**Facial Expression Recognition**

*Dokumentacija za predmet: Projektovanje elektronskih uređaja na sistemskom nivou*

1. **Specifikacija**

Ideja projekta “*Facial expression recognition* (FER)” je implementacija algoritma koji identifikuje i kategoriše emocije na osnovu izraza ljudskog lica. Konačan algoritam kombinuje više ML algoritama i gotovih biblioteka. Za detekciju lica korišten je HOG algoritam, za izvlačenje karakterističnih tačaka dlib biblioteka, a za klasifikaciju korišteni su AdaBoostClassifier i DecisionTreeClassifier.

Dio algoritma koji zahtjeva najviše vremena je HOG, stoga se priča u nastavku odnosi na njegovo ubrzavanje dok će ostatak algoritma biti ostavljen u softveru. Osnovna ideja koja stoji iza ovog algoritma je pronaći skup HOG vektora na dijelu originalne slike koji je sličan skupu HOG vektora šablon slike (*Slika 1.2*). Algoritam ekstrahuje HOG šablona slike kojoj se mijenjaju dimenzije od minimalne dimenzije originalne slike do jedne trećine minimalne dimenzije originalne slike da bi se omogućilo prepoznavanje lica bez obzira na njegovu veličinu na originalnoj slici. Sa različitim veličinama HOG-a šablona slike algoritam prolazi kroz originalnu sliku i pamti one koji su iznad određenog praga (threshold).

*Slika 1.1 Originalna slika Slika 1.2 Šablon slika Slika 1.3 Izlazna slika*

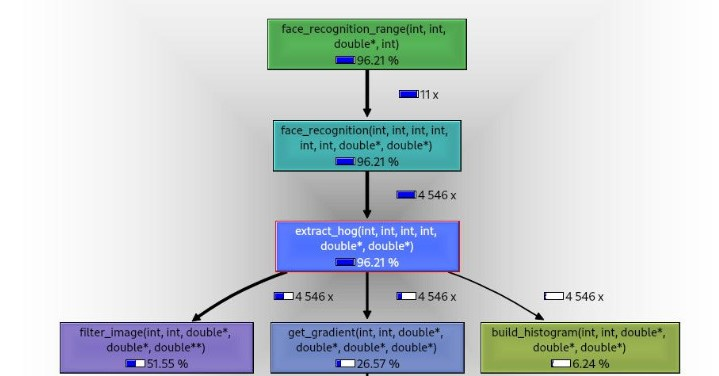
1. **Profajliranje**

Nakon što su funkcije HOGa prevedene u jezik C, utvrđeno je da se naduže izvršava **filter\_image** funkcija (51.55%) stoga je potrebno ubrzati taj dio softvera. Rezultati profajliranja prikazani su u *Tabeli 2.1.*

| **Funkcija** | **Potrošeno procesorsko vrijeme** |
| --- | --- |
| **face\_recogntion\_range** | **95,21%** |
| **face\_recognition** | **95,21%** |
| **extract\_hog** | **95,21%** |
| **filter\_image** | **51.55%** |
| **get\_gradient** | **26.57%** |
| **build\_histogram** | **6.24%** |

*Tabela 2.1 – Tabela rezultata profajliranja*

Funkcije prikazane u tabeli pozivaju jedna drugu, redoslijedom kojim su napisane (*Slika 2.2*). Na primjer, funkcija extract\_hog troši ukupno 95.21% vremena, od čega funkcija fiter\_image troši 51.55%, get\_gradient 26.57%, a build\_histogram 6.24% vremena.



