Team 3: Realizacija geometrijske transformacije slike

Geometrijske transformacije slike

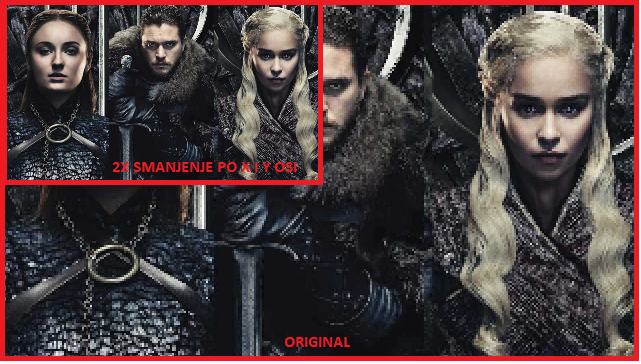
Geometrijske transformacije slike mijenjaju prostorne odnose piksela  
na slici, odnosno, kod geometrijskih transformacija, se piksel sa pozicije **(x,y)** pomjera na poziciju **(x‚,y‚)**. Sa stanovišta digitalne obrade slike, geometrijska transformacija slike se sastoji iz dvije operacije:

1. Dvodimenzionalne *prostorne transformacije* koja određuje kako se  
   pikseli pomjeraju na slici i
2. *Interpolacije intenziteta* kojom se određuju vrijednosti piksela na  
   određenim koordinatama.

Neka je data slika *f* čije su koordinate piksela (*x, y*). Prostorna transformacija, *T* , na osnovu slike *f*, određuje sliku *g* čije su koordinate piksela **(x‚,y‚)**, tako da je

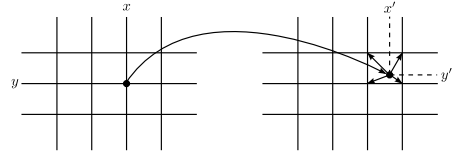


Na primjer, transformacija određena sa *r* (*x, y*) = *x/*2 i *s* (*x, y*) = *y/*2 predstavlja smanjenje veličine slike dva puta po horizontalnoj i vertikalnoj osi.

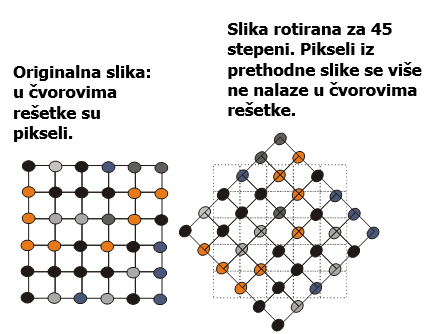


**Slika 1.0:** *Primjer smanjenja slike dva puta.*

U skladu sa prethodnim, sada je prostornu transformaciju moguće primijeniti na datu sliku tako što se za svaki piksel ulazne slike izračunaju rezultujuće koordinate  
na izlaznoj slici. Ovaj algoritam je poznat pod  
nazivom *preslikavanje unaprijed*. Međutim, koordinate piksela u digitalnoj slici moraju biti cjelobrojne, a prethodnim algoritmom se mogu  
dobiti necjelobrojne vrijednosti koordinata. Ovo znači da se pikseli slike  
*f* preslikavaju u tačke između piksela slike *g*. Vrijednosti piksela slike *g* se sada određuju interpolacijom intenziteta. Suštinski, to znači da će vrijednost ulaznog piksela uticati na vrijednosti više izlaznih piksela, te se kao česta pojava, dešava da neki ulazni pikseli mapiraju van granica izlazne slike.

