Vaja 7 – PWM z Nucleo

1. Cilj naloge: S pomočjo programskega okolja STM32CubeIDE in HAL knjižnicami sprogramirajte mikroprocesor

tako, da boste generirali PWM signal in ga tudi krmilili na izbranem izhodu Nucleo. Za preizkus potrebujete

osciloskop.

2. Postopek inicializacije periferije.

a) Zaženite STM32CubeIDE in ustvarite nov STM32 projekt (pod zavihkom information Center). V zavihku

Board selector s pomočjo filtrov Type in MCU/MPU Series izberite ustrezno razvojno ploščo (v našem

primeru NUCLEO-L476RG), kliknite Next, projekt poimenujte vaja7_PWM_sk_X in kliknite Next, v oknu za

knjižnice izberite Add necessary library files as reference .. ter gumb Finish (na možnosti opcije za

prenastavitev periferije izberite Yes, izbrana naj bo tudi opcija perspektive za STM32CubeMX).

b) V levem Pinout oknu razširite nabor možnosti za Timers ter za časovnik TIM1. Clock Source nastavite kot

Internal Clock. Prvi kanal aktivirajte kot PWM Generation CH1. Kateri pin ste omogočili? _____PA8_____. Kaj

se izpiše poleg pina? _____«TIM1_CH1«_____.

c) V Clock Configuration spremenimo takt časovnika APB1 Timer Clocks (MHz) na 16 MHz.

d) V Oknu Configuration kliknemo za TIM1 Vrednost Prescaler (Parameter settings) v zavihku Counter

Settinngs določite tako, da bo časovnik delal s frekvenco 1 MHz. Koliko je vrednost Perscaler (namig;

delitelj) ? _____16_____.

e) Parameter Counter Period nastavimo na 100 in s tem še dodatno znižamo takt časovnika. Koliko znaša

sedaj? _____kHz.

f) V PWM Generation Chanel nastavite Pulse (16 bits value) na 50. Kaj pomeni ta parameter?

_____To pomeni 16-bitno vrednost, ki določa dolžino visokega nivoja signala (duty cycle) znotraj ene PWM
periode._____

Namig – največja vrednost je lahko 100 odstotkov (znak za odstotek v polja ne pišemo).

g) Na izbrani izhodni PWM pin priključite sondo osciloskopa (ne pozabite sondo ozemljiti na GND). Vključite

osciloskop in ustrezno nastavite merilno območje za x in y os.

h) Sedaj generirajte kodo tako, da enostavno kliknete ikono Save in po potrebi še enkrat potrdimo

generiranje kode.

3. Programiranje v IDE:

a) V skrajno levem oknu Project Explorer poiščemo main.c datoteko pod Core → Src → main.c (dvokliknite

na datoteko, odpre se tekstovni urejevalnik za main.c).

b) Poiščite prenastavljeni parameter ..Pulse (vrednosti je nastavljena na 50) v vaši kodi in prepišite ukaz,

ki ga je generiral CubeMX: _____ `sConfigOC.Pulse = 50;`_____.

c) V User code begin 2 inicializiramo časovnik za PWM z ukazom:

HAL_TIM_PWM_Start(&TIM_HANDLE_TYPE, TIM_CHANNEL_1);

Ime tega paramtera boste našli v vrstici pod Private variables ----. Del zgornje kode v rdečem torej

zamenjajte z imenom te ročice (TIM_HandleTypeDef XXXX;). Drugi parameter je izbana številka

kanala za generiranje PWM signala.

d) V User code begin 4 ne napišemo ničesar!!!

4. Naložitev kode (Run) in opazovanje delovanja:

a) Kodo preverite s tipko Build (ikona za kladivce - Build). Ko je preverjanje končano lahko preverimo, če smo

med pisnjem kode naredili kakšno napako sintakse, sicer se pod kodo v oknu Console izpiše 0 errors.

b) Priklopite STM32F4Discovery na vaš računalnik preko USB kabla.

c) S tipko Run (zelena ikona za play – Run as STM…) prenesete program na STM32F4.

Vaja 7 – PWM z Nucleo

d) V oknu Edit Configuration kliknemo OK. Nekaj sekund bo na ploščici STM izmenično utripala zelena in

rdeča LED, ko je program naložen, sveti LED rdeče.

e) Z osciloskopom preverite izhodni signal (uporabite sondo za priklop na ustrezni pin, ne pozabite na GND)

in delovanje posnemite s telefonom (MP4 datoteka). Prav tako fotografirajte signal.

f) Iz osciloskopa določite čas ene periode T = _____ in čas aktivnega pulza TON = _____ .

5. Dodatne nastavitve kode v programu IDE:

a) V kodi spremenite vrednost širine pulza na 25 %. Zapišite popravljeni ukaz v kodi:

_____ `sConfigOC.Pulse = 25;`_____ .

Ponovno naložite program in s telefonom posnemite signal na zaslonu osciloskopa.

g) V User code begin 1 deklariramo spremenljivko :

uint16_t dutyCycle = 10;

V User code begin 3 prepišemo naslednje ukaze:

htim1.Instance->CCR1 = dutyCycle;

dutyCycle+=10;

if(dutyCycle>90) dutyCycle=10;

HAL_Delay(1000);

Ponovno naložite program in s telefonom posnamite signal na zaslonu osciloskopa.

Zapišite kaj počnejo ukazi v 1.,2. in 3. vrstici (v user code begin 3):

1. S tem ukazom spremeniš **širino PWM impulza** na TIM1 Channel

2. Poveča vrednost spremenljivke `dutyCycle` za 10

3. Preveri, če je dutyCycle večji od 90, če je, potem nastavi spremenljivko na 10.

Komentar:
Stvari deluje kot mora, bilo je dosti enostavno.