SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 80

IZVEDBA ALGORITAMA RAČUNALNOG VIDA ZA UGRADBENE SUSTAVE

Anđelo Martinović

Zagreb, lipanj 2010.

Zahvala g. Davoru Kovačecu, direktoru tvrtke Xylon d.o.o., na ustupljenoj LogiTAP razvojnoj platformi, bez koje ovaj rad ne bi bio moguć.

Sadržaj

1. Uvod

U današnjim integriranim sustavima računalnog vida koriste se digitalne video kamere ili slični uređaji kako bi pretvorili upadnu svjetlost u informaciju koju se može obrađivati. Postupkom uzorkovanja u video senzoru dobiva se izlazna informacija u obliku pravokutne rešetke. Svaki element te rešetke naziva se "slikovni element" ili "piksel" (skraćeno od eng. *picture element*). Ako promatramo samo slike sivih razina, vrijednost svakog piksela bit će proporcionalna jačini upadne svjetlosti.

Današnji trendovi u obradi slike očituju se u povećavanju rezolucije kamera, čime se direktno utječe na količinu informacija koju je potrebno obraditi. Primjerice, standardna VGA rezolucija iznosi 640x480=307 200 piksela, dok današnje Full HD kamere imaju rezoluciju od 1920*1080=2 073 600 piksela. Povećanjem količine informacija nužno se povećava i vrijeme obrade slike, što naročito dolazi do izražaja kod obrade video signala u realnom vremenu (25-30 slika u sekundi).

Obrada slike može se podijeliti u četiri faze [1]: *pretprocesiranje, segmentacija, ekstrakcija značajki* i *prepoznavanje*. Pretprocesiranje povećava kvalitetu slike uklanjanjem nepotrebnih informacija, uklanjanjem šuma, normalizacijom svjetline itd. U fazi segmentacije objekti se odvajaju od pozadine i jedni od drugih. Ekstrakcija značajki se provodi nad svakim objektom kako bi se smanjila količina informacija na reprezentativnu listu, tj. vektor značajki. Konačno, u fazi prepoznavanja, određuje se semantika dobivenih vektora značajki, njihovo značenje u kontekstu računalnog vida.

Iz prethodnog odjeljka vidljivo je da faze niže razine zahtijevaju da se obradi velika količina informacija koristeći jednostavne operacije (npr. usporedba svakog piksela s nekom vrijednošću), dok svaka od narednih faza zahtijeva sve manje podataka, ali koristi zahtjevnije i sofisticiranije algoritme. Dizajneri sustava računalnog vida su iz ovog razloga imali tendenciju implementirati niže razine procesiranja u sklopovlju. Reprezentativni primjer tog trenda je procesor NOA u kartici Genesis tvrtke Matrox [2].

S druge strane, tržište procesora već duže vrijeme prati tzv. Mooreov zakon koji kaže da se prosječan broj tranzistora u integriranim sklopovima udvostručava svakih 18 mjeseci (u početku je Moore postavio period udvostručenja na dvije godine, no ubrzo nakon toga zakon je korigiran) [3]. No, dok su procesori opće namjene kroz devedesete godine prošlog stoljeća samo povećavali brzinu i frekvenciju rada, trendovi zadnjih deset

godina pokazuju tendenciju paralelizacije operacija kroz povećanje broja jezgara u procesorima [4].

Kod implementacije integriranog sustava postoji više mogućnosti odabira konačne implementacijske platforme: ASIC (eng. *Application Specific Integrated Circuit*), aplikacijsko specifični standardni procesori (ASSP), procesori za digitalnu obradu signala (DSP) i FPGA (eng. *Field Programmable Gate Array*).

Od navedenih sklopova svakako treba izdvojiti FPGA uređaje, budući da postaju sve zanimljiviji širem spektru korisnika. Njihova moć procesiranja raste čak i brže od procesora opće namjene zbog osjetnijeg povećanja gustoće tranzistora [5]. Ovi programirljivi uređaji mogu se koristiti kao jednostavni elementi za izračun operacija niže razine zbog inherentnog paralelizma arhitekture, no imaju i potencijala za implementaciju puno sofisticiranijih postupaka. Velika gustoća logičkih vrata i mogućnost "*in-system*" programiranja (ISP) uz niske cijene omogućuju programerima korištenje FPGA sustava kao korisne platforme za brzi razvoj prototipova (eng. *rapid prototyping*) [6].

Mogućnost rekonfiguracije FPGA sklopovlja omogućava njihovu primjenu na širokom spektru operacija, kao što su operacija konvolucije [7], 2D Fourierova transformacija [8], praćenje točaka u realnom vremenu [9], implementacija stereo sustava računalnog vida [10], praćenje i brojanje ljudi [11] itd.

1.1. Integrirani sustav

U ovom radu razvijen je sustav računalnog vida za obradu video signala u realnom vremenu uz korištenje FPGA platforme. Sustav podržava veći broj algoritama računalnog vida izvedenih u sklopovlju uz zadržavanje programabilnosti.

U drugom poglavlju opisana su osnovna počela ugradbenih računalnih sustava, njihova podjela i način razvoja. Poseban naglasak stavljen je na FPGA sustave i njihovu primjenu, kao što je implementacija procesora s mekom jezgrom.

Postupak obrade slike započinje s njezinim dohvatom. Treće poglavlje opisuje kako se iz video signala u realnom vremenu dohvaća informacija o slici te prikazuje način generiranja signala za sinkronizaciju videa.

U poglavlju 4 opisani su neki od osnovnih algoritama računalnog vida, kao što su binarizacija, filtriranje i detekcija rubova, a u petom poglavlju je opisan način njihove izvedbe u sklopovlju. Šesto poglavlje bavi se generiranjem video izlaza, problemima sa sinkronizacijom i mogućnostima ulančavanja više jedinica za obradu.

2. Ugradbeni računalni sustavi

Ugradbeni računalni sustav obično se definira kao računalni sustav koji provodi jednu ili manji broj funkcija, često uz zahtjeve za performanse u realnom vremenu. Obično se ugrađuje u dio većeg uređaja koji sadrži sklopovlje i mehaničke dijelove. S druge strane, računala opće namjene (npr. osobna računala) dizajnirana su s ciljem fleksibilnosti i zadovoljavanja najrazličitijih korisničkih zahtjeva.

Ugradbena računala pogoni jedna ili više procesorskih jezgri, koje su najčešće mikrokontroleri ili DSP procesori. Razlika između mikrokontrolera i mikroprocesora nije potpuno definirana, iako mikrokontroleri u većini slučajeva osim centralne procesne jedinice sadrže ROM, RAM i periferije, dok mikroprocesori takve elemente ne posjeduju, osim eventualno manje količine priručne (eng. *cache*) memorije. Granica između termina "mikroprocesor" i "procesor" obično se povezuje sa širinom podataka koje procesor obrađuje ili s veličinom sabirnica u procesoru, a obično se postavlja na 16 bita [12].

Kod složenijih ugradbenih računala često se teži implementirati što je moguće veći dio elektroničkog sustava u jednom integriranom čipu. Takvi sustavi nazivaju se SoC (eng. *System on Chip*) i mogu sadržavati više procesora, memorija i različitih sučelja na jednom čipu. Najčešće korištene tehnologije za razvoj SoC sustava su ASIC i FPGA.

2.1. ASIC i FPGA

ASIC je integrirani sklop prilagođen za konkretnu ulogu, npr. čip za govor u dječjim igračkama. Prvi ASIC čipovi koristili su tzv. GA (eng. *Gate Array*) tehnologiju. GA je unaprijed proizvedena silicijska matrica koja "čeka" naručiteljev uzorak za povezivanje da bi postala funkcionalna. Povezivanje tranzistora i logičkih vrata u složenije funkcije izvodilo se na kraju proizvodnog procesa. Nedostatak ovog pristupa bio je dug razvojni ciklus i skupe promjene u logici. Osim GA tehnologije, postoji i “Full Custom” princip razvoja, u kojem se čipovi dizajniraju "od nule". Ovaj pristup svakako nudi najveću fleksibilnost, ali i daleko najveće nepovratne troškove - NRE (eng. *Non-Recurring Expense*). Daljnjim razvojem ASIC čipova pojavile su se tzv. standardne ćelije. Svaki proizvođač ASIC čipova bio je u mogućnosti kreirati funkcionalne blokove s poznatim karakteristikama, a zatim su se takve ćelije kombinirale u dizajnu uređaja. Ovaj

pristup nalazi se između GA tehnologije i Full Custom pristupa, kako po NRE trošku, tako i po trošku pojedine komponente.

Pojavom PLD uređaja (eng. *Programmable Logic Devices*) ASIC komponente su doživjele napredak u smislu kraćeg vremena oblikovanja i mogućnosti dizajniranja sklopova s manjim nepovratnim troškovima. PLD uređaji se mogu konfigurirati tako da se dobije komponenta za specifičnu aplikaciju, no ne rabe se posebno oblikovane maske ili ćelije, već se pomoću velikog broja programirljivih međuveza spajaju matrice logičkih makroćelija. Makroćelije se obično sastoje od neke vrste programirljivog logičkog polja kao što je PAL (eng. *Programmable Array Logic*), bistabila i zapornih sklopova.

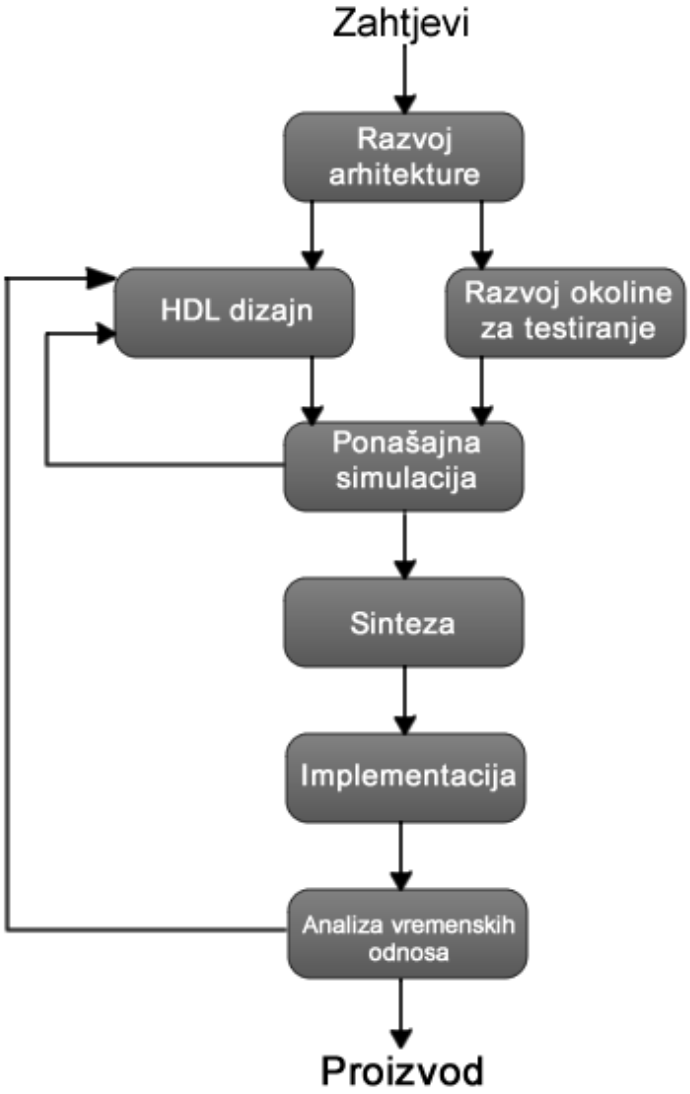
Kombiniranjem više PLD sklopova uz pomoć prospojnih matrica i uz dodatak blokova za ulaz/izlaz stvoreni su Complex PLD sklopovi (CPLD). Kod CPLD uređaja prisutna je mogućnost programiranja čipa dok je priključen na tiskanoj ploči. Ovaj postupak naziva se ISP (eng. *In-System Programming*) a izvodi se prijenosom "programa" s računala preko posebnog kabela. Najčešće se radi o tzv. JTAG (eng. *Joint Test Action Group*) sučelju.

Godine 1985. osnivači tvrtke Xilinx izumili su prvo komercijalno uspješno programirljivo polje ćelija (FPGA): XC2064 [13]. Navedeni uređaj imao je programirljive ćelije i programirljive prospojne matrice između ćelija. Upravo ta programirljiva prospojna matrica je ono što razlikuje FPGA uređaje od CPLD uređaja. Zbog toga su FPGA uređaji fleksibilniji u smislu mogućnosti implementacije raznolikih dizajna, ali ujedno i kompleksniji za razvoj. Današnji FPGA uređaji u sebi imaju integrirane i funkcije viših razina u vidu namjenskog sklopovlja kao što su zbrajala, množila i ugrađene memorije.

Povijesno gledano, FPGA uređaji su bili sporiji, manje energetski učinkoviti i imali manju funkcionalnost od njihovih ASIC kopija. Razvojem tehnologije izrade, uvođenjem tehnoloških novosti, povećanjem gustoće logičkih vrata i ulazno/izlaznih mogućnosti, razlika između dviju tehnologija sve više se smanjuje. Danas FPGA uređaji nude niz prednosti, kao što su kraće vrijeme izrade (eng. *Time to Market*), mogućnost reprogramiranja nakon završetka proizvodnog procesa u svrhu otklanjanja grešaka i manje nepovratne troškove [14]. Proizvođači se mogu odlučiti i na kompromis: razviti dizajn na FPGA uređaju te proizvesti konačni sklop tako da se dizajn više ne može mijenjati.

2.2. Proces razvoja

Kod razvoja složenih digitalnih sustava, postupak se može podijeliti u više faza. Na početku se analiziraju zahtjevi projekta, izvodi dekompozicija problema i provodi funkcijska simulacija, ukoliko je moguća. Rezultat prve faze je dokument koji opisuje buduću arhitekturu sustava, strukturne blokove te njihove funkcionalnosti i sučelja.



Slika 1. Proces razvoja digitalnog sustava

Druga faza se sastoji od dvije paralelne aktivnosti: jedan tim razvija arhitekturu sustava koristeći neki od jezika za opis sklopovlja, kao što su VHDL i Verilog. Taj tim stvara takozvani RTL (eng. *Register Transfer Level*) model, koji se može implementirati u ciljanoj tehnologiji. Drugi tim paralelno stvara modele ponašanja i funkcijske simulacije koristeći jezike za opis sklopovlja i specifične alate za simulaciju, kao što je

npr. ModelSim tvrtke Mentor Graphics. Svrha ispitne okoline je verifikacija razvijenog HDL modela.

U trećoj fazi provodi se ponašajna simulacija razvijenog sklopa na temelju RTL modela i ispitne okoline. Odziv RTL modela uspoređuje se s očekivanim odzivom, te se model korigira ukoliko su primijećena neprihvatljiva odstupanja.

Četvrta faza jest logička sinteza. Ona podrazumijeva pretvaranje RTL modela u listu povezanih standardnih logičkih sklopova (eng. *netlist*). U ovu svrhu obično se koristi baza sklopova koju daje proizvođač ciljne tehnologije. Ti sklopovi su najčešće jednostavni logički elementi, poput invertora, I/ILI vrata, bistabila, multipleksora i slično. Logičku sintezu provodi posebni programski alat. Neki od poznatijih sintetizatora su Design Compiler tvrtke Sinopsis (za ASIC) i Xilinx Synthesis Tool (XST) tvrtke Xilinx.

Nakon sinteze provodi se faza implementacije, koja se obično dijeli u više podfaza:

1. Mapiranje liste na ciljanu platformu (eng. *mapping*)
2. Razmještanje elemenata iz liste (eng. *placing*)
3. Električko povezivanje elemenata (eng. *routing*)

Mapiranje prilagođava listu konkretnom sklopovlju na kojem se radi implementacija. U fazi razmještanja optimira se duljina vodova, vremena propagacije signala i utrošena površina. Povezivanje stvara "žice" koje povezuju elemente liste. Konačan rezultat implementacije će, ovisno o tehnologiji, biti ili maska za proizvodnju integriranog sklopa (ASIC) ili datoteka s konfiguracijom sklopa (eng. *bitstream*).

Konačno, provodi se vremenska simulacija sklopa. Tijekom vremenske analize, program za simulaciju provjerava zadovoljava li razvijeni sklop korisnički zadana vremenska ograničenja, kao što je npr. frekvencija takta.

2.3. VHDL

Godine 1981., na inicijativu Ministarstva obrane Sjedinjenih Američkih Država, započeo je razvoj specijaliziranog jezika za opis sklopovlja. Ova inicijativa je bila primarno motivirana činjenicom da su tadašnji proizvođači ASIC sklopovlja imali nestandardne načine dokumentiranja ponašanja njihovih sklopova. Svaki uređaj je posjedovao velike i kompleksne priručnike koji su opisivali ne samo sučelja, već često i način same implementacije uređaja. Tvrtke IBM, Texas Instruments i Intermetrics su tako 1983. godine započeli s projektom izrade prve verzije VHDL jezika (od eng. *VHSIC Hardware Description Language; VHSIC- Very High Speed Integrated Circuits*).

Danas postoje dva standarda VHDL jezika, imenovana po godini kad je IEEE (*Institute of Electrical and Electronic Engineeers*) objavio standarde: VHDL87 i VHDL93.

Osnovne značajke jezika VHDL su [15]:

1. VHDL je jezik za oblikovanje i opisivanje sklopovlja
    - Kako su komponente u digitalnom sustavu istovremeno aktivne, VHDL mora podržavati konkurentnost obavljanja zadataka
2. Potpora hijerarhijskom oblikovanju
    - Komponenta se može opisati funkcijski ili strukturno, pomoću komponenti niže razine
3. Potpora bibliotekama
4. Sekvencijske naredbe
    - Za izvođenje sekvencijalnih naredbi sličnih programskom kodu (petlje, uvjeti…)
5. Mogućnost generiranja generičkog opisa komponenti
6. Potpora različitih tipova podataka
7. Uporaba potprograma
8. Upravljanje vremenskim vođenjem
9. Strukturna dekompozicija sklopovlja na svim razinama

Glavna prednost VHDLa u kontekstu dizajna digitalnih sustava jest da omogućuje opisivanje i verifikaciju ponašanja željenog sustava prije nego što ga alati za sintezu prevedu u stvarno sklopovlje. U VHDLu je moguće modelirati ispitno okruženje (eng. *testbench*) za verifikaciju sklopa, u kojem se definiraju pobudni signali, interagira s komponentom koja se testira (eng. *UUT-Unit Under Test*), te se provjerava ispravnost izlaza.

2.4. Xilinx FPGA

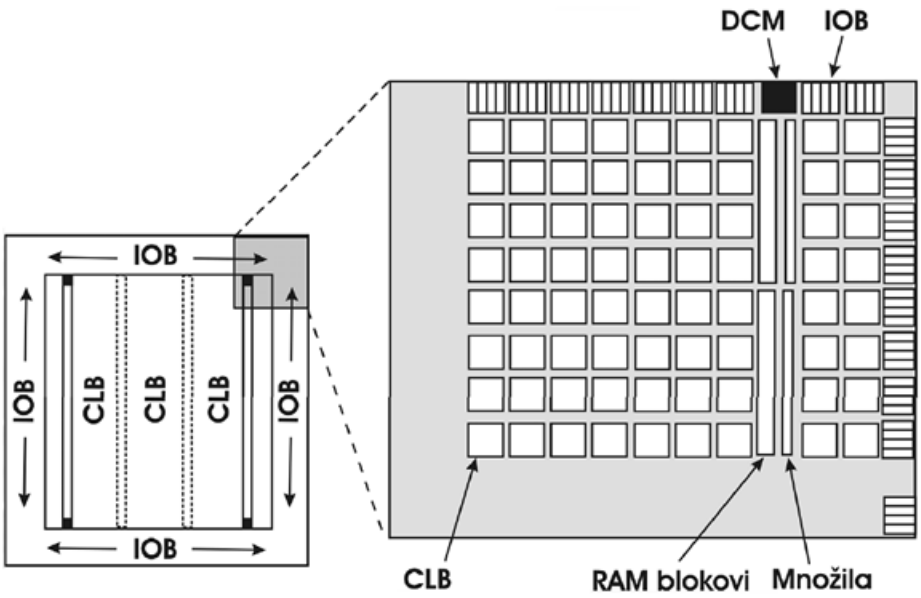
Glavni proizvođači FPGA sklopova u trenutku pisanja ovog teksta su Xilinx i Altera. Te dvije organizacije kontroliraju preko 80% tržišta, a sam Xilinx zauzima preko 50% [16]. Glavni proizvodi Altere su Cyclone, Arria GX i Stratix FPGA uređaji, dok su dvije glavne porodice Xilinxovih uređaja Virtex i Spartan. Kako je u ovom radu korišten Spartan-3 FPGA, u daljnjem tekstu bit će dan kratki pregled mogućnosti te porodice čipova.

