## Laboratrijska vježba 3:

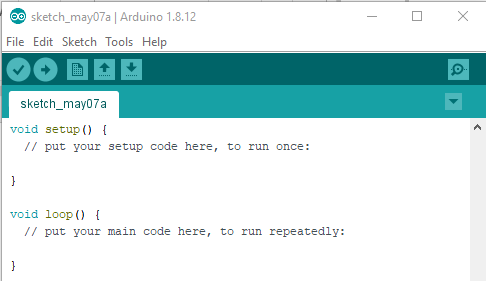
## Korištenje Arduino razvojnog okruženja za STM32F103

## Instalacija programskog paketa Arduino IDE

Osnovu razvojnog okruženja za Arduino čini programki paket “Arduino IDE”. On se može dohvatiti s interneta i instalirati lokalno na disk osobnog računala, ili se može koristiti online. Ukoliko se program koristi online instalacija nije potrebna, već se odmah može krenuti s postupkom konfiguriranja koji je detaljno opisan u narednom poglavlju.

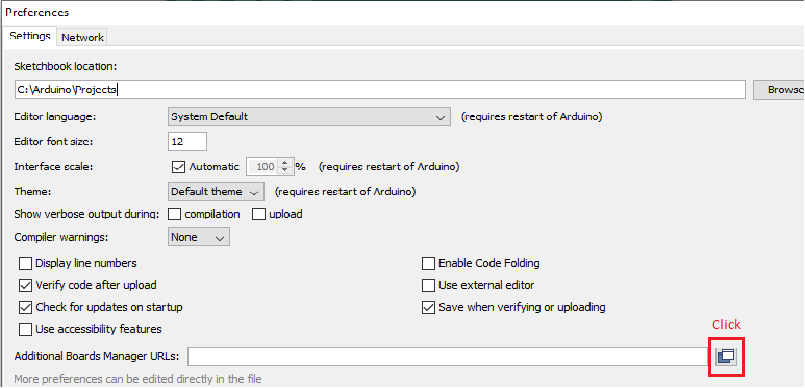
Instalacijski program za programski paket Arduino IDE može se dohvatiti s internetske stranice <https://www.arduino.cc/en/main/Software>, a pritom treba odabrati operacijski sustav na kojem će se paket koristiti. Daljnje upute podrazumjevaju korištenje operacijskog sustava Windows 10, no postupak je vrlo sličan i pri korištenju drugih podržanih operacijskih sustava (Linux i Mac OS).

Nakon dohvaćanja instalacijskog programa **arduino-1.8.12-windows.exe** (Windows installer), treba pokrenuti njegovu instalaciju te odabrati folder kamo će paket biti instaliran, npr. **C:\Arduino**. Po završetku instalacije, pokretanjem programa Arduino IDE utvrđujemo da li je instalacija uspješno provedena.

