika 4), ki je seveda lahko tudi drugačne barve (priporočljiva je črna), pri čemer svetujem, da za napajanje vedno uporabimo žičke enakih barv za plus oziroma minus. Tako bo vezje bolj pregledno, hkrati pa boste laže odkrili morebitne napake. Pozitivnega pola napajanja zaradi poenostavitve vezja v našem primeru ne bomo priklopili v napajalno vrstico.

V kontakt desno od vezne žičke vstavimo eno od obeh nožic upora (slika 5). Drugo nožico upora zapičimo v enak stolpec bloka kot prvo nožico, in sicer v vrstico D. Pri uporu ni pomembno, katero od obeh nožic elementa vstavimo v katerega od kontaktov. Zaradi lepšega videza in preglednosti je priporočljivo tolerance uporov, ki jo običajno predstavlja obroček zlate ali srebrne barve, vedno obrniti v isto smer.

Pri tem projektu bomo uporabili tudi svetlečo diodo (slika 6). Pomembno je, da jo pravilno priklopimo, saj drugače ne bo svetila. Ena od obeh nožic je daljša in označuje anodo. Ker nožice običajno prispajkamo in skrajšamo, je katoda, ki je povezana z drugo nožico, označena še z zarezo na obodu ohišja. Da katodo laže prepoznamo, je v članku dodatno označena s srebrnim flomastrom (slika 6).

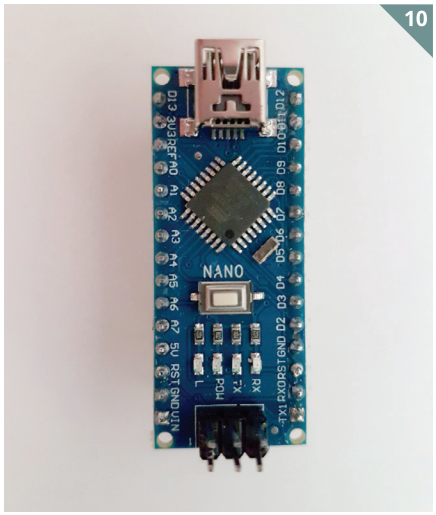
Svetlečo diodo vstavimo v preizkusno ploščico tako, da bo katoda vstavljena v isti stolpec neposredno k uporu, anoda pa v kontakt v istem stolpcu, toda v drugem bloku (slika 7).

V kontakt v istem stolpcu in v drugem bloku k anodi diode dodamo še rdečo vezno žičko (slika 8). To povežemo na pozitivni pol 9-voltne baterije (oznaka + na ohišju baterije), modro pa na negativni pol (slika 9). Če smo vse pravilno povezali, bi morala svetleča dioda zasvetiti. Če se to ni zgodilo, preverimo, ali so vezne žičke in nožice sestavnih elementov dobro vstavljene v posamezne kontakte. Zdaj zamenjamo obe vezni žički in preverimo, če svetleča dioda v tem primeru res ne sveti.

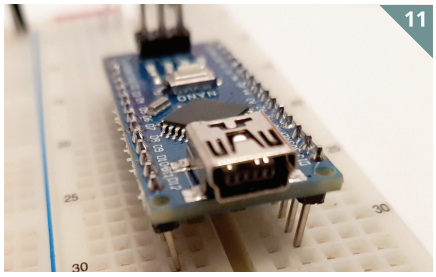
Kaj pa, če želimo, da bi svetleča dioda utripala? Zelo preprosto bi lahko eno od veznih žičk pritiskali in odmikali od pola baterije ali namestili tipko. Ta rešitev seveda ni najbolj elegantna, v ta namen raje uporabimo krmilnik Arduino. Pred tem odklopimo baterijo, saj je ne bomo več potrebovali. Diodo, upor in obe vezni žički bomo uporabili pozneje, zato jih lahko pustimo na preizkusni ploščici.

