ARM Delavnica *

Tomaž Perčič, Jernej Podlipnik

April 2013

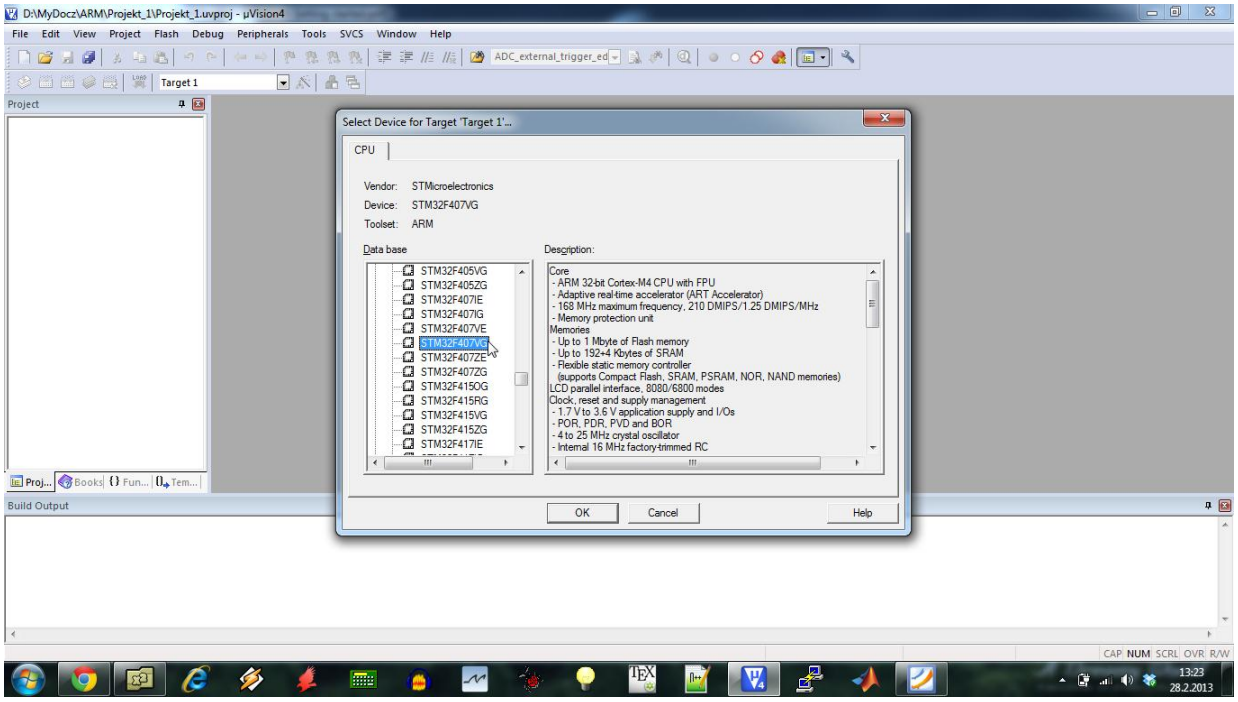
1 Vzpostavitev delovnega okolja

Razvojno ploščico STM32F4 Discovery je mogoče programirati v več razvojnih okoljih. Na tej delavnici bomo uporabljali KEIL uVision 4.6/4.7. Za začetek bomo vzpostavili delovno okolje. Pri vzpostavitvi delovnega okolja za našo razvojno ploščico se držimo naslednjih korakov.