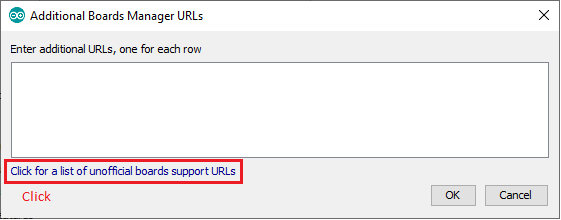


## Konfiguracija programskog paketa Arduino IDE za korištenje STM32 mikrokontrolera

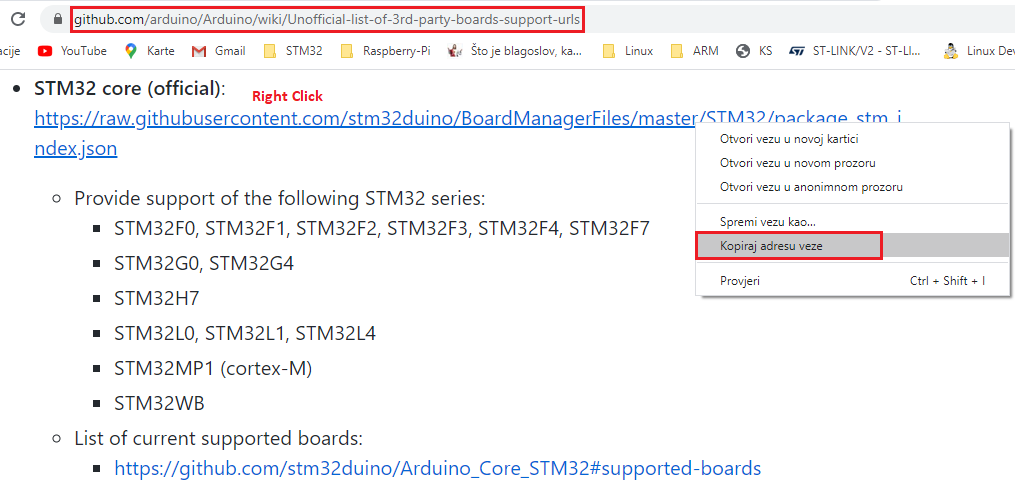
Programu Arduino IDE treba naznačiti tip mikrokontrolera ili razvojne pločice koji će se koristiti u Arduino IDE projektu. To se ostvaruje na način da odaberemo **File -> Preferences** te u prozoru **Preferences** koji se potom pojavljuje kliknemo na dolje označenu ikonu:



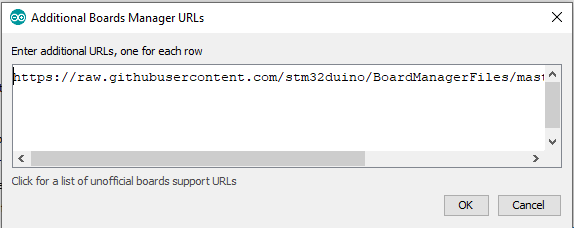
nakon čega se otvara prozor “**Additional Board Manager URLs**”:



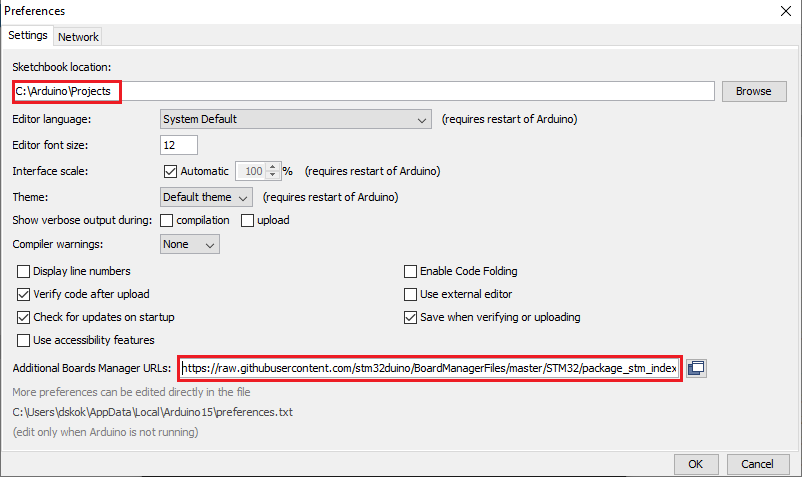
Pritiskom na “**Click for a list of unofficial board support URLs**” otvorit će se u internet pregledniku stranica <https://github.com/arduino/Arduino/wiki/Unofficial-list-of-3rd-party-boards-support-urls> na kojoj treba pronaći odsječak koji se odnosi na **STM32 Core**. U tom odsječku nalazi se poveznica koju treba kopirati