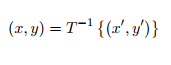


**Slika 1.1:** *Prikaz preslikavanja unaprijed.*

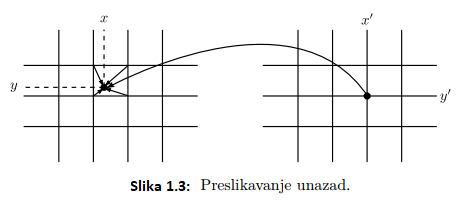


**Slika 1.2**: *Prikaz preslikavanja uz izlazak iz granica slike.*

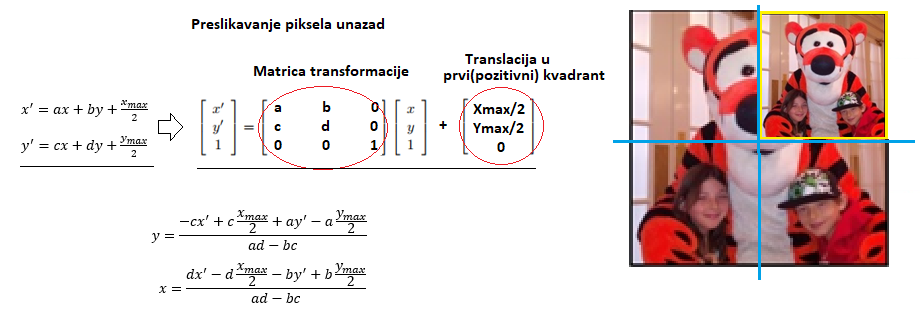
Efikasniji pristup je da se prostorna transformacija posmatra kao postupak određivanja vrijednosti piksela izlazne slike. U ovom slučaju je potrebno koordinate izlaznog piksela (*x0, y0*) preslikati u koordinatni sistem ulazne slike. Da bi se ovo postiglo potrebno je poznavati inverznu  
transformaciju



Ovaj pristup poznat je pod nazivom *preslikavanje unazad*, te je njegova reprezentacija prikazana na sljedećoj slici.



I kod preslikavanja unazad je moguće da se cjelobrojne koordinate piksela izlazne slike preslikaju u necjelobrojne koordinate u koordinatnom sistemu ulazne slike. U ovom slučaju se vrijednost intenziteta izlaznog piksela izračunava interpolacijom vrijednosti intenziteta ulaznih piksela.



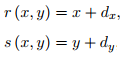
**Slika 1.4:** *Formule koje definišu opšti oblik preslikavanja unazad.*

Neke od najčešće korišćenih transformacija, su:

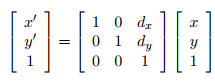
* Translacija
* Rotacija
* Obrtanje
* Skaliranje
* Smicanje, odnosno distorzija i slično.

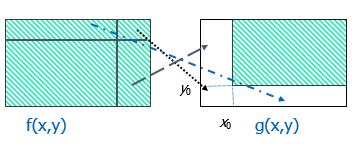
**Translacija slike**

Translacija tačke određena je transformacijom oblika



Predstavljeno u matričnom obliku, imamo sljedeće,





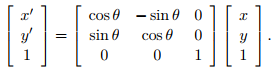
**Slika 1.5**: *Primjer translacije slike, uz dopunjavanje bijelom bojom.*

**Rotacija slike**

Rotacija tačke oko ishodišta za ugao *θ* se može opisati sljedećim jednačinama,



Predstavljeno u matričnom obliku, imamo sljedeće,

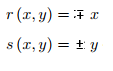
**



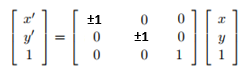
**Slika 1.6**: *Primjer rotacije originalne slike za ugao od 35 stepeni ulijevo.*

