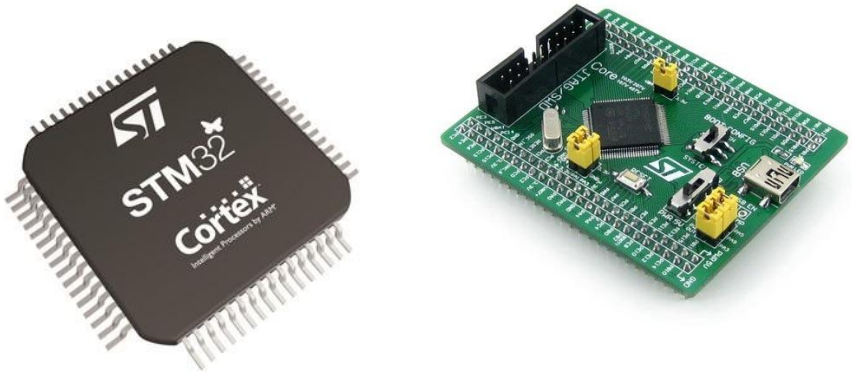
1. Tjedan - Uvod u STM32 & Izlazni pinovi

Uvod u svijet mikrokontrolera

Mikrokontroleri su **digitalni** elektronički uređaji vrlo slični računalima, samo s puno manje snage i memorije, koje je moguće isprogramirati kako bi izvodili neki određeni zadatak. U daljnjem tekstu mikrokontrolere ćemo skraćeno nazivati MCU-ovima. Kod MCU-ova potrebno je uvijek imati u vidu da su oni ograničeni sa svojim svojstvima kao i svaki drugi elektronički uređaj. Programer kod izrade i projektiranja nekog uređaja treba dobro poznavati njihova svojstva i ograničenja kako bi taj uređaj pravilno radio (primjerice, MCU ne može direktno napajati žarulju već je potrebno da on upravlja relejem koji onda napaja žarulju). Ograničenja MCU-ova postepeno će se upoznati kroz ovu edukaciju.

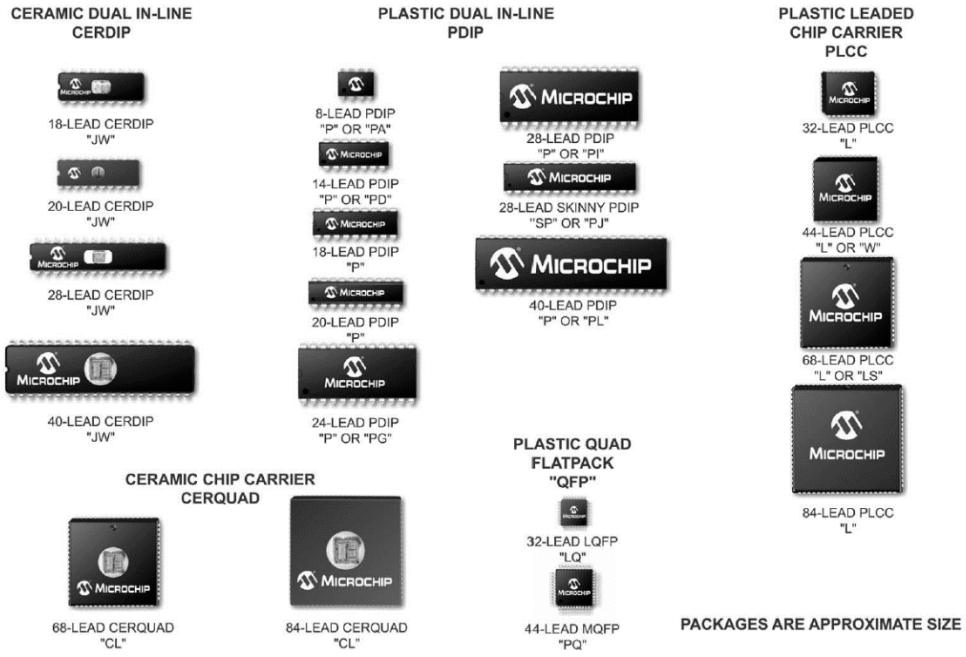
Danas na tržištu postoji niz različitih MCU-ova koji se razlikuju po broju pinova, memoriji, procesorskoj snazi i broju različitih komunikacijskih sabirnica. MCU često sadrži veći broj pinova, gdje svaki pin ima neku svoju definiranu funkciju. Određeni pinovi namijenjeni su za napajanje (VCC i GND), drugi za komunikaciju s vanjskim uređajima ili senzorima, a neki pinovi služe za spajanje na vanjske oscilatore (kvarcovi) itd.