i zalijepiti na za to predviđeno mjesto u “**Additional Board URLs**” prozoru

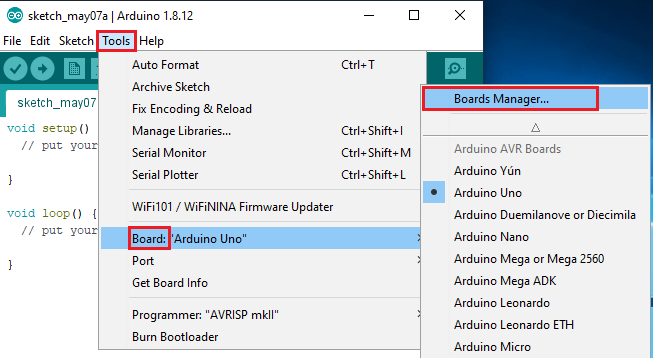


Pritisak na “**OK**” gumb vraća nas nazad na “**Preferences**” prozor, u kome dodatno treba unijeti folder u koji će se spremati Arduino projekti (Sketchbook location), npr **C:\Arduino\Projects**.

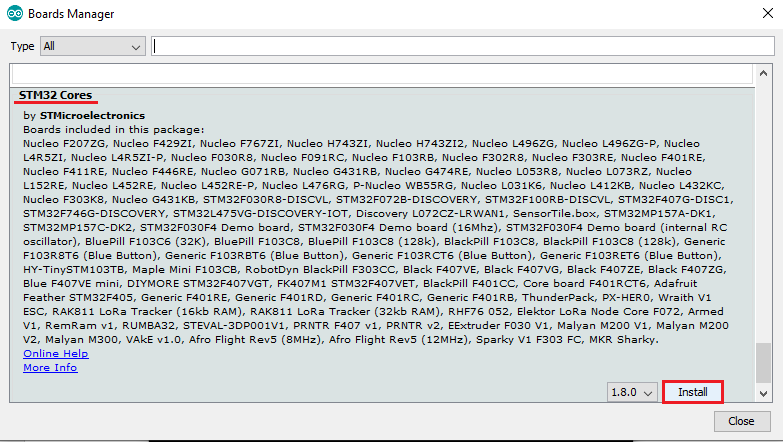


Gore opisani postupak prvi je korak kojim se omogućuje korištenje razvojnih pločica i mikrokontrolera tvrtke STM u Arduino projektima. Taj je korak nužan jer BluePill razvojna pločica koristi STM32F103 mikrokontroler tvrtke STM.

Za razvoj projekata zasnovanih na STM mikrokontrolerima, STM je, kao i mnogi drugi proizvođači, implementirao programski dodatak za Arduino IDE, koji to omogućuje. Slijedeći korak koji treba napraviti je instalacija navedenog programskog dodatka. On se instalira i integrira u Arduine IDE na način da se prvo odabere **Tools -> Board -> Board Manager…**, kao na slici dolje:

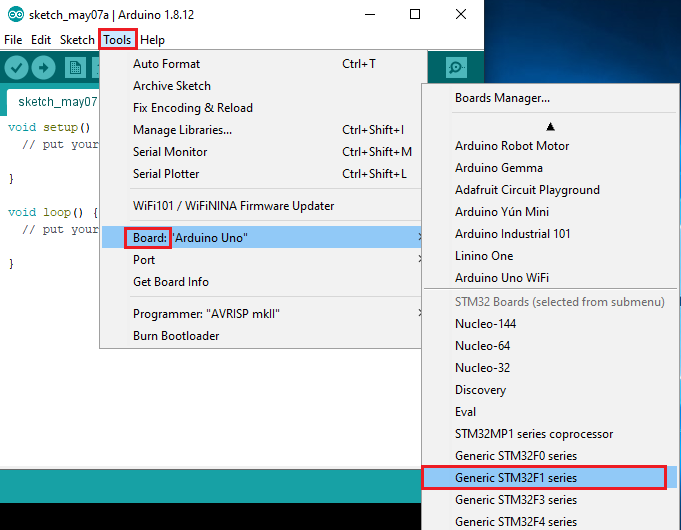


Time se otvara “**Boards Manager**” prozor u kome treba, koristeći klizač za navigaciju, doći do odsječka “**STM32 Cores**” i pritisnuti gumb **Install**:

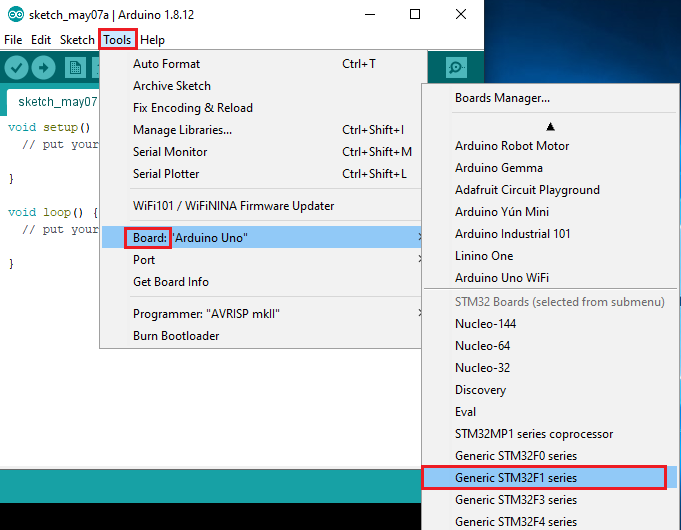


Programski dodatak **STM32 Cores** poprilično je velik te treba pričekati određeno vrijeme za njegovo preuzimanje s interneta i lokalnu instalaciju.

Sada konačno možemo odabrati mikrokontroler, odnosno skupinu kojoj mikrokontroler pripada. **BluePill** pločica koristi **STM32F103C8** mikrokontroler, koji pripada, iz oznake je vidljivo, skupini STM32F1:



Na kraju odabiremo konkretnu pločicu, odnosno mikrokontroler:



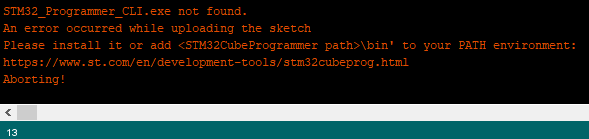
## Konfiguracija sklopa za programiranje i debugiranje

## Prijenos programa s razvojnog računala u mikrokontroler vrši se nakon unosa i prevođenja programa, no ovdje je taj postupak opisan ranije kako bi se zaokružio opis konfiguriranja razvojnog okruženja.

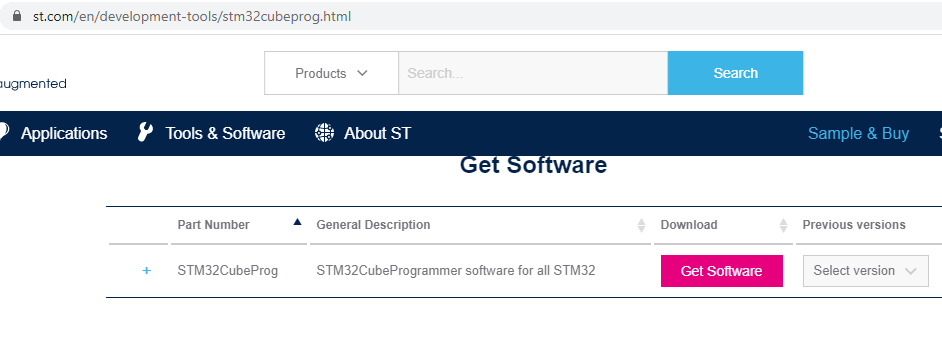
## Komunikacija razvojnog računala odnosno Arduino IDE programa s BluePill pločicom ostvarena je pomoću ST-Link sklopa, kao i u slučaju korištenja Atollic TrueSTUDIO programa, kako je već ranije opisano u vježbi 1. Program koji omogućuje korištenje ST-Link u tu svrhu nosi naziv “STM32CubeProgrammer” i treba ga odabrati na način prikazan na slici dolje:

## 

Ukoliko na računalu nije ranije instaliran programski paket „STM32CubeProgrammer“, koji služi za prijenos programa sa PC računala u FLASH memoriju mikrokontrolera na BluePill pločici, tada će se tijekom pokušaja prijenosa prevedenog programa u mikrokontroler pojaviti poruka o grešci koja obavještava da navedeni program treba instalirati:



Poruka o grešci pokazuje na internet stranicu s koje se može preuzeti program koji nedostaje, <https://www.st.com/en/development-tools/stm32cubeprog.html> :



Na navedenoj stranici treba klikniti na gumb „**Get Software**“. Prije preuzimanja bit ćete zatraženi da se prijavite ili, ukoliko nemate ranije otvoren korisnički račun kod tvrtke STM-a, da se registrirate. Procedura je jednostavna i treba jednostavno slijediti online upute.

Nakon izvršene prijave na stranici je omogućeno dohvaćanje datoteke **en.stm32cubeprog\_v2-4-0.zip.** Datoteka je kompresirana u .zip format te je nakon preuzimanja treba raspakirati u privremeni folder. Navedena zip datoteka sadrži instalacijski program za Windowse (**SetupSTM32CubeProgrammer-2.4.0.exe**) i za Linux (SetupSTM32CubeProgrammer-2.4.0.linux). Nakon raspakiranja treba pokrenuti instalacijski program SetupSTM32CubeProgrammer-2.4.0.exe .

Pri instalaciji može biti zatražena instalacija Java Runtime Environmenta (JRE) pa je u tom slučaju instalirajte koristeći poveznicu na prozoru koji se pojavljuje ukoliko je instalacija potrebna. Nakon instalacije JRE-a treba ponovno pokrenuti instalacijski program SetupSTM32CubeProgrammer-2.4.0.exe te slijediti jednostavan postupak instalacije.

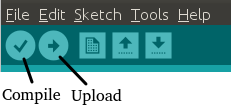
Na kraju treba još vrijednost (sadržaj) environment varijable **PATH** proširiti s nazivom foldera u kome se nalazi komandno-linijska inačica programa STM32CubeProgrammer (**STM32\_Programmer\_CLI.exe**). Navedeni program pokreće se od strane Arduino IDE programa tijekom prijenosa prevedenog programa u FLASH memoriju mikrokontrolera. Ukoliko je STM32CubeProgrammer program npr. Instaliran u u folder **C:\STM32\STM32CubeProgrammer**, tada vrijednost PATH envoronment varijable treba proširiti sa nazivom foldera **C:\STM32\STM32CubeProgrammer\bin** jer se STM32\_Programmer\_CLI.exe nalazi u **bin** subfolderu.

Naokon provedenih svih koraka treba restartati Arduino IDE kako bi se te promjene aktualizirale.

## 4.Unos, prevođenje i prijenos programa u mikrokontroler

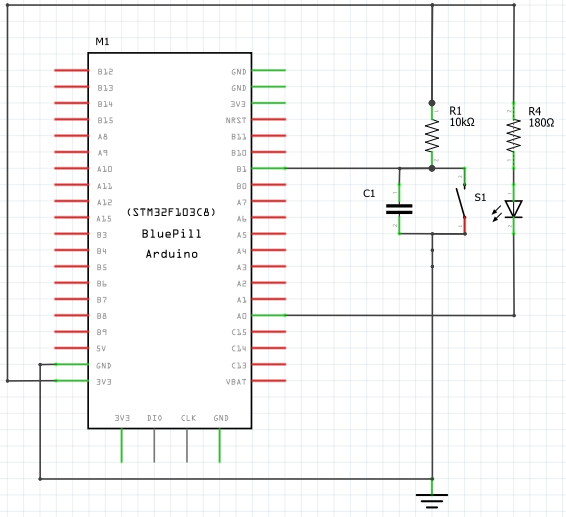
Postupak unosa prevođenja i prijenosa prevedenog programa je jednostavan I intuitivan i ne treba ga detaljnije opisivati.

