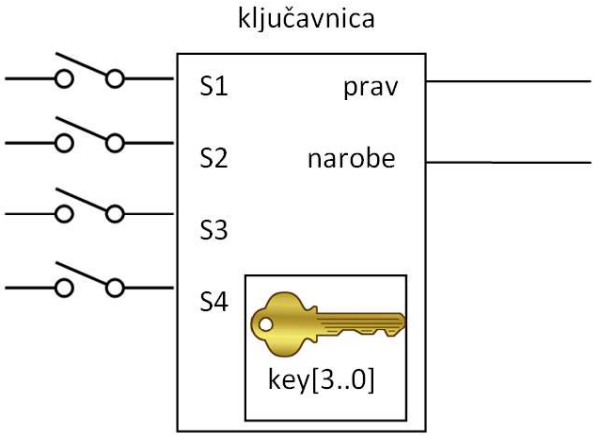
Vaja 2

Elektronska ključavnica

Izdelajte digitalno elektronsko ključavnico na razvojnem sistemu s programirljivim vezjem.

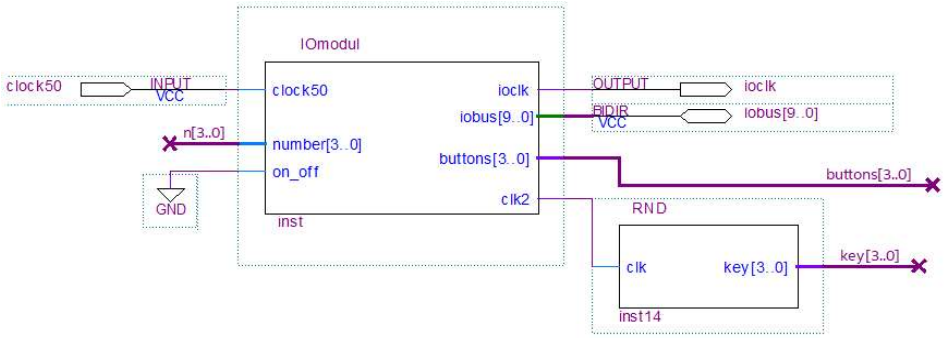


Slika 2.1: *Shema elektronske ključavnice.*

Naredite kombinacijsko vezje, ki bo v elektronski ključavnici preverjalo, ali je uporabnik pritisnil pravo kombinacijo tipk, ki se ujema s *ključem* ključavnice. *Ključ* vaše elektronske ključavnice je štiribitna beseda shranjena v vezju (na primer "0100"), ključavnico pa uporabnik odklene tako, da pritisne kombinacijo tipk **S1-S4**, ki se ujema s ključem.

V primeru ključa **key[3..0]**="0100" mora uporabnik za odklepanje pritisniti samo tipko **S2**. Da bo naloga bolj zanimiva, je ključavnica narejena tako, da vsake 4 sekunde zamenja ključ po zaporedju, ki ga ne poznate.

2.1 Kaj imate že na razpolago?



Slika 2.2: *Začetna shema z vmesnikom in generatorjem ključev.*

V predlogi projekta iz e-učilnice je glavna shema z dvema komponentama: vhodno-izhodnim vmesnikom *IOmodul* in generatorjem naključnih števil *RND*. Ključ je v shemi viden kot štiribitni signal z oznako **key[3..0]**, mi pa smo že poskrbeli, da se vrednost ključa zamenja vsake štiri sekunde. Zaporedje ključev je takšno, da se ključ 0000 ne more nikoli pojaviti. Kot v prejšnji vaji, imate na razpolago signal iz štirih tipk, z oznako **buttons[3..0]**. Na svetlečih diodah **led4-led7** se prikazuje vrednost trenutnega ključa, da lahko preverite pravilnost delovanja vezja. Pravi uporabnik elektronske ključavnice seveda ključa nikoli ne bi videl!

2.2 Kaj morate narediti vi?

Z logičnimi vrati naredite vezje, ki bo preverilo, ali se kombinacija pritisnjenih tipk ujema s ključem in prikazalo rezultat na svetlečih diodah.

Če se kombinacija tipk ujema s ključem, naj se signal z oznako **prav** postavi na logično 1, sicer pa naj se na 1 postavi signal **narobe**. Zapišite logično funkcijo obeh signalov v odvisnosti od stanja tipk **S1**, **S2**, **S3** in **S4** ter bitov ključa **K0**, **K1**, **K2**, **K3**. Uporabite skrajšan zapis: **key[0]**=**K0**, **key[1]**=**K1**, **buttons[0]**=**S1**, itd.

prav =

narobe =

Narišite shemo vezja z izbranimi logičnimi vrati. Signal **prav** naj bo vezan na **led0** in **led1**, signal **narobe** pa na **led2** in **led3**.

2.3 Možnosti izvedbe

Da bo naloga bolj zanimiva upoštevajte dodatne pogoje pri načrtovanju logične funkcije in vezja.

a) Uporabite samo operacije ekskluzivni ali (XOR), logični in (AND) in negacijo (NOT).

b) Uporabite samo ekskluzivni ali (XOR), logični ali (OR) in negacijo (NOT).

c) Uporabite samo logični ali (OR), logični in (AND) in negacijo (NOT).

Vezje prevedite, ga naložite na razvojni sistem in preizkusite delovanje. Pazite na to, da bo vaše vezje ustrezalo logični funkciji, ki ste jo zapisali zgoraj – samo v tem primeru bo naloga pravilno rešena!

Razmisli

- Koliko je vseh kombinacij, ki opisujejo delovanje vezja za preverjanje ključa?

- Na kakšen način lahko pokažemo, da bo vezje pravilno delovalo pri vseh kombinacijah?

- Koliko in katere gradnike zasede vezje znotraj FPGA?