Naloga 2 - GPIO

Vaja obsega inicializacijo vhodno-izhodnih priključkov (*GPIO pins*) in njihovo ustrezno konfiguracijo. V tem sklopu imate na voljo več nalog, ki so različno ocenjene (pri vsaki nalogi je zapisano, koliko je največjo število točk, ki jih lahko dobite). Odločitev, katero nalogo boste reševali, je prepuščena vam. Točke se ne seštevajo.

1 Naloge

1.1 Gumb in ena LED dioda [največ 1T]

Cilj naloge je, da se ob pritisku na gumb spremeni stanje LED diode na plošči (če jih je na voljo več, izberite eno). Vsakič, ko uporabnik pritisne na tipko (tudi če tipko drži dlje časa), se LED dioda vklopi oziroma izklopi samo enkrat.

Pomoč: *Neprestano branje priključka, na katerem je povezan gumb, vam da signal. Cilj te naloge je, da iz tega signala razberete, kdaj je bila tipka pritisnjena oziroma spuščena.*

Pomoč: *Primerjajte predhodno in trenutno stanje gumba!*

1.2 Gumb in več LED diod [največ 2T]

Ta naloga je nadgradnja prejšnje (potrebno je implementirati zaznavo pritiska gumba) in sicer je potrebno vklapljati in izklapljati dve LED diodi v podani sekvenci/zaporedju (recimo, da LED diodi poimenujemo a in b):

$$obe\ ugasnjeni \rightarrow a \rightarrow b \rightarrow ab \rightarrow obe\ ugasnjeni$$

Za tem se vzorec ponovi. Za vklop in izklop diode MORATE OBVEZNO uporabiti register *BSRR*.

Za razmisliti: *Razmislite, kako bi prižiganje LED diod rešili brez uporabe switch/if stavkov za (namig: array).*

1.3 7-segmentni zaslon [največ 3T]

V tej nalogi je potrebno na razvojno ploščo priključiti 7-segmentni zaslon (podrobnejši opis najdete v poglavju 4). Shema, kako je potrebno ekran povezati na razvojno ploščo, je objavljena pri nalogi na sistemu eŠtudij. Podobno kot v prejšnji nalogi je potrebno implementirati zaznavo pritiska gumba in uporabiti register BSRR za vklop in izklop LED diod. Program naj ob kliku na gumb spreminja stanje zaslona po vzorcu (črke od *a-f* predstavljajo LED diode na ekranu):