2.4.1. Spartan-3

Spartan-3 porodica FPGA uređaja predstavljena je 2003. godine. Radilo se o prvom 90-nanometarskom FPGA čipu na svijetu koji je spadao u niži cjenovni rang. Projektirana je tako da ima što širu primjenu, uz što je moguće nižu proizvodnu cijenu. Predstavljeno je više serija čipova: Spartan-3A, s brzim ulazom/izlazom i slabijim logičkim mogućnostima, Spartan-3E, za logički zahtjevnije aplikacije, te Spartan-3 koji je nudio najbolje od 3A i 3E serije.

Spartan-3 porodica čipova nudi visoke performanse: na čipu se nalazi i do 5 milijuna logičkih vrata, uz maksimalni period takta od 326 MHz. Podržani su logički blokovi s pomičnim registrima, multipleksori, brza logika prijenosa i 18x18 bitna množila. Od memorije je dostupno do 1872 Kb blok memorije i 520 Kb raspodijeljene memorije.

Građa sklopa Spartan-3 prikazana je na slici 2.



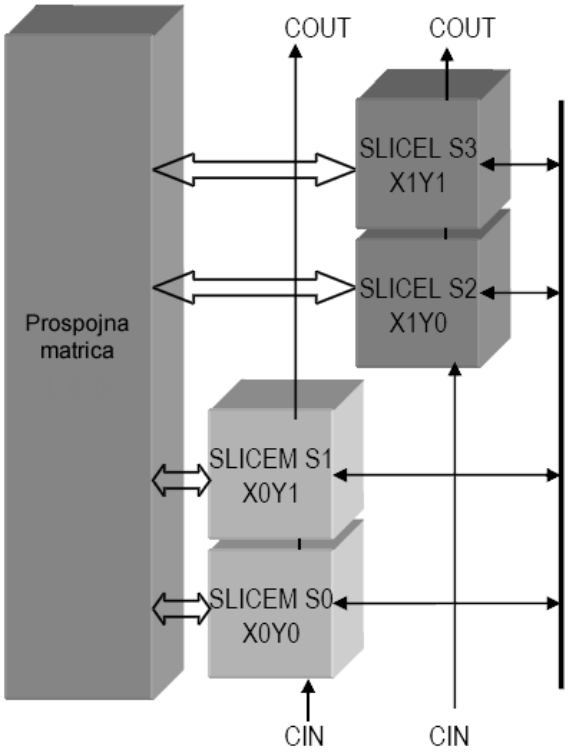
Slika 2. Građa Spartan-3 FPGA sklopa

Sklop se sastoji od sljedećih elemenata:

- CLB – konfigurabilni logički blok (eng. *Configurable Logic Block*)
- IOB – ulazno-izlazni blok (eng. *Input Output Block*)
- Blok RAM – konfigurabilna RAM memorija
- Množila opće namjene
- DCM – sklop za sintezu takta (eng. *Digital Clock Manager*)

- Mreža za povezivanje
- Logika za konfiguriranje

Blokovska shema CLB-a prikazana je na slici 3. Svaki CLB sastoji se od 4 odsječka (eng. *slice*), od čega su dva tipa SLICEM, a dva SLICEL. Razlika između dvaju tipova odsječaka jest da SLICEL odsječci mogu biti programirani da provode jednostavne logičke operacije ili da služe kao ROM memorija, dok SLICEM odsječci dodatno omogućavaju da budu konfigurirani kao distribuirani RAM ili kao posmačni registri.



Slika 3. Arhitektura CLB elementa

IOB blokovi kontroliraju protok podataka između ulazno/izlaznih pinova i unutarnje logike. Bitno je primijetiti da iako postoje različita pakiranja za isti tip FPGA uređaja, od kojih neki imaju manje pinova od drugih, broj IOB blokova za sve FPGA sklopove istog tipa je jednak. Stoga se neiskorišteni IOB blokovi svejedno mogu koristiti u dizajnu (npr. njihovi bistabili). IOB blokovi grupirani su u 8 banaka priključaka, i unutar svake banke priključci se konfiguriraju na isti način. Svaki IOB podržava dvosmjerni protok podataka i stanje visoke impedancije (eng. *Tri-State*). Podržane su 24 različite naponske razine na priključcima, od čega je 6 diferencijalnih standarda.

Blok-RAM memorije koriste se kad je u dizajnu potreban RAM većeg kapaciteta, a izvedene su kao nezavisne jedinice na sklopu FPGA čipa. Ovisno o tipu FPGA, na čipu može biti od 4 do preko 100 Blok RAM-ova, a svaki Blok RAM ima kapacitet od 18K bitova. Svaki Blok RAM može se konfigurirati da bude jednopristupni (eng. *single port*) ili dvopristupni (eng. *dual port*). Sintetizator može automatski iz VHDL koda prepoznati radi li se o Blok RAM-u ili o distribuiranoj RAM memoriji, pa je pri pisanju modela preporučeno koristiti predloške (eng. *language templates*).

Množila su također izvedena kao fizički nezavisne jedinice, ima ih od 4 do preko 100 ovisno o izabranom tipu uređaja i podržavaju predznačno množenje dva operanda od 18 bitova, s rezultatom od 36 bita.

DCM sklop služi za sintezu takta željene frekvencije, eliminaciju fazne greške takta i pomak faze takta.

Povezivanje svih elemenata ostvareno je bogatom programirljivom prospojnom mrežom. Konfiguracija sklopa provodi se učitavanjem konfiguracijskih podataka u statičku memoriju koja kontrolira funkcionalnost elemenata na sklopu i njihovu međusobnu povezanost. Kako su svi konfiguracijski elementi u osnovi SRAM memorije, sklop ne pamti konfiguraciju bez napajanja.

2.5. Izvedbe procesora

Svaki ugradbeni računalni sustav sadrži neku vrstu procesora koji vrši upravljanje i kontrolu. Postoje tri izvedbe takvog procesora:

1. Diskretni procesor
2. Procesori s tvrdom jezgrom
3. Procesori s mekom jezgrom

Diskretni procesor (eng. *off the shelf*) je zasebna komponenta, obično izvedena u ASIC tehnologiji. Postoji veliki izbor različitih proizvođača i funkcionalnosti sklopova, pa ispravan odabir odgovarajućeg procesora može biti nezahvalan zadatak.

Procesori s tvrdom jezgrom (eng. *hard processor core*) su posebni namjenski sklopovi unutar postojeće komponente (npr. u FPGA uređaju). Budući da su fiksno ožičeni, mogu raditi na viskom brzinama, ali im nedostaje mogućnost prilagodbe. No, kako im je okruženje u većini slučajeva rekonfigurabilno, okolina se može prilagoditi procesoru. Ovakva izvedba nije česta na tržištu, te postoji ograničen broj proizvođača i porodica FPGA sklopova koji podržavaju ovu tehnologiju.

Procesor s mekom jezgrom (eng. S*oft-Core Processor,* u daljnjem tekstu SCP) je onaj koji je potpuno implementiran u logičkim primitivima rekonfigurabilnog hardvera kao što je FPGA. Zbog svoje izvedbe, takav procesor u pravilu ne postiže brzine diskretnog ili procesora s tvrdom jezgrom. No, u velikom broju primjena ugradbenih sustava iznimne performanse nisu potrebne, već se iskorištava povećana funkcionalnost i fleksibilnost procesora s mekom jezgrom. Primjerice, mogu se koristiti u jednostavnim sustavima gdje je potrebna samo manipulacija s U/I uređajima, ali nekad se implementiraju i složenim sustavima koji pogone operacijski sustav uz korištenje sučelja poput Etherneta, PCI, DDR memorije i ostalih perifernih sklopova.

2.5.1. Procesori s mekom jezgrom

Razlozi korištenja procesora s mekom jezgrom mnogobrojni su. Već je spomenuta iznimna mogućnost fleksibilnosti kroz rekonfiguraciju FPGA uređaja. Ova prednost naročito dolazi do izražaja kod problema mijenjajućih zahtjeva. Idealni proces razvoja proizvoda, u kojem se svi zahtjevi identificiraju i analiziraju prije dizajniranja sustava rijetko se sreće u praksi. Zahtjevi se mijenjaju tijekom procesa dizajna, a često se dodaju i potpuno novi zahtjevi. SCP olakšava posao dizajnerima sustava jer ih ne ograničava na točno određeni skup funkcija i perifernih sklopova izabranih npr. kupnjom diskretnog procesora. U mnogim razvojnim sustavima za SCP-e moguće je dodati i konfigurirati periferni sklop kroz nekoliko kratkih koraka. Osim toga, postojanje široke palete već gotovih periferija omogućava komponentni dizajn sustava. Tipični periferni sklopovi dostupni od širokog broja proizvođača uključuju memorijske kontrolere, brojače, GPIO (eng. *General Purpose Input Output*), UART sklopove (eng. *Universal Asynchronous Receiver Transmitter*) i razne sabirnice kao što su PCI, RapidIO i HyperTransport. SCP i periferije dizajniraju se u jezicima za opis sklopovlja, što olakšava njihovu prilagodbu.

Za Xilinx FPGA sklopove razvijen je veliki broj SCP arhitektura, kao što su 8051, Z80, PIC, ATMEL i druge. Sam Xilinx razvio je dva SCP-a koji su optimizirani za Xilinx arhitekturu: PicoBlaze i MicroBlaze.

PicoBlaze je 8-bitni RISC mikroprocesor koji ima 8-bitnu adresnu i podatkovnu sabirnicu za pristup velikom broju periferija. Procesor je besplatan za korištenje zajedno s razvojnim alatima, iako radi samo na Xilinx FPGA uređajima. Dizajn procesora zahtijeva samo 96 odsječaka (*slice*) na Spartan-3 sklopu, što ga čini idealnim kontrolerom za jednostavnije primjene.

2.5.2. Microblaze

Microblaze je 32-bitni RISC procesor s mekom jezgrom razvijen za Xilinx FPGA sklopove. Mnogi aspekti procesora mogu biti konfigurirani: veličina priručne memorije, dubina protočnog sustava (3 ili 5), ugrađene periferije, MMU (eng. *Memory Management Unit*) i sabirnička sučelja. Primjerice, postoji prostorno optimizirana verzija Microblaze procesora, koja koristi 3-stupanjsku protočnu arhitekturu i žrtvuje frekvenciju takta za smanjenu potrošnju resursa na FPGA sklopu. S druge strane, verzija optimizirana za brzinu povećava protočnu arhitekturu na 5 stupnjeva i omogućava brzine do 210MHz na Virtex-5 FPGA sklopovima. Također je moguće selektivno isključiti procesorske instrukcije koje su sklopovski zahtjevne (npr. množenje, dijeljenje, rad s pomičnim zarezom). Uz korištenje jedinice za gospodarenje memorijom (MMU), podržano je izvođenje operacijskih sustava koji zahtijevaju sklopovsko zasnovano straničenje i zaštitu memorije, kao što je Linux kernel. Ako se MMU ne koristi, moguće je pokretanje jednostavnijih sustava, kao što je FreeRTOS.

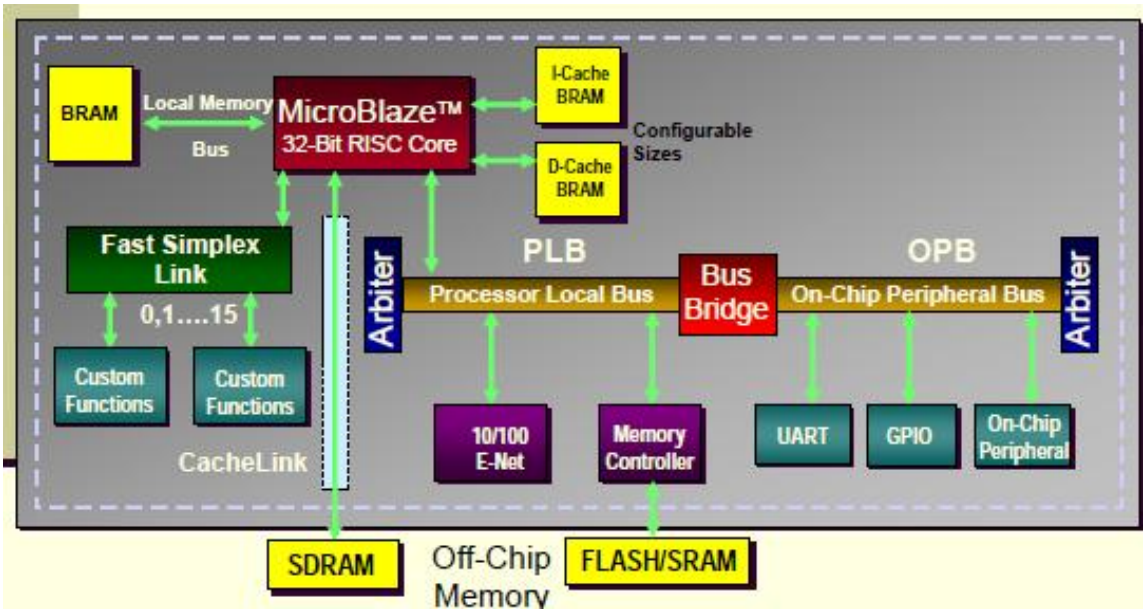
2.5.2.1. Sabirnički podsustav

Kao i u svakom sabirničkom sustavu, procesor i periferije komuniciraju preko sabirnica. U kontekstu sabirničkog protokola, svaka periferija može biti voditelj (eng. *master*), prateći (eng. *slave*), arbitar (eng. *arbiter*) ili most (eng. *bridge*).

Voditelj inicira komunikaciju na sabirnici, dok prateći može samo odgovarati na upite na sabirnici. Arbitar je sklop koji odlučuje kojem od vodećih sklopova će biti dodijeljeno upravljanje sabirnicom ukoliko više takvih sklopova šalje zahtjev za upravljanjem. Mehanizmi određivanja prioriteta mogu biti određeni fiksnim razinama prioriteta, *round-robin* algoritmom ili sličnim postupcima.

Sabirnička arhitektura korištena u sustavima s Microblaze procesorom je IBM CoreConnect. Navedeni standard definira tri sabirnice za spajanje uređaja, makro biblioteke i specijaliziranu logiku:

- Processor Local Bus (PLB)
- On-Chip Peripheral Bus (OPB)
- Device Control Register (DCR)



Slika 4. Microblaze procesorski sustav

Uređaji koji zahtijevaju visoke performanse spajaju se na PLB sabirnicu, koja nudi visoku propusnost (do 2.9 GB/s) i niske latencije. Sabirnica je sinkrona, s odvojenim sabirnicama za pisanje i čitanje, uz centraliziranu arbitražu. Koristi se 32-bitna adresa uređaja, dok podatkovna sabirnica može biti 32,64 ili 128-bitna.

Sabirnički PLB-OPB most prenosi PLB transakcije na OPB sabirnicu i obrnuto, a funkcionira kao prateći na PLB sabirnici i kao vodeći na OPB sabirnici.

OPB sabirnica predviđena je za sporije uređaje koji ne zahtijevaju veliku propusnost. Sabirnički protokol je jednostavniji od onog na PLB sabirnici: nema dijeljenja sabirničkih ciklusa ni protočne arhitekture. Adresna sabirnica je 32-bitna, kao i podatkovna sabirnica. Podržano je do 15 vodećih uređaja.

U Microblaze procesorskom sustavu postoji još nekoliko namjenskih sabirnica, prikazanih na slici 4. LMB sabirnica (eng. *Local Memory Bus*) omogućava procesoru jednociklusni pristup do Blok RAM memorije. Kako je arhitektura procesora harvardska, podatkovna memorija odvojena je od instrukcijske te stoga postoje dva sučelja: DLMB (podatkovno) i ILMB (instrukcijsko).

XCL (eng. *Xilinx Cache Link*) jednostavna je sabirnica za dohvat podataka u pričuvnoj memoriji, kako se podaci ne bi prenosili preko OPB ili PLB sabirnice. Može se spojiti na bilo koji memorijski upravljački sklop koji podržava FIFO memorije, kao što je MPMC (eng. *Multi-Port Memory Controller*).

FSL (eng. *Fast Simplex Link*) predstavlja FIFO sabirnicu za jednosmjernu komunikaciju od točke do točke. Najčešće se koristi kao koprocesorska sabirnica, kako bi se veća količina podataka prenijela na obradu koprocesoru, te s time rasteretio glavni procesor. Sabirnica je zasebna za svaki par uređaja koji je koriste i optimizirana za veliku brzinu, do 600 MHz. Širina sabirnice je 32-bitna, a dubina FIFO memorije može iznositi od 1 do 8193 mjesta. Postoje posebne naredbe u Microblaze asemblerskom i C kodu za prijenos podataka preko FSL sabirnice.

2.6. Razvojne platforme

Proizvođači FPGA sklopova često na tržištu nude tzv. razvojne platforme (eng. *development board, development kit*). U načelu, radi se o sustavima za evaluaciju i razvoj sustava pogonjenih FPGA čipovima, a sastoje se od jedne ili više tiskanih pločica na kojima se osim samog FPGA čipa tipično nalazi i izvor napajanja, generator takta, vanjske ROM i RAM memorije, razna U/I sučelja (PCI, UART, USB, Ethernet), LED diode i slično. Namjena ovih sklopova je demonstrirati što je moguće veći skup značajki i primjena primarnog proizvoda, te olakšati programerima i inženjerima projektiranje sustava zasnovanih na FPGA čipovima.

Naravno, nije nužno da razvojne platforme kreiraju isključivo proizvođači FPGA čipova. Postoji niz tvrtki na tržištu koje nude svoje razvojne platforme, obično prilagođene za specifičnu primjenu ili određenu tržišnu nišu. Primjer takve razvojne platforme je logiTAP razvojna platforma tvrtke Xylon, koja je korištena u izradi ovog rada.