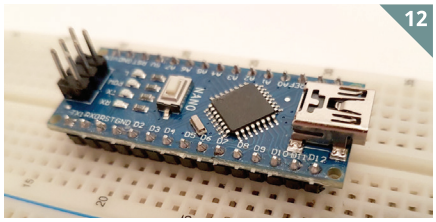
Obstaja več vrst krmilnikov, v našem primeru bomo uporabili krmilnik Arduino Nano (slika 10). To pa zato, ker ga lahko vstavimo neposredno na prototipno ploščico, s čimer je povezovanje z ostalimi elektronskimi elementi preprostejše. Pri vstavljanju moramo biti zelo previdni in nežni, da ne skrivimo priključnih nožic Arduina. V prototipno ploščico ga vstavimo tako, da ga najprej delno vstavimo na enem koncu (slika 11), nato še na drugem koncu in ga počasi potiskamo v kontakte (slika 12). Pri tem je polovica nožic Arduina v enem, druga pa v drugem bloku prototipne ploščice.

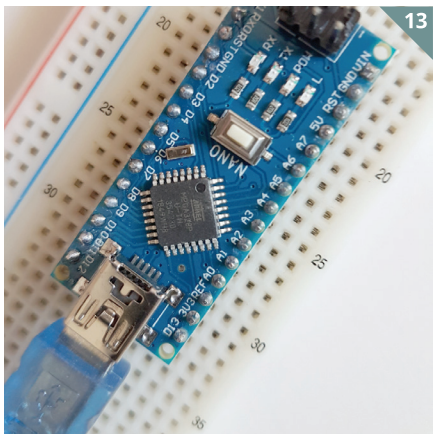
Pred prvo uporabo moramo na računalnik namestiti še programsko razvojno orodje Arduino IDE, ki je brezplačno dostopno na spletni strani *www.arduino.cc*. Po namestitvi in zagonu razvojnega okolja moramo najprej določiti, kateri krmilnik bomo uporabljali in prek katerih vhodnih vrat COM bo ta komuniciral. Pred priklopom Arduina v meniju Orodja/Vrata najprej preverimo, ali že obstaja kakšna naprava, ki komunicira prek vrat COM.

