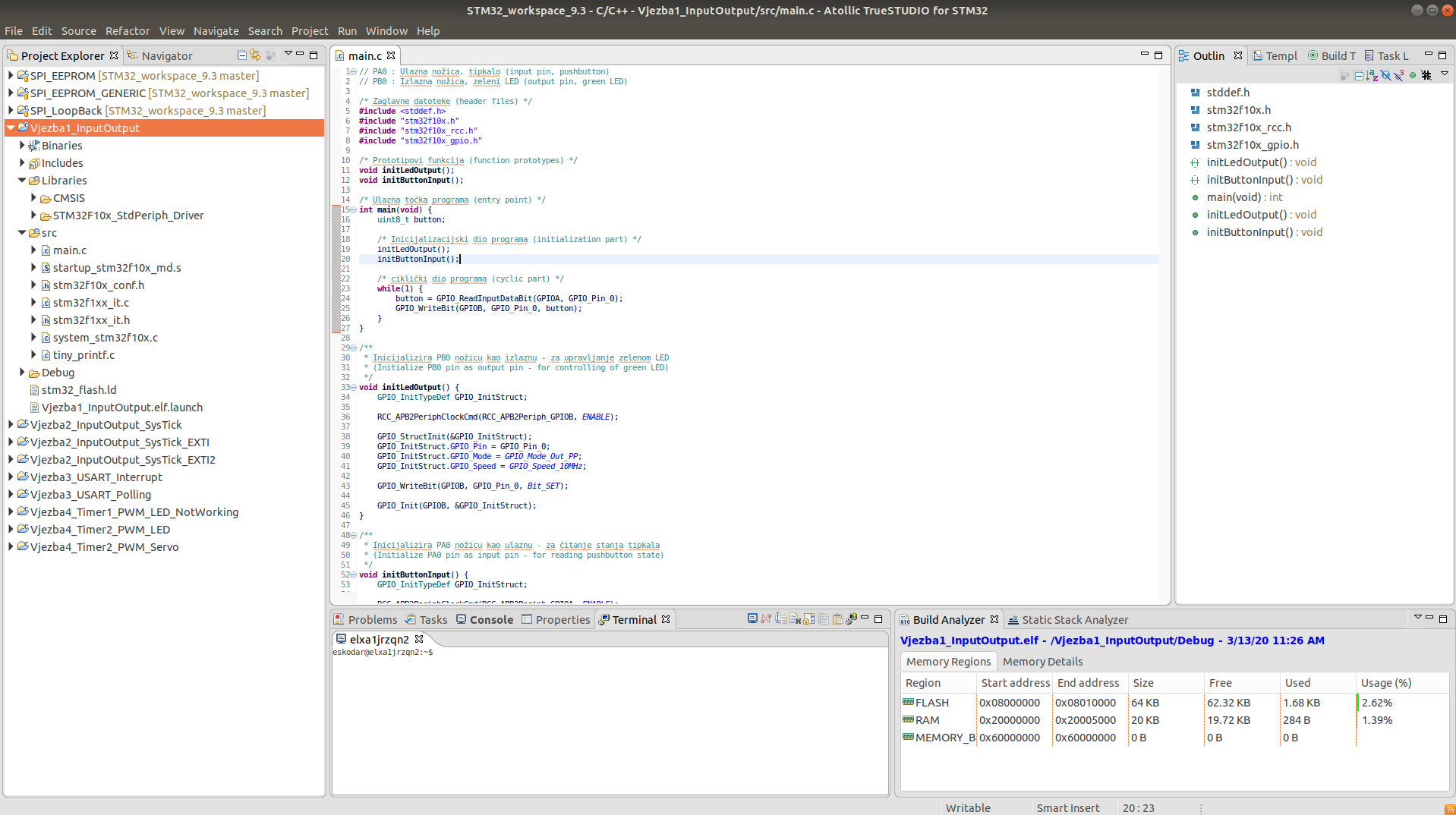
## Laboratorijska vježba 1:

## Upoznavanje s programskom i sklopovskom razvojnom okolinom

## Programska razvojna okolina

Vježbe se izvode korištenjem programskog paketa **Atollic TrueSTUDIO** za STM32 seriju mikrokontrolera.

Radi o cjelovitoj razvojnoj okolini (IDE), baziranoj na vrlo popularnoj *Eclipse* platformi, koja uključuje editor za unos programskog koda, prevodioc za C i asembler, debugger i mnoge druge pomoćne programske alate.



Instalacijski paket može se skinuti preko poveznice <https://atollic.com/resources/download/>, a pritom treba odabrati operacijski sustav na koji će se paket instalirati – podržani su Windowsi i Linux.

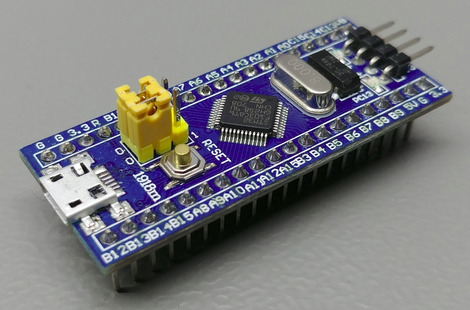
U programskom paketu dolazi i biblioteka *Standard Peripheral Library* čije korištenje znatno skraćuje i pojednostavljuje razvoj programa.

Atollic TrueSTUDIO paket je besplatan i kao takav može se slobodno koristiti kako u razvojne i obrazovne, tako i u komercijalne svrhe, bez ikakve naknade.

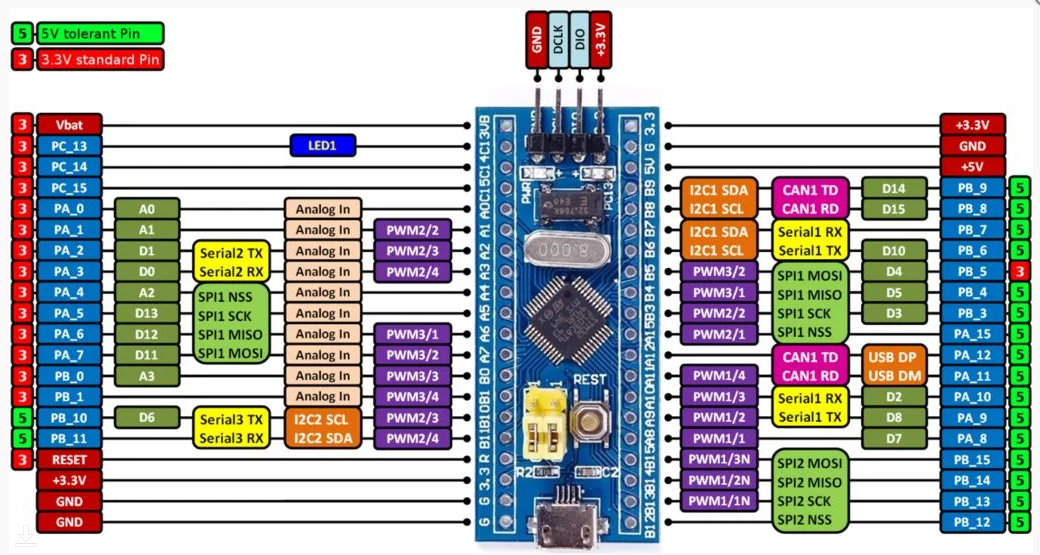
*Napomena:* Za nove projekte tvrtka ST preporučuje korištenje programskog paketa STM32CubeIDE umjesto Atollic TrueSTUDIO paketa, koji donosi razna poboljšanja i olakšava proces konfiguriranja mikrokotrolera korištenjem grafičkog alata.

## Sklopovska razvojna okolina

Na vježbama se koristi razvojna pločica popularnog naziva “BluePill“. Radi se o jednostavnoj i cijenom vrlo pristupačnoj pločici (oko 20 Kn) kojom upravlja mikrokontroler STM32F103C8T6.

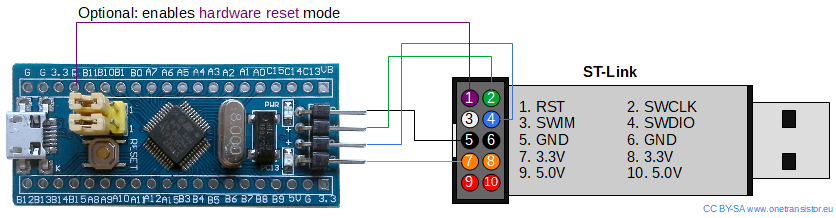


Sve nožice mikrokontrolera spojene su na dva jednoredna konektora duž pločice što omogućuje lako spajanje s vanjskim elementima – svijetlećim diodama, tipkalima, temperaturnim senzorima, servo-motorima, displayima i raznim drugim senzorima i izvršnim uređajima. ećina se nožica može programski konfigurirati za obavljanje jedne od više mogućih funkcija, a raspored nožica na konektorima kao i njihove mogućnosti prikazuje slijedeća slika.



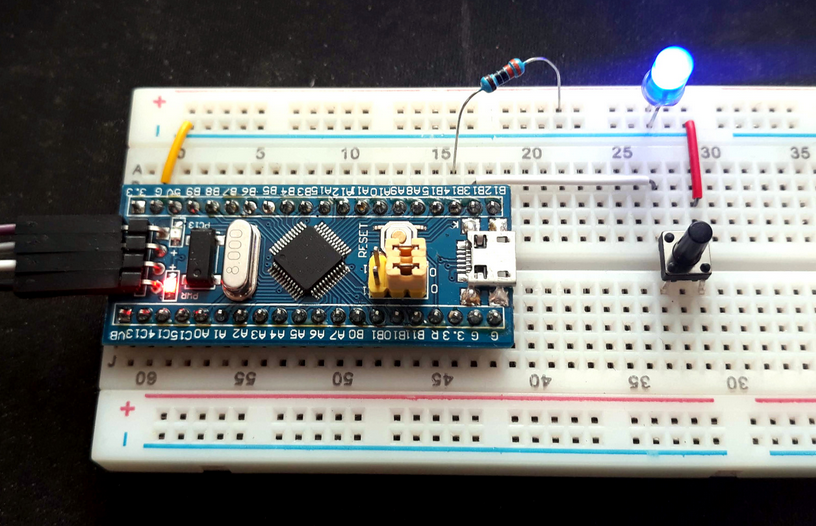
## Povezivanje programske i sklopovske razvojne okoline

Povezivanje programskog razvojnog okruženja, tj. programa Atollic TrueSTUDIO koji se izvodi na razvojnom računalu – najčešće PC računalu, sa sklopovskim razvojnim okruženjem, tj. sa BluePill pločicom, vrši se pomoću posebnog slopa, in-circuit debugger-a ST-LINK/V2 (u nastavku ST-LINK debugger). ST-LINK omogućuje ne samo prijenos programa sa razvojnog računala na STM32 mikrokontroler na BluePill pločici, već i njegovo debugiranje. STM32 mikrokontroleri podržavaju 2 načina povezivanja s debugerom: JTAG i SWD. SWD (Serial Wire Debug) je novijeg datuma i omogućuje sve što i JTAG i to korištenjem samo 4 linija za povezivanje: SWDIO, SWCLK, 3.3V I GND), kao što je prikazano na slici. Njegova je cijena oko 30 Kn.



## Izrada prototipa korištenjem protopločice

Korištenje protopločice (engl. Protoboard) omogućuje jednostavan i brz razvoj prototipova novih sklopova, jer se vanjski elementi (kondenzatori, otpornici, integrirani krugovi, servo-motori, displayi) povezuju s BluePill pločicom putem u protopločici ugrađenih veza, ili korištenjem vodiča za spajanje. Razvoj prototipa često zahtjeva manje ili veće izmjene prvotno zamišljenog sklopa. Pošto korištenje protopločice omogućuje povezivanje bez lemljenja, promjene na sklopu rade se jednostavnim dodavanjem, zamjenom, premještanjem ili drukčijim povezivanjem elemenata. Kad smo naposlijetku zadovljni radom sklopa možemo izaditi tiskanu pločicu i povezati elemente lemljenjem.

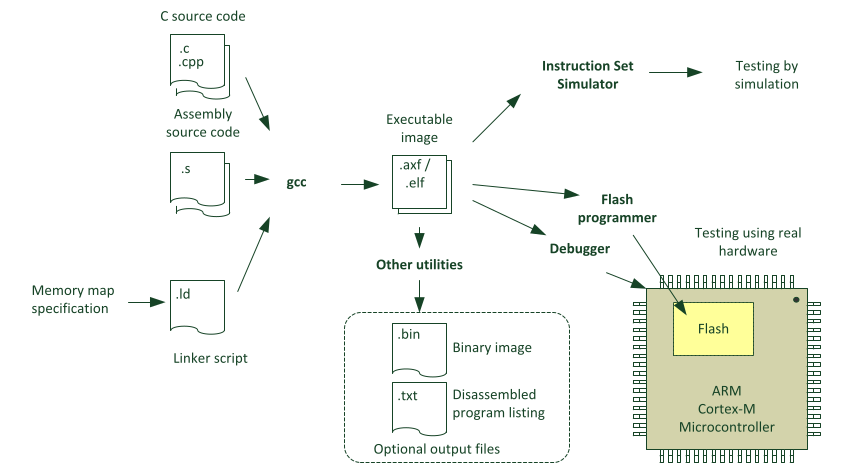


## Izrada programa

Osim programskog izvornog koda koji osmišljava programer, program u širem smislu čine i *Standard Peripheral Library* biblioteka, *linker skripta* koja definira način povezivanja programskog koda i *biblioteka* u izvršnu datoteku, kao i mnoge druge datoteke i postavke koje zajedničkim imenom nazivamo *Projekt*.

Mnogi programski alati, među kojima i Atollic TrueSTUDIO koriste tzv. *projektni način rada*. On podrazumjeva da u početku razvoja novog programa korisnik prvo kreira projekt u koji potom dodaje datoteke izvornog koda te po potrebi i ostale datoteke. Pri kreiranju projekta Atollic TrueSTUDIO zahtjeva od korisnika odabir raznih opcija koje koristi za konfiguriranje razvojnog okruženja. Time oslobađa korisnika od editiranja konfiguracijskih datoteka i omogućuje mu da se posveti samom kodiranju.

Nakon kreiranja projekta, korisnik upisuje izvorni kod, najčešće u programskom jeziku C, a ponekad i u asembleru. Potom se izvorni kod pomoću **gcc** prevodioca (engl. compiler) prevodi u tzv. objektni binarni kod, a nakon toga korištenjem programa za povezivanje **ld** (engl. linker) zajedno sa korištenim biblioteka povezuje u izvršnu binarnu datoteku (engl. executable file). Izvršna datoteka najčešće je u ELF formatu (Executable and Linkable Format) čiji sadržaj se korištenjem ST-LINK flash programatora upisuje u flash memoriju mikrokontrolera. Cijeli proces shematski je prikazan na slijedećoj slici.

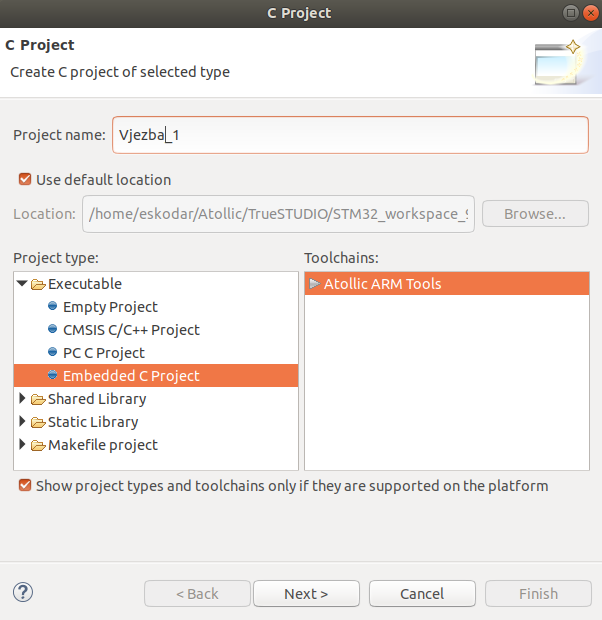


### 5.1 Kreiranje projekta

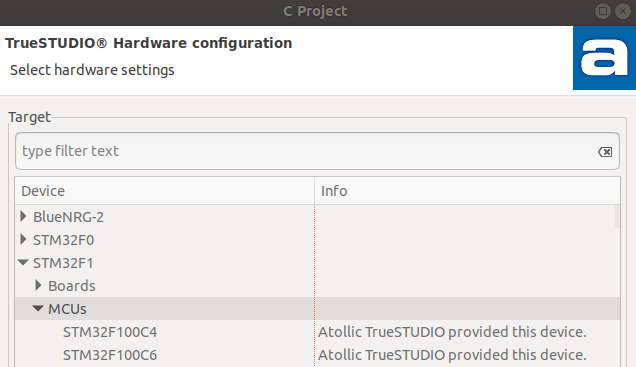
Nakon pokretanja programa Atollic TrueSTUDIO na jedan on načina koji to omogućuje opercijski sutav (dvoklik na ikonu, upis komande za pokretanje programa i sl.) na zaslonu računala pojavljuje se grafičko sučelje programa nalik onome na slici prikazanoj u poglavlju 1.

Postupak kreiranja novog projekta je slijedeći:

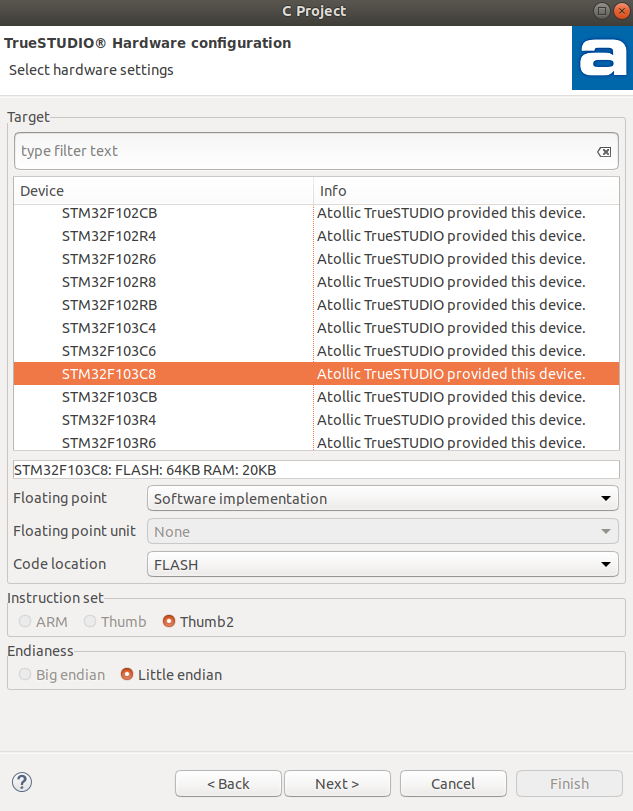
* Odaberite **File → New → C Project.** Time se otvara i na zaslonu prikazuje prozor *C Project*
* U novootvorenom *C Project* prozoru uradite slijedeće:
  + Za opciju *Project type* odaberite **Embedded C Project**
  + Za opciju *Toolchains* odaberite **Atollic ARM Tools**
  + Pod *Project name* upišite naziv projekta, npr.: *Vjezba\_1* i pritisnite gumb *Next*

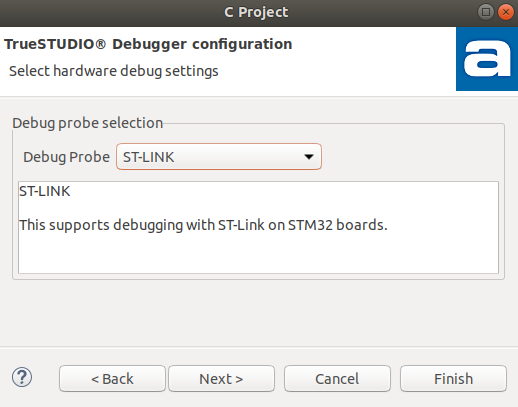


* Slijedi odabir korišenog mikrokontrolera. Na BluePill pločici nalazi se STM32F103C6T8 mikrokontroler. Za njegov odabir slijedite korake:
  + U panelu Target odaberite STM32F1 → MCUs

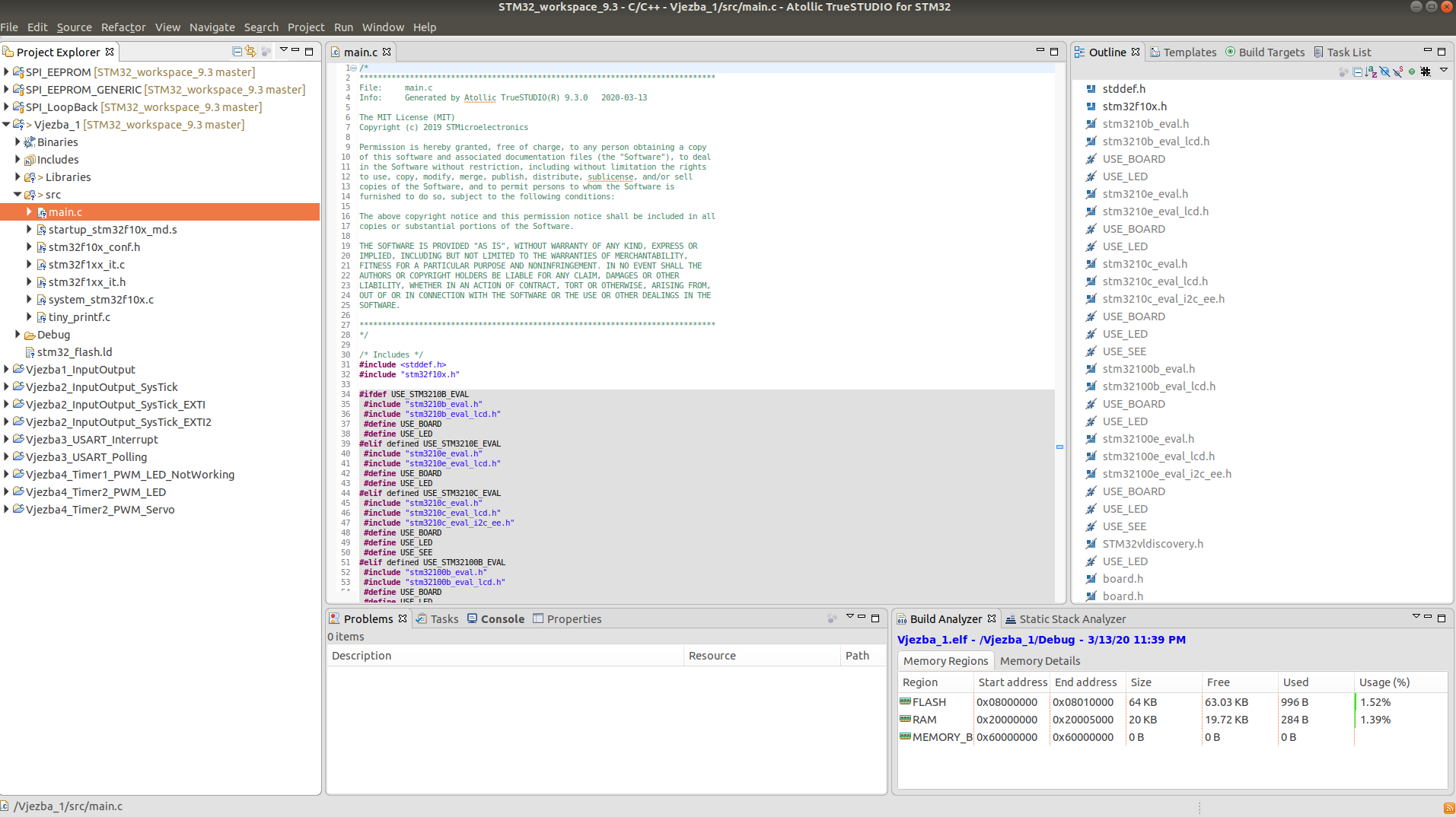


* + Odaberite **STM32F103C6T8** i kliknite *Next*



* Na slijećom prozoru ostavite sve opcije kako su podešene i kliknite samo *Next*
* U narednom prozoru kao *Debug Probe* odaberite **ST-LINK** i kliknite *Finish*

Nakon toga Atollic TrueSTUDIO automatski generira više datoteka, među kojima i **main.c** datoteku koja je predložak koji koristimo za upis željenog programa. Cijeli projekt se nakon toga automatski prevodi.



Uočite u *Project Explorer* panelu novokreirani projekt **Vjezba\_1** i **main.c** datoteku koja sadrži izvorni C kod. Build Analyzer panel (dolje desno) prikazuje zauzeće FLASH i RAM memorije mikrokontrolera od strane izvšnog programa **Vjezba1\_elf.**

Automatski generirana datoteka **main.c** sadrži mnogo linija koda koje uglavnom služe za inicijalizaciju serijskog porta i LCD display-a za tzv. STM32 Discovery razvojne pločice. Kako na vježbama koristimo BluePill razvojnu pločicu, većina generiranog koda u main.c datoteci je suvišna. Predložak koji možemo koristiti kao početni prikazan je i opisan u nastavku.

### 5.2 Predložak programa

### Osnovni predložak programa tj. main.c datoteke, koji ćemo kasnije nadograđivati, slijedećeg je sadržaja:

/\* Zaglavne datoteke (header files) \*/

**#include** <stddef.h>

**#include** "stm32f10x.h"

**int** **main**(**void**) {

// inicijalizacijski dio programa (initialization part)

// ciklički dio programa (cyclic part)

**while**(1) {

}

}

### Program započinje uključivanjem dviju zaglavnih datoteka. C Prevodioc će svaku od tih dviju linija zamijeniti (doslovce!) sadržajem navedenih datoteka. Datoteka stddef.h sadrži, između ostalog, definicije mnogih tipova podataka, koje možemo koristiti u programu. Datoteka stm32f10x.h sadrži simboličke nazive CPU registara kao i definicije struktura koje opisuju periferne sklopove.

### Svaki C program mora sadržavati main() funkciju. Funkcija main() ulazna je točka (engl. entry point) svakod C programa. Drugim riječima, izvršavanje C programa započinje pozivom main() funkcije. Tijelo main() funkcije sadrži dva glavna dijela, naznačena linijskim komentarima: *inicijalizacijski dio* i *ciklički dio*. U inicijalizacijski dio smješta se dio koda koji se treba izvršiti samo jednom, a to je najčešće inicijalizacija perifernih sklopova koje program koristi. Periferni sklopovi mogu biti ugrađeni u sam mikrokontroler, ili se na njega mogu spojiti izvana.

### Ciklički dio čini beskonačna petlja *while(1) {}* u kojoj se izvršava glavni dio programa. Bekonačna petlja, koja se najčešće samo pogreškom javlja u uobičajenim programima, postaje pravilo u programima za ugradbene sustave (engl. embedded systems). Za razliku od C programa namijenjenih osobnim računalima čijim radom upravlja operacijski sustav Windows, Linux, MacOS ili neki drugi, C programi namijenjeni ugradbenim sustavima poput BluePill-a koji koristimo na vježbama, često se izvršavaju bez posredstva operacijskog sutava. Njih izvršava direktno mikrokontroler. Takvi se programi na engleskom govornom području nazivaju *bare-metal p*rogramima.

### U slučaju izvršavanja programa posredstvom operacijskog sustava, nakon završetka izvođenja programa, program vraća kontrolu operacijskom sustavu. Kako na BluePill sklopu ne koristimo operacijski sustav, nameće se pitanje: kome u slučaju završetka programa predaje kontrolu program koji je netom završio izvršavanje. Za precizan i detaljan odgovor trebalo bi zaviriti “ispod haube” u dio programskog koda za koji je uobičajen naziv *startup code*. Ukratko: procesor bi najčešće nastavio s izvršavanjem nove prazne beskonačne petlju, ili bi pak završio u modu za obradu greške (engl. *fault interrupt handler*). Detaljnije informacije pronađite na poveznici[[1]](#footnote-1). Bare-metal program dakle ne završava svoje izvođenje dokle god je uređaj na kome se program izvršava priključen na napajanje.

### 5.3 Prevođenje i pokretanje programa

### Prevođenje programa vrši se na način da u Project Explorer prozoru odaberemo prvo projekt koji želimo prevesti. Nakon što desnim gumbom miša kliknemo na naziv projekta, javlja se izbornik na kome treba odabrati opciju Build Project.

### 

### Ukoliko nema sintaksnih pogrešaka, prevedeni se program može prenijeti na BluePill sklop i na njemu izvršiti na način opisan maločas, samo umjesto opcije Build Project treba odabrati Debug As, a potom Embedded C/C++ Application. Naravno, prije toga BluePill treba preko ST-Link debuggera treba spojiti s računalom.

### 

### Program će se nakon prenošenja na BluePill na njemu I pokrenuti I automatski zaustaviti na prvoj izvršnoj naredbi u main() funkciji, kako bi nam omogućio odabir načina izvođenja programa naredbu po naredbu, ili standardno izvođenje programa.

### 

### Pritiskom na pojedinu ikonu na alatnoj traci (engl toolbar) moguće je odabrati različite načine izvođenja programa koji mogu znatno olakšati traženje i popravljanje logičkih grešaka u programu. Istražite ih!

1. http://www.e-reading-lib.com/chapter.php/129272/37/michael-barr-programming-embedded-systems-in-c-and-c.html [↑](#footnote-ref-1)