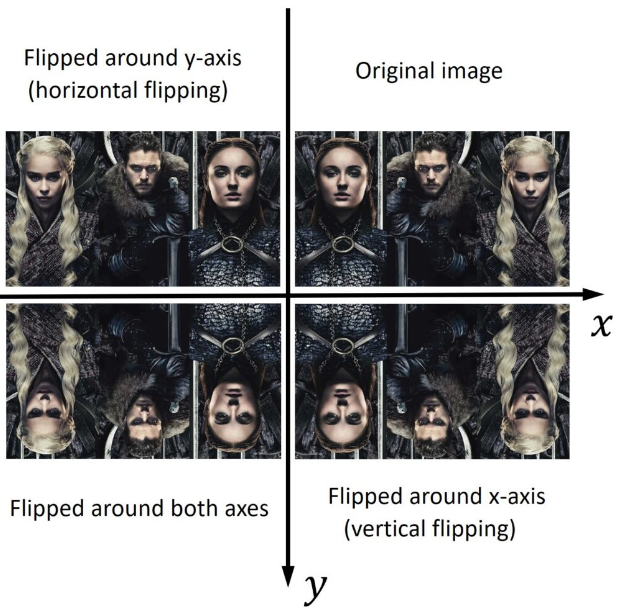
**Obrtanje(Flip) slike**

Obrtanje tačke određeno je transformacijom oblika

****

Predstavljeno u matričnom obliku, imamo sljedeće,

****

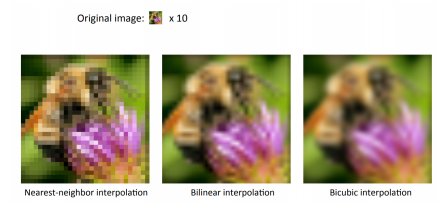
****

**Slika 1.7**: *Primjer slike okrenute oko x ose, y ose, te oko obe ose.*

**Interpolacija intenziteta piksela**

Kao što je pomenuto, može se desiti da, prilikom preslikavanja unazad, vrijednost piksela izlazne slike zavisi od vrijednosti intenziteta tačke sa necjelobrojnim koordinatama. U ovim slučajevima je potrebno interpolirati vrijednost intenziteta te tačke na osnovu vrijednosti intenziteta susjednih piksela. Definišemo nekoliko vrsta interpolacije,

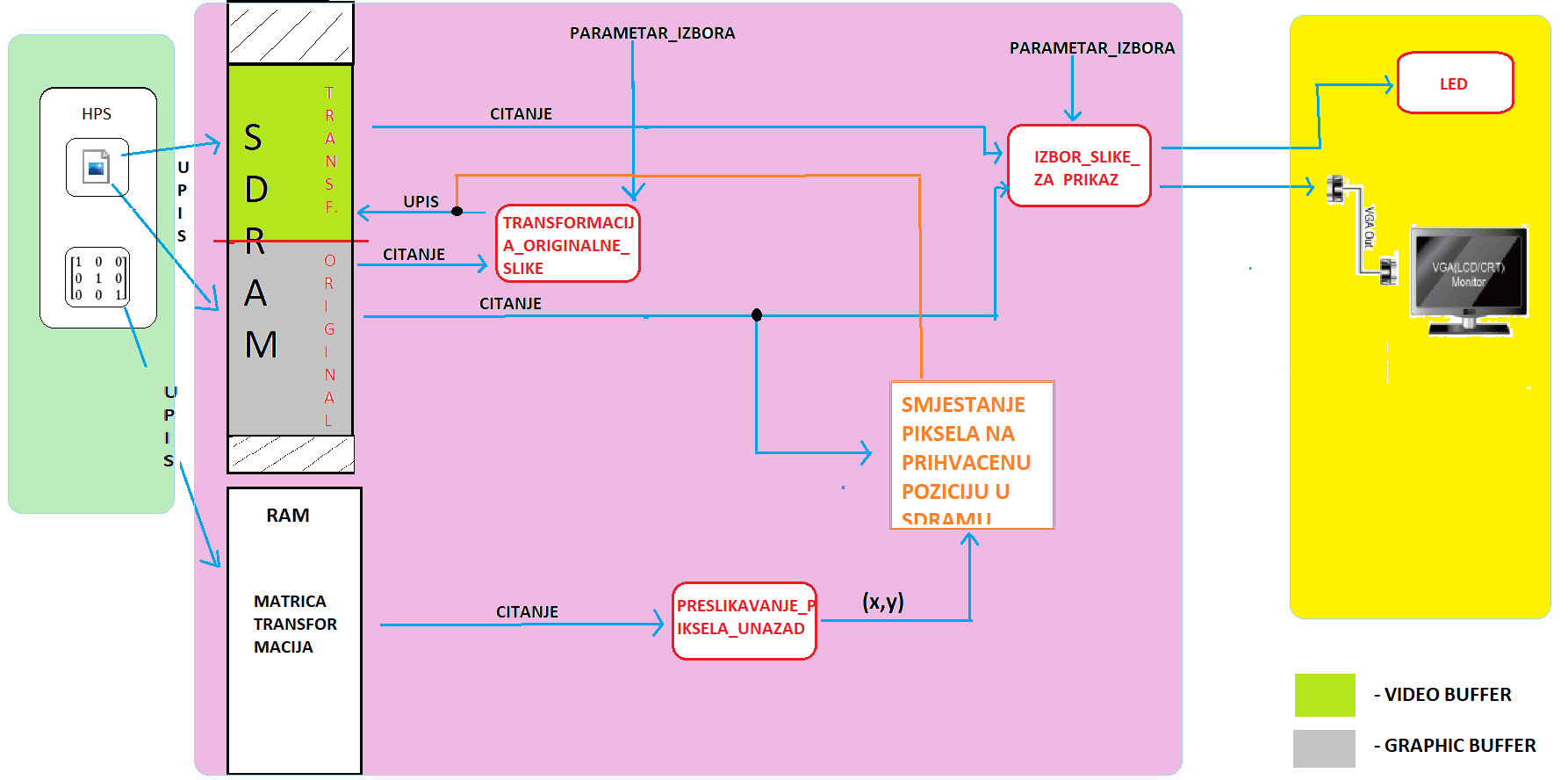
* **Interpolacija na osnovu najbližeg susjeda,** vrijednost izlaznog piksela je jednaka vrijednosti onog ulaznog piksela koji je najbliži lokaciji na koju se preslikava izlazni piksel. Ipak, ovakva interpolacija može uvesti artifakte na slikama koje sadrže fine strukture i gdje se nivo intenziteta znatno mijenja od piksela do piksela
* **Bilinearna interpolacija,** značajno složenija u odnosu na prethodnu, te se bazira na funkcijama više promjenjivih.
* **Bikubična interpolacija**