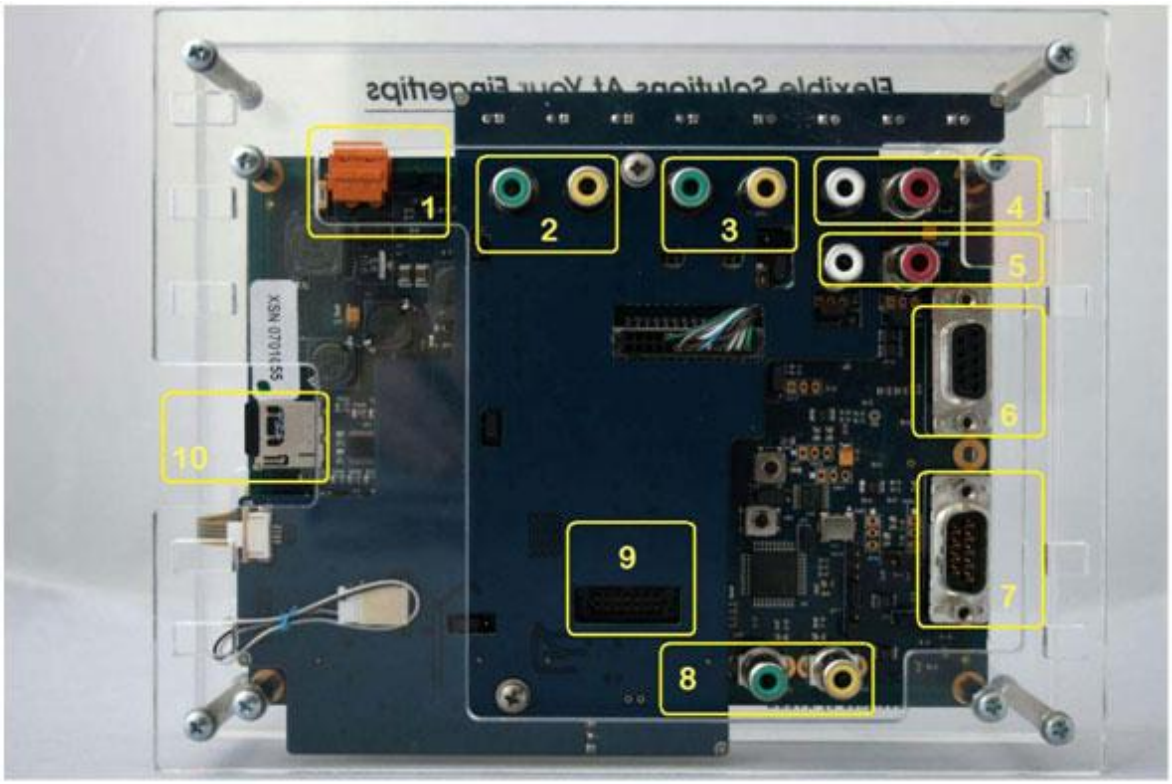
2.6.1. LogiTAP razvojna platforma

LogiTAP razvojna platforma zasnovana je na Xilinx Spartan-3E FPGA uređaju, a osnova joj je logiCRAFT3 razvojna ploča tvrtke Xylon.

Slika 5. LogiTAP razvojna platforma

Razvojna platforma upakirana je u kućište od pleksiglasa u kojem se nalaze razvojna ploča logiCRAFT3 i LCD ekran visoke rezolucije osjetljiv na dodir. Specifikacije ploče dane su u nastavku.

- Xilinx Spartan-3E XC3S1200E FPGA
- 7" 800x480 LCD zaslon u boji osjetljiv na dodir
- 32MB SDRAM memorije
- 8MB NOR Flash memorije
- 2GB SD Card Flash memorije
- 4 kompozitna i 2 S-Video ulaza
- LVDS izlaz za LCD
- 4 stereo audio ulaza, ulaz za mikrofon, stereo audio izlaz, 2 izlaza za slušalice, 2 wireless izlaza za slušalice
- RS232 sučelje



Slika 6. LogiTAP razvojna platforma: pogled straga

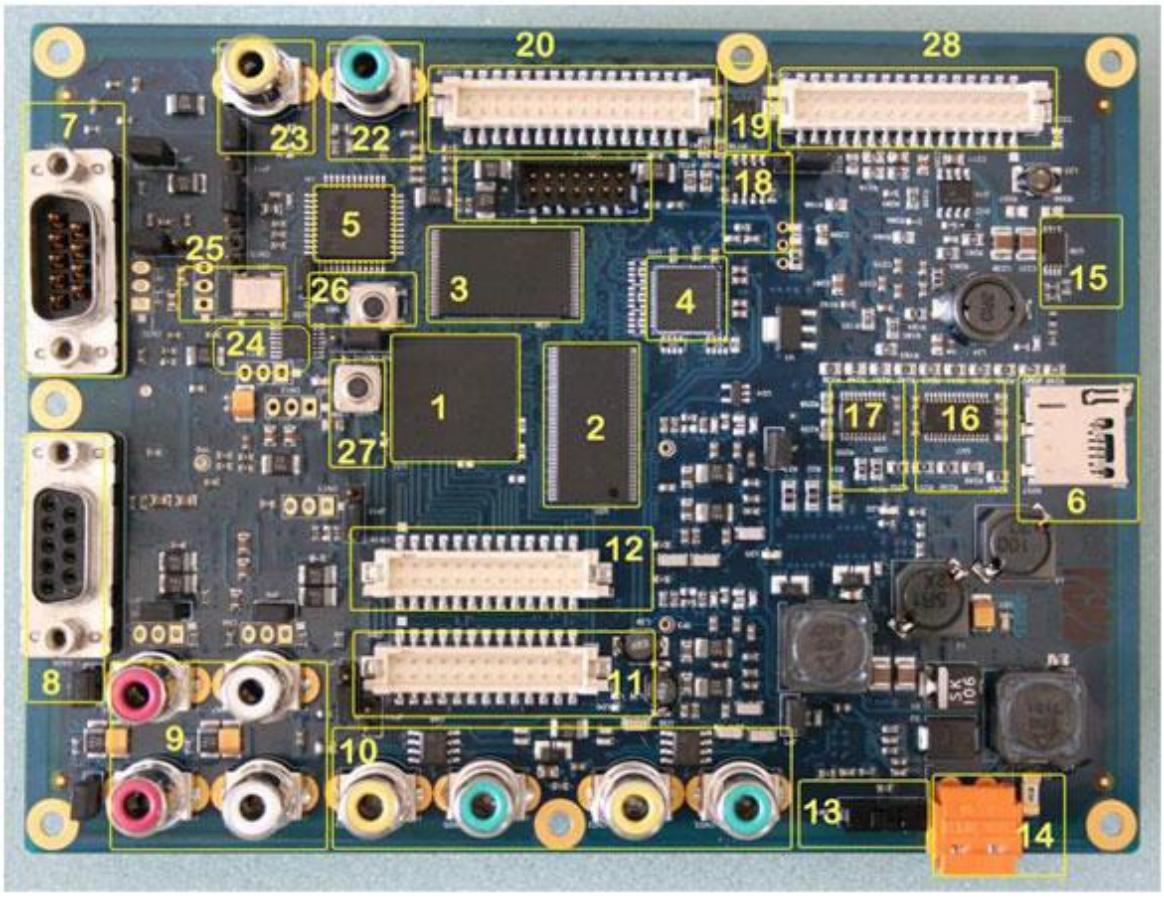
Ulazno izlazni priključci prikazani su na slici 6:

1. Napajanje
2. S-Video i kompozitni ulaz 1

3. S-Video i kompozitni ulaz 2
4. Audio ulaz 1
5. Audio ulaz 2
6. RS232 sučelje za debug
7. CAN
8. S-Video i kompozitni izlaz
9. JTAG
10. SD kartica

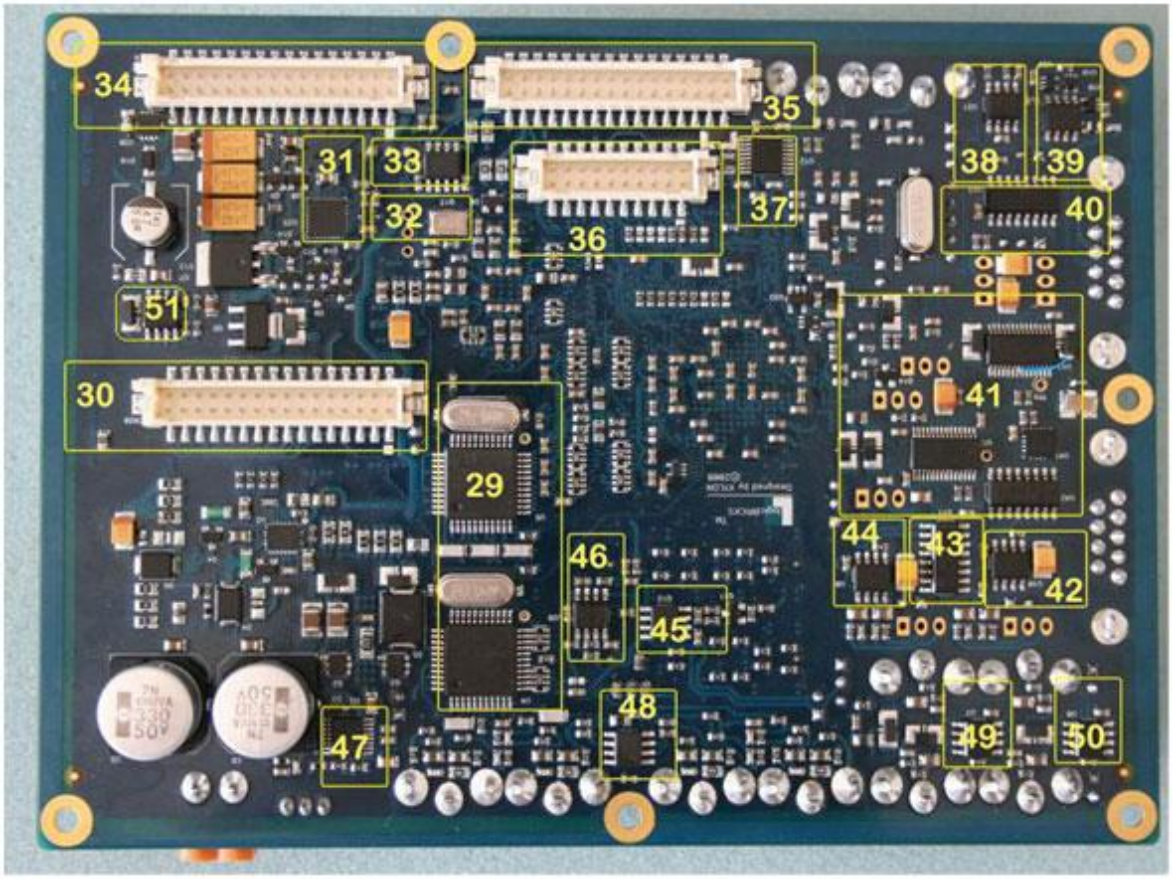
Referentni dizajn zasnovan je na Xilinx MicroBlaze procesoru koji se koristi kao mikrokontroler zadužen za nadzor svih funkcija sustava. Demonstracijski programi koji se izvršavaju na MicroBlaze procesoru nalaze se na SD kartici, te se mogu pokretati iz glavnog izbornika korištenjem ekrana osjetljivog na dodir.

Centralni dio logiTAP platforme je, kao što je već spomenuto, logiCRAFT3 tiskana pločica, prikazana na slici 7. U nastavku je dan opis svih bitnijih komponenti koje su korištene u stvaranju ovog rada.



Slika 7. LogiCRAFT3 razvojna ploča

- (1) FPGA uređaj XC3S1200E-5
    - Spartan-3E porodica, 8672 odsječka, 136Kbit DRAM, 504Kbit BRAM, 28 dediciranih množitelja, 8 DCM sklopova
    - FT256 pakiranje (eng. *Fine Pitch, Thin Ball Grid Array – FBGA*) sa 190 konfigurabilnih IOB blokova
- (2) 32-bit SDRAM, 32 MB
    - Glavna eksterna memorija iz koje se izvršavaju svi programi za Microblaze koji ne mogu stati u BRAM
- (3) Paralelni NOR Flash, 8MB
    - Izbrisiva perzistentna memorija u kojoj je pohranjena konfiguracija FPGA uređaja i prevedeni C program koji će se izvršavati na Microblaze procesoru nakon prebacivanja u SDRAM
- (5) Philips SAA7121H digitalni video enkoder
    - Čip koji obavlja D/A pretvorbu video signala (PAL, NTSC)
- (8) RS232 konektor
    - Sučelje za *debug* prema računalu, na njega su preusmjereni standardni ulaz i izlaz C programa koji se izvršava na Microblaze procesoru
- (10) Video ulazi
    - Koristi se samo jedan kompozitni video ulaz za primanje analognog PAL video signala
- (21) JTAG konektor
    - Sučelje za programiranje pločice preko Xilinx USB Platform kabela. Služi za direktno programiranje FPGA uređaja ili Flash memorije
- (23) Video izlazi
    - Koristi se jedan kompozitni video izlaz za analogni PAL video signal



Slika 8. LogiCRAFT3 razvojna ploča, pogled straga

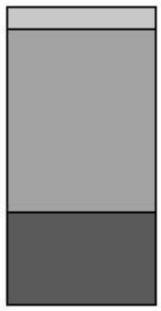
- (29) Philips SAA7113H procesor video ulaza
    - Čip koji obavlja A/D pretvorbu, tj. digitalizaciju ulaznog PAL video signala
    - Postoje dva identična čipa spojena na različite video ulaze, koristi se samo jedan

Uz logiTAP platformu isporučuje se i biblioteka razvijenih periferija tvrtke Xylon, tj. IP jezgri (eng. *intellectual property*) pod nazivom logicBRICKS. Optimizirani sklopovi sadržani u toj biblioteci omogućuju brži razvoj kompleksnijih ugradbenih računala implementiranih u Xilinx FPGA uređajima. Svaka od IP jezgri isporučuje se s punom dokumentacijom, ispitnim okruženjima i referentnim dizajnima. U ovom radu korištene su dvije IP jezgre iz logicBRICKS biblioteke: logiMEM i logiFLASH.

LogiMEM je fleksibilni sinkroni DRAM kontroler koji podržava SDR (eng. *Single Data Rate*) i DDR (eng. *Double Data Rate*) memorije. Na ostatak sustava spaja se preko PLB ili OPB sabirnice, dodatno koristeći XCL sučelje i XMB sučelje tvrtke Xylon.

LogiFLASH je memorijski kontroler za NOR tip Flash memorija. Može se spojiti na različita sistemska sučelja (OPB, XMB), a podržava paralelne, asinkrone flash uređaje s podatkovnim širinama od 8, 16 i 32 bita.

LogiTAP Flash memorija služi za spremanje konfiguracijskog *bitstreama* FPGA sklopa i softverske aplikacije koja se izvršava na procesoru. Pri uključivanju sustava, FPGA se automatski konfigurira iz Flash memorije. Ovaj način konfiguracije naziva se BPI (eng. *Byte Peripheral Interface*). U *bitstreamu* je integriran i program pod nazivom *bootloader*. On je spremljen u Block RAM memoriji, a Microblaze ga izvršava nakon resetiranja. *Bootloader* je zadužen za čitanje izvršne datoteke aplikacije iz Flash memorije i njeno prebacivanje u SDRAM. Nakon toga program skače na početnu adresu aplikacije u SDRAM-u i počinje je izvršavati. Razlog korištenja *bootloadera* umjesto direktnog zapisivanja aplikacije u BRAM je jednostavan: ukupna količina BRAM memorije je veoma ograničena (manje od 60 KB), pa veće aplikacije ne stanu u memoriju.

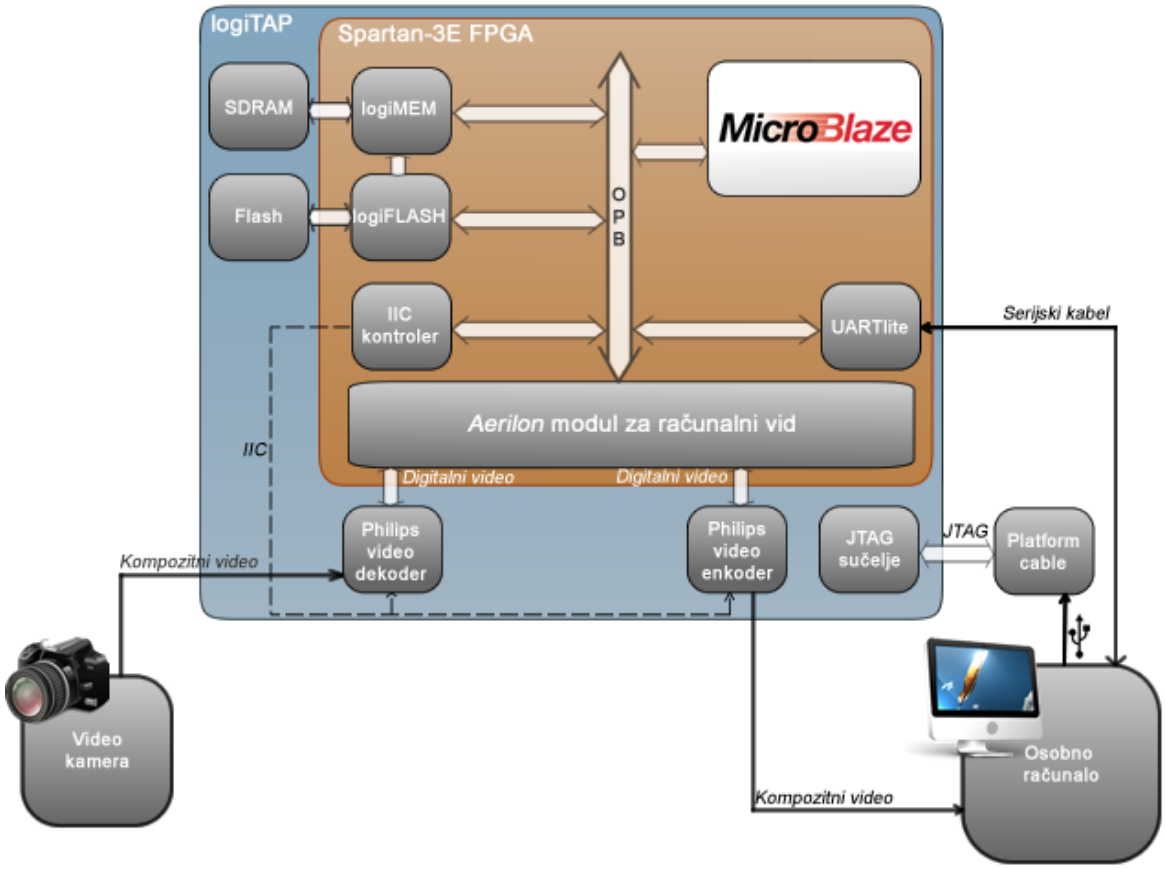


Slika 9. Organizacija Flash memorije u logiTAP sustavu

Slika 9 prikazuje organizaciju Flash memorije. Prvi sektor od 2 kB drži trajno pohranjene parametre sustava koji moraju biti dostupni prilikom pokretanja. Drugi sektor od 7.4 MB rezerviran je za programsku aplikaciju, dok treći sektor od 600 kB pohranjuje konfiguracijski *bitstream* FPGA sklopa.

2.7. Dizajn sustava za obradu videa u realnom vremenu

U prošlim potpoglavljima opisane su tehničke karakteristike platforme korištene za izradu sustava računalnog vida. U nastavku je prikazan razvijeni sustav.



Slika 10. Shema sustava

Slika 10 daje pregled arhitekture rješenja. LogiTAP platforma spojena je na računalo preko JTAG kabela, koji služi za programiranje FPGA i Flash memorije te RS232 sučelja, koje služi za komunikaciju s računalom pri izvođenju programa. Na jedan kompozitni ulaz spojen je izvor video signala, dok je na kompozitni izlaz spojena video kartica kako bi rezultati bili vidljivi na računalu u realnom vremenu.

Izvor video signala može biti bilo koji uređaj s kompozitnim video izlazom. U ovom radu je u tu svrhu korišten digitalni fotoaparat Sony CyberShot DSC-W120. Analogni PAL signal se preko kompozitnog priključka prosljeđuje do Philips SAA7113H čipa, koji pretvara analognu informaciju u digitalne signale. Digitalizirani video šalje se na ulazne pinove FPGA sklopa.