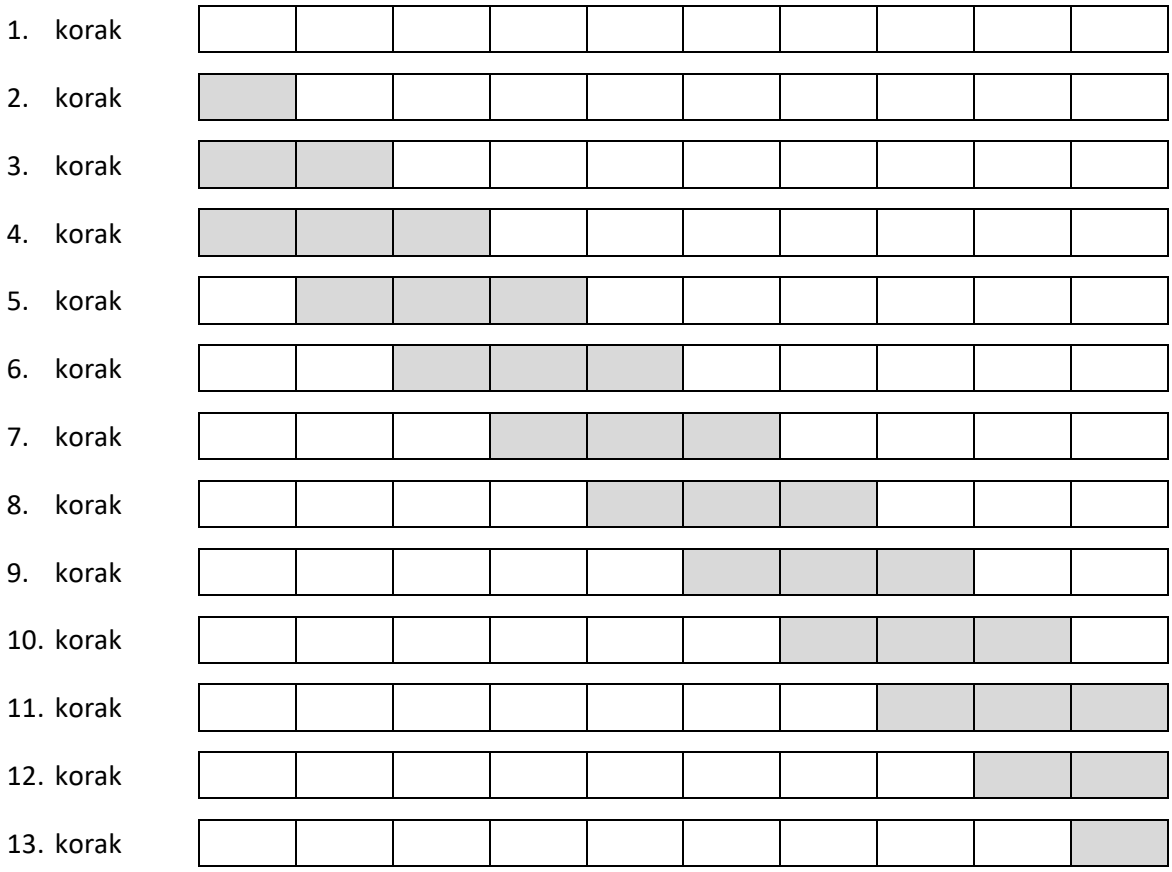
$$vse\ ugasnjene \rightarrow a \rightarrow g \rightarrow d \rightarrow f \rightarrow b \rightarrow e \rightarrow c \rightarrow vse\ vklopljenje \rightarrow vse\ ugasnjene$$

Za tem se vzorec ponovi.

Za razmisliti: *Razmislite, kako bi prižiganje LED diod rešili brez uporabe switch/if stavkov za (namig: array).*

1.4 Premikanje lučk [največ 5T]

Pri tej nalogi boste na razvojno ploščo priključili 10 LED diod (lahko tudi *t.i. LED bar* z 10 LED diodami), ki jih boste potem prižigali in ugašali po tri zaporedne diode skupaj. Cilj naloge je, da se trojčki LED diod spreminjajo s krajšim časovnim zamikom (*delay* – lahko implementirate z uporabo zank). Ob pritisku na gumb, se mora ta časovni zamik skrajšati (stanje lučk se spreminja hitreje). Implementirajte dve različni hitrosti (počasi in hitro) [tudi tukaj se cikel potem ponovi]. Podobno kot v prejšnji nalogi je potrebno implementirati zaznavo pritiska gumba in uporabiti register BSRR za vklop in izklop LED diod.



Vzorec se ponovi

Za razmisliti: *Razmislite, kako bi to nalogo rešili brez uporabe switch/if stavkov za prižiganje LED diod (namig: array).*

2 Dodatni pogoji

Pri implementaciji upoštevajte naslednje:

- **uporaba knjižnic ni dovoljena** (vsa koda mora biti *vaša*),
- **uporaba Cube generatorja kode ni dovoljena,**
- obvezna ločitev programske kode glede na funkcionalnost v različne datoteke (*Primer: koda za inicializacijo, vklop in izklop LED diod naj bo v datoteki led.[c in h]).*