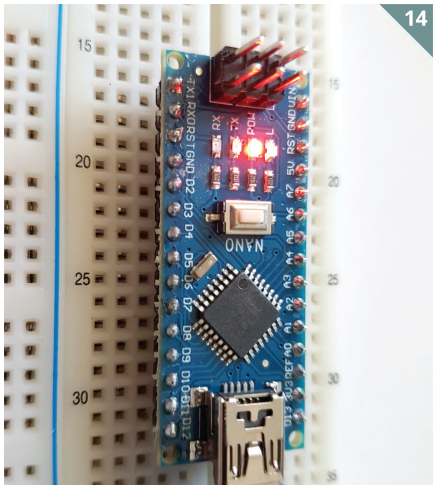


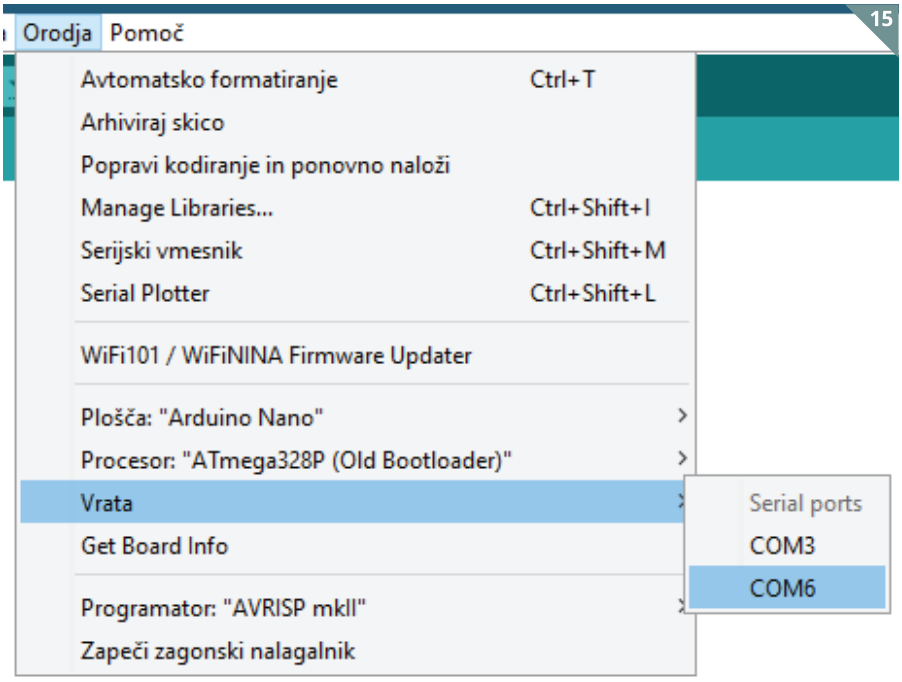
10


11


12


13


14



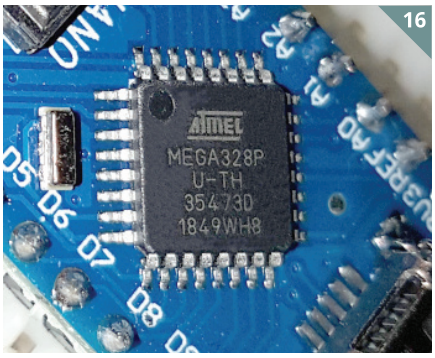
15

Taki so na primer tiskalnik, USB-ključek, zunanji disk ali podobna naprava. Za vsako oznako COM je navedena številka vrat (na primer COM3). Vrednosti si zapomnimo in zapremo razvojno okolje Arduino IDE.

V krmilnik vstavimo kabel z mini USB-kontaktom (slika 13), drugi konec pa v računalnik. Če je vse v redu, bi morala na Arduinu zasvetiti majhna svetleča dioda (slika 14) z oznako POW. Pri nekaterih različicah Arduina je oznaka drugačna, na primer PWR. V novejših operacijskih sistemih se ob priklopu največkrat uspešno izvede tudi nameščanje ustreznih gonilnikov, pri starejših pa moramo za to poskrbeti sami. Ustrezna navodila in gonilnike za naš operacijski sistem in naš Arduino lahko poiščemo na spletu, dodatne informacije v slovenskem jeziku pa tudi na *www.drti.si*.

Ko so gonilniki za Arduino uspešno nameščeni, lahko ponovno zaženemo razvojno okolje Arduino IDE. V meniju Orodja/Vrata so ob uspešni namestitvi navedena nova vrata, na primer COM6 (slika 15). S klikom izberemo ta vrata (pogosto so že izbrana), saj bo Arduino prek njih komuniciral z računalnikom. V meniju Orodja/Plošča preverimo, ali je med navedenimi krmilniki pravilno izbran Arduino Nano. Običajno moramo izbrati tudi tip mikrokrmilnika, ki je nameščen in ga uporablja naš Arduino. Tip razberemo iz napisa na čipu, v našem primeru Atmega328P podjetja Atmel (slika 16). V našem primeru v meniju Orodja/Procesor izberemo Atmega328 (Old Bootloader). Za test izberemo Orodja/Get Board Info, pri čemer se na zaslonu v odvisnosti od vrste Arduina izpišejo osnovni podatki (slika 17).

Ko povezava z Arduinom deluje, se lahko lotimo programskega dela. Najprej napišemo program, ki bo prižigal testno svetlečo diodo na Arduinu z oznako L, ta je ob svetleči diodi za napajanje POW in je v vezju Arduina povezana tudi s pinom (nožico) z oznako D13, ki ga bomo uporabili pozneje. Programsko se nanj sklicujemo s