Unutar FPGA sklopa implementirana je posebna komponenta za provođenje algoritama računalnog vida, pod razvojnim nazivom *Aerilon*. Navedena komponenta

provodi obradu videa u realnom vremenu i na izlazu daje digitalni PAL video signal. Sa slike je vidljivo da je komponenta spojena na OPB sabirnicu Microblaze procesorskog sustava. Preko OPB sabirnice ostvaruje se programirljivost *Aerilon* komponente na način da Microblaze adresira unutarnje, korisnički definirane registre komponente te iz njih ili čita podatke ili u njih upisuje nove podatke. OPB sabirnica izabrana je iz razloga što je količina podataka koja se razmjenjuje s komponentom razmjerno mala te nisu potrebne brže sabirnice, kao što je npr. FSL.

Izlazni digitalni video signal s vanjskih se priključaka FPGA sklopa prosljeđuje u Philips SAA7121H enkoderski čip, koji provodi D/A konverziju. Izlaz video enkodera spojen je na kompozitni video izlaz, te se prema vani šalje analogni video signal. Prikaz slike može se dobiti na bilo kojem uređaju s kompozitnim video ulazom, kao što su televizijski prijemnici, video capture kartice itd.

3. Dohvat video signala

Prvi korak u obradi videa jest njegov dohvat iz izvora video signala. U ovom poglavlju opisani su razni video standardi koji se danas koriste u svijetu, specifikacije PAL standarda, načini dohvata vrijednosti piksela iz toka video podataka, te generiranje sinkronizacijskih signala. Konačno, prikazano je kako se izdvojeni pikseli šalju na obradu procesnoj jedinici računalnog vida.

3.1. Video standardi

U svijetu postoje brojni standardi analognog i digitalnog videa. Mnoge televizijske kuće i centri za video produkciju koriste komponentni digitalni video pri stvaranju, spremanju i transportu video materijala. Komponentni digitalni video pogodan je za sažimanje uz korištenje standarda za digitalnu video kompresiju. Također ga je moguće kodirati u analogni kompozitni video za emitiranje. Danas su najčešći digitalni video standardi zasnovani na 4:2:2 shemi uzorkovanja [17]. Komponentni 4:2:2 digitalni video format koristi se u raznim standardima za 525-linijske sustave (NTSC), 625-linijske (PAL), NTSC i PAL širokog zaslona, te HDTV. Tablica 1 navodi neke poznatije standarde.

Tablica 1. Najčešći 4:2:2 standardi komponentnog digitalnog videa

SMPTE 125M(1) i ITU-R BT.601-5(2)	NTSC & PAL, omjer stranica 4:3, 4:2:2 komponentni digitalni video
SMPTE 267M	NTSC, omjer stranica 16:9, 4:2:2 komponentni digitalni video
SMPTE 260M	1125 linijski 60-Hz HDTV
SMPTE 274M	1920 x 1080 sken – progresivni i isprepleteni HDTV
SMPTE 293M	720 x 483 – HDTV s progresivnim skeniranjem
SMPTE 296M	1280 x 720 – HDTV s progresivnim skeniranjem

Napomena:

1. SMPTE – "Society of Motion Picture and Television Engineers"
2. ITU – "International Telecommunication Union"

3.1.1. Prostor boja

U početku razvoja televizije, video signali sadržavali su samo informaciju o intenzitetu svjetlosti (svjetlina ili *luma*, s oznakom Y), pa su televizijski prijemnici shodno tome prikazivali samo crno-bijelu sliku. Kad je razvojem tehnologije dodana informacija o boji, signal svjetline ostavljen je netaknut zbog kompatibilnosti s postojećom opremom, no dodane su dvije komponente informacije o boji, nazvane U i V. Te dvije komponente često se nazivaju i signalima razlike u boji jer se dobivaju iz razlike intenziteta boje i cjelokupnog intenziteta uzorka. "U" komponenta je, pojednostavljeno, razlika između plave boje i Y komponente. Komponenta "V" razlika je između crvene boje i Y komponente.

Današnji PAL i NTSC sustavi TV signala zasnovani su na YUV prostoru boja. U komponentnom digitalnom videu koristi se tzv. YCbCr prostor boja. Radi se o skaliranoj i pomaknutoj verziji YUV prostora s komponentom svjetline (Y) i dvije komponente boje (Cb i Cr). Komponenta Y sadrži nominalne 8-bitne vrijednosti od 16 do 235. Komponente boje mogu poprimiti vrijednosti od 16 do 240. Neke od vrijednosti ispod i iznad nominalnog raspona koriste se za kodiranje specijalnih signala, tj. "markera" u tok video podataka.

3.1.2. Uzorkovanje

Jedna od glavnih karakteristika digitalnih formata komponentnog videa jest shema uzorkovanja, označena s nizom brojeva odvojenih dvotočkama, kao što su 4:2:2 i 4:4:4.

Uzorkovanje po shemi 4:2:2 označava da na svaka 4 uzorka svjetline (Y), dolaze 2 uzorka signala boje, po jedan od Cb i Cr komponente. U videu standardne definicije (SD), svjetlina se uzorkuje frekvencijom od 13.5MHz, a svaka od komponenti boje dvostruko nižom frekvencijom. Ovaj način uzorkovanja iskorištava činjenicu da je ljudsko oko manje osjetljivo na promjene u boji nego u intenzitetu svjetlosti. Smanjenom frekvencijom uzorkovanja boje smanjuje se i količina podataka koji se moraju prenijeti, kao i zahtijevana propusnost (eng. *bandwidth*).

Ostale česte sheme uzorkovanja su 4:4:4, kod koje postoji jednak broj Y,Cb i Cr uzoraka, te 4:1:1, gdje se na svaka 4 uzorka svjetline uzorkuje tek jedan signal boje. Također postoji i shema oznake 4:2:0, koja podrazumijeva kompresiju komponenata boje u horizontalnom i vertikalnom smjeru. No, 4:2:2 je i dalje najčešća danas korištena shema uzorkovanja u centrima za emitiranje i produkciju video materijala [17].

3.2. PAL video format

U PAL video formatu, svaka linija slike sadrži 864 uzorka. Uzorak se sastoji od jednog bajta[1] Y komponente i jednog bajta komponente boje. Cb i Cr komponenta izmjenjuju se u uzastopnim uzorcima. Aktivni dio linije sastoji se od uzoraka označenih brojevima 0 do 719. Neaktivni dio linije naziva se horizontalni interval zatamnjenja (eng. *Horizontal Blanking Interval*) i sastoji se od uzoraka 720 do 863. Uzorčni parovi 720/721 i 862/863 sadrže specijalne oznake koje se nazivaju signalima vremenskog vođenja (eng. *Timing Reference Signals – TRS*). Par 720/721 sadrži TRS oznaku kraja aktivnog videa (eng. *End of Active Video – EAV*), dok par 862/863 sadrži TRS oznaku početka aktivnog videa (eng. *Start of Active Video – SAV*). Ovi signali koriste se za označavanje prijelaza između aktivnih i neaktivnih dijelova linije, ali sadrže i druge informacije o vremenskom vođenju. Prva tri bajta TRS simbola su 0xFF, 0x00 i 0x00. Četvrti bajt naziva se XYZ vrijednošću, i označava stanje F,V i H bitova; dok se četiri bita koriste za detekciju pogreške. Sadržaj XYZ bajta prikazan je na slici 11.

Bit	7	6	5	4	3	2	1	0
	1	F	V	H	P3	P2	P1	P0

Slika 11. XYZ bajt u TRS oznaci

Bitovi P3 do P0 su bitovi za zaštitu i računaju se po sljedećim formulama:

P3 = V XOR H

P2 = F XOR H

P1 = F XOR V

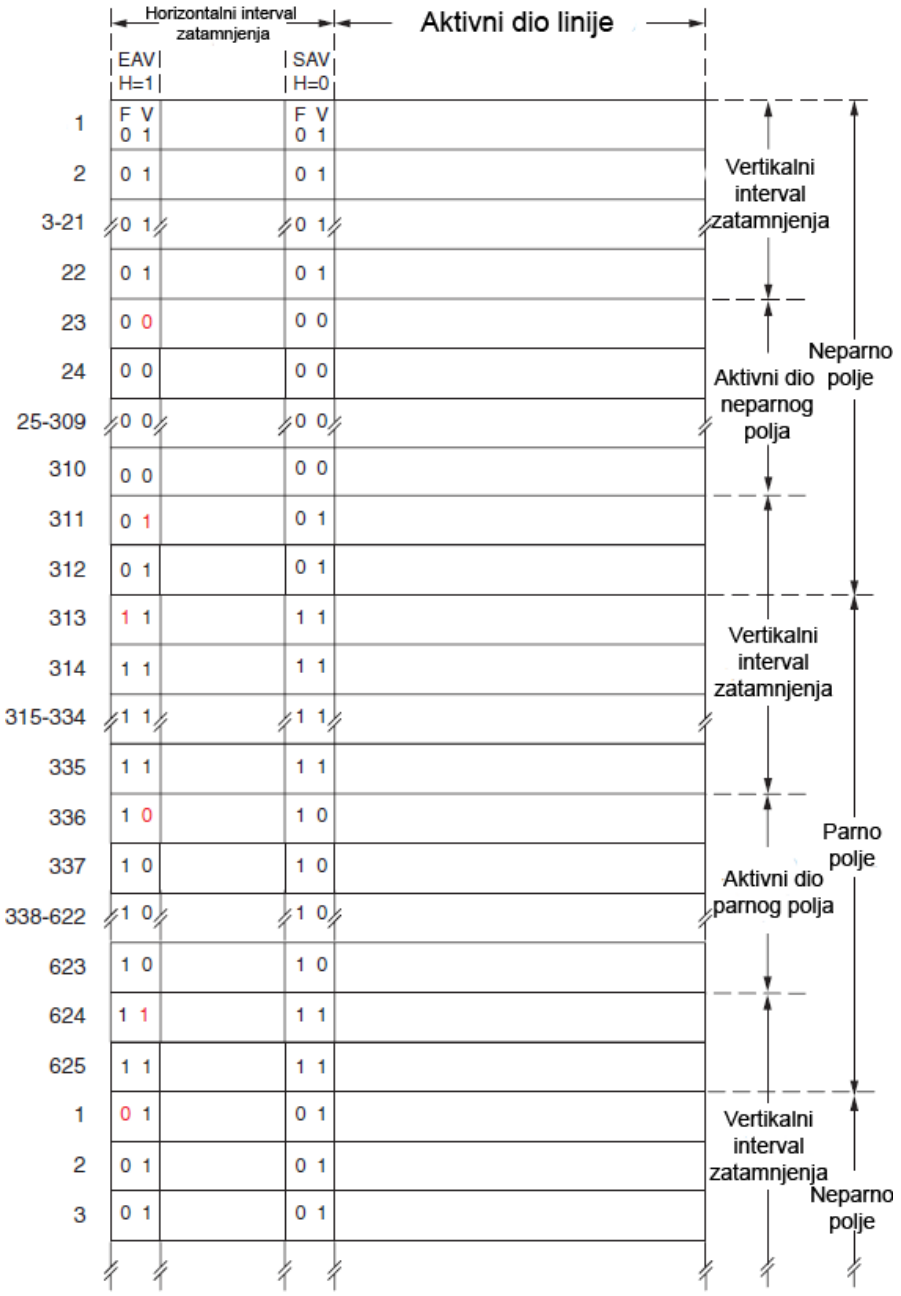
P0 = F XOR V XOR H

Bit F (od eng. *Field* - polje) označava radi li se o parnom ili neparnom polju isprepletene slike. Bit V (od eng. *Vertical blanking*) označava da li je trenutno aktivan vertikalni period zatamnjenja, a bit H (od eng. *Horizontal blanking*) aktivan je za vrijeme horizontalnog perioda zatamnjenja. Tijekom zatamnjenja naizmjence se šalju podaci 0x80, 0x10, 0x80, 0x10 za Cb, Y, Cr i Y komponente, respektivno.

[1] Napomena: postoji i 10-bitni standard, u kojem se informacija o svakom uzorku šalje u 10 bitova umjesto u 8, čime se postiže veća preciznost. Kompatibilnost s 8-bitnom verzijom izvodi se zanemarivanjem najniža dva bita, koja se smatraju frakcionalnim dijelom. U daljnjem tekstu će se podrazumijevati 8-bitna izvedba.

PAL video okviri sastoje se od 625 linija podijeljenih u dva isprepletena polja. Stopa osvježavanja slika u PAL signalu iznosi 25 Hz, što povlači da se polja iscrtavaju frekvencijom 50 Hz. Od 625 linija, 576 ih je aktivnih, što daje efektivnu rezoluciju slike od 720*576 piksela.

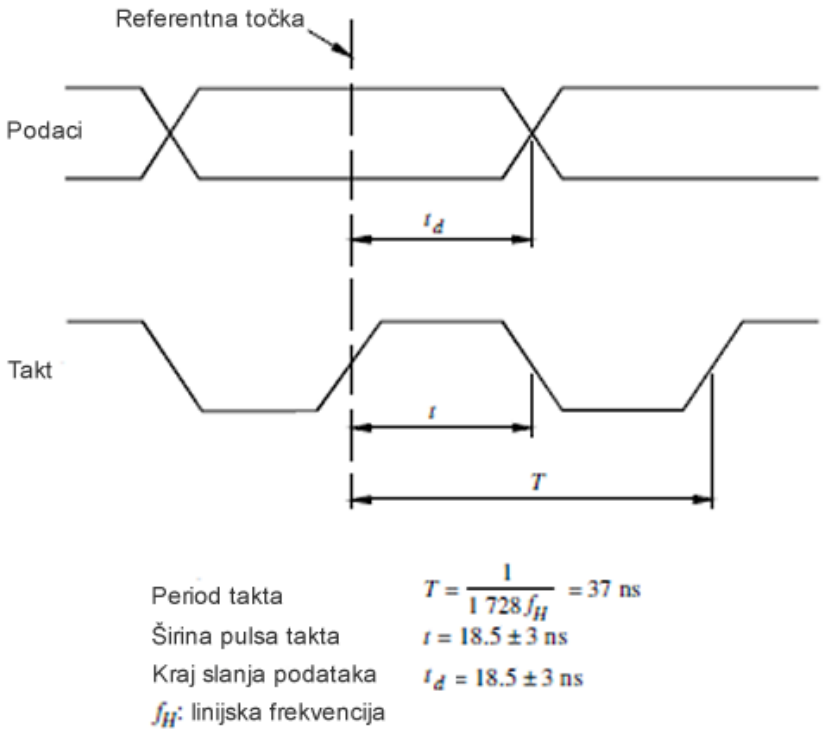
Slika 12 ilustrira uređenje vertikalnih regija u PAL digitalnom videu. Dijagram prikazuje brojeve linija u kojima bitovi F i V mijenjaju vrijednost. Na primjer, u prve 22 linije bit F je 0, a bit V je 1, jer se radi o intervalu vertikalnog zatamnjenja neparnog polja. U liniji 23 kreće aktivni dio neparnog polja i bit V se postavlja u 0.



Slika 12. Vertikalne regije u PAL signalu

3.2.1. Sučelje

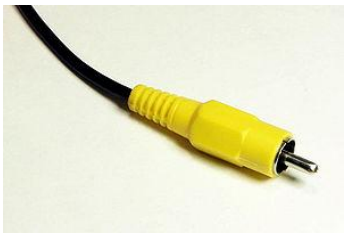
Bitovi digitalnih kodnih riječi koje opisuju video signal prenose se paralelno pomoću osam signalnih linija, a svaka prenosi multipleksirani tok bitova (jednake važnosti) svake komponente (Cb,Y,Cr,Y). Tih osam parica također prenose i dopunske informacije koje su multipleksirane u tok video podataka tijekom intervala zatamnjenja. Dodatna linija prenosi sinkroni signal takta od 27 MHz. Rastući brid signala takta događa se na pola puta između promjene podataka (slika 13).



Slika 13. Vremenski odnosi u digitalnom video signalu

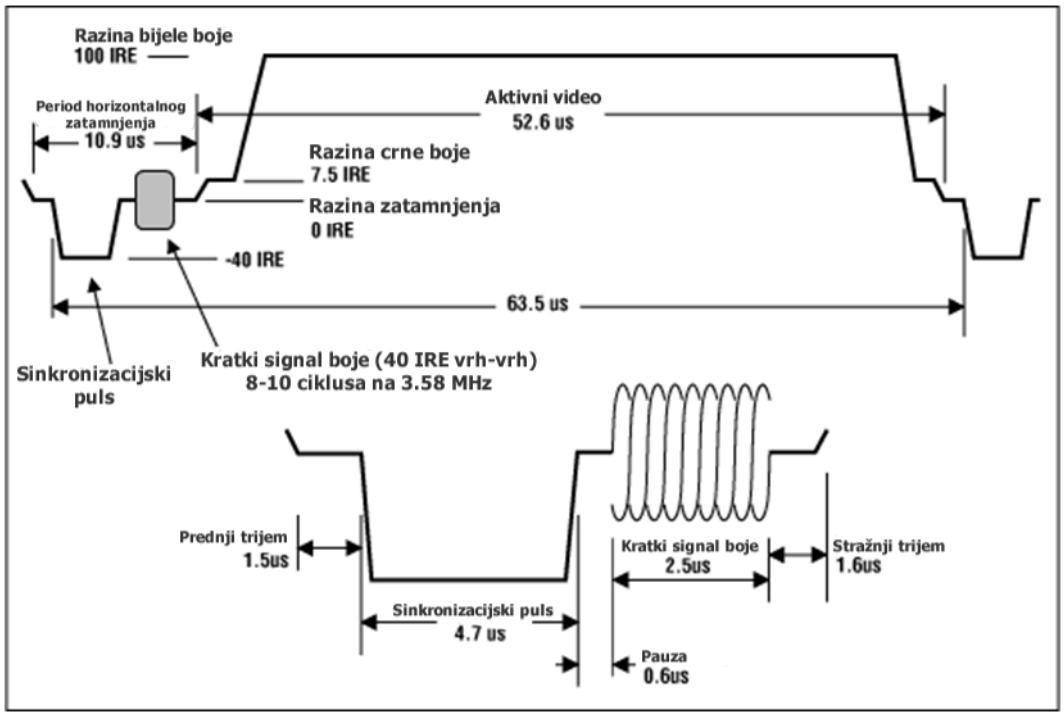
3.3. Analogni kompozitni video

Kompozitni signal najčešće je korišteno analogno video sučelje [18]. Kompozitni video se također naziva i CVBS (eng. *Color, Video, Blanking, Sync*), a kombinira informaciju o svjetlini, informaciju o boji i signale za sinkronizaciju na samo jednoj žici. Konektor je tipično RCA (slika 14), koji se ujedno koristi i za standardne audio spojeve.



Slika 14. Izgled RCA konektora

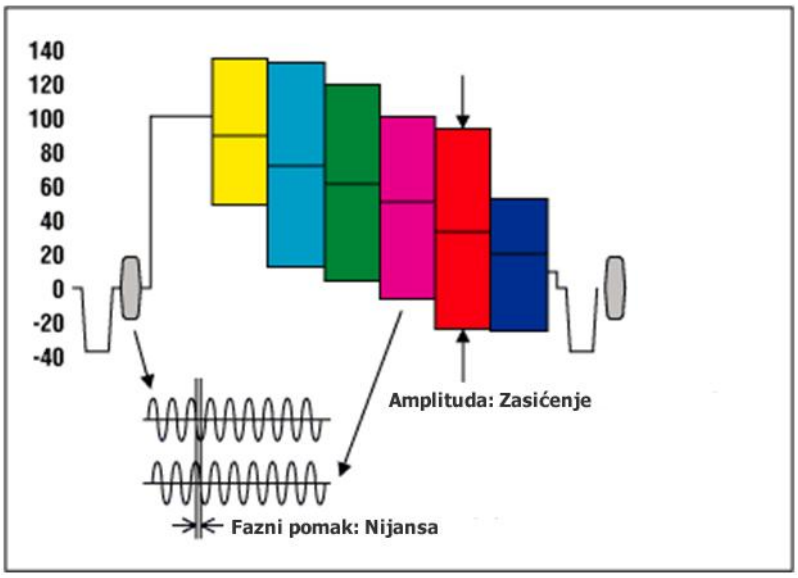
Tipični valni oblik kompozitnog video signala koji prikazuje čistu bijelu boju prikazan je na slici 15.



