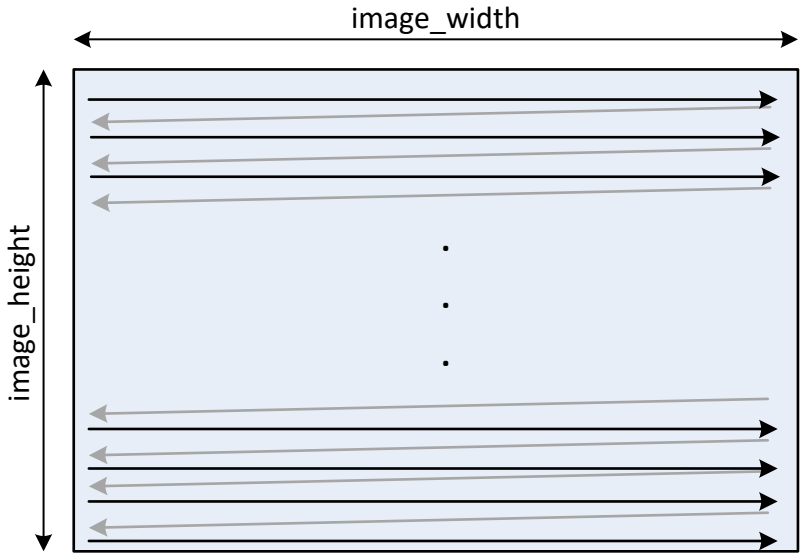
DIGITALNI VLSI SISTEMI 2018/2019 – PROJEKTNI ZADATAK

UBRZANJE ALGORITMA ZA EKVALIZACIJU HISTOGRAMA SLIKE

1. Kreiranje osnovnog sistema

Potrebno je realizovati sistem za ekvalizaciju histograma ulazne slike. Ulazna slika je u binarnom formatu pri čemu prva 4 bajta predstavljaju širinu slike, druga 4 bajta visinu slike dok nakon toga slede pikseli slike u **uint8** formatu. Slika u binarnom formatu se nalazi na host računaru i učitava se u SDRAM memoriju korišćenjem ***host file system*** mehanizma. Po uključivanju ***hostfs*** opcije u okviru BSP, moguće je koristiti standardne C funkcije **fread** i **fwrite** za pristup binarnim fajlovima koji se nalaze na host računaru. ***Voditi računa da ova opcija radi samo u Debug modu.*** Smatrati da su ulazni podaci 8-bitni pikseli čija vrednost se nalazi u opsegu 0-255.

Na Slici 1. je prikazan redosled upisivanja piksela u memoriju. Prvi piksel predstavlja gornji levi ugao slike, na narednu memorijsku lokaciju se upisuje prvi desni susedni piskel iz istog reda. Nakon upisivanja poslednjeg piksela prvog reda slike na sledeću memorijsku lokaciju se upisuje prvi piksel iz narednog reda. Tako da se slika zapravo u memoriji može posmatrati kao niz dimenzija $M \cdot N$ gde su M i N visina i širina slike.



Slika 1. Redolsed upisivanja piksela u memoriju

Prvi korak algoritma predstavlja određivanje histograma slike. Histogram predstavlja niz od 256 elemenata (pošto postoji ukupno 256 različitih vrednosti piksela) pri čemu svaki element niza sadrži broj piksela u ulaznoj slici čija je vrednost jednaka poziciji unutar niza. Na primer ***hist[5]*** sadrži informaciju koliko piksela u ulaznoj slici ima vrednost jednaku 5. Ako je ulazna slika označena sa ***x***, histogram sa ***hist*** i ako su dimenzije ulazne slike MxN onda je histogram slike definisan sledećim izrazom:

$$hist[k] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \left(x[m,n] == k \right)$$

Elementi ***hist*** niza su predstavljeni sa 32-bita. Preslikavanje koje definiše ekvalizaciju histograma je definisano funkcijom kumulativnog histograma predstavljenog sledećim izrazom:

$$cumhist[k] = round\left(\frac{255}{M \cdot N}\sum_{l=0}^{k} hist[l]\right)$$

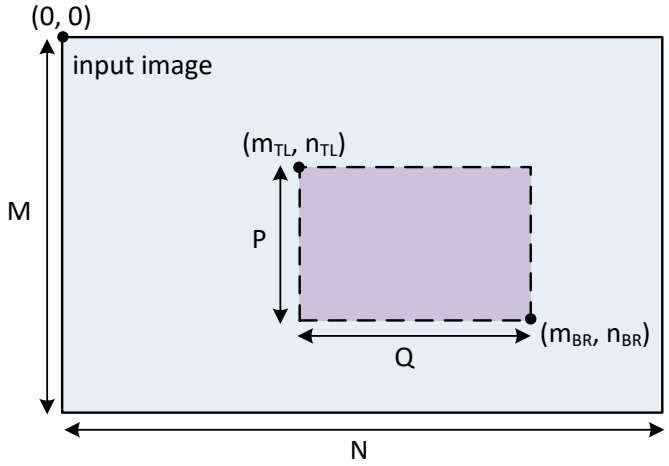
Niz ***cumhist*** sada predstavlja tabelu preslikavanja ulazne slike u sliku sa uniformnim histogramom. Na primer element ***cumhist[5]*** sadrži informaciju u koju vrednost treba da se preslika piksel koji u ulaznoj slici ima vrednost 5. Elementi ovog niza uzimaju vrednosti iz opsega 0-255 i zauzimaju po 8 bita. Izlazna slika ***y*** se određuje na osnovu izraza:

$$y[m,n] = cumhist\left[x[m,n]\right],\ \ m = 0..M-1,\ n = 0..N-1$$

<u>Omogućiti da se histogram računa na osnovu prozvoljnog regiona slike.</u> Region slike se zadaje koordinatama gornjeg desnog i donjeg levog piksela kao što je prikazano na Slici 2. Neka su koordinate gornjeg levog piksela m_{TL}, n_{TL} i donjeg levog piksela m_{BR}, n_{BR} i neka su dimenzije regiona od interesa $P = m_{BR} - m_{TL} + 1$ i $Q = n_{BR} - n_{TL} + 1$. Tada se izrazi za histogram i za kumulativnu sumu definišu na sledeći način:

$$hist[k] = \sum_{m=m_{TL}}^{m_{BR}} \sum_{n=n_{TL}}^{n_{BR}} \left(x[m,n] == k\right)$$

$$cumhist[k] = round\left(\frac{255}{P \cdot Q}\sum_{l=0}^{k} hist[l]\right)$$



Slika 2. Zadavanje proizvoljnog regiona unutar slike

Po završetku procesiranja izlazna slika se šalje nazad na host računar i upisuje u izlazni binarni fajl. Prva 4 bajta izlaznog fajla sadrže širinu isprocesirane slike, druga 4 bajta visinu isprocesirane slike, nakon toga slede pikseli izlazne slike u **uint8** formatu (svaki bajt sadrži informaciju o jednom pikselu izlazne slike). Kreiranje binarnih fajlova ulazne slike i učitavanje binarnih fajlova izlazne slike realizovano je skriptama ***write_input_bin_image.m*** i ***show_bin_img.m***.

Demonstrirati funkcionalnost sistema na slikama ***bright64.bin***, ***dark64.bin***, ***low_contrast64.bin***, ***orig64.bin***. Pri evaluaciji rezultata izvršiti poređenje sa procesiranjem u Matlabu. Za generisanje referentnih rezultata možete koristiti skriptu ***generate_ref_outputs.m***.

Korišćenjem komponenti ***performance counter***-a izmeriti brzinu rada (procesiranja slike) ove realizacije na slici ***dark512.bin***. Pirikazati odvojene rezultate za računanje histograma, određivanje kumulativne sume i finalno mapiranje izalzne slike.

2. Ubrzanje određivanja histograma

Kako bi se ubrzalo procesiranje potrebno je dodati hardverski blok kojim se omogućava računanje histograma proizvoljnog regiona ulazne slike. Protok podataka se obavlja pomoću Avalon-ST interfejsa. Ulazni Avalon-ST Sink interfejs prihvata 8-bitne ulazne podatke dok po izlaznom Avalon-ST Source interfejsu šalje vrednosti histograma grupisane u 32-btine podatke. Konfiguracija modula se obavlja pomoću Avalon-MM interfejsa. Modul poseduje registar u koji se upisuje broj piksela regiona koji se procesira. Takođe, modul treba da poseduje statusni i kontrolni registar.

Demonstrirati funkcionalnost sistema na slikama ***bright64.bin***, ***dark64.bin***, ***low_contrast64.bin***, ***orig64.bin***. Pri evaluaciji rezultata izvršiti poređenje sa procesiranjem u Matlabu.

Korišćenjem komponenti ***performance counter***-a izmeriti brzinu rada (procesiranja slike) ove realizacije na slici ***dark512.bin***.

Uporediti brzinu obrade u odnosu na čistu softversku realizaciju.

Koliko ubrzanje se dobije ukoliko se ulazna magistrala proširi na 32-bita tako da u jednom taktu u projektovani modul dolaze po 4 piksela? Ideja je da se unutar komponente generišu 4 histograma ***hist0***, ***hist1***, ***hist2*** i ***hist3*** koji se paralelno popunjavaju pri čemu je vrednost koja se šalje na izlaz zapravo zbir vrednosti akumuliranih u ova 4 histograma ($hist[k] = hist0[k] + hist1[k] + hist2[k] + hist3[k]$).

3. Ubrzanje mapiranja intenziteta

Kako bi se ubrzalo određivanje odbiraka izlazne slike potrebno je dodati hardverski blok kojim se omogućava mapiranje inteziteta piksela ulazne slike u vrednosti izlaznih piksela. Funkcija mapiranja je definisana nizom od 256 elemenata koji zapravo predstavlja LUT (look up tabelu). Protok podataka se obavlja pomoću Avalon-ST interfejsa. Ulazni Avalon-ST Sink interfejs prihvata 8-bitne ulazne podatke dok po izlaznom Avalon-ST Source interfejsu šalje vrednosti izlaznih piksela koji su takođe 8 bita. Elemente LUT-a je potrebno smestiti u on-chip memoriju. Memorijski blokovi se u IP katalogu nalaze pod ***Library → Basic Functions → On Chip Memory***. Modul poseduje registar u koje se može upisati broj piksela slike koja se procesira. Takođe modul treba da poseduje statusni i kontrolni registar. Ovim registrima se pristupa preko Avalon-MM interfejsa. Hardverski modul se može naći u jednom od 3 stanja: idle (slobodno stanje), konfiguracija (konfigurisanje tabele preslikavanja) ili procesiranje (stanje u kom se obrađuju pikseli ulazne slike). Konfiguracija se može obaviti pomoću Avalon-MM interfejsa ili Avalon-ST interfejsa. Studentima se ostavlja izbor da odaberu jednu od ove dve varijante. U izveštaju je potrebno argumentovati donetu odluku. Konfiguracija se započinje upisom odgovarajućeg bita u kontrolni registar. Započinjanje konfiguracije označava da je potrebno postaviti vrednosti svih 256 koeficijenata unutar tabele preslikavanja. Procesiranje ne može započeti dok je konfiguracija u toku. Procesiranje započinje upisom odgovarajućeg bita u kontrolni registar. Procesiranje traje sve dok se ne obradi svi pikseli ulazne slike. Obezbediti mogućnost da se sa procesora preko statusnog registra očita trenutno stanje periferije.

Kako bi se postiglo ubrzanje sistema potrebno je dodati projektovane hardverske blokove u sistem kao i odgovarajuće DMA kontrolera kojim se obezbeđuje prenos podataka između memorije i akceleratora.

Demonstrirati funkcionalnost sistema na slikama ***bright64.bin***, ***dark64.bin***, ***low_contrast64.bin***, ***orig64.bin***. Pri evaluaciji rezultata izvršiti poređenje sa procesiranjem u Matlabu.

Korišćenjem komponenti ***performance counter***-a izmeriti brzinu rada (procesiranja slike) ove realizacije na slici ***dark512.bin***. Pirikazati odvojene rezultate za računanje histograma, određivanje kumulativne sume i finalno mapiranje izalzne slike.

Uporediti brzinu obrade u odnosu na softversku realizaciju.

NAPOMENA: Potrebno je da svaka grupa kreira blog na koji će postavljati vesti o svom trenutnom progresu. **Uniformisati ime bloga: DVS18_GX.** Portali na kojima se može besplatno pokrenuti blog su https://www.blogger.com/ i https://wordpress.org/ . Po otvaranju bloga potrebno je poslati link predmetnim asistentima kako bi mogli da prate progres projekta. **Važno je da redovno osvežavate vaš blog!**

Na kraju projekta je potrebno da kreirati izveštaj koji predstavlja dokumentaciju projekta i opisuje arhitekturu sistema, realizaciju hardvera i softvera za svaki od delova projekta. Izveštaj predstavlja dokumentaciju projektovanog sistema i potrebno je da bude dovoljno detaljan kako bi neko uz izveštaj i kod mogao potpuno da reprodukuje rezultate i iskoristi delove projekta u većem sistemu.

Za pun broj poena potrebno je da pored potpune funkcionalnosti sistem bude smisleno particionisan i kod uredno napisan sa dovoljnim brojem komentara.