Ukratko: nakon unosa programa u editoru koji čini najveći dio radne površine Arduino IDE programa, isti se prevodi odabirom **Sketch -> Verify/Compile,** ili pritiskom na prvu ikonu slijeva na alatnoj traci. Prijenos prevedenog programa u mikrokontroler obavlja se odabirom **Sketch -> Upload**, ili pritiskom na drugu ikonu slijeva na alatnoj traci.



**4.1 Primjer 1 : GPIO digitalni ulaz I izlaz**

U ovom primjeru bit će prikazan način korištenja digitalnih ulaza i izlaza mikrokontrolera u Arduino razvojnom okruženju. Isti je primjer ranije prikazan u vježbi 2 korištenjem Atollic TrueSTUDIO razvojnog okruženja. Implementacija istog primjera u različitim okruženjima poslužit će za njihovu usporedbu. Prisjetimo se, zadatak programa je čitati stanje tipkala spojenog na PA0 nožicu mikrokontrolera i prikazati stanje tipkala na svijetlećooj diodi (LED) spojenoj na nožicu PB1. Svijetleća dioda treba dakle svijetliti kad je tipkalo pritisnuto, a biti ugašena kad tipkalo nije pritisnuto. Shema sklopa prikazana je na slici dolje:



Arduino program kojim se implementira željena funkcionalnost krajnje je jednostavan:

#define BUTTON PA0

#define LED PB1

int value = 0;

*/\* funkcija setup() izvrsava se samo jednom na pocetku programa \**/

void setup() {

pinMode(BUTTON, INPUT\_PULLUP);

pinMode(LED, OUTPUT);

}

*/\* funkcija loop() izvrsava se ciklicki \**/

void loop() {

value = digitalRead(BUTTON);

digitalWrite(LED, value);

}

Program sadrži dvije funkcije: **setup()** i **loop()**. Svaki Arduino program mora sadržavati barem dvije funkcije: **setup()** i **loop()**. Funkcija setup() poziva se jednom na početku izvođenja programa, a nakon toga se ciklički poziva funkcija loop(). U tijelu fukcije setup() smješta se inicijalizacijski dio programa, npr. inicijalizacija ulaznih i izlaznih portova. Tijelo funkcije loop() sadrži algoritam koji će se izvoditi cijelo vrijeme dok je mikrokontroler spojen na izvor napajanja ili dok se ne izvrši reset mikrokontrolera.

Funkcijom **pinMode()** vrši se inicijalizacija pojedinog GPIO porta, a funkcijama **digitalRead()** i **digitalWrite()** vrši se čitanje odnosno upis jednog bita sa/na GPIO port mikrokontrolera.

Detajnje informacije o navedenih funkcijama, kao i opise mnogih drugih, mogu se potražiti odabirom **Help→Reference.** Do uputa vezanih za Arduino razvojno okruženje dolazi se odabirom **Help->Environment** ili posjetom stranici <https://www.arduino.cc/en/Guide/HomePage>.

Već na navedenom kratkom primjeru vidljivo je da su programi pisani korištenjem Arduino razvojnog okruženja jednostavniji od onih koji koriste Atollic TrueSTUDIO okruženje. Razlog tome prvenstveno leži u većoj razini razini apstrakcije korištenoj u Arduino okruženju koja omogućuje konfiguriranje određenje nožice mikrokontrolera samo jednom linijom koda (funkcija pinMode()). Ta jednostavnost u korištenju Arduino okruženja dolazi još više do izražaja prilikom korištenja složenijih perifernih sklopova (ADC, PWM, SPI,...). Viša razina apstrakcije ima naravno i svojih nedostataka, a to su u prvom redu skromnije mogućnosti u pogledu performansi i nemogućnsot pristupa nekim dijelovima mikrokontrolera. S aspekta debugiranja, TrueSTUDIO s ugrađenim mogućnostima korištenja breakpointa, izvršavanja programa korak po korak i praćenja vrijednosti varijabli tijekom izvođenja programa te mnogim drugim pogodnostima uvelike nadilazi mogućnosti Arduino okruženja.