****

**Realizacija geometrijske transformacije slike**

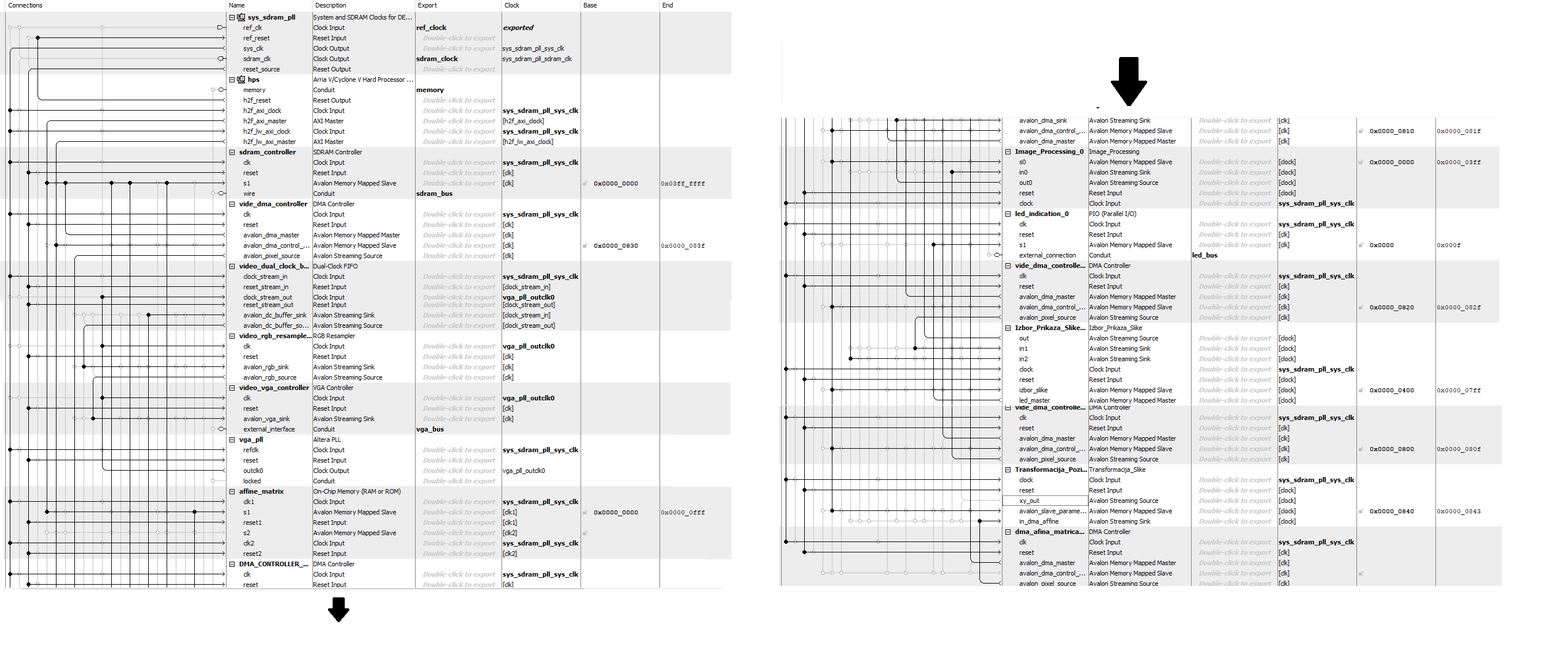
Ciljni zadatak, bazira se na obavljanju geometrijskih transformacija slike, koja je pritom smještena u SDRAM memoriji FPGA ploče, smještanje tako transformisane slike ponovo u SDRAM, te njeno prikazivanje na izlazu, za što će se iskoristiti VGA. U skladu sa tim, rezolucija slike iznosi 640x480, što definiše sam VGA, pri čemu se prilikom transformacije, odnosno obrade same slike, uzima u obzir da je slika predstavljena u 565 RGB formatu, što definiše da se po 5 bita koristi za R i B komponentu, odnosno 6 bita za G komponentu. Kako sam VGA ne prepoznaje taj format, zahtjeva se transformacija 565 u neki od poznatih VGA standarda, što može biti 10-10-10 RGB standard.