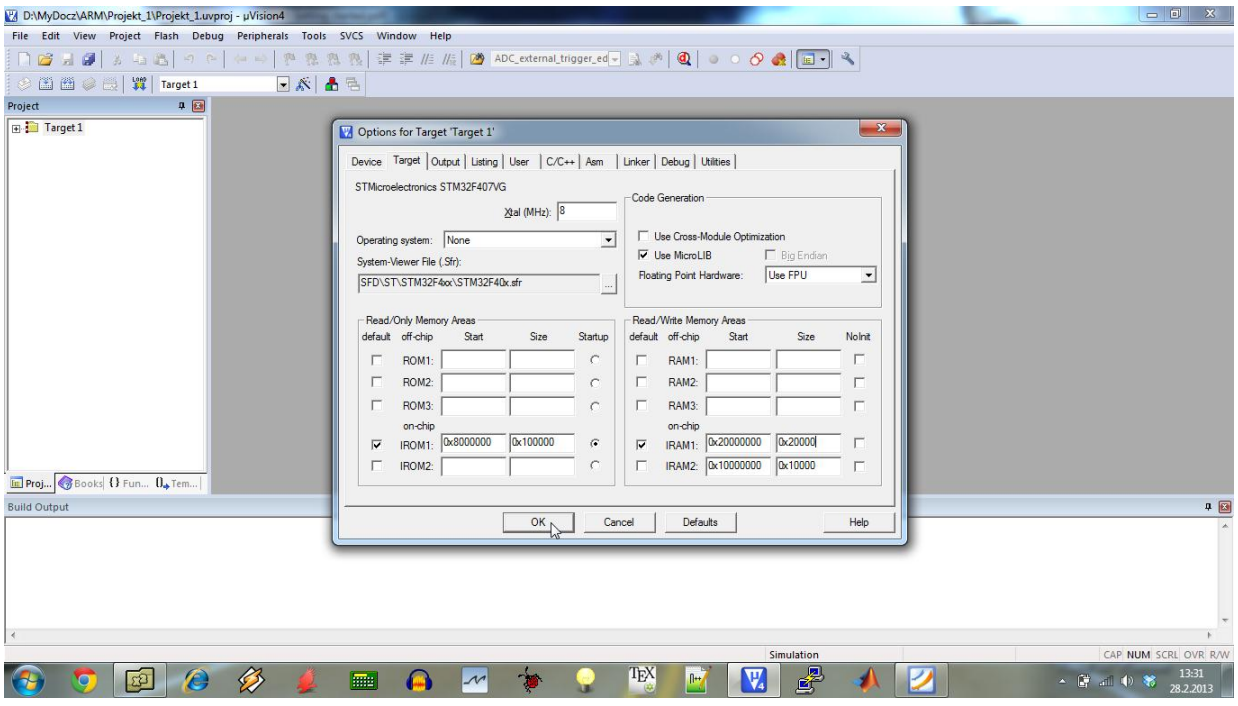
1. Najprej odpremo programsko okolje KEIL uVision.
2. Naredimo nov projekt s klikom na Project → New uVision Project ...

   in ga shranimo na primerno mesto na disku.
3. Odpre se nam naslednje okno (slika 1). Izberemo mikrokrmilnik STMicroelectronics → STM32F407VG.
4. Ob pojavnem oknu kliknemo "Yes". S tem dodamo zbirniško datoteko `startup_stm32f4xx.s`, ki poskrbi za zagon celotnega programa za mikrokrmilnik.
5. Na spletni strani `www.stromar.si` gremo pod Zapiski, Delavnice in prenesemo .zip datoteko s knjižnjicami. Te datoteke razširimo v isto mapo, v katero smo prej shranili projekt. Te knjižnice so dostopne tudi na uradni strani proizvajalca `www.st.com`.
6. Na levi strani pod "Project" desno-kliknemo na "Target1" in izberemo "Options for Target 'Target1' ..." Odpre se nam okno (slika 2).
7. Izberemo zavihek "Target" in preverimo frekvenco kristala (Xtal(MHz): 8), lokacije ROM pomnilnika (IROM1: 0x8000000, 0x100000) ter lokacijo RAM pomnilnika (IRAM1: 0x20000000, 0x20000).
8. Izberemo zavihek "C/C++" in kliknemo na gumb poleg polja "Include Paths". Dodamo poti `.\inc` in `.\STM32F4xx_StdPeriph_Driver\inc` (Slika 3).
9. Izberemo zavihek "Debug" in na desni iz izbirnega menija izberemo "ST-Link Debugger" ter kliknemo na krogec zraven "Use" pri tem izbirnem meniju. Nato kliknemo "Settings", nastavimo Port "SW" in kliknemo "OK" (Slika 4).