Slika 15. Kompozitni video signal

Na slici je prikazan signal koji predstavlja jednu horizontalnu liniju (radi se o NTSC signalu, razlika u odnosu na PAL je minimalna). Svaka linija sastoji se od perioda horizontalnog zatamnjenja i aktivnog videa. Aktivni video sadrži informacije o svjetlini i boji slike. Informacija o svjetlini je amplituda signala u bilo kojem trenutku. Mjerna jedinica za amplitudu naziva se IRE. Radi se o relativnoj jedinici, jer video signal može biti bilo koje amplitude. Vrijednost od 100 IRE je originalno definirana kao raspon od crne do bijele boje u video signalu. Vrijednost od 0 IRE odgovara naponu od 0 V tijekom perioda zatamnjenja. Sinkronizacijski puls obično se nalazi 40 IRE ispod vrijednosti 0, tako da bi bijeli signal od vrha do vrha trebao imati razliku od 140 IRE.

Informacija o boji modulira se na signal svjetline, a predstavlja ju sinusni valni oblik. Boje se određuju pomoću razlike u fazi između tog valnog oblika i tzv. *color-burst* referentne faze. Slika 16 prikazuje horizontalnu liniju koja sadržava stupce u boji.



Slika 16. Modulacija boje u kompozitnom signalu

Amplituda modulacije proporcionalna je količini boje (zasićenost), a informacija o fazi označava nijansu boje. Horizontalni period zatamnjenja uz sinkronizacijski puls sadrži i referencu boje (*color-burst*) koja počinje odmah nakon pozitivnog brida tzv. stražnjeg trijema (eng. *back porch*).

3.4. Dekodiranje ulaznih podataka

Ulazni podaci u sustav dolaze s kompozitnog priključka i u analognom su formatu. Prvi korak u dohvatu podataka bit će njihova digitalizacija. Nakon što su ulazni podaci u odgovarajućem digitalnom formatu, potrebno je ispravno protumačiti tok podataka i iz njega izlučiti sinkronizacijsku informaciju te vrijednosti pojedinih piksela koje se mogu slati na obradu.

3.4.1. Digitalizacija

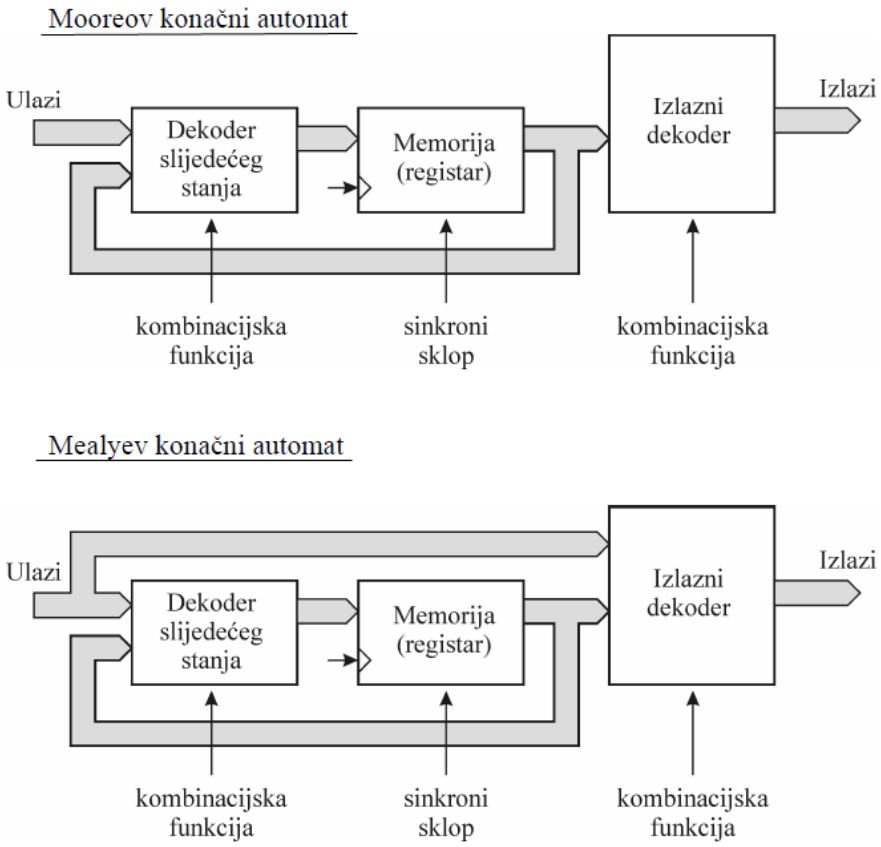
Proces digitalizacije provodi Philipsov čip oznake SAA7113H. Radi se o dekoderu analognog video signala, koji može primati CVBS ili S-Video analogni signal, te kao izlaz davati digitalni video po ITU-656 standardu. Sklop također generira signal video takta odgovarajuće frekvencije (27 MHz). Kontrola sklopa izvršava se preko I^2C sabirnice. Na izlazu sklopa dobivaju se 8-bitni video podaci s umetnutim signalima vremenskog vođenja (TRS) i signal takta.

Navedenih 8 signalnih linija spojene su na vanjske priključke FPGA sklopa, koji je konfiguriran tako da navedene priključke interpretira kao ulazne.

3.4.2. Generiranje sinkronizacijskih signala

Dizajn sklopa za implementaciju računalnog vida započinje s dohvatom odgovarajućih podataka iz digitalnog video toka. Međutim, kako ne postoje vanjski signali za sinkronizaciju koji nose informaciju o početku slike, kraju perioda zatamnjenja i slično, te signale potrebno je dobiti iz toka podataka interpretiranjem TRS oznaka.

Budući da radimo sa sinkronim dizajnom sklopovlja, potrebno je osmisliti sustav koji je u mogućnosti mijenjati stanje u ovisnosti o ulaznim podacima. U ovu svrhu koriste se strojevi s konačnim brojem stanja (eng. *Finite State Machine – FSM*). Automat karakterizira svojstvo da može pamtiti trenutno stanje, te da se prijelaz u iduće stanje obavlja na temelju trenutnog stanja i ulaznog signala. Dva su osnovna tipa takvih strojeva: *Mooreov* i *Mealyev* automat, a razlikuju se po generiranju izlaza. Izlaz Mooreovog automata ovisi isključivo o trenutnom stanju, dok izlaz Mealyevog automata ovisi o trenutnom stanju i trenutnim ulazima. Slika 17 prikazuje sheme sklopovskih realizacija ovih dvaju automata.



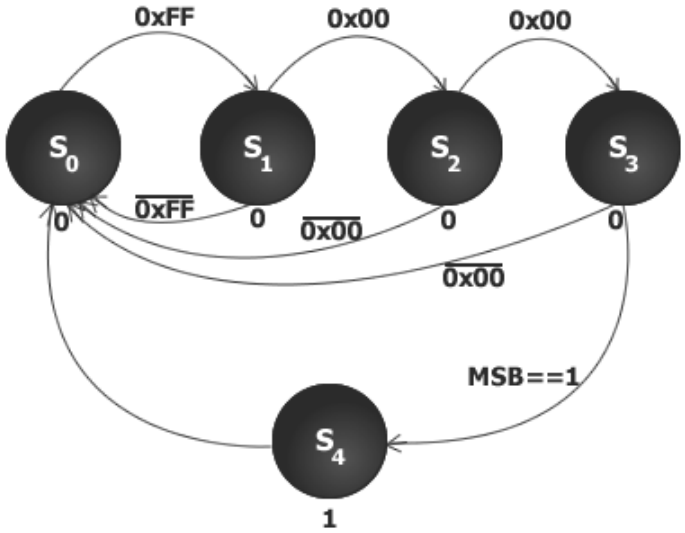
Slika 17. Mooreov i Mealyev konačni automat

Bitno je primijetiti kako se u oba tipa automata sklopovlje može podijeliti u dva dijela. Jedan dio je sinkroni, aktiviran signalom takta, i služi za pamćenje stanja. Drugi dio je kombinacijski i služi za određivanje prijelaza automata i njegovog izlaza.

Jezici za opis sklopovlja pogodni su za opis obje vrste dizajna: sinkronog i kombinacijskog. U VHDL-u se za dizajn automata tipično koristi šablona dvaju procesa. Jedan proces opisuje ponašanje kombinacijskog dijela sklopa, uključujući određivanje idućeg stanja u koje će automat preći, dok se drugi proces najčešće aktivira na rastući brid signala takta i mijenja stanje automata. Sintetizator će iz ovakvog opisa stvoriti odgovarajuće sklopovlje, optimizirajući brzinu i zauzeće logičkih elemenata. Stanje automata pohranjuje se u registru odgovarajuće širine. Svako stanje opisano je binarnim kodom. Način kodiranja odredit će sintetizator u ovisnosti o mogućnostima prijelaza iz stanja u stanje. Primjerice, moguće je koristiti Grayev kod za minimizaciju broja bitova koji se mijenjaju pri prijelazu u susjedno stanje.

Kao što je rečeno u poglavlju 3.2, TRS oznaka sastoji se od 4 okteta: 0xFF, 0x00, 0x00 i XYZ okteta, koji nosi informaciju o tipu TRS oznake. Važno je primijetiti da je nemoguće da se ovakav niz okteta pojavi negdje unutar niza video podataka jer se vrijednosti piksela nalaze unutar manjeg raspona.

Na slici 18 prikazan je Mooreov automat koji detektira pojavu TRS oznake u toku video podataka.



Slika 18. Mooreov automat za detekciju TRS oznake

Automat koristi signal video takta (27 MHz) za aktiviranje prijelaza. Skup stanja je jednostavan. Sekvenca 0xFF,0x00,0x00,XYZ prebacit će automat u stanje u kojem na

izlazu daje logičku jedinicu. Pojava bilo kakve druge oznake resetira automat. Postoji samo jedan izlazni signal koji se postavlja u jedinicu nakon detektiranja TRS oznake, a na idući pozitivni brid takta postavlja se u nulu. Ovaj signal koristi se kao ulaz drugog automata koji služi za dekodiranje stanja toka video podataka.

Detekcija TRS oznake prvi je korak u određivanju sinkronizacijskih signala. Kako bi bilo moguće odrediti kojem dijelu slike odgovaraju ulazni podaci, nužno je interpretirati TRS oznake. Korištenjem F,V i H bitova XYZ okteta u TRS oznaci moguće je odrediti pripada li nadolazeći tok podataka aktivnom dijelu slike ili periodu zatamnjenja.

Drugi Mooreov automat služit će određivanju sinkronizacijskih signala. Potrebno ga je dizajnirati tako da trenutno stanje automata odgovara poziciji unutar okvira. Sa slike 12 vidimo da postoji 12 disjunktnih regija unutar okvira. Svako stanje automata odgovarat će jednoj poziciji, uz dodatak početnog stanja. Tablica 2 prikazuje popis stanja i izlaza automata u svakom stanju.

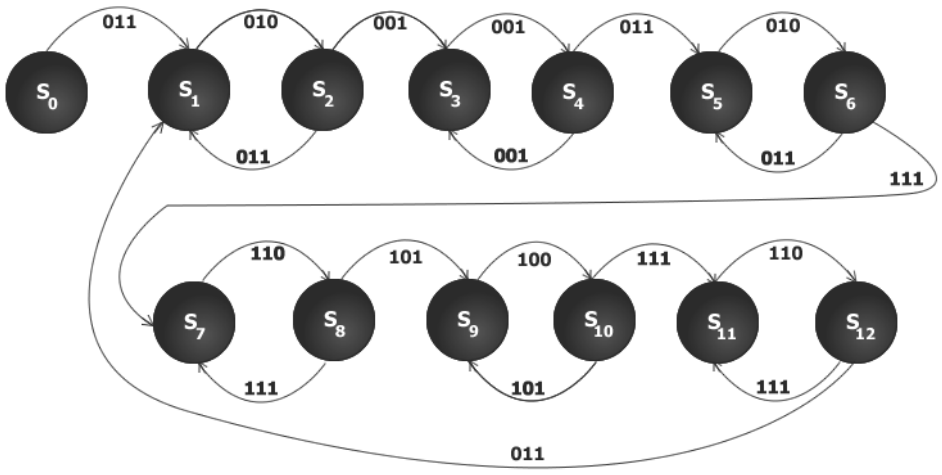
Tablica 2. Stanja i izlazi Mooreovog automata za dekodiranje vertikalnih regija

Stanje	**Pozicija**	**F**	**VSync**	**HSync**
S_0	(početno stanje)	X	X	X
S_1	VBLANK, HBLANK	0	1	0
S_2	VBLANK, ACTIVE	0	1	0
S_3	ACTIVE, HBLANK	0	0	0
S_4	ACTIVE, ACTIVE	0	0	1
S_5	VBLANK, HBLANK	0	0	0
S_6	VBLANK, ACTIVE	0	0	0
S_7	VBLANK, HBLANK	1	1	0
S_8	VBLANK, ACTIVE	1	1	0
S_9	ACTIVE, HBLANK	1	0	0
S_{10}	ACTIVE, ACTIVE	1	0	1
S_{11}	VBLANK, HBLANK	1	0	0
S_{12}	VBLANK, ACTIVE	1	0	0

Napomena:
VBLANK: vertikalni period zatamnjenja
HBLANK: horizontalni period zatamnjenja
ACTIVE: aktivne linije; aktivni dio linije
X: *„don't care“*

Automat ima tri izlazna signala. Signal F (eng. *field*) određuje radi li se o parnom ili neparnom polju u isprepletenoj slici. Signal VSync aktivan je za vrijeme vertikalnog perioda zatamnjenja i pada u nulu čim počnu pristizati podaci aktivnog dijela slike. Signal HSync aktivan je dok su na ulazu podaci aktivnog dijela linije, tj. dok u sustav ulaze čisti pikseli slike. Ovaj signal je ujedno i najbitniji za sinkronizaciju, jer se može koristiti kao jednostavan signal za omogućavanje (eng. *enable signal*) sklopa koji prima samo piksele slike.

Početno stanje automata je ono u kojem će se sklop naći nakon reseta. Stanja S_1-S_6 ekvivalentna su stanjima S_7-S_{12}, uz razliku signala F, koji razlikuje polja slike. U stanjima S_4 i S_{10} u sustav dolaze aktivni pikseli slike, dok ostala stanja razlikuju razne vrste zatamnjenja. Dijagram stanja prikazan je na slici 19.



Slika 19. Pojednostavljeni dijagram stanja automata za dekodiranje vertikalnih regija

Brojevi na prijelazima stanja predstavljaju bitove F,V i H u XYZ oktetu TRS oznake. Izmjenjivanje stanja S_1 i S_2 odgovara prijelazima iz horizontalnog intervala zatamnjenja u aktivni dio linije tijekom vertikalnog perioda zatamnjenja, kao i izmjenjivanje stanja S_5 i S_6. Stanja S_3 i S_4 odgovaraju linijama aktivnog dijela slike i prijelazima aktivnog dijela linije u period horizontalnog zatamnjenja. Analogno se ponašaju stanja S_7-S_{12} za drugo polje slike. Automat očito nema završno stanje, već se vrti u ciklusima prateći ulazni video.

U svrhu ispravnog praćenja podataka u sklopovlju je moguće implementirati i brojač piksela u liniji slike. Iz specifikacije PAL standarda poznato je da u aktivnom dijelu linije postoji 720 piksela. Svaki piksel predstavljen je s dva okteta (jedan za svjetlinu, drugi za boju), što daje 1440 okteta u aktivnom dijelu linije. Brojač od *n* bitova može brojati do

2^n-1, iz čega slijedi da je za potrebe brojanja okteta u liniji potrebno implementirati 11-bitni brojač koji se resetira na vrijednosti 1440. Ovaj brojač omogućuje kontrolu automata na način da resetira automat ukoliko on nije prešao u odgovarajuće stanje na kraju pojedine linije. Drugo korisno svojstvo brojača proizlazi iz promatranja najnižeg bita brojača. Taj bit će biti postavljen u logičku jedinicu kad je na ulazu oktet svjetline, a u nulu kad je na ulazu oktet koji opisuje boju. Na taj način moguće je jednostavno filtrirati informaciju o svjetlini slike i zanemariti podatke o boji, što se pokazuje pogodnim za razne algoritme računalnog vida koji koriste slike sivih razina.

Najniži bit brojača u kombinaciji sa sinkronizacijskim signalima aktivne linije (HSync) i vertikalne regije (VSync) može se koristiti kao signal takta koji omogućuje sklopovlju za obradu slike da se sinkronizira na nadolazeće piksele.

4. Obrada slike

U prošlom poglavlju opisan je način dohvaćanja piksela slike iz toka video podataka. Pokazano je kako se dekodiraju sinkronizacijske oznake i kako se generiraju signali koji omogućuju da sklopovlje za obradu slike bude aktivno u precizno određenim vremenskim trenucima. U ovom poglavlju bit će riječi o raznim operacijama i algoritmima računalnog vida koji se mogu provesti nad slikom, a da se efikasno implementiraju u sklopovlju.

4.1. Operacije nad pojedinim pikselima

Najjednostavniji algoritmi računalnog vida obrađuju sliku piksel po piksel. Takva obrada ne zahtijeva memorijske elemente za pamćenje piksela, a i jednostavna je za implementaciju u sklopovlju. Neke od operacija koje se izvode na ovaj način su: određivanje histograma, komplementiranje slike, povećavanje kontrasta, binarizacija itd. Za potrebe ovog rada kao primjer operacije izabrana je binarizacija slike pomoću usporedbe s pragom (eng. *thresholding*).

Kod sklopovske izvedbe algoritama koji operiraju nad pojedinim pikselima potrebno je osigurati da vrijeme izvođenja operacije nad pikselom nije duže od vremena koje je potrebno da sljedeći piksel dođe na obradu. Frekvencija takta PAL signala jednaka je 27 MHz. Kako se koriste samo informacije o svjetlini piksela, zbog čega se ulaz uzorkuje na svakom drugom periodu takta, podaci dolaze efektivnom frekvencijom od 13.5 MHz. Dakle, kašnjenje signala kroz sklopovlje za obradu ne smije biti veće od 74 ns.

4.1.1. Binarizacija slike

Binarizacija je jednostavan postupak segmentacije slike kojim se vrijednosti sivih razina u slici pretvaraju u ili crnu ili bijelu boju. Budući da su vrijednosti u novoj slici predstavljene samo sa dvije razine, možemo govoriti o binarnoj slici, gdje vrijednost '0' predstavlja crnu boju, a vrijednost '1' bijelu.

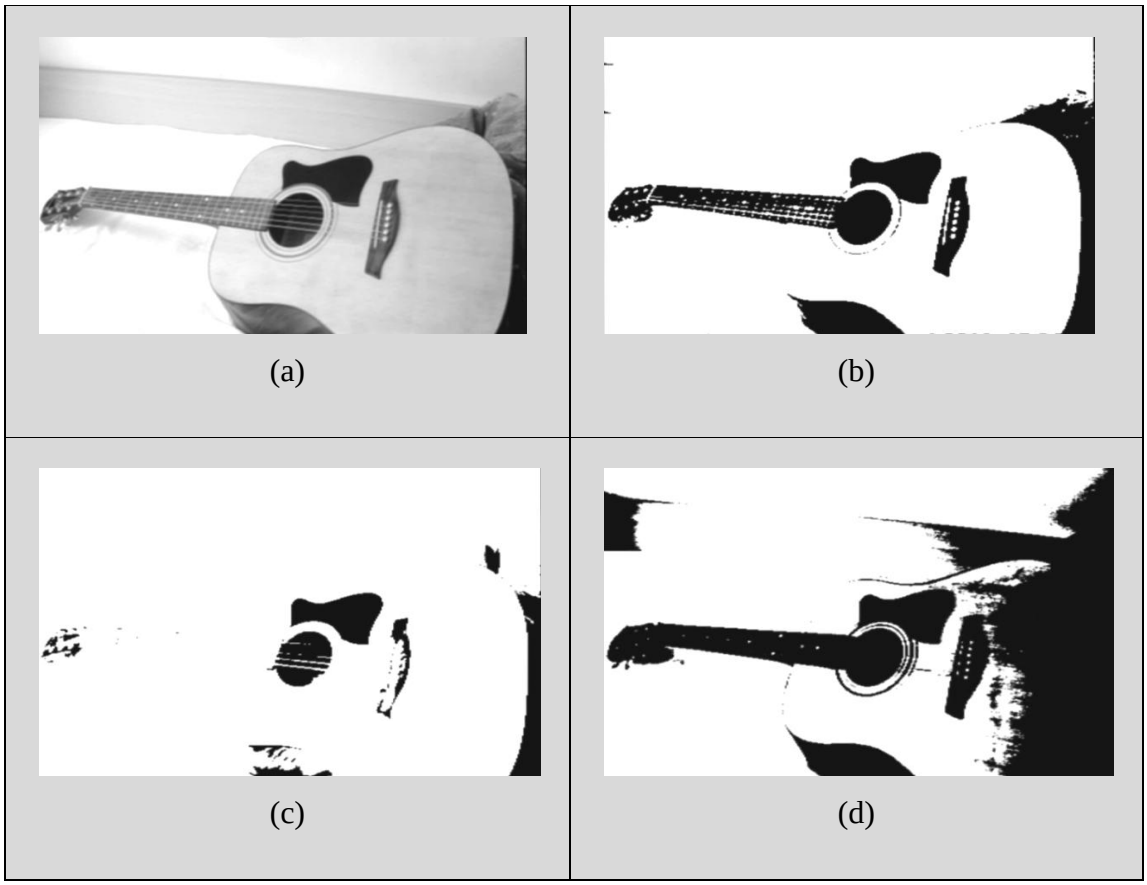
Postupak binarizacije najčešće se provodi algoritmima usporedbe s pragom. Dotični algoritmi mogu se razvrstati u nekoliko skupina:

- jednostavni (na temelju jednog praga)
- algoritmi koji koriste veći broj pragova
- algoritmi koji koriste automatski izbor praga u zavisnosti od značajki slike

Algoritam koji koristi jedan prag za segmentaciju slike vrlo je jednostavan. Potrebno je svaki piksel slike usporediti s pragom. Ako je vrijednost veća ili jednaka pragu, nova vrijednost postaje '1', inače postaje '0'. Ovaj pristup je najjednostavniji, ali i najmanje fleksibilan, jer očito ne postoji jedan univerzalni prag koji se može primijeniti na sve slike.

Kod implementacije sklopa za jednostavnu binarizaciju slike korisno je ostaviti otvorenom mogućnost naknadne promjene praga. Ovdje u igru ulazi programabilnost komponenti. Umjesto fiksnog „ožičenja“, tj. upisa stalne vrijednosti praga u npr. ROM memoriji, komponentu razvijenu za obradu slike moguće je konfigurirati tako da čita vrijednost praga iz korisnički definiranog registra. Korisnički definiran registar moguće je adresirati preko OPB sabirnice u Microblaze procesorskom sustavu i u njega programski upisati određenu vrijednost. Ideja ovakve izvedbe jest da se korisniku omogući promjena praga u bilo kojem trenutku te da se rezultati odmah vide na izlaznom toku video podataka.

Slika 20 prikazuje rezultate dobivene korištenjem programirljive komponente za binarizaciju korištenjem praga.



Slika 20. Postupak binarizacije: (a) Originalna slika (b) Prag na sredini raspona: 128 (c) Prag 112 (d) Prag 144

4.2. Operacije nad susjedstvom piksela

Složenije operacije obrade slike za određivanje rezultantne vrijednosti piksela osim informacije o trenutnoj vrijednosti promatranog piksela koriste i vrijednosti njemu susjednih piksela. Ovaj pristup podrazumijeva korištenje konteksta za razlikovanje slikovnih elemenata. Primjerice, definicija ruba kaže da je to element koji leži na granici dvaju područja. Iz toga slijedi da se rubni element može otkriti računanjem razlike susjednih slikovnih elemenata. Koristeći tu činjenicu, možemo dizajnirati algoritam koji promatra susjedstvo nekog elementa kako bi u slici pronašao rubove.

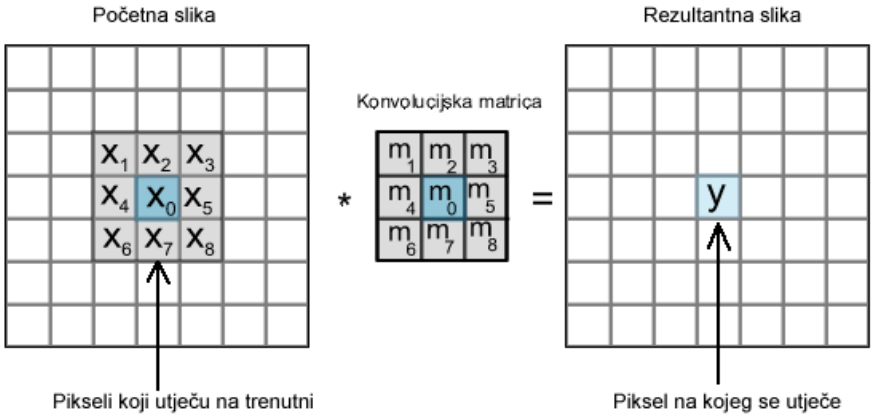
4.2.1. Konvolucija

Osnovna operacija nad susjedstvom piksela jest operacija konvolucije. Ako označimo vrijednosti intenziteta u izvornoj slici na položaju (x,y) s funkcijom f(x,y), tada operaciju konvolucije s konvolucijskom matricom m zapisujemo na sljedeći način:

$$f'(x,y) = f(x,y) * \mathbf{m}[x,y]$$

Vrijednosti u konvolucijskoj matrici definiraju utjecaj susjednih piksela na određenu točku slike. Konvolucijska je matrica najčešće puno manjih dimenzija od same slike. U primjeru na slici 21, intenzitet rezultantnog piksela *y* bit će izračunat sljedećom formulom:

$$y = m_0 \cdot x_0 + m_1 \cdot x_1 + m_2 \cdot x_2 + m_3 \cdot x_3 + m_4 \cdot x_4 + m_5 \cdot x_5 + m_6 \cdot x_6 + m_7 \cdot x_7 + m_8 \cdot x_8$$



Slika 21. Operacija konvolucije

4.2.2. Rubna segmentacija

Postupak rubne segmentacije podrazumijeva segmentiranje slike na temelju diskontinuiteta, tj. naglih promjena vrijednosti atributa. Za detektiranje rubova u slikama često se koristi tzv. rubni operator. Riječ je o matematičkom operatoru (ili njegovom

računskom ekvivalentu) s malim područjem djelovanja kojim se detektira lokalni rub. Postoji više vrsta rubnih operatora:

1. Rubni operatori koji aproksimiraju matematički gradijentni operator
2. Rubni operatori zasnovani na višestrukim maskama (kompasni operatori)
3. Rubni operatori koji podešavaju lokalne vrijednosti sivih razina s parametarskim modelom ruba

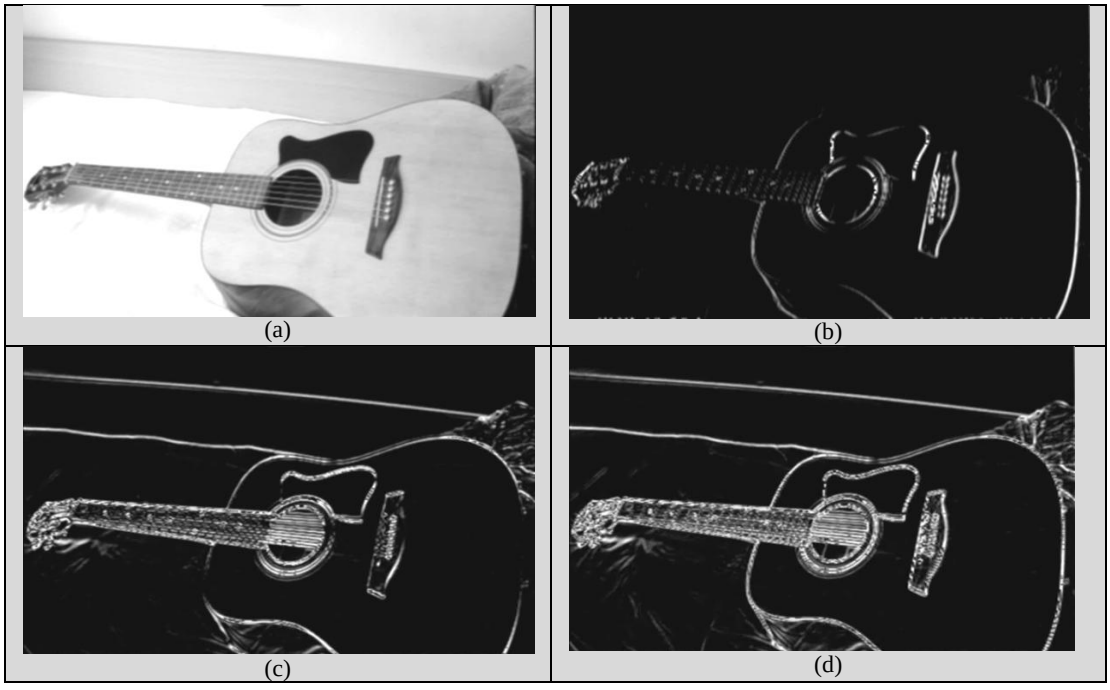
U daljnjem tekstu bit će više riječi o numeričkim aproksimacijama gradijentnih operatora, budući da su iznimno rašireni i zasnovani na operaciji konvolucije.

4.2.2.1. Sobelov operator

Sobel je 1970. godine predložio rubni operator koji izbjegava računanje gradijenta u interpoliranoj točki između dviju linija koristeći masku veličine 3x3. Amplituda gradijenta za Sobelov operator računa se kao $M = \sqrt{S_x^2 + S_y^2}$, a može se aproksimirati s računski manje zahtjevnim izrazom: $M' = |S_x| + |S_y|$, gdje se S_x i S_y mogu implementirati uporabom konvolucijskih maski:

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Prva konvolucijska maska detektira vertikalne rubove, dok druga detektira horizontalne rubove.



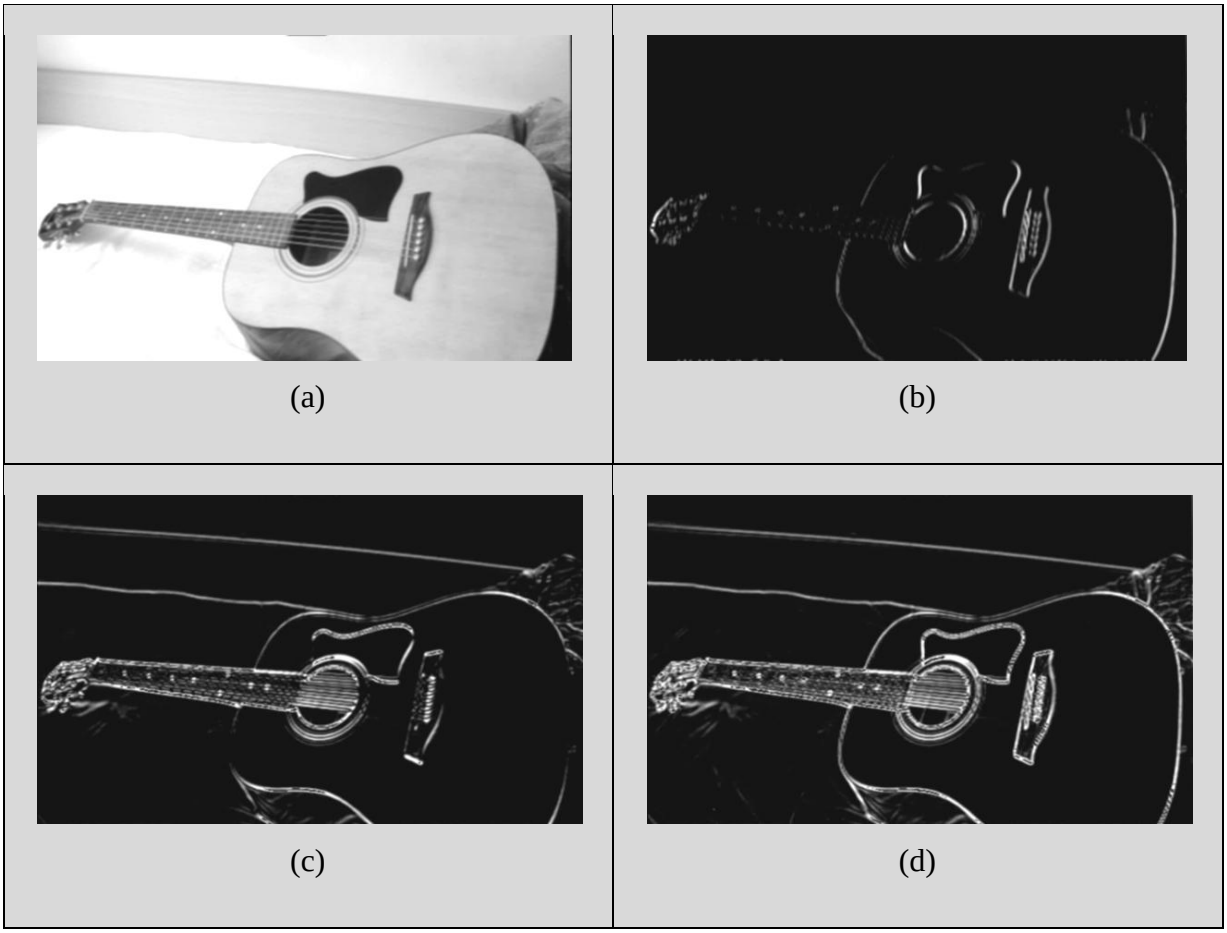
Slika 22. Primjena Sobelovog operatora: (a) Originalna slika, (b) Vertikalni Sobelov operator (c) Horizontalni Sobelov operator (d) Kombinacija dva operatora za općenitu detekciju rubova

4.2.2.2. Prewittov operator

Koriste se slične jednadžbe za S_x i S_y kao i za Sobelov operator:

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, S_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Na slici 23 prikazani su rezultati dobiveni korištenjem Prewittovog operatora nad ulaznom slikom.



Slika 23. Primjena Prewittovog operatora: (a) Originalna slika, (b) Vertikalni Prewittov operator (c) Horizontalni Prewittov operator (d) Kombinacija dva operatora za općenitu detekciju rubova

4.2.2.3. Laplacian

Znamo da su rubne vrijednosti u slici predstavljene vršnim vrijednostima prve derivacije. Za očekivati je da će druga derivacija u tim točkama imati vrijednost nula. Za detektiranje rubova pomoću druge derivacije treba pronaći mjesta prolaska kroz nulu (eng. *zero crossing*).

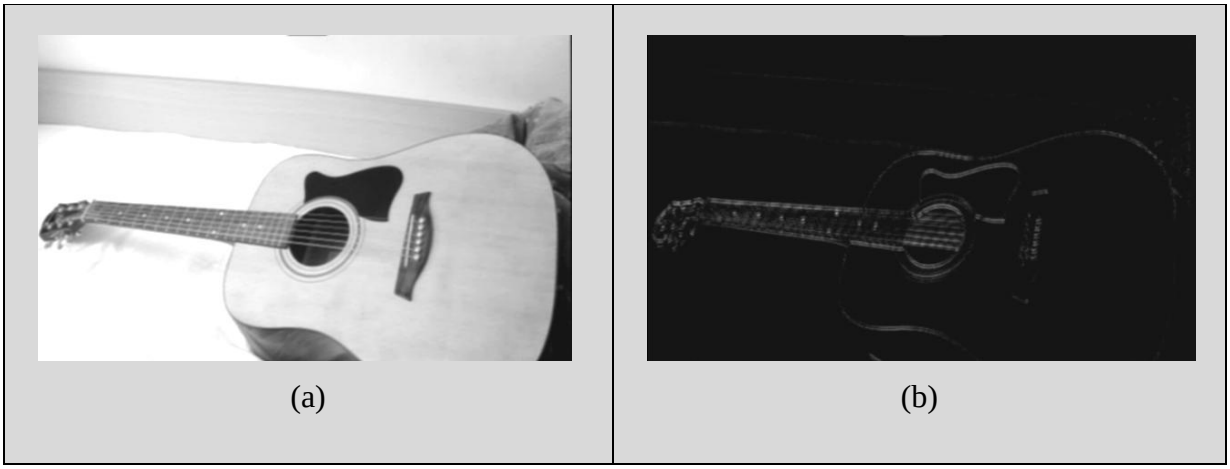
Laplacian je dvodimenzionalni ekvivalent druge derivacije funkcije *f(x,y)*:

$$\nabla^2 f = \frac{\delta^2 f}{\delta x^2} + \frac{\delta^2 f}{\delta x^2}$$

Korištenjem diferencija za aproksimaciju ovog izraza dobiva se maska:

$$\nabla^2 \approx \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Slika 24 prikazuje rezultate dobivene nakon primjene Laplaceovog operatora nad ulaznom slikom.



(a) (b)

Slika 24. Primjena Laplaceovog operatora: (a) Originalna slika (b) Slika nakon primjene operatora

4.2.3. Filtriranje slike

Ulazna slika često sadrži artefakte koji su nepoželjni u postupcima obrade slike, kao što je pojava šuma. Šum je slučajna varijacija intenziteta i boje svjetlosti na slikama, a javlja se kao nusprodukt senzora i sklopovlja za obradu u skenerima i kamerama. U frekvencijskoj domeni, šum će dodati nepoželjne više harmonike. Uklanjanje šuma moguće je izvesti pomoću niskopropusnog filtra koji uklanja visoke frekvencije. Slika nakon takvog postupka obično izgleda zamućeno zbog nedostatka viših harmonika, no odziv operatora za detekciju rubova poboljšava se zbog uklonjenih lažnih pozitiva.

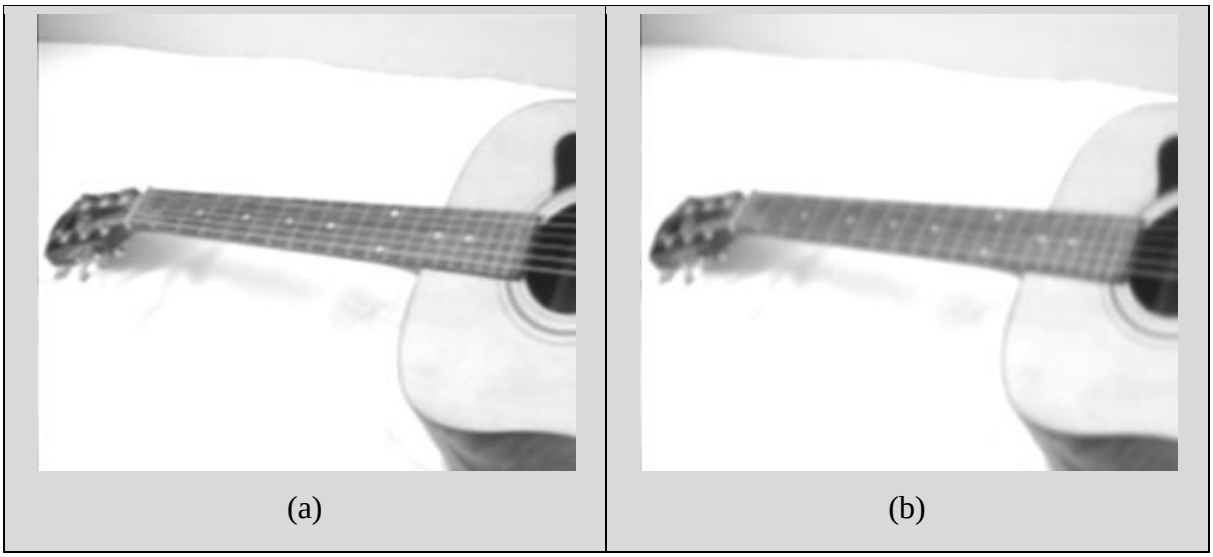
Zaglađivanje slike najčešće se izvodi primjenom Gaussove maske. Vrijednosti unutar matrice dobivaju se numeričkom aproksimacijom dvodimenzionalne Gaussove funkcije srednje vrijednosti nula i odabrane standardne devijacije.

Za veličinu maske od 3x3 piksela, koristi se sljedeća matrica:

$$G = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Budući da težine maske ne daju sumu 1, slikovni elementi nakon konvolucije moraju biti normalizirani. U gornjem primjeru, suma svih elemenata matrice iznosi 16, pa se točne vrijednosti dobivaju dijeljenjem s 16.

Slika 25 prikazuje uvećani detalj slike prije i poslije primjene Gaussovog filtra za zaglađivanje slike.



Slika 25. Primjena Gaussovog filtra: (a) Uvećani detalj originalne slike (b) Detalj nakon primjene filtriranja Gaussovom maskom

4.3. Ulančavanje operacija

Mnogi postupci računalnog vida izvode se sekvencijalnim provođenjem operacija nad ulaznom slikom. Kao primjer možemo uzeti napredniju detekciju rubova, prikazanu na slici 26.

Slika 26. Napredniji postupak detekcije rubova

Prvi korak u postupku jest filtriranje ulazne slike, najčešće Gaussovom maskom, u svrhu uklanjanja šuma. Nakon filtriranja slijedi korak naglašavanja rubova, primjerice korištenjem gradijentnih operatora, tj. maski za detektiranje ruba. U prošlom potpoglavlju opisana je izvedba operacije filtriranja i gradijentnih operatora korištenjem konvolucije.

Na kraju postupka primjenjuje se nelinearni operator koji računa iznos svakog elementa kombinirajući rezultate više različitih maski. Na primjer, ako su korištene maske

za naglašavanje rubova u horizontalnom i vertikalnom smjeru, konačni iznos gradijenta možemo izračunati korištenjem više različitih izraza.

$$G[i,j] = \sqrt{{S_x}^2 + {S_y}^2}$$

$$G[i,j] = |S_x| + |S_y|$$

$$G[i,j] = max\{S_x, S_y\}$$

Sekvencijalno izvođenje operacija u fiksnom redoslijedu može se predstaviti protočnom strukturom, u kojoj je izlaz svake prethodne razine spojen na ulaz iduće razine za obradu. Ovakav način opisa pogodan je za implementaciju u sklopovlju, o čemu će biti više riječi u daljnjem tekstu.

4.4. Algoritmi više razine

Složeniji algoritmi računalnog vida osim operacija nad pojedinim slikovnim elementima i susjedstvima često zahtijevaju višestruke prolaze kroz sliku, uz pamćenje različitih parametara u dodatnim podatkovnim strukturama. Primjerice, iterativni algoritam za traženje povezanih komponenti prolazi dva puta kroz sliku. U prvom prolazu pojedinim slikovnim elementima dodjeljuju se oznake, a u posebnoj tablici bilježe se ekvivalentne oznake. Drugi prolaz izjednačava ekvivalentne oznake piksela.

Već je spomenuto kako algoritmi niže razine obrađuju velike količine podataka i provode jednostavnije operacije, što ih čini idealnima za implementaciju u sklopovlju. Svaka viša razina redovito radi s manjom količinom podataka ali uz korištenje sofisticiranijih i kompleksnijih algoritama. Primjerice, algoritam niže razine koji određuje vertikalnu projekciju slike veličine N*N piksela radi s upravo tolikim brojem elemenata. Algoritam više razine koji provodi segmentaciju korištenjem vertikalne projekcije radit će samo s elementima vektora projekcije, kojih ima N puta manje. Operacije u nižoj razini ovakvog algoritma svode se na zbrajanje, dok viša razina obrade podrazumijeva korištenje raznih aritmetičkih operacija, usporedbi, memorijskih mjesta itd.

Iz prethodnog teksta može se zaključiti da je za izvođenje operacija više razine potrebna veća fleksibilnost i programirljivost operacija. Za takve zahtjeve logički se nameće izbor mikroprocesora kao jedinice za izvedbu algoritama.

5. Implementacija u sklopovlju

U prošlom poglavlju opisani su neki osnovni algoritmi računalnog vida, uz određenu količinu matematičke pozadine. Ovo poglavlje prikazuje načine implementacije dotičnih algoritama u FPGA sklopovlju uz pomoć jezika VHDL i paradigme simultanog oblikovanja sklopovlja i programske podrške.

5.1. Microblaze: kontroler ili procesor?

U drugom poglavlju opisane su karakteristike Microblaze procesora s mekom jezgrom. Iz specifikacija je očito da taj procesor nije u mogućnosti procesirati goleme količine podataka, no ono što mu nedostaje u brzini i propusnosti, nadoknađuje velikom fleksibilnošću. Alati tvrtke Xilinx za razvoj sustava baziranih na Microblaze procesoru pružaju mogućnost razvoja programske podrške za procesor u programskom jeziku C. Napisani program moguće je prevesti u asemblerski kod za Microblaze i ugraditi u konfiguracijsku datoteku za FPGA uređaj.

Najjednostavniji pristup problemu implementaciji algoritama računalnog vida u Microblaze procesorskom sustavu bilo bi jednostavno prepisivanje algoritma u čisti C kod koji bi se potom izvršavao na samom procesoru. Glavni nedostatak ovog pristupa je što uopće ne iskorištava mogućnost ubrzavanja operacija njihovim izvođenjem u sklopovlju, ni mogućnost paralelizacije koja je inherentna FPGA sustavima. Program na Microblaze procesoru izvodio bi se puno sporije nego na nekom procesoru opće primjene zbog velike razlike u procesorskoj moći.

Drugi pristup jest razvoj potpuno primjenskog sklopovlja u VHDL-u, kojeg bi se moglo kontrolirati pomoću Microblaze procesora. Microblaze bi u ovakvom scenariju izvodio najosnovnije operacije upisa podataka u registre periferija i komunikacije s korisnikom. Nedostatak ovakvog pristupa jest otežana implementacija složenijih algoritama, budući da bi se morali projektirati namjenski sklopovi samo za tu svrhu.

Kompromis između gornja dva pristupa jest korištenje namjenskog sklopovlja za operacije s velikim količinama podataka, dok se na Microblaze procesoru izvodi algoritam obrade više razine. Ovaj pristup ublažava problem brzine procesora, budući da operacije koje procesor izvodi nisu podatkovno zahtjevne. Algoritam za procesor moguće je napisati

u višem programskom jeziku, a razvoj namjenskog sklopovlja olakšan je zbog smanjene složenosti operacija koje rade s većom količinom podataka.

5.2. Komunikacija s periferijom

Periferni sklopovi spojeni na Microblaze procesorski sustav različiti su po namjeni. Neki od njih zahtijevaju veće količine podataka koje se prenose između periferije i procesora, a neki tek povremeno čitanje ili upis u registar. Za spajanje manje zahtjevnih sklopova, čija primjena nije vremenski kritična, dovoljna je OPB ili PLB sabirnica. Ako je brzina izmjene podataka između procesora i korisničkog sklopa vremenski kritična ili se prenosi veća količina podataka, preporučuje se korištenje FSL sabirnice.

Primjerice, računanje jednodimenzionalne inverzne diskretne kosinusne transformacije (korištene u algoritmu dekodiranja JPEG slike) iznimno je računski zahtjevna operacija. No, moguće je dizajnirati namjenski sklop koji prima 8 vrijednosti kao ulaz i jednu kao izlaz transformacije, uz određeno kašnjenje. Korištenje FIFO strukture (kao što je FSL sabirnica) za prosljeđivanje vrijednosti prema periferiji i od periferije u ovom slučaju nameće se kao najbrže i najfleksibilnije. Dok periferija računa rezultat, procesor može neometano izvoditi ostatak programa.

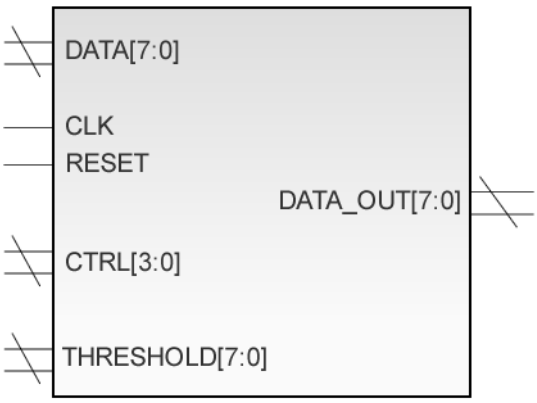
S druge strane, sklop koji provodi obradu slike u realnom vremenu (npr. binarizacija) i prosljeđuje izlaz na izlazne pinove FPGA sklopa ne zahtijeva učestalu komunikaciju s procesorom, budući da podaci ne idu kroz procesor. Spajanjem takve periferije na OPB sabirnicu postiže se jednostavna komunikacija s procesorom koja nije vremenski kritična. U takvom scenariju procesor igra ulogu kontrolera koji postavlja parametre perifernog sklopa.

5.3. Sklopovska obrada slike

U potpoglavlju 3.4. opisan je način dohvata piksela iz toka video podataka. Prikazan je i postupak generiranja sinkronizacijskih signala. U ovom potpoglavlju opisuje se sklop za obradu slike u realnom vremenu koji koristi prethodno definirane signale kao podatke i takt.

5.3.1. Sučelje sklopa

Na slici 27 prikazano je vanjsko sučelje sklopa za obradu slike.



Slika 27. Sučelje sklopa za obradu slike

Osmobitni signal *data* predstavlja sabirnicu podataka na kojoj se pojavljuju pikseli slike. Njihovo pojavljivanje sinkronizirano je signalom *clk*. Pojava rastućeg brida *clk* signala označava da su novi podaci stabilni na *data* sabirnici. Ove signale generira sklopovlje za dohvat iz toka video podataka.

Sklop može biti resetiran pojavom jedinice na *reset* ulazu. *Ctrl* ulazi služe za kontrolu sklopa. Pomoću njih određuje se operacija koju sklop izvršava. *Threshold* predstavlja 8-bitnu vrijednost praga s kojom će se uspoređivati slikovni elementi u postupku binarizacije.

Izlaz iz sklopa je također 8-bitni podatkovni signal, sinkroniziran s *clk* ulazom. Kašnjenje kroz sklop za obradu ekvivalentno je prolazu dvije linije piksela (dakle 1440 perioda video takta), ali protočna struktura osigurava sinkrono funkcioniranje sklopa. Kad se protočna struktura napuni, vrijeme potrebno za obradu svakog novog piksela jednako je jednom periodu takta.

5.3.2. Struktura sklopa

Za provođenje operacije konvolucije u jednom taktu potrebno je u određenom trenutku imati spremne podatke o svim pikselima u susjedstvu. Za masku veličine 3x3 to znači da je osim promatranog piksela potrebno znati koji pikseli se nalaze u prethodnoj i idućoj liniji. Budući da pikseli na ulaz sklopa dolaze jedan nakon drugog, očito je da će

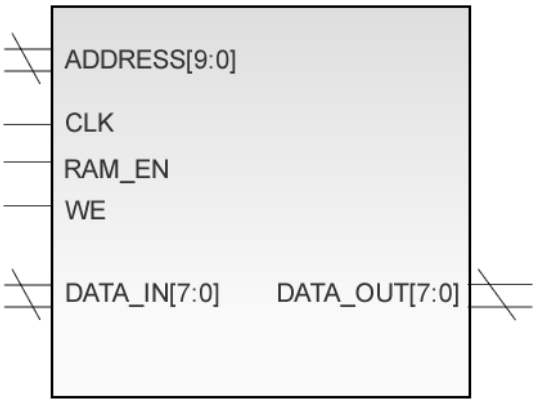
morati biti privremeno pohranjeni unutar sklopa za obradu kako bi bili iskorišteni za računanje konvolucije.

Prirodna struktura za pohranu serijskih podataka je FIFO međuspremnik. Pikseli koji su prvi došli na ulaz sklopa bit će oni koji prvi izlaze. Veličina ulazne slike iznosi 720x576 piksela. Kad bi htjeli držati cijelu sliku u međuspremniku, količina zauzete memorije iznosila bi 720*576*8 bita = 414 720 B = 406 KB, što je daleko više od raspoložive memorije (64 KB BRAM memorije u XC3S1200E FPGA sklopu), pa je pamćenje cijele slike moguće jedino u eksternoj memoriji (SDRAM ili Flash). No, budući da je maska za konvoluciju konačne širine, dovoljno je pamtiti samo dio slike, tj. dvije linije za masku veličine 3x3.

5.3.2.1. Linijski međuspremnici

FIFO međuspremnik koji sadrži jednu liniju slike naziva se linijski međuspremnik (eng. *line buffer*). Za primjenu u obradi slika po PAL standardu, veličina takvog međuspremnika iznosi 720 piksela, tj. 720 bajtova. Za implementaciju međuspremnika u Spartan-3 arhitekturi najpogodniji su blokovi BRAM memorije. Kao što je već spomenuto, svaki blok RAM ima kapacitet od 18432 bitova, što je i više nego dovoljno za spremanje jedne linije slike. Osim toga, blok je moguće konfigurirati tako da se istovremeno u memoriju upisuje nova i čita prethodna vrijednost.

Sučelje linijskog međuspremnika prikazano je na slici 28. *Clk* je signal takta, *ram_en* omogućava memoriju, a *we* (*write enable*) omogućava pisanje u memoriju. Adresna sabirnica označena je s *address*, a podatkovnih sabirnica su dvije: *data_in* za upisivanje u memoriju i *data_out* za čitanje iz memorije.



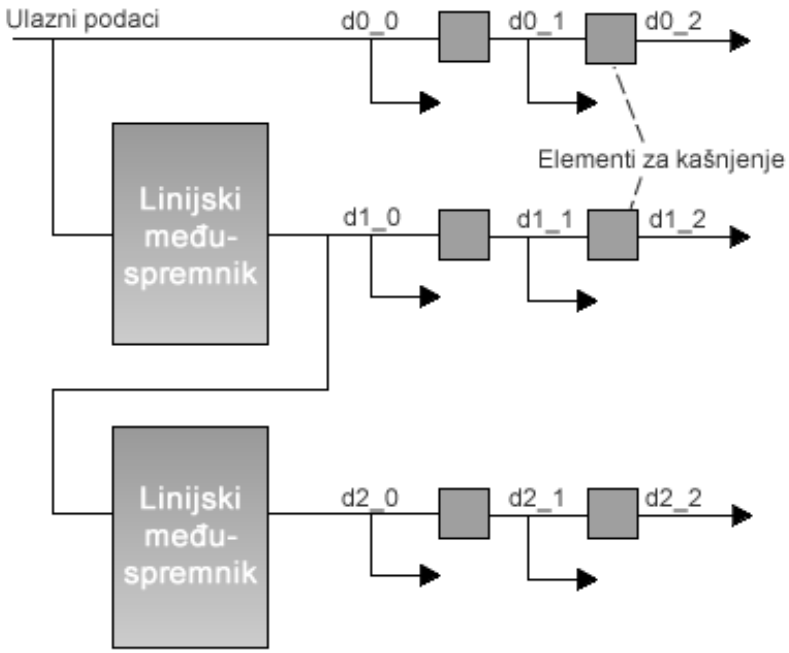
Slika 28. Linijski međuspremnik

Memorija je sinkrona, te se na rastući brid takta provjeravaju signali za omogućavanje. Ako je *ram_en* u logičkoj nuli, neće doći ni do čitanja ni do upisivanja. U suprotnom, na izlaznoj podatkovnoj sabirnici pojavit će se podaci zapisani na adresi određenoj riječju s adresne sabirnice. Dodatno, ako je *we* u logičkoj jedinici, nakon čitanja će se na istu adresu upisati novi podaci s podatkovne sabirnice.

Za potrebe korištenja Blok RAM memorije kao linijskog međuspremnika, na ulaze *ram_en* i *we* uvijek će se dovoditi logička jedinica. Osim toga, potrebno je osigurati da memorija koristi samo 720 adresnih mjesta. U tu svrhu koristi se sinkrono brojilo programirano da broji do 720, te se nakon toga resetira na početnu vrijednost (nula). Izlaz ovog brojila spojen je na adresnu sabirnicu memorijskog međuspremnika, što osigurava da će adrese na ulazu u Blok RAM uvijek biti od 0 do 719. Iz ovoga slijedi da širina adresne sabirnice memorije mora biti 10 bita (2^{10}=1024).

5.3.2.2. Implementacija

Općenito, za implementaciju konvolucije s veličinom maske NxN potrebno je instancirati N-1 linijskih međuspremnika. Zbog ograničenog broja Blok RAM memorija u Spartan-3 sklopu, bit će raspravljena samo implementacija sklopa koji podržava veličinu maske 3x3. Poopćenje se izvodi na trivijalan način, dodavanjem dodatnih međuspremnika u kaskadu. Arhitektura sklopa prikazana je na slici 29.



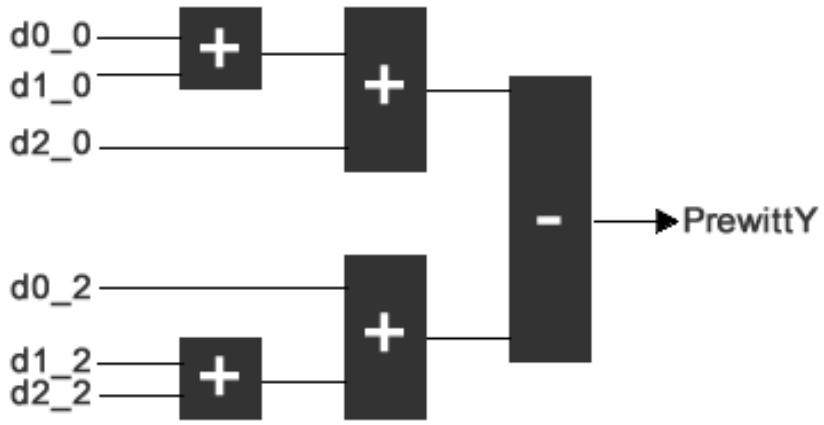
Slika 29. Kaskadiranje linijskih međuspremnika

Ulaz sklopa spojen je na ulaz prvog linijskog međuspremnika. Izlaz prvog međuspremnika spojen je na ulaz drugog, te je dobivena protočna struktura dubine 1440 piksela. Na ulazni signal i na izlaze oba međuspremnika spojena su po dva bloka za kašnjenje u seriju. Svaki blok za kašnjenje zakasnit će ulazni signal za jedan period takta. Ovi blokovi za kašnjenje pamte susjedne piksele u jednoj liniji.