Generalizovana struktura datog sistema, nalazi se na sljedećoj slici:



**Slika 1.8**: *Struktura Sistema za transformaciju slike.*

Slika koja dati sistem reprezentuje pri realizaciji u QSYS Tool-u, data je u prilogu.



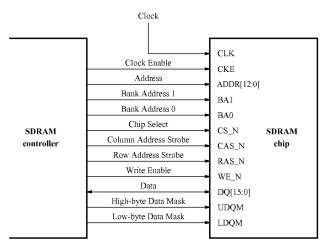
**Slika 1.9**: *Struktura Sistema QSYS alat.*

Na slici koja reprezentuje generalizovani oblik sistema, možemo uočiti nekoliko komponente(blokova, modula), koje su zaslužne za geometrijsku transformaciju slike, realizaciju grafičkog te video bafera, kao i ostvaritavnje pamćenja transformacionih matrica, koje definišu samu transformaciju,

1. SDRAM komponenta
2. On-Chip RAM (Affine Matrix)
3. Izbor slike za prikaz
4. Transformacija originalne slike
5. Preslikavanje unazad piksela originalne slike
6. Smjestanje prethodno prihvacenih piksela, u SDRAM
7. Led indikator(LED diode)
8. VGA

**SDRAM(Synchronius Dynamic Random Access Memory)**

SDRAM čip na DE1-SoC ploči ima kapacitet od 512 Mbits (64 Mbytes). Svaki čip je organizovan u 8M x 16 bits x 4 banks. Signali neophodni za komunikaciju sa datim čipom, dati su na sljedećoj slici,



**Slika 2.0**: *SDRAM signali.*

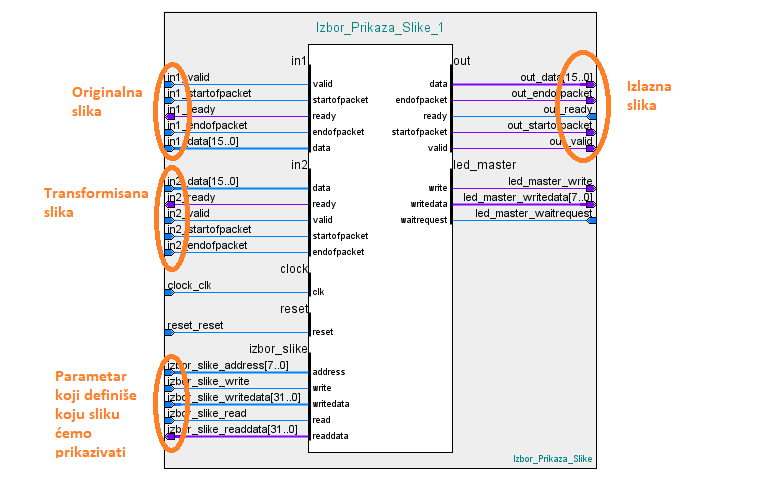
Za komunikaciju sa samom memorijom, koristi se SDRAM Controller, koji se generiše korišćenjem već postojeće komponente u Qsys toolu, uz podešavanja kapaciteta memorije, vremena pristupa, te sličnih parametara. Kao što je moguće vidjeti na slici koja reprezentuje strukturu Sistema, SDRAM memorija se koristi kroz dvije funkcije. Prva je uloga grafičkog bafera, što je dio memorije odvojen za smještanje najčešće “nepromjenjive, konstantne” slike, te je često kod transformacija, nazivamo originalna slika. Druga je Image Buffer, ili kako je navedeno Video Buffer, što je suštinski dio memorije sa promjenjivim slikama. U ovom slučaju, tu će se smještati sadržaj koji se pušta na izlaz, te nad kojim se obavljaju transformacije. Da bismo ostvarili pristup sadržaju memorije, koristimo DMA kontroler, odnosno njegovu izvedenu verziju Video kontrolera, kome navodimo širinu I visinu slike, funkciju(čitanje/upis), te početnu adresu, na osnovu čega se vrši kontinualno čitanje/upis sadržaja memorije. Po dostizanju posljednjeg piksela slike, “pokazivač” se vraća na početak, te se nastavlja čitanje/upis. U skladu sa tim, kako nismo bili u mogućnosti da ostvarimo pristup memoriji sa kompleksnošću O(1), gdje bismo na osnovu adrese željenog piksela, direktno mu pristupili preko DMA, te ga pročitali(DMA za povezivanje sa SDRAM kontrolerom koristi Avalon Memory Mapped Master/Slave, u kome se indirektno na osnovu prethodnog definiše kontinualni pristup, te se sadržaj generiše preko Avalon-ST Source-a), ostvarujemo kompleksnost 0(N), što će za kompleksnije geometrijske transformacije, ostvariti značajan pad performansi izvršavanja.

**On-Chip RAM (Affine Matrix)**

Dio memorije SoC-a, smješten na FPGA dijelu, namjenjen za čuvanje matrice transformacije neophodne za obavljanje transformacije slike. Matrica transformacija, njena veličina i sadržaj, biće pojašnjeni u odljeljku za Geometrijske transformacije slike. Suštinski, upis matrice transformacije se obavlja iz HPS dijela sistema, na osnovu korisnički izabranog parametra koji definiše tip transformacije, dok se njeno čitanje i procesiranje, obavlja na FPGA dijelu, preko DMA kontrolera, odnosno procesiranje se obavlja u bloku Transformacija\_Slike. Prednosti korišćenja date memorije za smještanje matrice transformacije, ogledaju se u najvećem dostignutom throughput-u, od svih FPGA memorije, kao i najmanjem vremenu kašnjenja podataka. Nedostaci se uglavnom ogledaju u relativno malom kapacitetu memorije, koji se kreće od 15kB pa do 2MB.

**Izbor slike za prikaza**

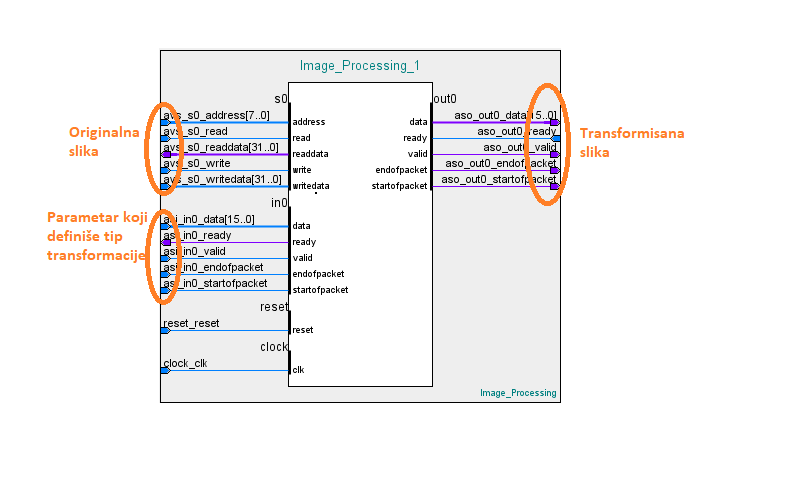
Komponenta koja ima za zadatak da obezbjedi čitanje, te prikaz originalne slike iz Grafičkog bafera, ili transformisane slike iz Image bafera(često se koriste termini originalna slika, te kopija, odnosno transformacija). Koja slika će se prikazivati na izlazu, definiše korisnik preko korisničkog parametra unutar C programa HPSa, što se prenosi preko AXI mastera ka datoj komponenti, koja prihvaćeni parameter procesira, te obezbjeđuje prikaz.U skladu sa tim, pored parametra izbora, data komponenta treba da ima pristup oba dijela SDRAMa, kako bi se prethodno ostvarilo, te se kao rezultat prihvaćanja parametra, ostvaruje prikaz na LED diodama FPGA, za šta se koristi Avalon MM Master, kome prosljeđuje vrijednost upisa.