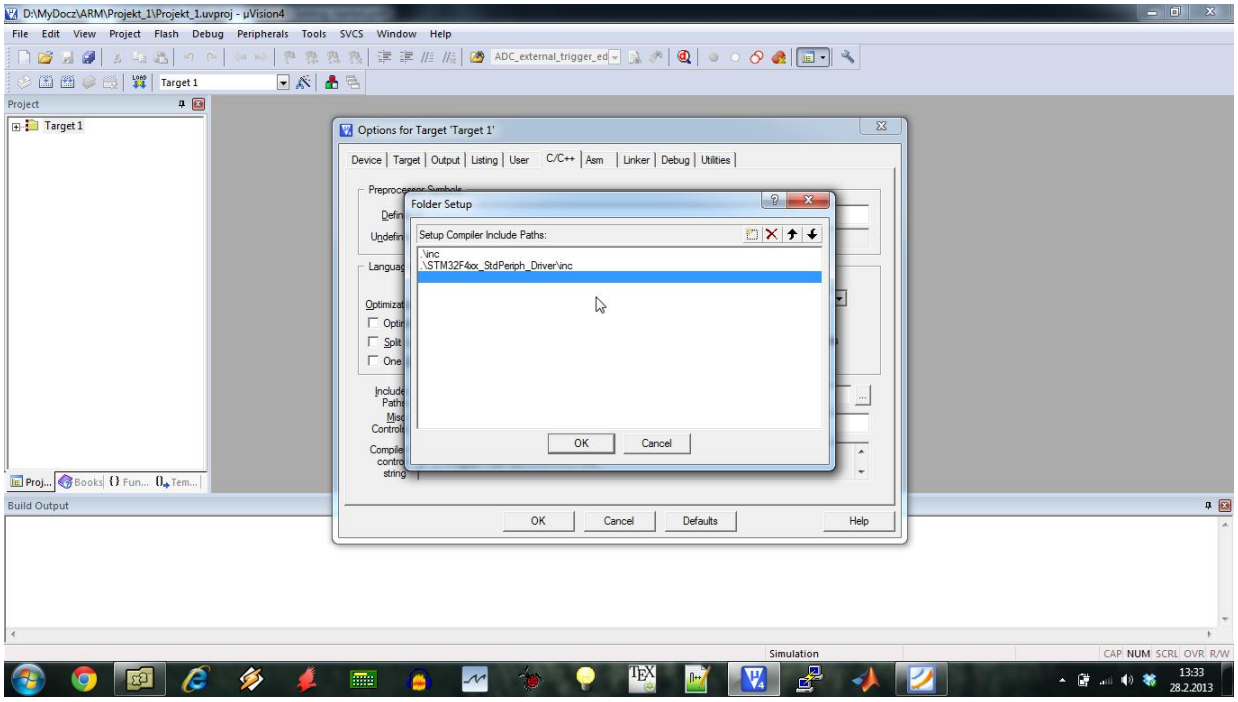
*Različica 0.1a (neprečiščeno besedilo).