MCU-ove, same po sebi, ne može se jednostavno „samo“ spojiti na napajanje i sa njima raditi. Potrebno je točno određeno napajanje da bi oni radili, a zbog propada napona usred veće potrošnje energije mora se blizu MCU-ova postaviti i kondenzatore koji služe kao dodatni izvor energije kako bi se to spriječilo. Zbog svega toga MCU-ove se uvijek lemi na PCB (engl. *Printed Circuit Board*) na kojemu se uz MCU nalaze sve ostale potrebne komponente da bi MCU mogao normalno raditi. **Bitno je zapamtiti da MCU-ovi ne mogu raditi bez ostalih elektroničkih komponenti**.



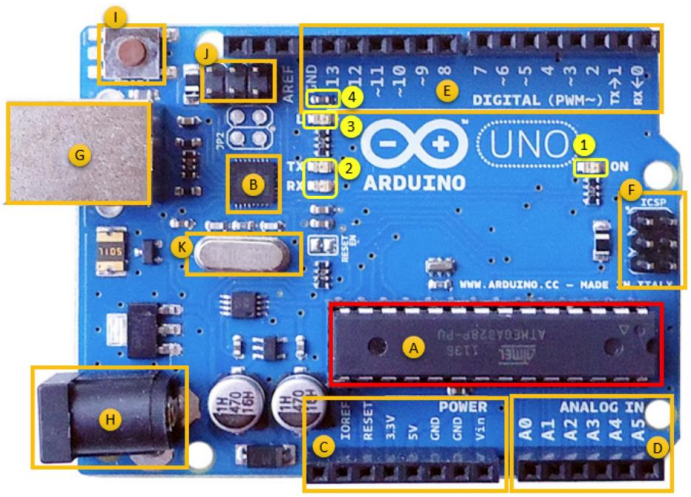
Danas se MCU-ove može kupiti u raznim „pakiranjima“. „Pakiranje“ označava kako MCU fizički izgleda. Izgled može varati, naime MCU s istim karakteristikama (procesor, memorija, periferija itd.) može drugačije izgledati. Tako za neki MCU može postojati više pakiranja s istim ili različitim brojem pinova, međutim sve ostale karakteristike su mu jednake. Jednako tako vrijedi da se u istom

pakiranju mogu nalaziti dva različita MCU-a sa sasvim različitim karakteristikama, u tom slučaju razlika se može raspoznati jedino čitanjem imena MCU-a otisnutog na kućištu.