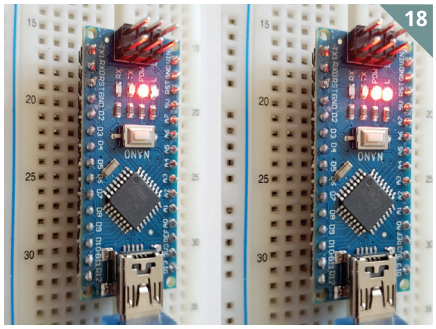


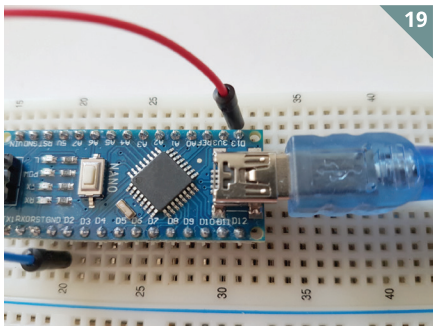
16

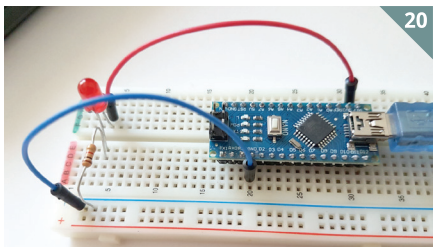
17

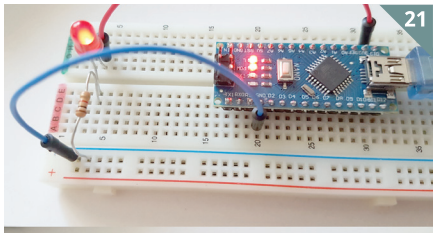
konstanto LED_BUILTIN, ki ima številčno vrednost 13, enako, kot je oznaka pina D13. Pri prepisovanju moramo paziti, da res natančno prepišemo kodo, sicer program ne bo deloval.

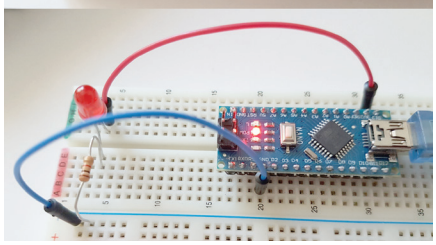
```
// Program Test

void setup() {
  pinMode(LED_BUILTIN, OUTPUT); //digitalni
  izhod za vgrajeno LED na krmilniku
}
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // LED
  zasveti
  delay(1000); //čakanje na naslednji ukaz v
  trajanju eno sekundo
  digitalWrite(LED_BUILTIN, LOW); // LED ne
  sveti več
  delay(500); // čakanje v trajanju 0,5 sekun
  de, potem se izvajanje vrne na prvo vrstico
  za loop
}
```


18


19


20


21



void setup() { } je podprogram, ki se zažene le enkrat ob zagonu programa. Tedaj se izvede vsa koda, ki jo navedemo med zavitima oklepajema. Običajno tu nastavimo, katere vhode oziroma izhode Arduina bomo uporabljali. V našem primeru bomo prižigali LED_BUILTIN svetlečo diodo na Arduinu, zato bomo signal pošiljali iz procesorja. To dosežemo z ukazom pinMode(LED_BUILTIN, OUTPUT);. Pri tem bodimo pozorni, da na koncu ukaza ne pozabimo dodati podpičja, saj je to ena od najpogostejših napak pri tipkanju kode. Pazite tudi na pravilen zapis velikih in malih črk.

void loop() { } je podprogram, ki se izvede in se nato ves čas ponavlja. To je bodoče srce našega programa. Celotna koda, ki jo navedemo med zavitima oklepajema, se namreč vseskozi ponavlja. V našem primeru smo z ukazom digitalWrite(LED_BUILTIN, HIGH); digitalni izhod LED_BUILTIN svetleče diode postavili na HIGH, kar pomeni, da se bo na izhodu Arduina pojavila napetost in bo dioda posledično zasvetila. Z ukazom delay(1000); smo določili, da program čaka 1000 milisekund (ms), preden se izvede naslednji ukaz. Sledi digitalWrite(LED_BUILTIN, LOW);, s katerim izhod Arduina postavimo na LOW oziroma na 0 V, s čimer tudi dioda ne bo več svetila. Z delay(500); ponovno določimo čas čakanja. Ker se vse dogaja znotraj podprograma void loop(), se postopek po izteku čakanja ponovi in se nato ves čas ponavlja.

Ko vse pretipkamo, program shranimo z Datoteka/Shrani, nato pa s Skica/Naloži prevedemo in prenesemo na Arduino. Če smo se pri prepisovanju programa zmotili, nam razvojno okolje to izpiše na spodnjem delu zaslona, običajno z informacijo, v kateri vrstici je napaka in s kratkim opisom. Če je vse v redu, začne LED_BUILTIN dioda utripati (slika 18).