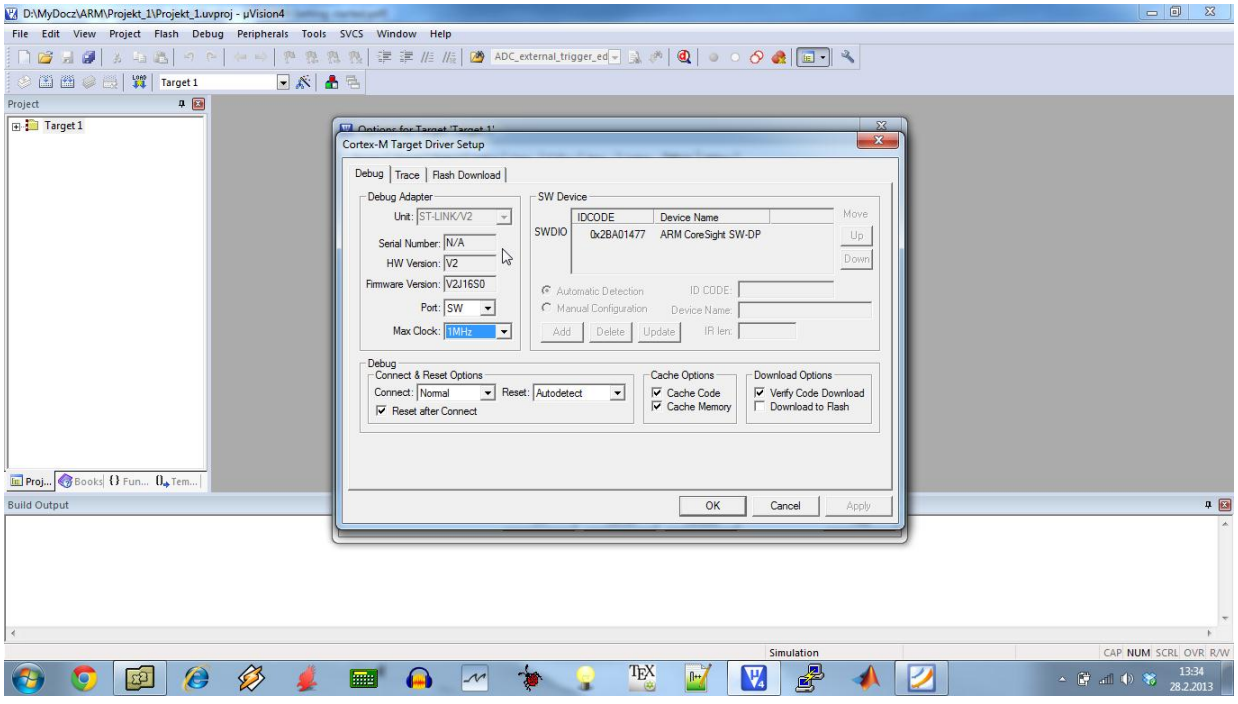
Slika 1: Izberemo ciljni mikrokrmilnik



Slika 2: Nastavitve programatorja 1



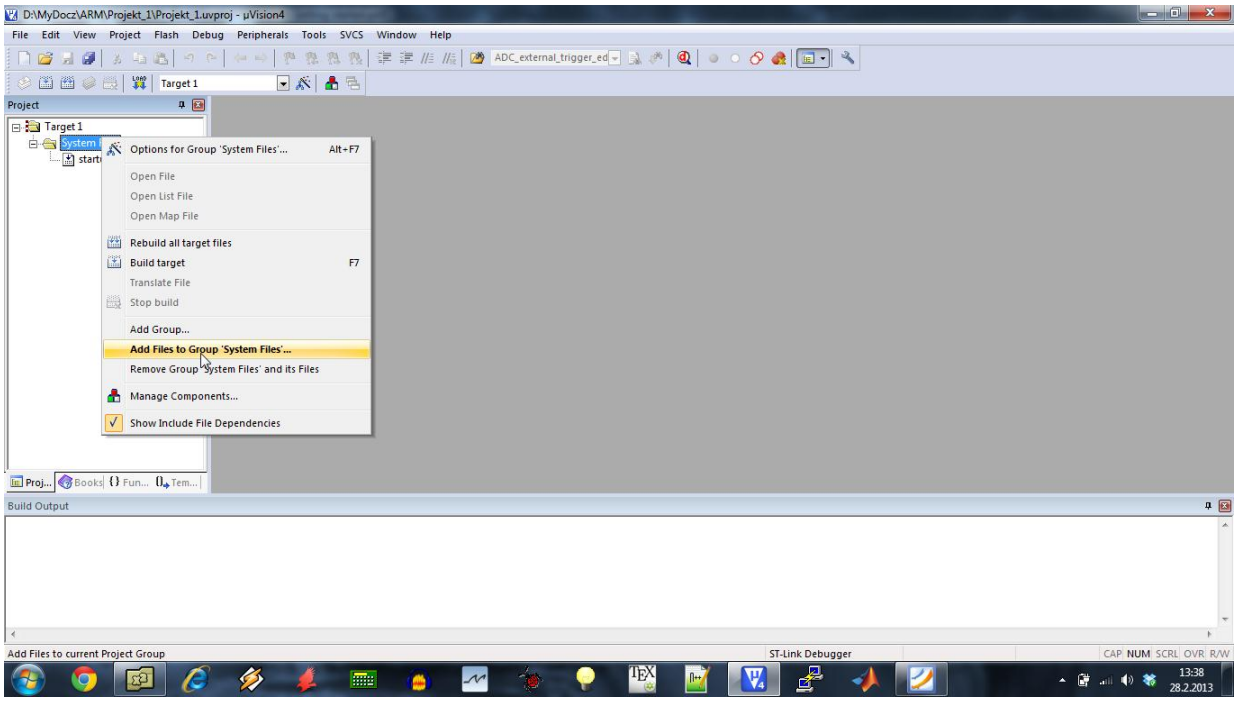
Slika 3: Nastavitve programatorja 2



Slika 4: Nastavitve programatorja 3

10. Izberemo zavihek “Utilities” in iz prvega izbirnega menija izberemo “ST-Link Debugger”. Kliknemo “OK” in s tem zapremo pojavno okno.

