# Vaja 4

# Generator ključev

Naredili ste že preprosto elektronsko ključavnico, pri kateri je moral uporabnik uganiti skriti štiribitni ključ s pritiskom na ustrezno kombinacijo vhodnih tipk. Ključ se je zamenjal vsakih nekaj sekund. Da je bila rešitev naloge lažja, smo vam takrat že pripravili generator ključev.

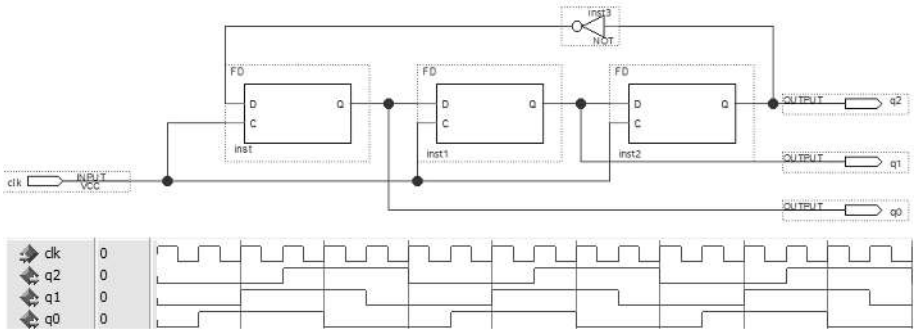
Pri tej vaji boste naredili sekvenčno vezje, ki deluje kot generator naključnih števil in ga vključili v shemo iz prejšnje vaje. Generator ključev naj nadomesti blok RND, ki določa izhodno kombinacijo na vodilu **key[3..0]**.

## 4.1 Sekvenčno vezje

Sekvenčno vezje sestavljajo pomnilni elementi flip-flopi, ki za en cikel ure shranijo stanje na izhodu. Generator ključev spada med sekvenčna vezja s povratno zanko, kjer je izhod vezja preko logike povezan na vhod.

Pri generatorju naključnih vrednosti se izhodi ne spreminjajo v pravilnem zaporedju. Najbolj preprosti so digitalni generatorji naključnih vrednosti s pomikalnim registrom. Sestavljeni so iz zaporedno povezanih flip flopov in logike, ki določa vhod prvega flip-flopa. Takšni generatorji niso popolnoma naključni, ker se vrednosti na izhodih flip-flopov po določenem številu ciklov ponavljajo, zato jim pravimo generatorji psevdonaključnih vrednosti.

Slika 4.1 prikazuje vezje iz treh D flip-flopov in negatorja v povratni vezavi. Flip-flopi na shemi imajo skupno uro (signal clk) in delujejo tako, da spreminjajo vrednost na izhodu le ob prehodu ure iz 0 na 1. Izhod posameznega flip-flopa je zakasnjen glede na podatkovni vhod D za en cikel ure.




**Slika 4.1:** *Shema in simulacija 3-bitnega pomikalnega registra s povratno zanko.*

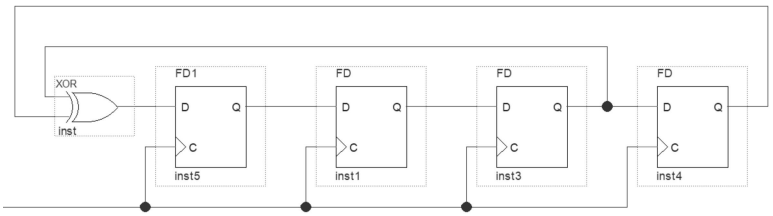
Vsi izhodi flip-flopov naj bodo na začetku simulacije postavljeni na 0. Zaradi negatorja, je vhod prvega flip-flopa na 1, zato se ob naslednjem ciklu postavi **q0** na 1. Naslednji cikel ure se ta vrednost prenese na izhod flip-flopa **q1**, nato pa še na **q2**. Zaradi negatorja, bo sedaj na vhodu prvega flip-flopa logična 0, ki se bo pomikala, kot prikazuje simulacija. Spreminjanje stanj flip-flopov predstavimo s tabelo:

| cikel | q0 | q1 | q2 |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 |
| 2 | 1 | 1 | 0 |
| 3 | 1 | 1 | 1 |
| 4 | 0 | 1 | 1 |
| 5 | 0 | 0 | 1 |
| 6 | 0 | 0 | 0 |

Stanja se po šestih ciklih začnejo ponavljati. Katere kombinacije manjkajo? Če bi na naredili podobno vezje iz štirih flip-flopov, bi videli, da se ponavlja 8 od 16 mogočih kombinacij. Za vezje, ki generira daljše in bolj naključno zaporedje potrebujemo namesto negatorja drugačno logično funkcijo v povratni vezavi.

## 4.2 Štiri-bitni psevdonaključni generator

Shema 4-bitnega generatorja z maksimalnim zaporedjem je na sliki 4.2. Vezje je sestavljeno iz štirih D flip-flopov in logičnih vrat XOR. Vsi flip-flopi so vezani na isto uro, njihovi izhodi pa predstavljajo stanje vezja.




**Slika 4.2:** *Pomikalni register s povratno zanko za generiranje naključnih vrednosti.*

Prvi flip-flop na shemi ima nekoliko drugačno oznako (FD1), ker je njegovo začetno stanje 1, ostali flip-flopi (FD) pa imajo začetno stanje 0. Preden začnete z reševanjem naloge naredite analizo vezja na papirju. Začetno stanje vezja je kombinacija 1000 (skupaj napisani izhodi vseh flip-flopov). Ob naslednjem ciklu ure se vrednost iz prvega prenese v drugega, iz drugega v tretjega, ... prvi pa dobi vrednost, ki jo določa funkcija XOR. Napiši stanje vezja ob zaporednih ciklih ure:

| cikel | stanje | cikel | stanje |
|---|---|---|---|
| 0 | 1 0 0 0 | 10 | |
| 1 | | 11 | |
| 2 | | 12 | |
| 3 | | 13 | |
| 4 | | 14 | |
| 5 | | 15 | |
| 6 | | 16 | |
| 7 | | 17 | |
| 8 | | 18 | |
| 9 | | 19 | |

## 4.3 Kaj morate narediti vi?

- Naredite analizo generatorja naključnih vrednosti na papirju.
- V projektu iz prejšnje vaje izbrišite gradnik RND z glavne sheme.
- Narišite shemo izbranega vezja z gradniki FD oz. FD1. Dodajte vhodne in izhodne priključke in v orodju **Quartus** generirajte HDL opis (*File, Create, Create HDL Design*), ki ga potrebujete za simulacijo. Po navodilih izvedite simulacijo vezja.
- Novo shemo pretvorite v simbol (*File, Create, Create Symbol Files*), dokončajte glavno shemo in preizkusite delovanje na razvojnem sistemu.

Ugotovi po kakšnem številu ciklov se začnejo kombinacije ponavljati.
Zakaj potrebujemo flip-flop, ki ima začetno stanje 1 ?

## 4.4 Možnosti izvedbe

a) Naredite naključni generator s funkcijo XOR, kot prikazuje slika 4.2.

b) Naredite naključni generator s štirimi flip-flopi FD in negatorjem v povratni zanki.

c) Naredite naključni generator s štirimi flip-flopi FD in funkcijo XNOR v povratni zanki.

### Razmisli

- Kaj se zgodi, če se vezje zaradi kakšne motnje znajde v stanju, ki ni med kombinacijami v tabeli?
- Kako bi naredil generator z daljšim psevdonaključnim zaporedjem, ki imel še vedno le 4-bite na izhodu?