**4.2 Primjer 2 : Treptalo**

U slijedećem primjeru želimo napraviti treptalo. Svijetleća diode spojena na PA0 nožicu mikrokontrolera , kao u prethodnom primjeru, treba se ciklički paliti i gasiti frekvencijom 1 Hz, s time da je trajanje perioda kada diode svijetli jednako trajanju perioda kada dioda ne svijetli.

Program izgleda ovako:

/\* Program svakih pola sekunde mijenja stanje na

\* svijetlecoj diodi (LED) koja je spojenan na nozicu PB1.

\*/

// umjesto naziva PB1 zelimo koristiti LED

#define LED PB1

// funkcija setup() izvrsava se samo jednom na pocetku programa

void setup() {

// inicijalizacija LED nozice (PB1) kao izlazne

pinMode(LED, OUTPUT);

}

// funkcija loop() izvrsava se ciklicki

void loop() {

// upis HIGH ('1') na LED

digitalWrite(LED, HIGH);

// pauza 500 ms

delay(500);

// upis LOW ('0') na LED

digitalWrite(LED, LOW);

// pauza 500 ms

delay(500);

}

Program prikazuje korištenje funkcije delay(), kojoj se kao ulazni parameter proslijeđuje vrijeme u milisekundama.

**4.3 Primjer 3 : Treptalo s potenciometrom za promjenu frekvencije treptanja**

U slijedećem primjeru želimo dodati treptalu iz primjera 2 mogućnost promjene frekvencije treptanja. Promjenu frekvencije ostvarit ćemo dodavanjem potenciometra vrijednosti otpora 10 kOhma na analogni ulaz mikrokontrolera. Svijetleća diode i dalje je spojena na PA0 nožicu mikrokontrolera , a klizač potenciometra spojit ćemo na nožicu A0 (PA0. Trajanje perioda kada diode svijetli jednako je trajanju perioda kada dioda ne svijetli.

Program izgleda ovako:

/\* Program mijenja frekvenciju treptanja svijetlece diodi (LED)

\* koja je spojena na nozicu PB1, ovisno o kutu zakreta potenciometra

\* (POT) otpora 10 KOhma čiji je klizac spojen na PA0 nozicu mikrokontrolera.

\* Nozicu PA0 (A0) konfigurirati kao analogni ulaz

\* Napomena: funkcija analogRead() za citanje vrijednosti napona s potenciometra

\* daje vrijednosti u rasponu [0, 1023]. Tu vrijednost uzeti kao

\* vrijeme trajanje pauze pri promjeni stanja na LED.

\* Radi se o dakle o pretvorbi napona u vrijeme (trajanje pauze)

\*/

#define POT PA0

#define LED PB1

int pot;

void setup() {

// inicijalizacija LED nozice (PB1) kao izlazne

pinMode(LED, OUTPUT);

}

void loop() {

pot = analogRead(POT);

digitalWrite(LED, HIGH);

delay(pot);

digitalWrite(LED, LOW);

delay(pot);

}

**4.4 Primjer 4 : Treptalo povezano s osobnim računalom putem serijske veze**

U narednom primjeru želimo dodati treptalu iz primjera 3 mogućnost slanja osobnom racualu spojenom putem serijske veze nana BluePill pločicu, informaciju o podešenom vremenu trajanja pauze na potenciometru. Serijsku vezu ostvarujemo korištenjem konvertorskog sklopa “USB to RS-232/TTL converter ” koji se operacijskom sustavu osobnom računala predstavi kao serijski port (COMx).

Program izgleda ovako: