# ORT2

# FPGA

# Projekat

# januar 2018 / 2019

# Studenti:

# Anja Marković 0420/2017

# Vukašin Drašković 0455/2017

# Marko Divjak 0084/2017

# Tematika projekta

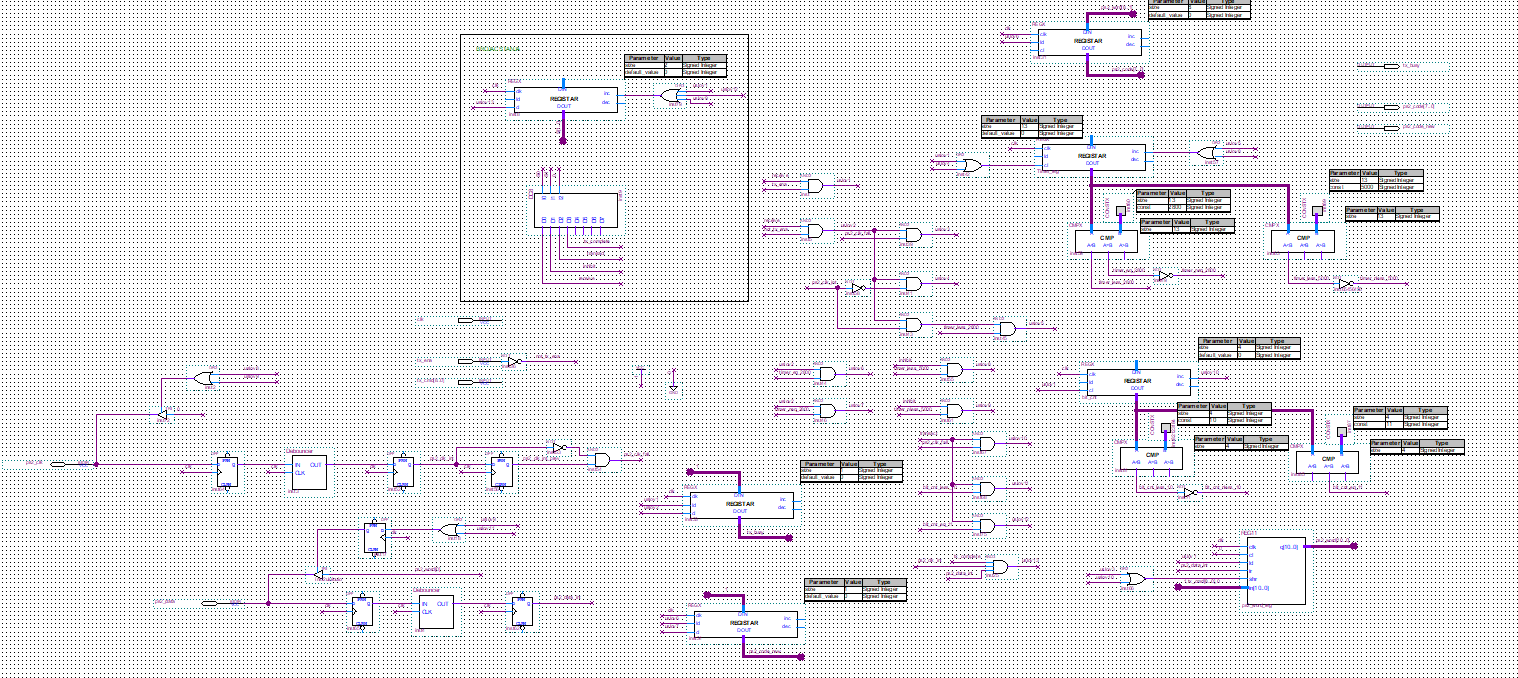
Ovaj projekat predstavlja implementaciju igrice kojoj je kao ideja poslužila kompjuterska igrica „Icy tower“. Igru igraju dva igrača. Cilj je da igrač skoči sa jedne pločice na narednu i da ide što je moguće više, bez pada ispod poslednje pločice.

Igrači započinju igru na prizemlju tornja. Pločice kule su nepokrente. Kada oba igrača skoče na prvu pločicu, one počinju da se kreću nadole. Za penjanje uz toranj, igrači moraju da skoče sa pločice na pločicu. Igrači mogu da prođu preko pločice odozdo, ali slete na njih kada padaju odozgo. Kada igrač padne na pločicu, može da se kreće po njenoj površini pomeranjem miša, ali ne može da ide dalje od zidova koji se nalaze na levoj i desnoj strani ekrana. Ukoliko jedan igrač padne, povećavaju se poeni drugog igrača, a igrica počinje ispočetka. Cilj je da igrač sakupi više poena od protivničkog igrača. S obzirom na to da je toranj beskonačne visine, nemoguće je doći do vrha.

# 2. Opis korišćenih protokola

## PS/2 protokol za komunikaciju sa mišem

### PS/2 Transceiver modul



Modul služi za slanje komandi mišu. Komande koje se šalju mišu imaju osam bita. Na ulazu ovog modula se nalazi bus *tx\_cmd[8..0]* koji sadrzi osmobitnu komandu i deveti bit koji je bit parnosti. Ulazi su još i *tx\_ena* koji je jednak jedinici kada treba da se komanda *tx\_cmd[8..0]* pošalje mišu i signal clk koji predstavlja signal takta uređaja.

Modul ima četiri stanja: *receive, inhibit, transact* i *tx\_complete*. Transceiver se nalazi u stanju *receive* sve vreme dok ne dođe signal *tx\_ena*. Ukoliko je aktivan signal *tx\_ena*, aktivira se signal *tx\_busy* koji signalizira da je modul trenutno u fazi slanja komande i ne može da šalje niti čita nove podatke. Registar u kome se čuvaju podaci se briše i brojač koji broji primljene bite se briše. Nakon toga se prelazi u stanje *inhibit*.

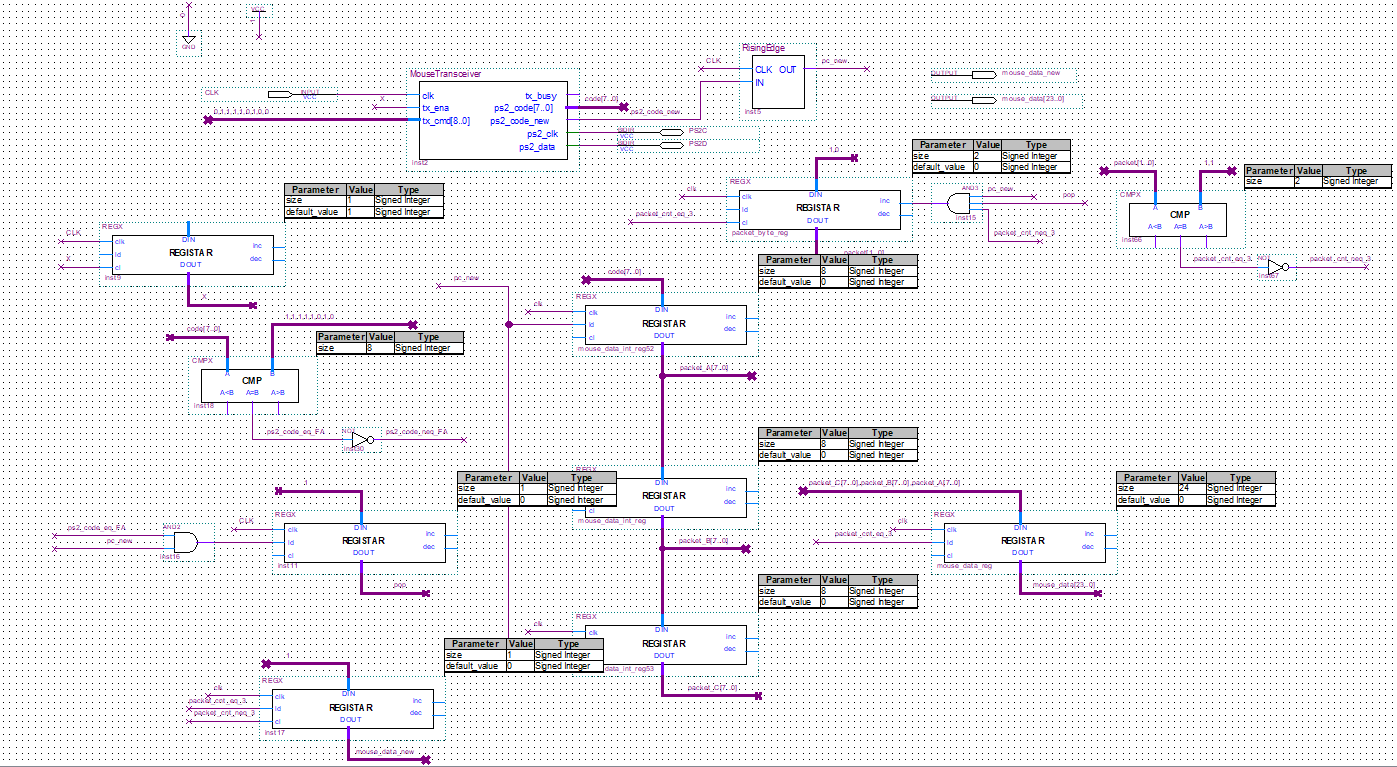
Ukoliko nije aktivan signal *tx\_ena* signal *tx\_busy* je neaktivan. Na svaki falling edge takta miša se učitava novi bit šiftovanjem registra podataka u desno. Ukoliko je takt miša jednak jedinici inkrementira se brojač. Kada taj brojač dođe do 2800 šalje se jedinica na signal *ps2\_code\_new* čime se signalizira da je dostupan novi paket koji je poslao miš. Ako je na liniji takta miša signal 1 u trajanju dužem od pola periode takta miša, onda je miš prestao da generiše signal takta što znači da nema više podataka koje treba pošalje. Polovina periode takta miša iznosi 55 µs, što je 2800 taktova uređaja.

U stanju *inhibit* se inkrementira registar *timer* koji broji do 5000. Da bi se poslala komanda mišu potrebno je da uređaj pošalje nulu u trajanju od 100 μs na liniju takta miša. Dok je *timer* manji od 5000 bafer propušta signal nula na takt miša. Kada brojač dođe do 5000 prošlo je 100 µs i prelazi se u stanje *transact*.

U stanju *transact* se šalje osmobitna komanda i bit parnosti na liniju *ps2\_data*. Bafer koji propušta nulu na *ps2\_clk* se zatvara i miš počinje da generiše signal takta. Na svaki falling edge signala takta miša registar komande se šiftuje u desno i najdesniji bit se propušta na liniju *ps2\_data*. U isto vreme se inkrementira brojač koji broji koliko je bitova poslato. Bafer kod linije *ps2\_data* je sve vreme otvoren i propušta najlakši bit registra u kome se nalazi komanda. Mišu se šalje 11 bita (9 bita komande, kao i start i stop bit koji su 0 i 1). Kada brojač dođe do 11 prelazi se u stanje *tx\_complete*.

U stanju *tx\_complete* se ostaje sve dok uređaj ne pošalje vrednost 1 i na liniji *ps2\_clk* i na liniji *ps2\_data* u isto vreme. Kada se to desi komanda je poslata, miš je odgovorio da je uspešno primljena i prelazi se u stanje *receive*.

### PS/2 Mouse Interface Modul



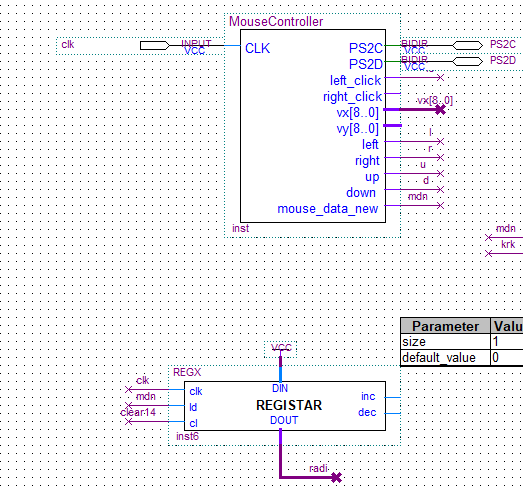
Kada se pokrene šema na pločici u registru *X* se nalazi vrednost 1. Ta vrednost signalizira da treba da se pošalje komanda mišu. Ova vrednost će biti vezana na ulaz *tx\_ena MouseTransceivera* i u istom taktu će se obrisati tako da će se samo u prvom taktu poslati komanda za slanje podataka.

Da bi miš počeo da šalje pakete potrebno je da mu se pošalje komanda 0xF4 koja je kodirana na ulazu *MouseTransceivera* zajedno sa bitom parnosti. U prvom taktu počeće da se šalje komanda mišu. Nakon što miš primi komandu odgovoriće sa paketom 0xFA.

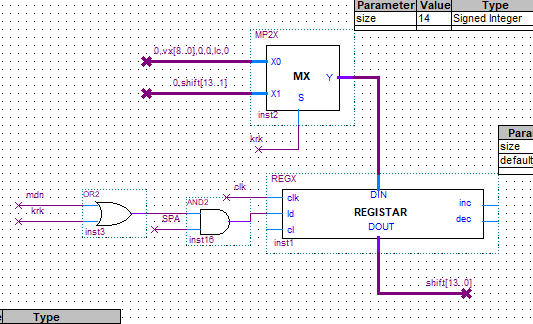
Kada miš prvi put odgovori paketom 0xFA učitaće se vrednost 1 u registrar *pop*. Od tog trenutka novi paketi će se učitavati u registre *packet\_A[7..0], packet\_B[7..0], packet\_C[7..0].* Pri prijemu svakog paketa inkrementiraće se vrednost registra *packet\_byte\_reg.* Kada ovaj brojač dostigne vrednost 3 znači da su primljena 3 paketa koji će se učitati u registar *mouse\_data\_reg*. U tom trenutku će se učitati vrednost 1 u registar *mouse\_data\_new* čiji je izlaz na outputu modula. U narednom taktu će vrednost poslednja 3 paketa biti na izlazu *mouse\_data[23..0],* a na izlazu *mouse\_data\_new* će biti vrednost 1.

## Komunikacija dve pločice – asinhrono slanje podataka

U semi s*aljem* u postoji modul *MouseController* koji šalje signale miša koje treba proslediti drugoj pločici. MouseController šalje i signal *mdn* koji govori da je stigao novi paket signala koji treba proslediti. Kada je on aktivan signal radi postaje 1 i označava da treba otpočeti slanje.



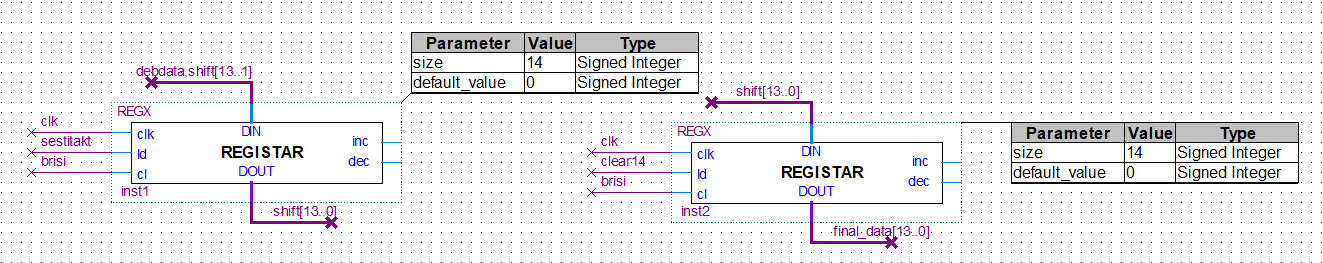
Signal *radi* se šalje drugoj pločici kao cap signal koji označava da treba otpočeti primanje bita. U ovaj registar se učitava ili novi paket ili vrednost registra šiftovana u desno za jedno mesto. Šiftovanje se radi kako bi se na output data uvek slao nulti bit shift registra. Sadržaj registra se pomera u desno samo kada je aktivan signal SPA.