3 Nasveti

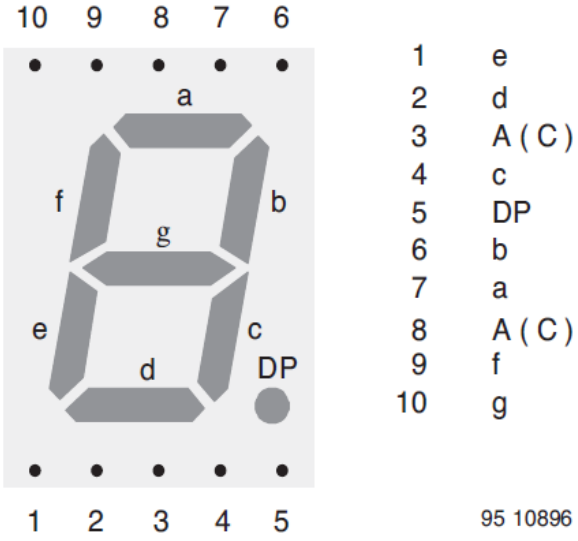
Kako začeti:

- v razvojnem okolju ustvarite projekt, ki bo omogočal zagon programa na razvojni plošči
- v specifikacijah razvojne plošče preverite, katera vrata (*port*) in priključke (*pin*) lahko uporabite za povezavo gumbov in LED diod (na primer *port F*)
- v specifikacijah procesorja preučite poglavje namenjeno vhodno/izhodnim povezavam (GPIO)[1],
- v specifikacijah procesorja poiščite, na katerih lokacijah se nahajajo registri za upravljanje vhodno/izhodnih povezav[2]
- zaradi varčevanja z energijo procesor STM32F429 privzeto izklopi uro periferijam. Zato morate pred uporabo priključka posameznim vratom omogočiti uro s pomočjo registra *RCC_AHB1ENR*
- ustvarite in implementirajte funkcije za LED diode in gumbe

4 7-segmentni zaslon

7-segmentni zaslon je sestavljen iz sedmih LED diod razporejenih v obliki pravokotnika, kot je prikazano na Slika **1**. Včasih je dodana še osma dioda, ki predstavlja piko.
LED diode imajo dva priključka – anodo (+) ter katodo (-). Tok lahko teče samo iz smeri anode proti katodi. Obstajata dve vrsti 7-segmentnih ekranov: s skupaj povezanimi katodami (CC) ali s skupaj povezanimi anodami (AC). Pri *CC* so vsi LED segmenti skupaj povezani na logično 0 (*ground*). Za vklop posameznega segmenta je potrebno na priključek preko upora dati logično vrednost 1 (*high*). Pri *AC* so vsi LED segmenti povezani na logično 1 in je potrebno za vklop posamezne LED diode preko upora posamezen priključek nastaviti na logično 0 (*low*).

[1] za procesor STM32F429 najdete to informacijo v poglavju 8 (*GPIO*) na strani 269
[2] za procesor STM32F429 najdete to informacijo v poglavju 2.3 (*Mermoy map*) na strani 64



Slika 1: 7-segmentni ekran

Na vajah se bo uporabljal ekran TDSR5150, ki je tudi prikazan na zgornji sliki. Spada v kategorijo AC 7-segmenith ekranov. Ima 10 priključkov, kjer je 8 namenjenih nadzoru LED diod, dva (priključka 3 in 8) pa sta za skupne povezave. Priključek 3 med seboj poveže priključke od 1 do 5, priključek 8 pa priključke od 6 do 10. AC povezava pomeni, da je potrebno priključka 3 in 8 povezati na priključek VCC na plošči (5V). Ostale priključke je potrebno z uporom povezati na priključke plošče (na primer vrata F, priključki 0-6). Za vklop posameznega segmenta je potrebno na priključek postaviti vrednost 0 (low).

5 Oddaja naloge

Pred oddajo naloge **projekt počistite** (*Clean project*) in ga združite v eno datoteko s formatom *zip*.