*Slika 2.2 – Rezultati profajliranja*

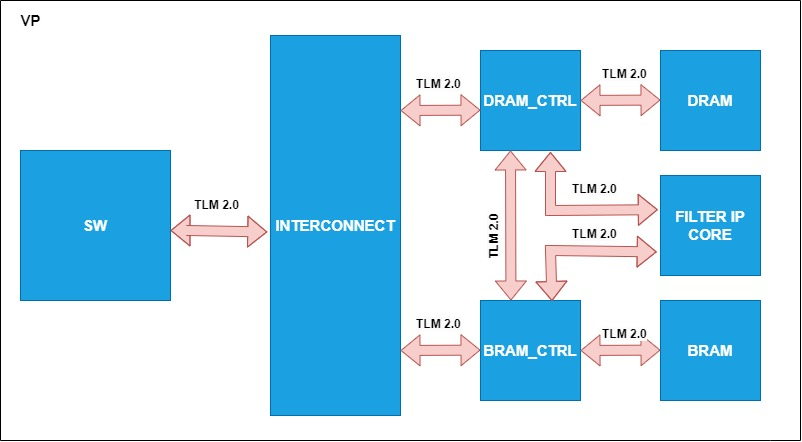
1. **Bitska analiza**

U okviru funkcije filter\_image kao ulaz i izlaz koriste se brojevi sa pokretnim zarezom. Urađena je dinamička bitska analiza. Utvrđeno je da je dovoljna preciznost od **16 bita** za predstavljanje ulaznih i izlaznih vrijednosti. Ulaz u funkciju su normalizovane pozitivne vrijednosti opsega [0, 1], pa za njihovo predstavljanje koristimo 1 bit za cijeli dio, a ostalih 15 za decimalni dio. Izlazi funkcije mogu imati vrijednosti u opsegu [-4, 4] i zbog toga za njihovu reprezentaciju koristimo 4 bita za cijeli dio, dok se preostalih 12 koristi za predstavljanje decimalnog dijela broja.

Pored toga analizirana je širina promjenljive koja se koristi za izračunavanja unutar ove funkcije. Ta promjenljiva je imROI i tretira se isto kao i ulaz u funkciju, 16 bita širine gdje jedan bit odlazi na predstavljanje cijelog dijela broja. Na kraju, tu je i niz konstanti koje predstavljaju filtar\_x i filtar\_y. S obzirom da uzimaju vrijednosti u opsegu od -2 do 2, potrebno je svega 3 bita za njihovo predstavaljanje.

1. **Virtualna platforma**

Na *Slici 4* prikazana je blok šema virtualne platforme koja izvršava HOG algoritam.



*Slika 4.1 – Blok dijagram virualne platforme*

Platforma se sastoji od sljedećih modula:

* **Softver modul** - opisuje ponašanje softvera
* **Interconncet modul** - obezbjeđuje memorijsko mapiranje
* **BRAM modul** - memorija koju koristi akcelerator
* **BRAM CTRL modul** - upravlja komunikacijom izmedju BRAM-a i drugih modula
* **DRAM modul** – memorija za skladišenje ulaznih i izlaznih piksela
* **DRAM CTRL modul** - upravlja komunikacijom između DRAM-a i drugih modula
* **Hardver modul** - implementira funkciju za ubrzavanje

Sistem prikazan na Slici 4 funkcioniše na sljedeći način:

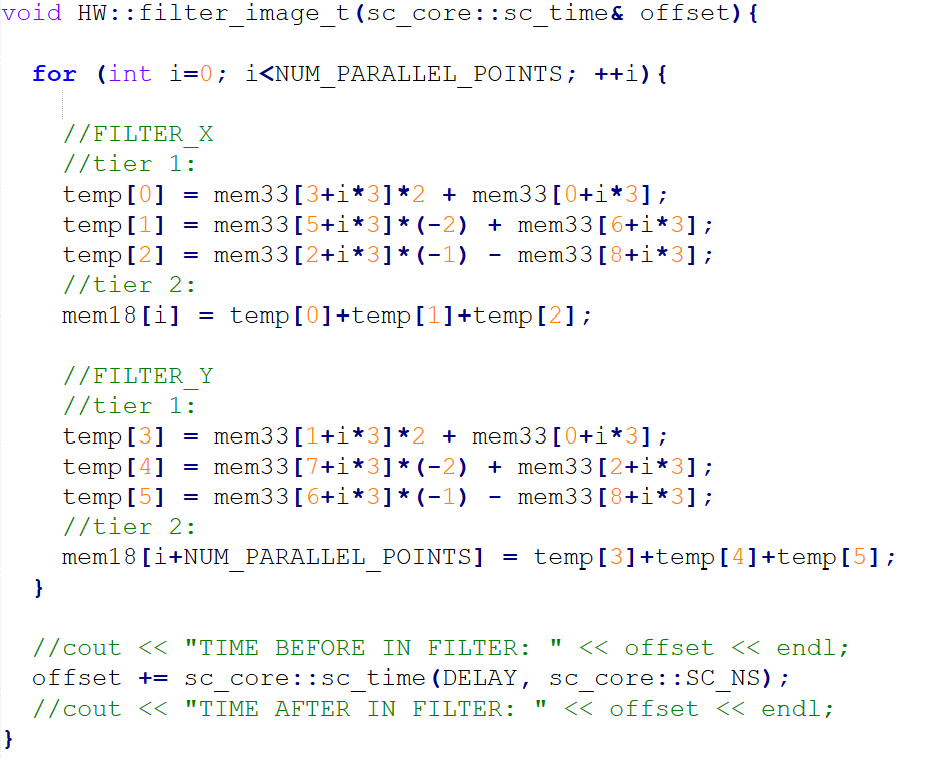
U softverskom modulu se učitava i priprema slika za obradu do filter transformacije. Pripremljena slika se smješta u DRAM zbog svoje veličine, a iz DRAM-a se smiješta maksimalan broj podataka koji može da stane u BRAM. Kada se obrade svi pikseli iz BRAM-a, pozove se ponovo incijalizacija i doprema se nova tura piksela koji će biti obrađeni. Postupak se ponavlja sve dok ima neobrađenih piksela u DRAM-u. Prije nego što hardver započne transformaciju, konfigurišu se registri u Filter IP Core-u koji su neophodni za proces izvršavanja. Podizanjem *start* bita počinje sa radom Filter IP Core. U prvoj fazi pikseli prebačeni iz BRAM-a se obrađuju putem pipelajnovanih MAC operacija. Nakon te faze se obrađeni pikseli iz registra prebacuju u DRAM. Nakon što je obrada u IP jezgru završena, podizanjem *ready* bita obavještava se softver da može poslati nove podatke za obradu.

1. **Filter Image IP Core modul**

| **Naziv registra** | **Adresni offset** | **Širina registra** | **Opis** |
| --- | --- | --- | --- |
| **picture\_width** | **0x00** | **10 bita** | **širina slike** |
| **picture\_height** | **0x02** | **11 bita** | **visina slike** |
| **start** | **0x04** | **1 bit** | **vrijednost 1 - početak rada jezgra**  **vrijednost 0 - modul je započeo obradu ili čeka komandu** |
| **ready** | **0x05** | **1 bit** | **vrijednost 1 - modul spreman za obradu**  **vrijednost 0 - moduli nije završio obradu** |
| **accumulated\_loss** | **0x06** | **11 bita** | **gubitak koji se javlja prilikom inicijalizacije zbog preskočenih BRAM blokova** |

Tabela 5 – Registarska mapa

Dio modula koji je izabran za ubrzavanje prikazan je na *Slici 5.1*. Konfiguriše se iz softvera popunjavanjem odgovarajućih registara prikazanih u *Tabeli 5* i setovanjem *start* bita. *Ready* registar se koristi za provjer završetka obrade od strane softverske komponente.



*Slika 5.1 – Kod za Filter Image IP Core modul*

Ovaj modul je povezan sa DRAM i BRAM memorijom. Iz BRAM memorije se uzimaju pikseli koji su potrebni za obradu slike (paralelno se uzima 8 piksela za jedan filter i 8 piksela za drugi filter), dok se u DRAM pohranjuju obrađeni pikseli (analogno, pohranjuje se 16 piksela paralelno). U oba slučaja se radi o 16-bitnim vrijednostima.

1. **Performanse Sistema**

Filter Image IP Core modul sadrži operacije množenja, sabiranja i oduzimanja što je pogodno za mapiranje na DSP module na FPGA ploči. Nakon analize najduže kombinacione putanje unutar modula, procjenjena frekvencija na kojoj sistem može da radi je **133.33 MHz**. (nedovršeno)