Vrednost *klokic[4..0]* koja se čuva u registru predstavlja usporeni takt koji se realizuje brojanjem 16 taktova pločice i za to vreme drži vrednost bita *shift[0]* na outputu data kako bi druga pločica mogla pravilno da ga preuzme. Ovaj registar se briše kada signal *krk* označi da je izbrojano 16 taktova.

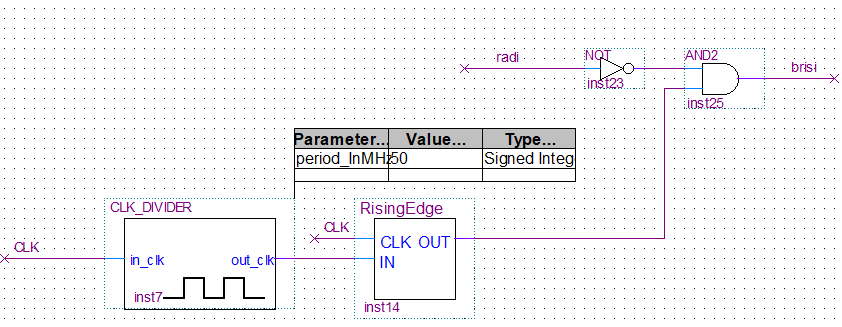
Vrednost *poslati[4..0]* koja se čuva u registru predstavlja broj bita koji su dosada poslati outputom data. Briše se kada se izbroji 14 poslatih bita. Tada se i signal *radi* koji označava da treba slati bite postavlja na 0 jer je slanje završeno.

U šemi *primam* se brojanje koliko je bita primljeno vrši na isti način kao i u šemi *saljem*.



Kada *klokic[4..0]* bude 6, šiftuje se dotadašnje stanje registra sa bitom debdata kao najvišim bitom, koji označava bit koji je pročitan sa ulaza koji se šalje sa druge pločice. Kada se izbroji 14 primljenih bitova registar se učita u drugi registar koji čuva vrednost *final\_data[13..0]* koji sada sadrži paket bita spreman za korišćenje na ovoj pločici.

Signal *brisi* je aktivan na svakih 100 ms kada se briše vrednost dotada primljenog paketa.



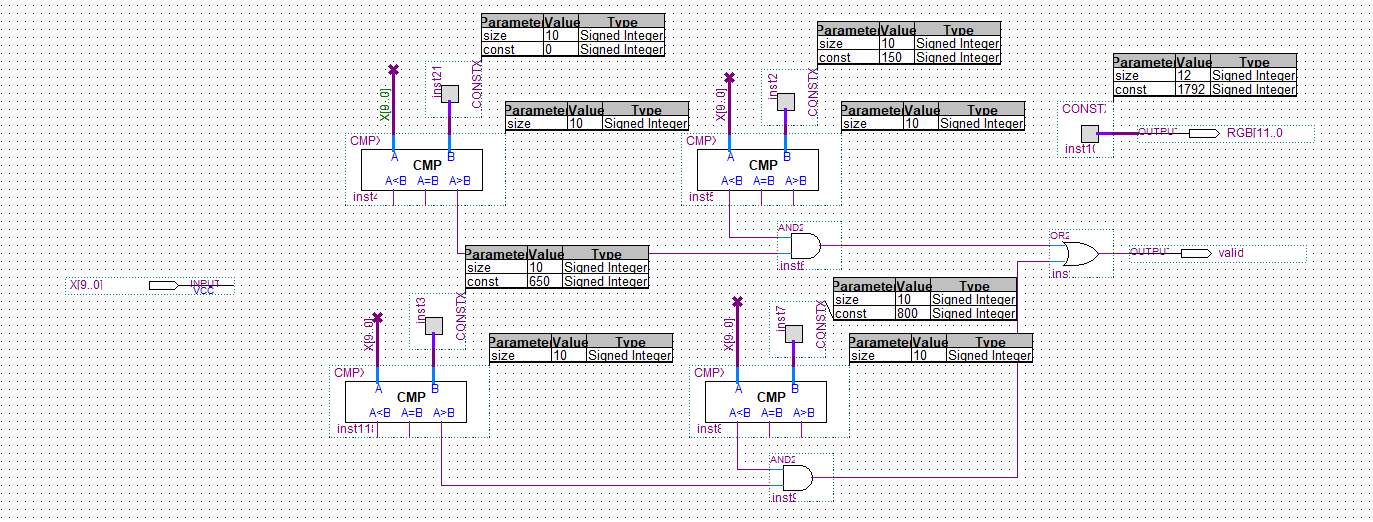
# 3. Sledeći koraci u razvoju i poboljšanju projekta

U cilju unapređenja i daljeg razvoja projekta mogu se uvesti sledeće promene i poboljšanja:

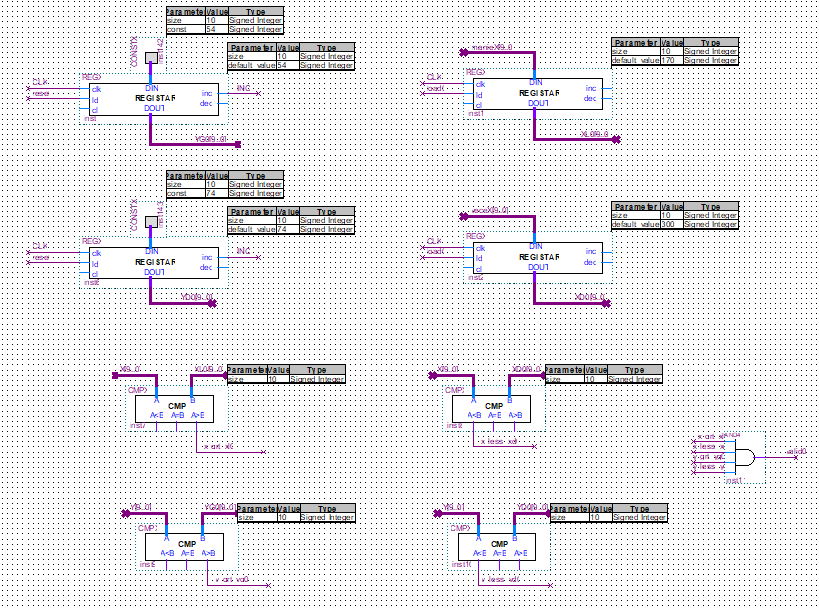
1. Iscrtavanje pokemona korišćenjem memorije
2. Poboljšanje funkcije generisanja nasumičnih brojeva u cilju ostvarivanja uniformnije raspodele slučajnih brojeva
3. Realizovanje jedinstvenog modula za iscrtavanje pokemona na osnovu ulaznih početnih koordinata
4. Otežavanje igre u vidu ubrzanja pločica na koje skaču pokemoni u zavisnosti od ukupnog broja pločica koje su preskočili
5. Uvođenje specijalnih bonus skokova koji mogu da odbace pokemona više
6. Poboljšanje grafike korišćenjem bolje slike za pozadinu
7. Uvođenje specijalnih efekata kao rotacija pokemona u skoku, bljuvanje vatre, mahanje krilima itd.

# 4. Realizovane šeme

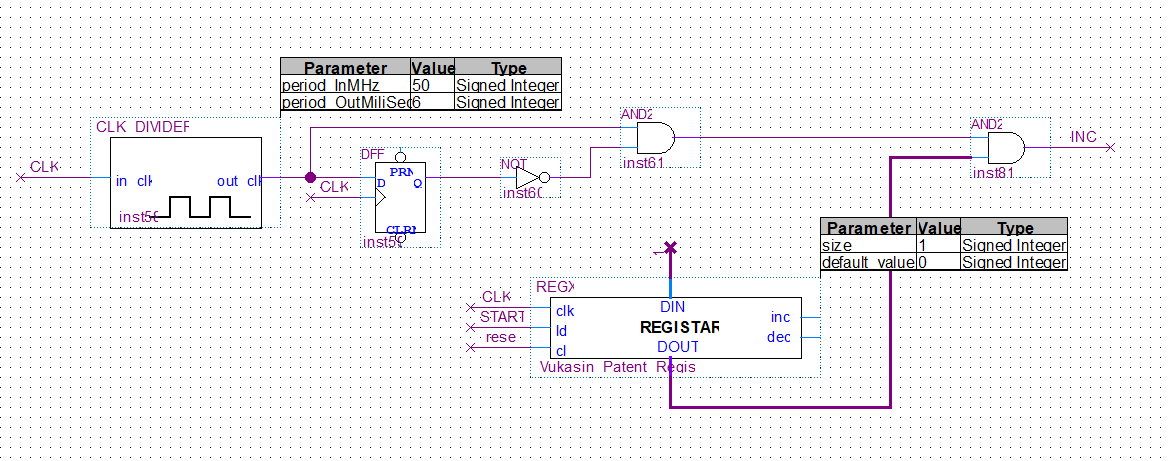
## Modul Zidovi

Zidovi su napravljeni uz pomoć pravougaonika, nacrtanih pomoću komparatora koji određuju granice datih pravougaonika na *x* i *y* osi. Kada je piksel u datom opsegu signal *valid* je aktivan i piksel se iscrtava zadatom bojom *RGB[11..0]*.

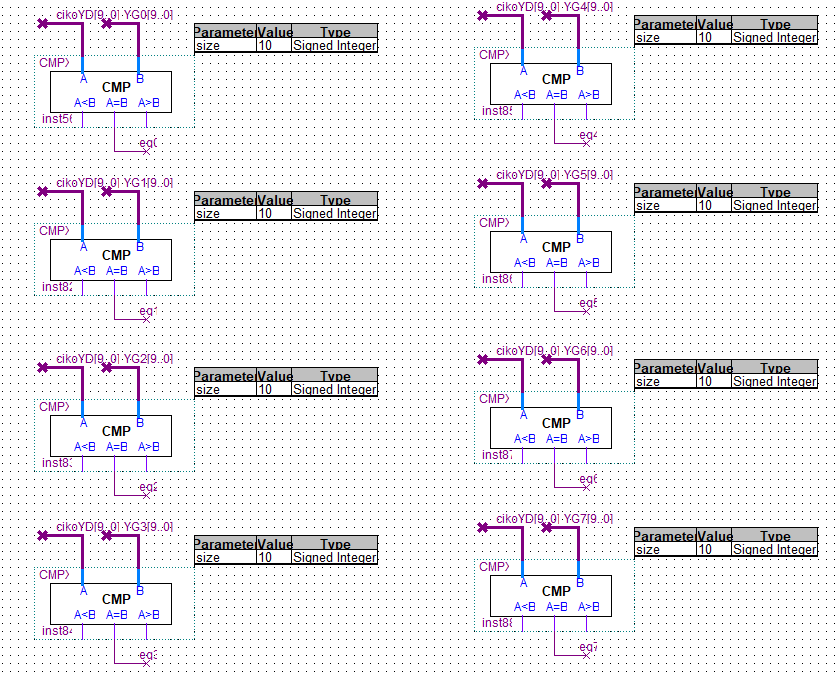
## Modul Pločice

Pločice su napravljene u obliku pravougaonika uz pomoć registara koji imaju početne vrednosti koordinata datih pločica, i komparatora koji određuju opsege pločica na *x* i *y* osi.

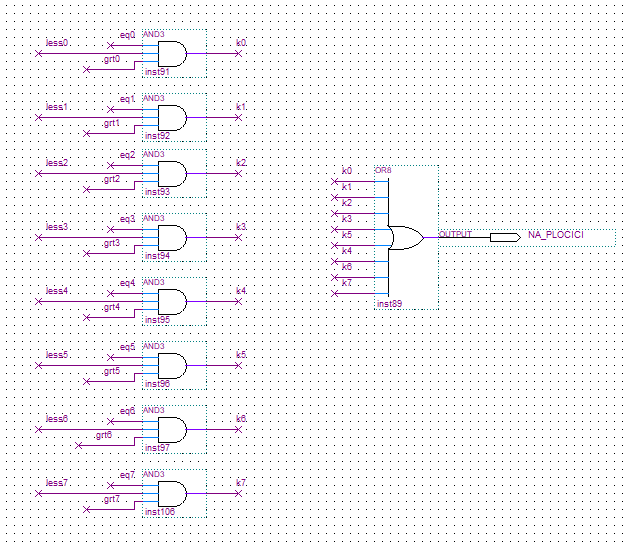
Pločice počinju da padaju kada oba igrača skoče. Tada se u registar *Vukasin\_Patent\_Registar* upisuje vrednost 1. Na svakih 6 ms generisaće se signal *INC*. Taj signal se dovodi na ulaze *inc* registara koji određuju *y* koordinate pločica. Inkrementiranjem njihovog sadržaja se dobija efekat padanja pločica.



Pomoću komparatora se proverava da li je neki od igrača na nekoj od pločica, tako što se upoređuju donje koordinate igrača sa gornjim koordinatama pločica.

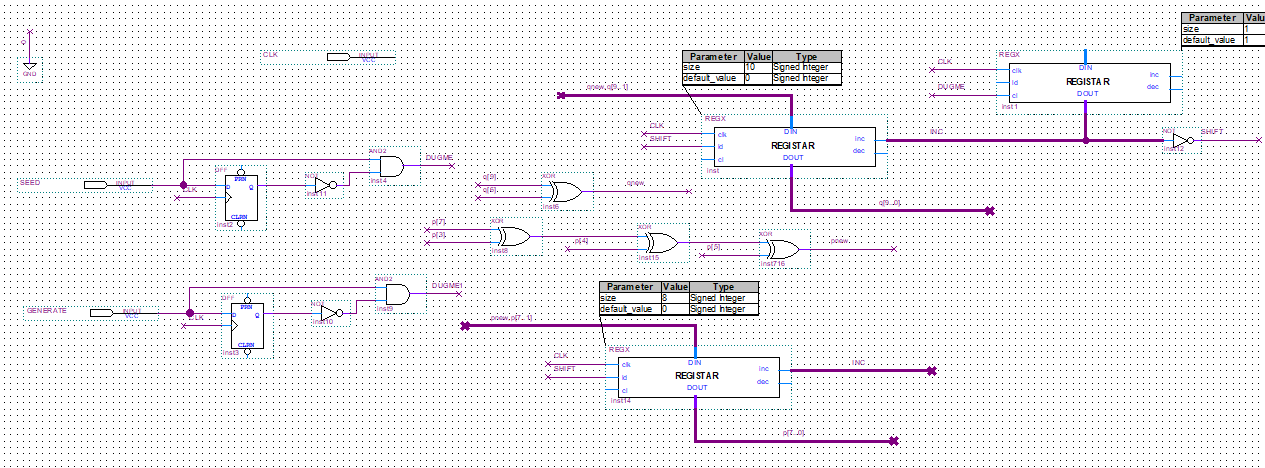


Analogno se porede leva i desna *x* koordinata sa odgovarajućim koordinatama svake od pločica. Ukoliko je proizvod bilo koja odgovarajuća tri signala aktivan na izlazu će biti aktivan signal *NA\_PLOCICI* tj. *NA\_PLOCICI\_B* (za drugog igrača).



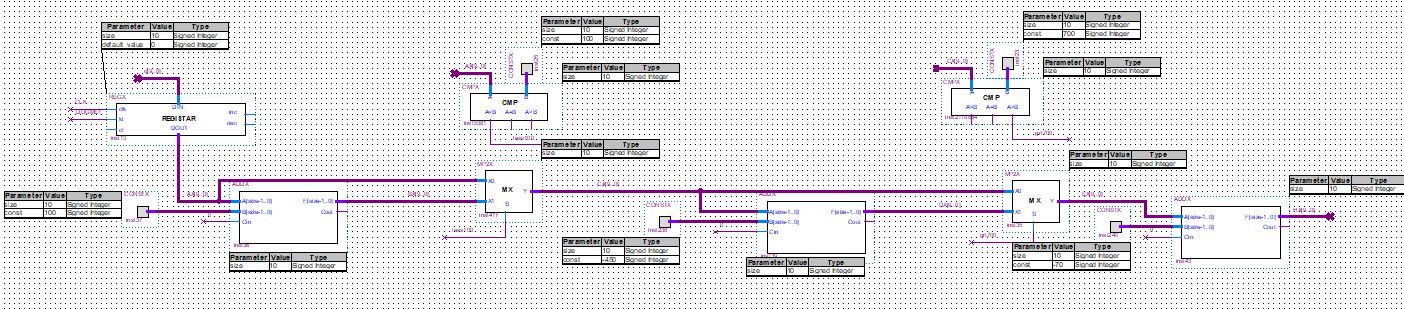
Kada pločice padaju i donja *y* koordinata neke pločice dodje do 630 kao njene leve i desne *x* koordinate se učitaju nasumični brojevi.

## Modul Random



U ovom modulu se nalaze dva registra koji služe za generisanje nasumičnih brojeva. Registri se inkrementiraju dok se ne pritisne dugme. Kada se pritisne dugme brise se vrednost registra koji označava inkrementiranje i od toga trenutka se registri pomeraju u desno, dok je *IR* bit generisan unapred definisanom permutacijom pozicija bita nad kojima se izvrši operacija isključivog ili (*xor*).

Jedan registar je desetobitni i služi za generisanje koordinate leve ivice pločice, dok je drugi registar osmobitni i služi za generisanje dužine pločice. Kada je ulazni signal *GENERATE* aktivan tada se u druga dva registra učitaju trenutne nasumične vrednosti u registrima za pomeranje.

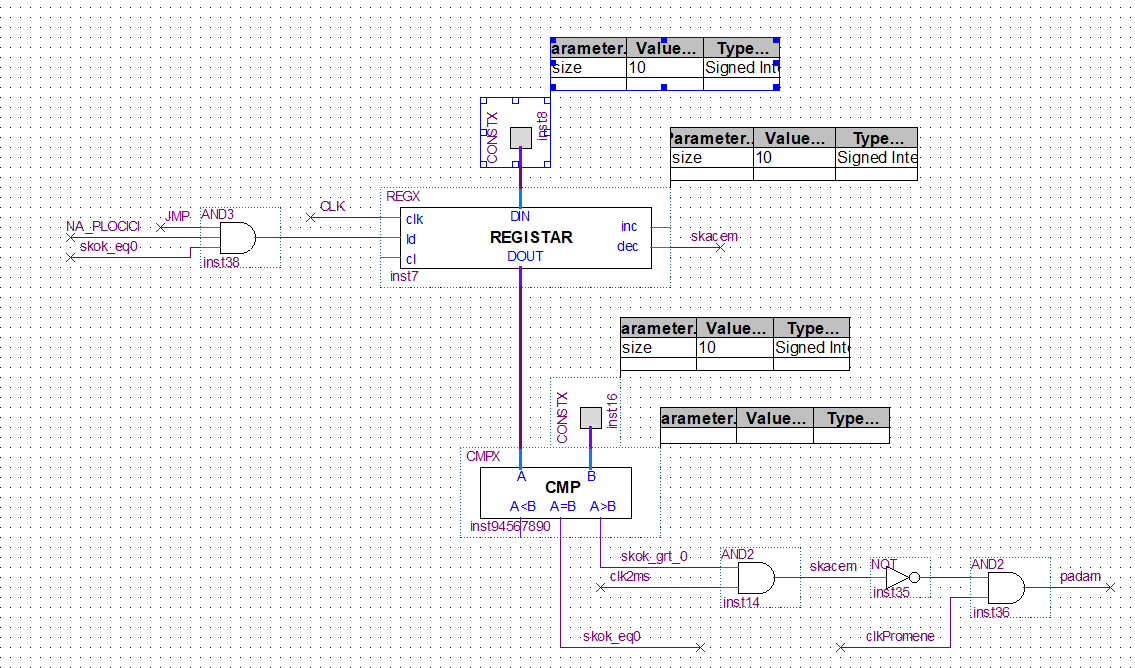


Nakon što se učitaju nasumične vrednosti u registre, njihove vrednosti prolaze kroz fazu obrade da bi se dobio željeni opseg koordinata. Ukoliko je koordinata leve ivice manja od 100 na nju se dodaje 100. Koordinata leve ivice ne sme biti manje od 100 jer se tu nalaze zidovi. Zatim ukoliko je koordinata leve ivice veća od 700 (takođe se nalaze zidovi), oduzima se 450. Slične provere se vrše i za dužinu pločice. Dužina pločice neće biti manje od 30 piksela niti veća od 170 piksela.

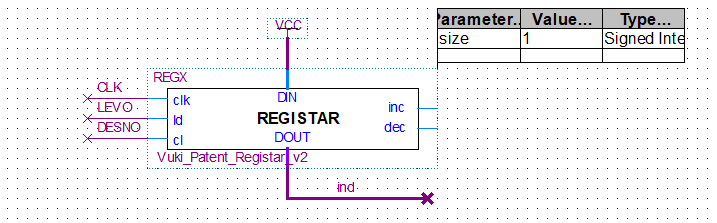
## Modul Carmander

U šemi *main* postoje dva modula *carmanderLevoDesno* i *carmanderLevoDesno2* koji predstavljaju dva igrača.

U modulu *carmanderLevoDesno* postoji registar koji čuva vrednost *start* koja označava da igrica treba da počne. Dva clock dividera predstavljaju dve brzine kojim igrač pada kada je na pločici i kada nije na pločici.



Ovaj deo predstavlja registar za učitavanje vrednosti skoka igrača. Kada se desi signal za LOAD, u registar se učitava vrednost 150 koja predstavlja jačinu skoka. Na osnovu izlaza registra se generišu signali koji označavaju da igrač skače ili pada (koji se koriste za inkrementiranje i dekrementiranje na registrima koji čuvaju vrednosti *Y* koordinata igrača).

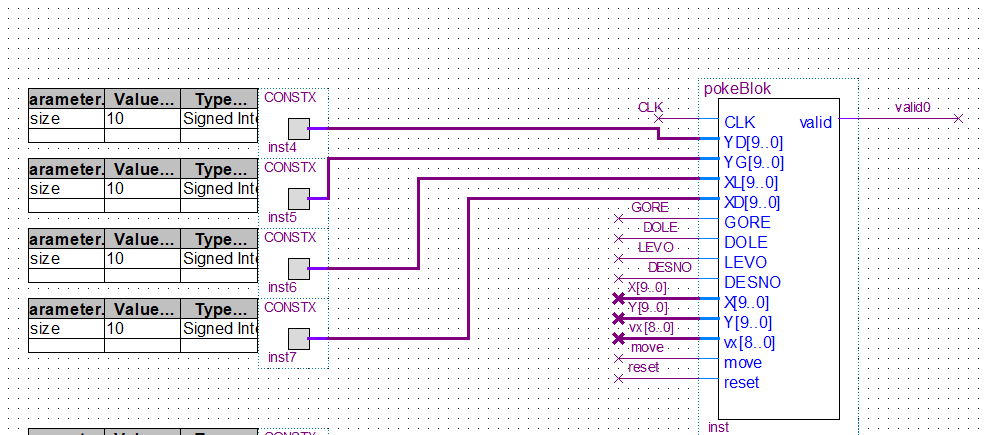


Ovaj registar čuva vrednost *ind* koja se koristi kao signal multipleksera za odlučivanje koja sličica igrača je validna u tom trenutku (*carmander* ili *flipovaniCarmander*). Ostatak ovog modula se koristi za korektno čuvanje granica koordinata igrača.

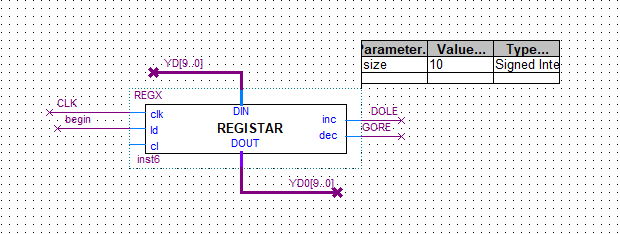
U ovom modulu postoje i dva modula *carmander* i *flipovaniCarmander* (simetrični su).

U modulu *carmander* postoje moduli *crniPokeBlok*, *narandzastiPokeBlok*, itd. koji imaju sličnu strukturu. Svaki od njih predstavlja deo sličice igrača koji je te boje.

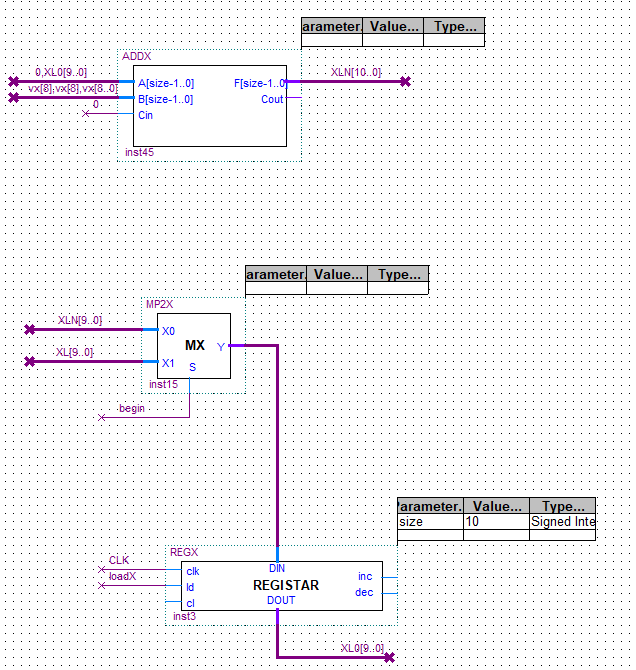
Modul *crniPokeBlok* sadrži više modula *pokeBlok* koji zajedno iscrtavaju crni deo sličice igrača.



U modul *pokeBlok* se šalju koordinate jednog crnog dela sličice igrača (XL, XD, YD, YG) koje služe za resetovanje koordinata igrača kada jedan igrač umre. Šalje se vrednost *vx[8..0]* i signal *move* koji označavaju za koliko i da li se ovaj *pokeBlok* pomera. Signali *GORE, DOLE, LEVO, DESNO* označavaju u kom smeru se pomera pokeBlok.



U modulu *pokeBlok* *Y* koordinata (ovde donja granica) se čuva u registru koji se inkrementira ili dekrementira u zavisnosti da li dolazi signal *DOLE* ili *GORE*.



U modulu *pokeBlok* *X* koordinata se čuva u registru u koji se loaduje početna vrednost (ako treba resetovati igrače) ili nova vrednost koja nastaje sabiranjem trenutne vrednosti i vrednosti za koju treba pomeriti igrača po *X* osi.