Zdaj priklopimo še svetlečo diodo z uporom, ki ga imamo že od prej. Pred tem Arduino odklopimo od računalnika. Kot smo že povedali, je LED_BUILTIN svetleča dioda povezana z nožico (pinom) z oznako D13, zato bomo tja povezali tudi rdečo vezno žičko, ki smo jo imeli prej povezano s pozitivnim polom baterije. Pin D13 je v skrajnem desnem kotu Arduina, kamor v kontakt nad njim v preizkusno ploščico vstavimo rdečo vezno žico (slika 19). Obstoječo modro vezno žico priklopimo pod pinom GND (slika 19). Če si pozorno ogledamo Arduino, bomo opazili, da je na nasprotni strani še en pin GND. Oba pina GND sta med seboj povezana, tako da je vseeno, katerega uporabimo.

Po priklopu svetleče diode in upora (slika 20) lahko Arduino prek USB-kabla ponovno povežemo z računalnikom. Ker smo program že naložili in je ta tudi po izklopu ostal v pomnilniku Arduina, začne poleg LED_BUILTIN diode utripati še svetleča dioda (slika 21). S tem je testni program končan.

Zdaj moramo program prilagoditi tako, da nam bo s pomočjo Morsejevih svetlobnih znakov pošiljal sporočilo za stisko SOS. Sporočilo je sestavljeno iz pik in črtic, in sicer »pika, pika, pika, črta, črta, črta, pika, pika, pika«, ali krajše kot · · · – – – · · ·. To pomeni tri kratke, tri dolge in ponovno tri kratke svetlobne ali zvočne utripe. Rešitev je preprosta, najbolje pa je predelati kar obstoječi testni program. Najprej ga z Datoteka/Shrani kot... shranimo pod drugim imenom, na primer Program SOS. Sledi prilagoditev za tri kratke utripe, kar pomeni, da obstoječo kodo v podprogramu void loop() { } kopiramo in dvakrat prilepimo. Nato spremenimo vrednost za čas utripa in čas čakanja 500 ms v vseh treh primerih. Nato skopiramo in prilepimo celoten blok za utripanje ter spremenimo čas utripa v dolgi utrip 1000 ms. Ko je to narejeno, prvi blok s kratkimi utripi kopiramo in prilepimo še enkrat. Na koncu programa dodamo še čas čakanja treh sekund (oziroma 3000 ms), da se po ponovitvi znotraj podprograma void loop() { } končno utripanje loči od začetnega. Z nekaj predznanja lahko program poenostavimo, kar pa presega namen tega prispevka. S tipko Reset, ki je na sredini Arduina (velja za krmilnik Arduino Nano), lahko program kadar koli prekinemo in ponovno zaženemo. S tem smo projekt uspešno pripeljali do konca.

```
// Program za Arduino IDE

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
  // ...
  digitalWrite(LED_BUILTIN, HIGH);
  delay(500);
  digitalWrite(LED_BUILTIN, LOW);
  delay(500);
  digitalWrite(LED_BUILTIN, HIGH);
  delay(500);
  digitalWrite(LED_BUILTIN, LOW);
  delay(500);
  digitalWrite(LED_BUILTIN, HIGH);
  delay(500);
  digitalWrite(LED_BUILTIN, LOW);
  delay(500);

  // - - -
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(500);
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(500);
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(500);

  // ...
  digitalWrite(LED_BUILTIN, HIGH);
  delay(500);
  digitalWrite(LED_BUILTIN, LOW);
  delay(500);
  digitalWrite(LED_BUILTIN, HIGH);
  delay(500);
  digitalWrite(LED_BUILTIN, LOW);
  delay(500);
  digitalWrite(LED_BUILTIN, HIGH);
  delay(500);
  digitalWrite(LED_BUILTIN, LOW);
  delay(500);

  delay(3000);
}
```

## Zaključek

Dandanes je uporaba krmilnika Arduino zelo razširjena. Arduino je primeren za izvajanje projektov, kjer spremljamo dogodke v okolju (različna stikala, svetloba, zvok, toplota ...) in glede na to krmilimo različne zunanje elemente (motorčki, elektromagneti, releji ...). Arduino uporabljajo v številnih hobijskih projektih, zadnje čase pa se čedalje pogosteje pojavlja tudi v industrijskih aplikacijah.

V prihodnjih številkah bomo predstavili še nekaj projektov z Arduinom. Ti bodo čedalje zahtevnejši, s čimer bo raslo naše znanje in (upamo) tudi veselje do dela z njim.