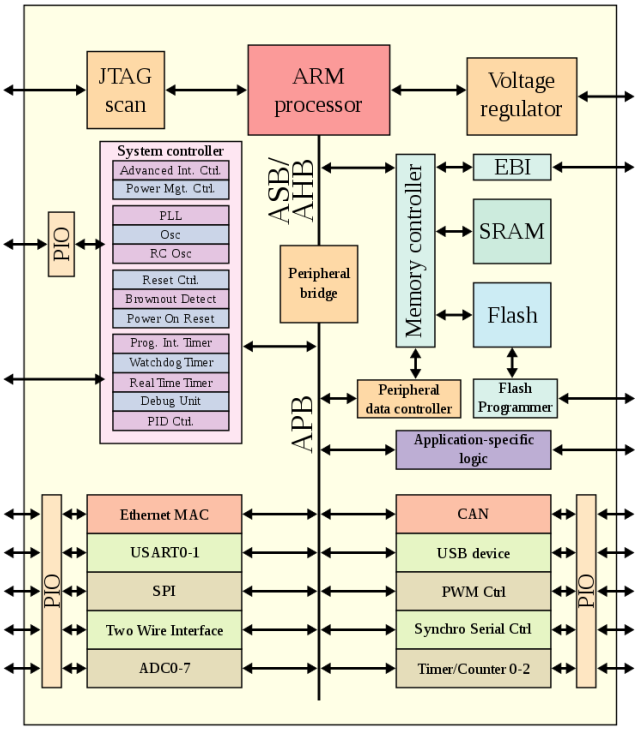
Arduino

U svijetu prepunom raznih elektroničkih kitova, svakako vrijedi spomenuti Arduino kao primjer uređaja koji se probio na tržište i danas je svakako najkorišteniji kit za hobiste. Arduino spada u elektroničke kitove/module koji se sastoje od PCB-a na kojem se nalaze sve ključne komponente kako bi njegov MCU radio. Tu se nalazi USB priključak iz kojeg MCU dobiva 5V napajanje, određene LED-ice koje označavaju da je uređaj aktivan, a tu je i sam MCU **ATmega328** kao glavni čip kojeg programiramo i s kojim radimo. Bitno je za napomenuti da uz glavni MCU na PCB-u nalazi se i još jedan „pomoćni“ MCU ATmega16U2 koji je već isprogramiran i njegov kod nije moguće mijenjati, a njegova zadaća je da se program s računala prebaci na **ATmega328**. Dakle ima funkciju programatora. Sa svake strane PCB-a nalaze se ženski konektori na koji su izvedeni pinovi MCU-a kako bi korisniku bili fizički dostupni. Iz opisa može se zaključiti da je Arduino zapravo **skup više elektroničkih komponenti**. Najveća snaga Arduino proizvoda je njegova ogromna zajednica na internetu gdje hobisti mogu pronaći sve materijale i instrukcije (software-ski kod) za izradu svojih prototipova.



ARM

Svaki MCU unutar sebe sastoji se od niza blokova koji ga definiraju. Najvažniji dio svakog MCU-a njegova je jezgra koja definira kako su određene funkcionalnosti implementirane, način kako se handle-aju prekidi, na koji se način upisuje u registre, gdje se nalaze registri u memoriji, dostupne sabirnice u sustavu itd. Osim jezgre tu postoje i periferijski izlazi za komunikaciju, PLL, RAM, ROM itd. Svi navedeni blokovi zajedno čine MCU. Ovi blokovi kroz edukaciju detaljno će se proučiti i objasniti, a trenutno ih spominjemo da se dobije „šira“ slika sustava.



ARM je tvrtka koja je specijalizirana samo za dizajn arhitekture jezgre MCU-a. Imaju zanimljiv poslovni model s kojim su ostvarili velike prihode, a sastoji se u tome da firme za izradu MCU-a otkupljuju ARM-ovu licencu za korištenje njihovog dizajna jezgre. Tvrtke proizvode MCU-ove pod svojim imenom, ali s ARM arhitekturom. Danas je ARM firma u vlasništvu SoftBank-a (japanska tvrtka) koja je tu akviziciju platila **23,4 milijarde** britanskih funti (2016.). MCU-ovi s ARM jezgrom zastupljeni su u velikoj većini današnjih elektroničkih uređaja (kao i u Apple-ovim procesorima).

ARM je dizajnirao više različitih arhitektura jezgara. Ona s kojom će se raditi kroz edukaciju je „**Cortex-M**“ namijenjena real-time sustavima s brzim odzivom, ali zato s malo resursa (RAM i ROM). S druge strane vjerojatno najpopularnija je „**Cortex-A**“ arhitektura jezgre namijenjena grafičkim aplikacijama gdje brzina odziva nije kritična i koristi se u većini mobitela, ali i kod nekih poznatih elektroničkih modula kao **BeagleBone** i **Raspberry Pie**. Cortex-M arhitektura namijenjena je „**hard real time**“ sustavima, a Cortex-A je namijenjena „**soft real time**“ sustavima. „Hard real time“ sustavi su sustavi u kojima je jako bitan odziv koji mora biti deterministički. Većina sustava koji su zasnovani u „hard real time“ domeni koriste se u industriji, pogotovo u strojevima koje koristi čovjek (automobil, avion, senzori visoko naponske mreže, senzori detekcije vlakova i spuštanja rampi, vojni uređaji i strojevi itd), gdje je pravovremenost i determinizam reakcije ključan da se ne dogodi ozljeda. „Soft real time“ sustavi većinom su komercijalni sustavi za široku potrošnju, a najbolji primjeri su mobiteli i računala. Kod računala odziv miša nije deterministički, on se nekad desi za 1ms, a ponekad za 2ms. Za takav sustav to je prihvatljivo ponašanje jer u slučaju da se prekorači traženo vrijeme odziva, sustav se neće urušiti niti će doći do ozljeda ili bilo kakvih štetnih posljedica. Temeljne razlike između Cortex-A i Cortex-M najbolje prikazuje donja usporedba:

ARM Cortex-M4	ARM Cortex-A8
Hard real time sustavi	**Soft** real time sustavi
Programira se u **C-u** (prelazi se na **C++**)	Programira se u **C++**
Može se staviti real-time operacijski sustav, al i ne mora („*bare metal*“)	Ima **Linux** OS
Clock do **180MHz** (STM32F427)	Clock do **1GHz** (AM335x na Beaglebone-u)
256 Kbyte RAM (STM32F427)	**512 MB** RAM (AM335x na Beaglebone-u)
2 MB Flash-a (STM32F427)	**4 GB** eMMC-a (AM335x na Beaglebone-u)

MCU s Cortex-A arhitekturom se vrti na Linux OS-u zbog većih resursa procesora, među kojima su i komponente za obradu dinamičke alokacije memorije (engl. *Memory Managment Unit*).

STM32

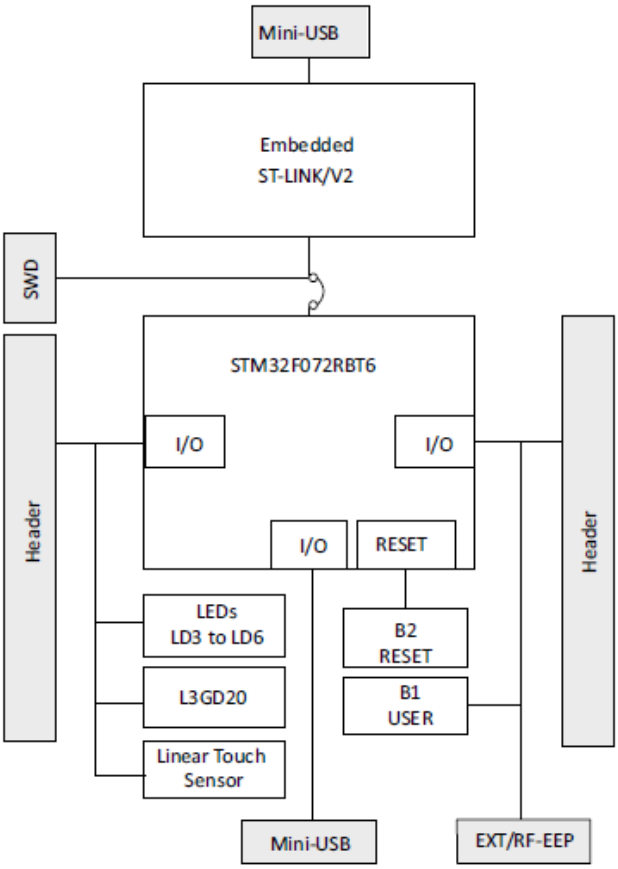
STM32 je proizvođač MCU-ova koje ćemo koristiti kroz cijelu ovu edukaciju. STM32 vjerojatno je najveći i najzastupljeniji proizvođač MCU-ova s ARM **Cortex-M** jezgrom. On u svojoj ponudi, uz same MCU-ove, nudi i niz elektroničkih kitova i modula za hobiste, ali i za firme kako bi njihovi programeri počeli što prije pisati kod (brži „*time to market*“). U prvom dijelu edukacije koristi se **STM32 Discovery kit**, odnosno elektronički modul nalik na Arduino koji na sebi ima već sve komponente potrebne za rad. STM32 puno je napredniji od Arduina, nudi puno više kontrole i fleksibilnosti svih sustava, ali zbog toga je i kompliciraniji za rad, naročito u početku. Kasnije tijekom edukacije prelazi se na **STM32 Nucleo** koji je vrlo sličan Discovery kit-u, ali opet u nekim detaljima različit. STM32 ima svoju nomenklaturu označavanja naziva MCU-ova. Tako se u njihovoj paleti proizvoda mogu pronaći MCU-ovi naziva: STM32F0, STM32F4, STM32L0 itd. Slova i brojevi zapravo označavaju o kojoj **Cortex-M** jezgri se radi. Na ispod priloženoj tablici može se vidjeti kategorizacija čipova. U pravilu što je veći broj to označava jači MCU s više procesorske snaga i više memorije.

STM32F072 Discovery kit (32F072BDISCOVERY)

STM32F072 Discovery kit, službenog naziva **32F072BDISCOVERY** sastoji se od cijelog niza elektroničkih dijelova čiji je cilj osigurati pravilan rad MCU-a **STM32F072RBT6** (**ARM Cortex-M0**), kojega programiramo i koji izvodi naše naredbe. Objasnit ćemo samo osnovne dijelove ovog Discovery-a, dokle kompletnu funkcionalnost će se postepeno objašnjavati kroz daljnju edukaciju.

Discovery se napaja putem USB-a od kuda napajanje vuče i sam MCU. Pinovi MCU su izvedeni sa svake strane PCB-a. Sadrži dvije tipke, od kojih je jedna za resetiranje („RESET“) MCU-a, dokle druga tipka („USER“) je spojena na ulazni pin MCU-a čija se namjena može isprogramirati. Dodatno sadrži 4 LED-ice koje se nalaze oko žiroskopa koji je povezan s MCU-om preko komunikacijske sabirnice, a na dnu Discovery-a nalazi se linearni senzor dodira. Jedna od stvari koje se kroz ovu edukaciju uče je i pronalaženje te čitanje manual-a i datasheet-a za određene senzore i MCU-ove kako bi ih polaznici znali koristiti. Na temelju „**User Manual**“-a **UM1690** (dostupan na LMS-u) u **tablici 6** mogu se vidjeti svi pinovi koji su izvedeni s MCU-a na krajeve PCB-a. Iz tablice se može uočiti da svaki pin se označava prvo slovom pa onda brojem (pr. PA7, PB0…), to je standardna nomenklatura pinova na većini MCU-ova. Slova uz broj pina (A,B,C…) nazivaju se „**Port**“-ovi koji uz sam broj čine **ID pina**. Iz tablice je moguće pročitati koji pinovi su spojeni na određene tipke i senzore, a koji pinovi su slobodni. Imena pinova su otisnuti i na PCB-u.

Većina pinova MCU provedeni su do konektora, bitno je za naglasiti da **pojedini pinovi osim što su provedeni do konektora na PCB-u, spojeni su i na određene senzore, LED-ice, tipke itd**. Na donjoj slici blokovski je prikazan način na koji je to spojeno.



MCU **STM32F072RBT6** sadrži sljedeće karakteristike:

STM32F072RBT6	
Jezgra:	**ARM Cortex-M0**
Max frekvencija:	**48 MHz**
Flash memorija	**128 KB**
SRAM memorija:	**16 KB**
Broj pinova:	**64**
Pakiranje:	**LQFP64**

Početak rada i Prvi program

Za programiranje Discovery-a koriste se dva software-a koje je potrebno instalirati:

- Keil MDK 5
  - Keil je IDE (engl. *Integrated Development Environment*), odnosno razvojni program u kojemu pišemo naš C kod, debuggiramo te compile-iramo
  - Keil je **službeni IDE alat od ARM-a**, iako osim Keila STM32 može se programirati putem niza ostalih IDE alata tipa Eclipse, IAR itd.
  - Za instalaciju Keil-a pratiti prezentaciju: ***Keil instalacija i licenca.pdf***
  - Dodatno za kratko objašnjenje Keil razvojnog okruženja pogledati: ***Keil osnove razvojnog okruženja.pdf***
- STM32CubeMX
  - Cube je razvojni alat za STM32 koji omogućava brži **početni** razvoj programa za MCU. Ovaj alat razvijen je od strane STM32 s ciljem olakšavanja posla programerima, jer se na temelju konfiguracije postavljene u Cube-u generira početni kod. Međutim ima i par nedostataka o kojim će biti govora u sljedećim tjednima.
  - Za instalaciju Cube-a pratiti prezentaciju: ***STM32CubeMx instalacija.pdf***

Uz instalaciju ova dva software-a potrebno je instalirati i **driver** za STM32 kako bi računalo prepoznalo STM32 Discovery kada ga priključimo (***ST-link driver instalacija.pdf***).

Gore navedene upute za instalaciju moguće je skinuti sa službene **LMS Immersa web stranice pod *1. Uvod u STM32 & Izlazni pinovi* u folderu *Software instalacija***.

O Keil-u i Cube-u moguće je naći dodatne informacije na internetu, na njihovim službenim web stranicama.

Nakon instalacije Keil-a, Cube-a i driver-a, za postavljanje svog prvog projekt pratiti upute iz prezentacije: ***STM32CubeMx postavljanje projekta.pdf***. Prije nastavka čitanja potrebno je postaviti svoj prvi projekt kako je opisano u dokumentu.

Nakon što je Cube izgenerirao potreban kod za Keil, može se krenuti s njegovim objašnjavanjem. Program kreće iz **main.c** datoteke i odmah se može uočiti da na dnu main funkcije postoji **beskonačna while petlja**. Za razliku od normalnih programa koji mogu imati kraj, kod MCU-ova **uvijek postoji** while petlja gdje se naš kod izvršava u beskonačnosti dok se MCU ne zagasi, odnosno izgubi napajanje. Bez obzira na okolnosti, svi programi za MCU **moraju** imati beskonačnu petlju, kod se ne smije nikada prestati izvršavati.

Unutar **main funkcije** stvoreno je niz ostalih funkcija koje inicijaliziraju određenu periferiju MCU-a. Ispod su navedene i objašnjene samo neke od funkcija :

- HAL_Init()
  - Inicijaliziraju se prekidi u slučaju greške, periodični prekid od system timera itd.
- SystemClock_Config()
  - Inicijalizira se clock (frekvencija) sustava i svih sabirnica

    - Popunjavaju se tri strukture sa određenim vrijednostima ovisno o postavkama u Cube-u (define-ovi)
- MX_GPIO_Init()
    - Inicijalizira pinove koji su postavljeni u Cube-u
    - Popunjava se određena struktura sa tim vrijednostima
    - Funkcija je detaljnije objašnjena u prezentaciji ***MX_GPIO_Init - objasnjenje koda.pdf***

Kratkim pregledom koda bitno je za uočiti par stvari koje se će ponavljati kroz cijelu edukaciju. Kod se zasniva **na puno struktura** koje definiraju opcije određenog pina, komunikacijske sabirnice, timer-a itd. Većina vrijednosti kojima se strukture popunjavaju su **define**-ovi, odnosno određene konstante kojima punimo te strukture te preko kojih MCU inicijalizira određenu periferiju. Koja konstanta će se koristiti ovisi o našim postavkama u Cube-u.

Osim struktura i konstanti bitno je za uočiti i niz već definiranih funkcija koje se pozivaju. To su **HAL funkcije** (engl. *Hardware Abstracion Layer*) koje su napisane od strane STM32 i koje programerima olakšava pisanje koda. Prije je bilo potrebno za postaviti neki pin u logičku jedinicu upisati niz vrijednosti u određene registre. Zbog takvog načina pisanja koda, s direktnim upisom u registre, razvoj cijelog *firmware*-a (naziv za software u MCU-u) je tekao jako sporo. Kako bi se to ubrzalo većina proizvođača MCU-a napravila je svoje library-e s funkcijama koje brinu o upisu u registre, kako korisnik o tome ne bi trebao brinuti. Primjerice, upis vrijednosti u izlazni pin uz pomoć HAL library-a danas se izvodi preko jedne funkcije, međutim da nema te gotove funkcije bili bi prisiljeni pisati vrijednosti direktno u registre što bi značajno produljilo vrijeme razvoja proizvoda. HAL funkcije koriste se apsolutno u svim primjerima, te ujedno i Cube kao razvojni alat upravo se temelji na njima.

Izlazni pinovi

Na početku dokumenta spominjala su se određena ograničenja MCU-ova koja će se kroz edukaciju upoznati. Jedno od prvih svojstva MCU je da je **digitalan**. To znači da kod njega vrijedi samo 0 ili 1. Konkretno, izlazni pinovi na MCU-u koji su upravljivi od strane MCU-a (ne pinovi napajanja) mogu biti postavljeni ili na **3,0V (što je jednako naponu napajanja MCU-a)** ili na **0V**. **Dakle samo dva stanja postoje**. Ujedno izlazni pinovi, osim napona, imaju ograničeni iznos struje koji mogu dati od maksimalno **25mA** (ograničenja u napajanju MCU-a, kao i izlaznim strujama detaljno su opisani u datasheet-u MCU-a).