11. V projekt vključimo knjižnjice iz mape .\inc (slika 5). To naredimo z desnim klikom na “System Files” in ukazom “Add Files to Group”. Vključimo datoteki `stm32f4xx_conf.h` in `system_stm32f4xx.c`.



Slika 5: Vključevanje zunanjih knjižnjic

12. Odpremo nov dokument in ga shranimo kot `main.c`. Nato to datoteko dodamo pod “Target1” s pomočjo desnega klika na “System Files” in ukaza “Add Files to Group...”. Sedaj imamo pred seboj vzpostavljeno okolje.

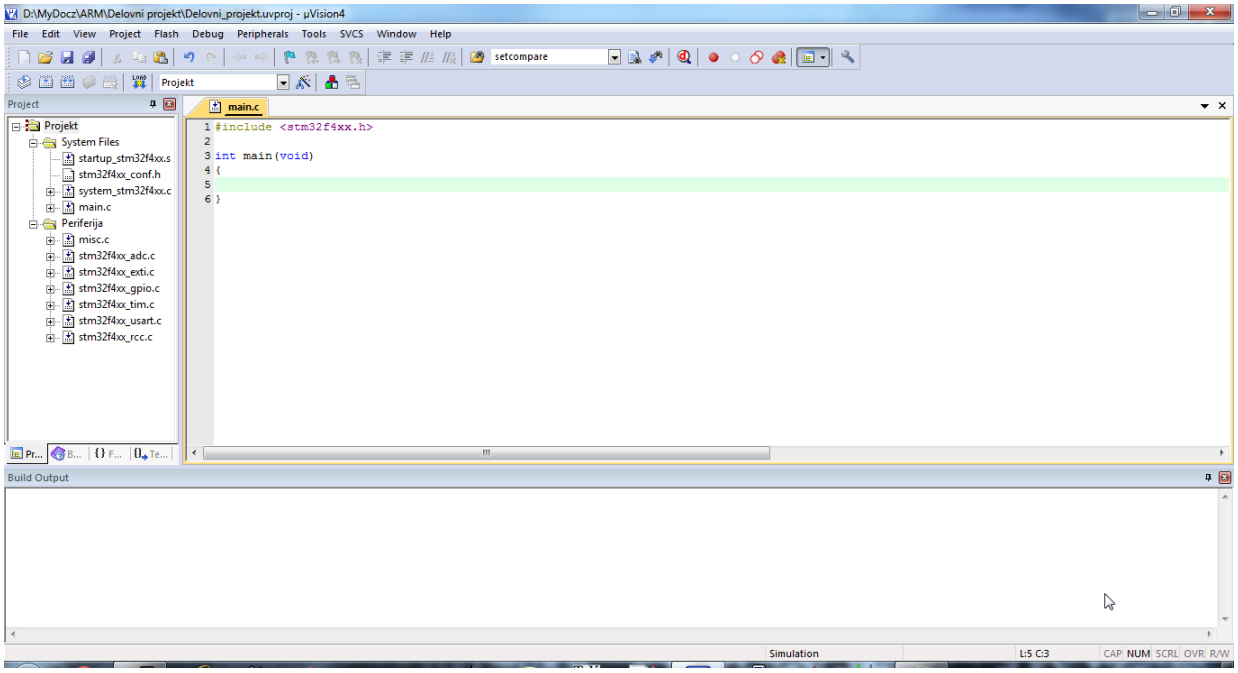
13. Na vrhu `main.c` datoteke dodamo vrstico `#include <stm32f4xx.h>`. V tej datoteki so vključitve za periferijo.

2 LED Utripalnik

2.1 Vključitev dodatnih funkcij

Sedaj ko imamo vzpostavljeno delovno okolje, lahko napišemo naš prvi program. Kot narekuje naslov bomo naredil LED utripalnik. Najprej moramo dodati knjižnjice za delo s periferijo. Iz mape \STM32F4xx_StdPeriph_Driver\src dodamo naslednje knjižnjice (slika 6):

- stm32f4xx_rcc.c - služi za vklapljanje ure posamezni periferiji. Šele z vklopom ure dotično periferijo pravzaprav vklopimo. V našem primeru bomo potrebovali periferijo GPIO, kamor imamo priklopljene svetleče diode.



Slika 6: Pripravljeno delovno okolje 2

- stm32f4xx_gpio.c - služi za delo z vsemi vhodno/izhodnimi pini (GPIO = General Purpose Input/Output). V tej knjižnici so spisane funkcije za inicializacijo pinov in za pisanje ter branje iz pinov.

Knjižnici smo dodali, programu pa moramo še povedati, da ju želimo vključiti. Odprimo datoteko `stm32f4xx_conf.h` z desnim klikom na njo in "Open Document...". Najdemo željeni knjižnici in ju odkomentirajmo. Vsako knjižnico, ki jo potrebujemo, dodamo na tak način. Ob prvi uporabi Keil uVision, moramo za vključitev standrdnih knjižnjic v datoteki `stm32f4xx.h` odkomentirati vrstico 90 (`#define USE_STDPERIPH_DRIVER`).

Ker pa je to uvodno predavanje, smo vam vnaprej pripravili knjižnjico za inicializacijo in prižganje ter ugašanje svetlečih diod. Knjižnica se nahaja v isti stisnjeni datoteki, katero ste že prenesli iz interneta. Dodajmo izvorno datoteko `ledfunction.c` in "header file" `ledfunction.h` v mapo, kjer imamo projekt. Nato .c datoteko dodamo pod "System Files" na enak način, kot smo prej vključili ostale knjižnice.

Sedaj moramo našemu programu povedati, da želimo vključiti zunanjo knjižnico. Na vrhu programa main.c dodamo vrstico, v kateri vključimo željeno zunanjo knjižnico (`#include "ledfunction.h"`).

Da bo prevajalnik prebral in upošteval vse vključitve znotraj datotek, moramo program prevesti. To naredimo s pritiskom na gumb **Build** ali tipko F7. Sedaj imamo avtomatsko vključeno še datoteko ledfunction.h. V tej datoteki so definirani vsi prototipi funkcij in vse bližnjice (makroji). Definirane imamo naslednje funkcije:

- `LEDInit()` - nam inicializira izhode, kamor so priključene diode.
- `setLED()` - nam postavi port, kamor je priključena dioda (LED1, LED2, LED3 ali LED4) na visok nivo. Funkcija kot argument sprejme LED1, LED2, LED3 ali LED4, glede na željeno

prižgano LED.

- `clrLED()` - nam postavi port, kamor je priključena dioda (LED1, LED2, LED3 ali LED4) na nizek nivo. Funkcija sprejme enake argumente kot prej.
- `delay()` - preprosta zakasnitvena funkcija, ki traja približno enako dolgo, kot je vnešena številka v milisekundah.
- `keyInit()` - funkcija, ki nam inicializira gumb (USER gumb na ploščici). Deluje s pozitivno logiko.

2.2 Sedaj pa zares!

Najbrž vsi poznate funkcije iz prototipnega sistema Š-ARM, ki smo ga uporabljali v prvem letniku pri predmetu programiranje 2. Funkcije, ki jih bomo uporabljali pri našem projektu so podobne tistim, ki ste jih uporabljali na sistemu Š-ARM.

Ves nadaljnji program se bo dogajal v datoteki main.c v funkciji `int main(void);`. Najprej inicializirajmo svetleče diode s funkcijo `LEDInit()`. Kot je že zgoraj navedeno, nam ta funkcija inicializira porte, kamor so priključene diode LD3, LD4, LD5 in LD6. Funkcija je tipa void in ne sprejema argumentov.

V programu potrebujemo neko neskončno zanko `while(1)`, v kateri bomo prižigali in ugašali diode z nekim zamikom `delay();`. Prižiganje diod izvedemo s funkcijo `setLED();`, ugašanje pa s `clrLED();`. Obe funkciji kot argument sprejmeta oznako svetleče diode, kateri želimo spreminjati stanje (LED1 - zelena, LED2 - rdeča, LED3 - oranžna ali LED4 - modra).

Na koncu bi morala naša koda zgledati nekako takole:

```
#include <stm32f4xx.h>
#include "ledfunction.h"

int main(void) {

LEDInit();

  while(1) {
  setLED(LED1);
  delay(1000);
  clrLED(LED1);
  delay(1000);
  }

}
```