****

**Slika 2.1**: *Struktura signala bloka*

**Transformacija originalne slike(Image\_Processing)**

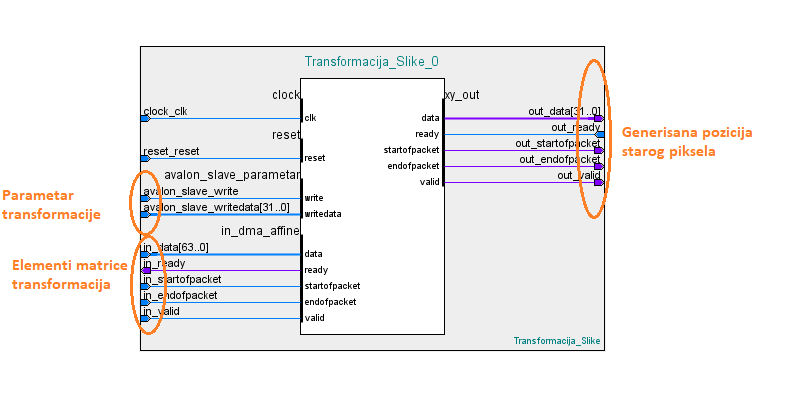
Specijalan predefinisan blok, odnosno modul, koji ima za zadatak da po prihvatanju parametra koji definiše tip transformacije, čita originalnu sliku piksel po piksel, te u zavisnosti od toga koju transformaciju vrši, dati piksel upisuje u memoriju kao piksel transformisane slike, ili ga jednostavno ne koristi. Ovaj pristup, je već pomenut kod SDRAMa, gdje smo rekli da usljed neostvarene 0(1) kompleksnosti, zahtjeva se 0(N), što hoće reći, ako nam treba piksel transformisane slike u poziciji (123,222), trebamo proći kroz originalnu sliku do tog piksela kontinualno, pri čemu sve njih prihvaćamo u datom bloku, ali ne koristimo, te kad dostignemo piksel originalne slike, upišemo g ana mjesto gdje se DMA kontroler za upis prethodno zaustavio. Na ovaj način, ostvaruje se preslikvanje piksela unazad, ali dati blok ne koristi matricu transformacija, već ima predefinisane algoritme za već pomenute korišćene transformacije.

****

**Slika 2.2**: *Struktura signala bloka*

**Preslikavanje unazad piksela originalne slike**

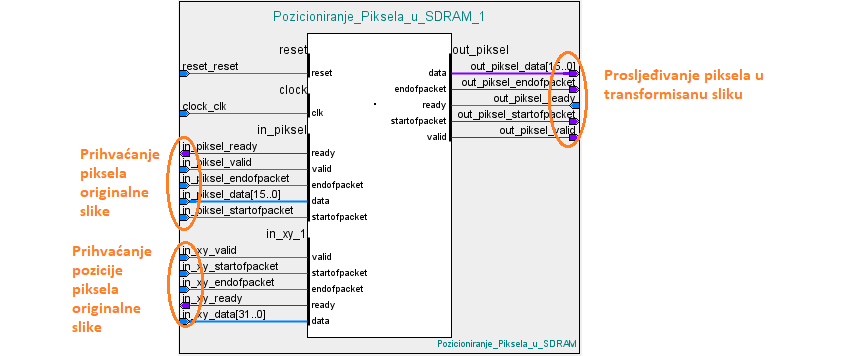
U skladu sa prethodnim blokom, gdje smo vršili predefinisanje željenih transformacija, što nije najkorisnije rješenje, druga varijanta se bazira na korišćenju matrice transformacije, njenom procesiranju, te ostvarivanju preslikavanja unazad. Već je pomenuto kako se matrica transf. generiše, gdje se smješta te kako, ali ono što nije pomenuto, je gdje se čita, odnosno procesira. Upravo za to se koristi dati modul, na način da znamo dimenzije slike, broj bita koji se odvaja za svaki piksel(njegov format), generišemo virtuelnu matricu datih dimenzija kroz koju prolazimo, ten a osnovu pozicija date matrice, na osnovu formula za preslikavanje piksela unazad, generišemo nove pozizicije. Generisane pozicije prosljeđujemo bloku koji će zapravo odraditi memorijsko mapiranje, te reorganizaciju originalne slike, na osnovu prihvaćenih pozicija.



**Slika 2.3**: *Struktura signala bloka*

**Smjestanje prethodno prihvacenih piksela, u SDRAM**

Po dobijanju podatka sa prethodno definisanog bloka, što je zapravo uređeni par (x,y) koji definiše poziciju piksela originalne slike, podaci se dobijaju u formatu 32 bita, te je neophodno njihovo formatiranje, i adekvatna dodjela deklarisanim promjenjivim. U skladu sa prethodnim, treba da blokiramo pristizanje novih pozicija, dok se data pozicija ne procesira. Procesiranje pozicije, bazira se na prihvaćanju piksela iz memorije(originalna slika), te njihovo „brojanje“, odnosno po uspostavljanju prihvaćene pozicije, neophodno je dati prihvaćeni piksel proslijediti na izlaz, što će preko DMA kontrolera biti upisano kao piksel transformisane slike.



**Slika 2.4**: *Struktura signala bloka*

VGA

Kao izlaz na osnovu koga ćemo provjeriti prethodno objašnjeno, koristimo VGA koji je sastavni dio FPGA dijela SoCa. VGA, je već na neki način pomenut na početku, gdje smo govorili o formatu piksela, te formatu njegovog prikazivanja. Odnosno, pikseli se smještaju/čuvaju/procesiraju u formatu 5-6-5 RGB, dok sam VGA zahtjeva neki od drugih formata, što zapravo zahtjeva transformaciju formata piksela. U ovom slučaju, koristi se format 10-10-10 RGB. Da bismo pored prethodnog, zapravo prikazali rezultat našeg rada, neophodi su sljedeći elementi, koji su podržani u Qsys alatu,

* PLL, generator takt signala neophodnog za VGA(25MHz)
* Dual-Clock FIFO Buffer, neophodan zbog razlikovanja takta VGA od ostatka sistema, pa se zahtjeva smještanje piksela za prikaz.
* VGA Resampler, koristimo za promjenu formata piksela.
* VGA Controller, omogućava pristup VGA izlazu, te prikaz sadržaja.

Softversko obavljanje transformacija

Pored prethodno definisanog načina za obavljanje transformacije nad slikom, koga možemo posmatrati kao hardverski način obavljanja transformacije, transformacije možemo obavljati i softverski. Ovaj način, bazira se na tome da originalnu sliku koju učitamo pomoću HPSa u SDRAM, možemo i pročitati unutar programa, te tako pročitanu sliku softverski unutar C programa obraditi, dobiti transformisanu sliku, koju ćemo smjestiti u Image buffer. Na ovaj način, sa FPGA strane ostaje samo da tu sliku čitamo, i prikazujemo na VGA, ili neki drugi izlaz.

**Literatura**

1. *DE1-SoC User Manual*, адреса: www.terasic.com pristupljeno Oktobar. 2020
2. *Using the SDRAM*, адреса: <http://using_the_SDRAM> pristupljeno Oktobar. 2020
3. DMA Controller: <https://www.intel.com/content/www/us/en/programmable/solutions/partners/partner-profile/eureka-technology-inc-/ip/dma-controller.html> pristupljeno Oktobar. 2020
4. *Making Qsys component*, адреса: <http://making_qsys_component> pristupljeno Oktobar. 2020
5. *Rotating Images*, адреса: <http://datagenetics.com/blog/august32013/index.html>
6. pristupljeno Oktobar. 2020
7. Resize, Translate, Flip and Rotate an Image, адреса: <http://datahacker.rs/003-how-to-resize-translate-flip-and-rotate-an-image-with-opencv/> pristupljeno Oktobar. 2020
8. *TEAM#2:* [*https://gitlab.com/mladen.knezic/videobox/-/wikis/Team2-report#sdram*](https://gitlab.com/mladen.knezic/videobox/-/wikis/Team2-report#sdram)pristupljeno Oktobar 2020
9. *Avalon Interface Specification* : <https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/manual/mnl_avalon_spec.pdf> pristupljeno Oktobar 2020