Promotrimo signal d2_2 na slici. Pretpostavimo da se na njemu nalazi prvi piksel slike. Signal d2_1 nalazi se ispred bloka za kašnjenje, što znači da je to drugi piksel u protočnoj strukturi. Analogno tomu, d2_0 predstavlja treći piksel slike. Signal d1_2 je signal koji kasni točno 720 taktova za d2_2, što znači da se radi o pikselu koji se nalazi u drugom retku i prvom stupcu originalne slike. Slično se pokazuje i za ostale piksele. Signali d0_0, d0_1,...d2_2 predstavljaju 3x3 susjedstvo srednjeg piksela.

Budući da svih devet signala mijenja stanje sinkrono s taktom, moguće je dizajnirati kombinacijske sklopove koji koriste tih devet signala kao ulaz. Na primjer, za implementaciju Prewittovog operatora u x-smjeru potrebno je zbrojiti tri piksela u trećem stupcu matrice susjedstva i oduzeti vrijednosti tri piksela u prvom stupcu matrice susjedstva.

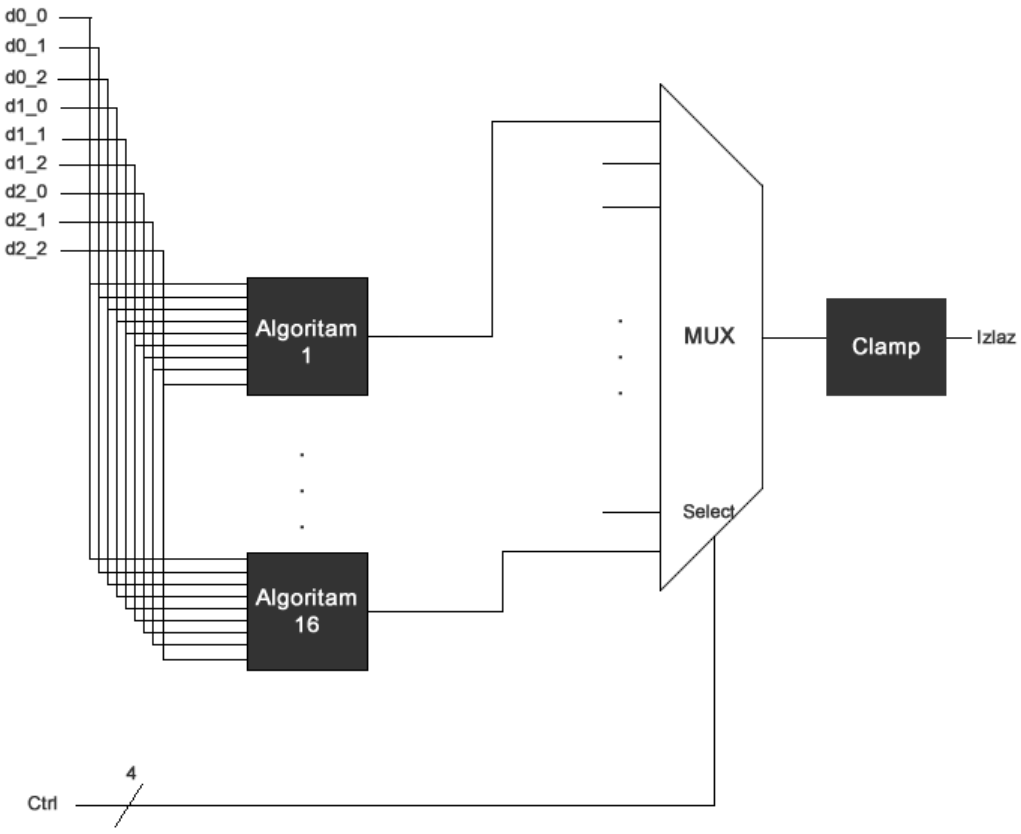
Koristeći ponašajni opis u VHDL-u, moguće je napisati odgovarajući kombinacijski izraz koji će sintetizator prevesti u stvarno sklopovlje. Moguća implementacija Prewittovog operatora prikazana je na slici 30. Konačni izgled sklopovlja bit će drukčiji zbog raznih optimizacija koje se provode u postupku sinteze (eliminacija višestrukih signala, uklanjanje nedostupnih puteva podataka itd.)



Slika 30. Arhitektura sklopa za implementaciju Prewittovog operatora u y-smjeru

Operacija binarizacije ne zahtijeva nikakve memorijske elemente, te ima samo jedan ulaz. Ovaj ulaz uspoređuje se s fiksnom vrijednošću koja se nalazi na ulazu sklopa (signal *threshold*) pomoću 8-bitnog komparatora.

Implementacija svake operacije nad susjedstvom iz 4. poglavlja može biti izvedena kao zaseban sklop s devet ulaza i jednim izlazom. U ovisnosti o kontrolnim bitovima na ulazu sklopa za računalni vid, na ulaz će se propustiti rezultat samo jedne operacije. Shema ovakvog rješenja prikazana je na slici 31.



Slika 31. Multipleksiranje operacija

Nakon izlaznog multipleksora nalazi se dodatni element za ograničavanje izlaza. Budući da PAL signal definira da je najniža vrijednost signala 16, a najviša 235, vrijednosti koje su izvan ovih granica treba zaokružiti na najbližu dopuštenu vrijednost. Izlaz iz sklopa bit će obrađeni pikseli slike unutar dopuštenih granica PAL standarda koji kasne za ulaznim pikselima za točno dva reda slike. Dodatno sklopovlje bit će potrebno da bi se od ovakvog izlaza ponovo generirao PAL signal.

5.4. Brzina rada i sklopovski zahtjevi

Sklop za obradu slike početno je razvijen zasebno u VHDL-u, uz korištenje simulatora za verifikaciju ispravnosti. U hipotetskoj situaciji gdje je takav sklop jedini implementiran na FPGA sklopu, sintetizator dojavljuje iskorištenje FPGA sklopa od 360 odsječaka (4% od ukupnog broja), i 4 Blok RAM memorije (14%). Maksimalna teoretska frekvencija sklopa na kojoj bi sklop mogao raditi iznosila bi oko 215 MHz.

Nakon ugradnje sklopa kao periferije u Microblaze sabirnički sustav, povećava se količina zauzetih resursa zbog dodatne logike zadužene za komunikaciju preko OPB sabirnice. Nakon sinteze, zauzeće periferije iznosi 672 odsječka (7%) i 4 Blok RAM memorije. Smanjuje se i maksimalna moguća frekvencija takta i iznosi oko 103 MHz.

Cjelokupni Microblaze procesorski sustav, koji osim procesora uključuje i logiMEM i logiFLASH memorijske kontrolere, generator takta, IIC kontroler, te 3 instance sklopa računalnog vida implementiran je na FPGA sklopu. Zauzeće sklopa prikazano je tablicom 3.

Tablica 3. Zauzeće raspoloživih resursa na FPGA sklopu

	Zauzeto	**Dostupno**	**Postotak iskorištenja**
Odsječci (*slices*)	5060	8672	58%
BRAM	24	28	85%
Množila	3	28	10%
DCM	5	8	62%
I/O blokovi	96	190	50%

U usporedbi s procesorima opće namjene, čiji takt dostiže frekvencije i preko 3 GHz, može se činiti da su FPGA uređaji iznimno spori. No, ako se usporedi broj taktova koji je potreban za obavljanje operacija, razlika se drastično smanjuje. Primjerice, sklop za obradu slike razvijen u ovom radu može obraditi susjedstvo piksela u samo jednom taktu, dok bi ekvivalentna programska operacija zahtjevala veliki broj procesorskih instrukcija. Nadalje, uz dovoljnu količinu resursa u FPGA sklopu, moguće je postići paralelizacije takve razine da se cijela slika obrađuje u samo jednom periodu takta.

Sudeći po trendovima na tržištu, za očekivati je da će se maksimalne frekvencije takta FPGA uređaja nastaviti povećavati, što će im dati dodatnu prednost. Također je bitno zamijetiti da manji takt i veći broj operacija po periodu takta povlači i manju potrošnju i zagrijavanje sklopa.

6. Generiranje video signala

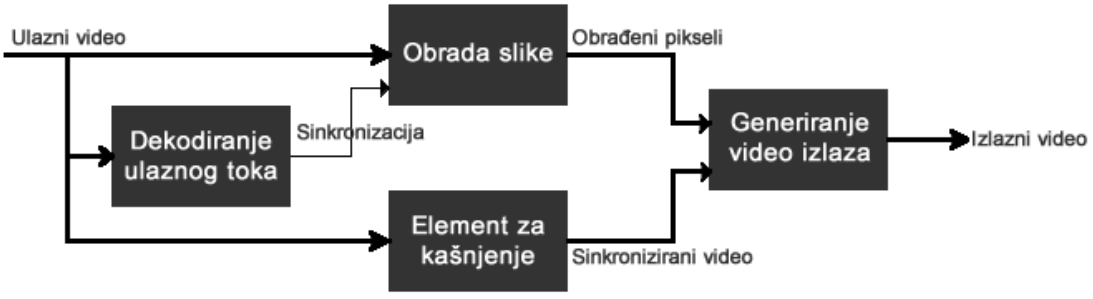
U prošlom poglavlju bilo je riječi o implementaciji operacija računalnog vida nad ulaznom slikom. U ovom poglavlju bit će opisano kako se obrađeni podaci umeću u tok video podataka po PAL standardu.

6.1. Sinkronizacija puteva podataka

Kao što je opisano u poglavlju 3, za dekodiranje informacija iz toka video podataka moguće je koristiti konačne automate. Sklop za obradu slike na ulazu prima originalni video uz signale za sinkronizaciju koje generiraju automati. Zbog protočne prirode obrade podataka, izlazni podaci iz sklopa za obradu slike kasne za ulaznim podacima za točno dvije linije slike.

Budući da na ulazu sklopa već postoji potpuni tok video podataka, idealna obrada u realnom vremenu bi jednostavno mijenjala podatke koji opisuju intenzitete slikovnih elemenata u originalnom toku. Ako se ograničimo na obradu slika sive razine, sve komponente boje možemo ostaviti jednakima originalu ili ih postaviti u neutralnu vrijednost (0x80). Sinkronizacijske oznake morale bi ostati nepromijenjene.

Problem nastaje u sinkronizaciji između originalnog video signala i obrađenih podataka koji pune protočnu strukturu. U trenutku kad na izlaz sklopa za obradu stigne vrijednost piksela u redu *i*, u originalnom toku nalazimo se već u retku *i+2*. Kako bi sinkronizirali ova dva toka podataka, originalni signal propuštamo kroz element za kašnjenje. Radi se o protočnoj strukturi ekvivalentnoj međuspremnicima linije, samo s dvostrukim kapacitetom. Slika 32 prikazuje organizaciju puteva podataka.



Slika 32. Put podataka u sustavu

Sklop za generiranje video signala prima originalni video koji kasni dvije linije i podatke iz sklopa za obradu slike, koji su međusobno sinkronizirani. Zaduženje ovog sklopa jest da multipleksira obrađene piksele u originalni video tok, te da ih šalje na izlaz. Arhitektura sklopa za generiranje video signala slična je sklopu za dohvat podataka. Također sadrži dva konačna automata za dekodiranje videa, no umjesto generiranja sinkronizacijskih signala kod detektiranja piksela, na izlaz se propušta piksel iz faze obrade. Dakle, ovaj sklop djeluje kao „pametna sklopka“, čije se djelovanje može pojednostavljeno opisati na ovaj način:

$$Izlaz = \begin{cases} \textbf{Obrađeni piksel}, ako\ je\ Ulaz = Intenzitet\ piksela \\ \textbf{Ulaz}, ako\ je\ Ulaz \neq Intenzitet\ piksela \end{cases}$$

Izlazni signal u potpunosti je u skladu s PAL standardom, i može se proslijediti na ulaz nekog drugog sklopa koji očekuje digitalni PAL signal, a može se i spojiti na izlazne pinove FPGA sklopa. Digitalni signal je potom moguće kodirati u analogni signal, npr. kompozitni ili S-Video.

6.2. Ulančavanje sklopovlja

Razvijeni periferijski sklop na ulazu prima digitalni video, te ga generira na svom izlazu. Budući da su ulaz i izlaz potpuno kompatibilni, moguće je spojiti više sklopova za obradu u cjevovod. Unutar cjevovoda nalaze se instance istog sklopa, sve spojene na OPB sabirnicu. Procesor može zasebno adresirati svaku instancu i isprogramirati je za različitu operaciju. Ovakvim pristupom moguće je direktno implementirati sekvencijsku obradu slike, ali je vrijeme izvođenja značajno brže jer se opet, u principu, radi o protočnoj strukturi. Npr. dok jedan sklop izvodi transformaciju nad prve tri linije slike, drugi sklop provodi neku drukčiju transformaciju nad sljedeće tri linije. Budući da se operacije opisane u poglavlju 4. mogu izvesti u jednom periodu takta, vrijeme obrade svakog piksela i dalje ostaje konstantno, ali se obrada paralelizira.

U demonstracijskom projektu implementirane su tri instance istog sklopa za obradu slike i organizirane u cjevovod. Programska podrška omogućava da korisnik zasebno isprogramira svaki sklop. Primjerice, moguće je postići naprednu detekciju rubova ako se prvi sklop isprogramira za zaglađivanje slike Gaussovim filtrom, a drugi za detekciju rubova Sobelovim algoritmom.

Razina paralelizacije algoritma ovisi o njegovoj pogodnosti pretvorbe u protočnu strukturu i količinom dostupnog sklopovlja u FPGA čipu. Uz dovoljno velike memorijske međuspremnike može se postići da npr. jedan sklop obrađuje jednu sliku, a svaki sljedeći narednu sliku iz slijeda. Za primjenu u sličnim zahtjevnim aplikacijama pogodni su FPGA sklopovi iz Virtex porodice tvrtke Xilinx, gdje količina Blok RAM memorije iznosi i do 38304 Kb [19]. Za usporedbu, Spartan-3 FPGA korišten u izradi ovog rada sadrži ukupno 504 Kb Blok RAM memorije.

6.3. D/A konverzija i video izlaz

Obrađeni video signal nalazit će se na izlaznim pinovima FPGA sklopa. Kako bi taj signal pretvorili u oblik pogodan za prijenos ili prikaz na analognim uređajima (npr. na TV prijemniku), potrebno je izvršiti digitalno/analognu konverziju. Za postupak kodiranja digitalnih podataka o svjetlini i boji u analogne CVBS i S-Video signale na logiTAP razvojnoj platformi zadužen je Philipsov digitalni video enkoder oznake SAA7121H. Njegove osnovne funkcije sastoje se od generiranja vala podnosioca (eng. *subcarrier*), moduliranja boje i ubacivanja sinkronizacijskih signala. Moguće ga je konfigurirati preko I^2C sučelja, a podržava veliki broj postavki i parametara, kao što su: određivanje razine crne boje i perioda zatamnjenja, frekvencija vala nosioca boje, korištenje PAL ili NTSC standarda, način dekodiranja sinkronizacijskih markera itd.

Izlaz iz video enkodera spojen je na analogne S-video i kompozitne priključke. Za potrebe verifikacije razvijenog sklopovlja, izlaz je testiran s dva uređaja: Quadro TV prijemnikom (preko CVBS-SCART adaptera) i „NOT LV5E“ USB karticom za dohvat videa na računalo.

7. Zaključak

Cilj ovog rada bio je prikaz mogućnosti sklopovske implementacije karakterističnih operacija računalnog vida u svrhu njihove primjene u ugradbenim sustavima. Naglasak je stavljen na FPGA uređaje, budući da se radi o inherentno paralelnim sustavima koji omogućavaju postizanje većih brzina obrade njenom podjelom na više simultanih ili ulančanih operacija.

Dobiveni rezultati pokazuju da je u FPGA sklopu niske cijene u relativno kratkom vremenu moguće dizajnirati sustav koji provodi osnovne operacije računalnog vida. Obrada se pri tome može odvijati u realnom vremenu, što dokazuje da su ugradbeni sustavi primjenjivi u područjima računalnog vida koja se bave obradom video signala. Mogućnost rekonfiguracije FPGA sklopovlja dodatno omogućuje jednostavnu nadogradnju već razvijenog sklopa s novijom inačicom, čime postupak proizvodnje postaje sličniji razvoju programske podrške na osobnim računalima.

Nedostaci sklopovske implementacije postaju izraženiji s povećanjem kompleksnosti algoritama. Izraženiji zahtjevi za memorijskim prostorom često uvjetuju korištenje vanjskih memorijskih čipova, što povećava složenost izvedbe i cijenu sklopa. Potreba za razvojem kompleksnijeg sklopovlja povećava i vrijeme potrebno za razvoj. Osim povećanja složenosti samog sklopovlja rastu i vremena potrebna za sintezu i simulaciju digitalnog sustava, što vodi do duljih ciklusa u postupku dizajna digitalnih sustava.

Razvijeni digitalni sustav i prateća programska podrška dostupni u prilogu ovog rada omogućuju demonstraciju opisanih algoritama i mogućnosti primjene sustava.

8. Literatura

[1] Moreo, A.T.,Lorente, P.N., Valles F.S., Muro J.S., Andres C.F., **Experiences on developing computer vision hardware algorithms using Xilinx system generator**, Microprocessors and Microsystems 29 (2005), str. 411-419.

[2] Matrox Inc., http://www.matrox.com, datum pristupa: 14. svibnja 2010.

[3] Intel Inc., http://www.intel.com/technology/mooreslaw/index.htm, datum pristupa: 14. svibnja 2010.

[4] Wolfe, A., **Interview: Intel Talks Multicore Processor Trends**, Chip Tech, 2. ožujka 2009., http://www.informationweek.com/blog/main/archives/2009/03/interview_intel.html, 13. svibnja 2010.

[5] Krupnova, H., Saucier, G., **FPGA technology snapshot: current devices and design tools**, 11th International Workshop on Rapid System Prototyping (2000), str. 200-205.

[6] Edwards, M., Fozard, B., **Rapid prototyping of mixed hardware and software systems**, Euromicro Symposium on Digital System Design 2002, str. 118-125

[7] Wiatr, K., Jamro, E., **Implementation of image data convolutions operations in FPGA reconfigurable structures for real-time vision systems**, International Conference on Information Technology: Coding and Computing (2000) str. 152–157.

[8] Uzun, I.S., Bouridane, A.A.A., **FPGA implementations of fast Fourier transforms for real-time signal and image processing**, IEEE International Conference on Field Programmable Technology (FPT 2003), str. 102–109.

[9] Shijian, L., Li, G., Junzhu, Z., Feng, L., **A CPLD design of real time system for point targets detection in infrared image sequences**, Proceeding of the Fifth International Conference on ASIC (2003), str. 906–909.

[10] Jin, S., Cho, J., Pham, X.D., Lee, K.M., Park, S.-K., Kim, M., Jeon, J.W., **FPGA Design and Implementation of a Real-Time Stereo Vision System**, IEEE Transactions on Circuits and Systems for Video Technology, sv. 20, br. 1, 2010, str. 15-26.

[11] Vicente, A.G., Munoz, I.B., Molina, P.J., Galilea, J.L.L., **Embedded Vision Modules for Tracking and Counting People**, IEEE Transactions on Instrumentation and Measurement, sv. 58, br. 9, 2009, str. 3004-3011.

[12] Vučić, M. **Alati za razvoj digitalnih sustava**, predavanja, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu

[13] Xilinx, **Our History**,http://www.xilinx.com/company/history.htm, 16. svibnja 2010.

[14] Biran, D., **Reduce Cost, Risk and Time-to-Market with an FPGA-to-Structured ASIC Strategy,** http://rtcmagazine.com/articles/view/100669, 21. travnja 2010.

[15] Pencek, B., **Reconfigurable Application-Specific Computing: How Hybrid Computer Systems using FPGAs are Changing Signal Processing**, Industrial Embedded Systems, 5.svibnja 2010.

[16] Seeking Alpha, **Altera and Xilinx Report: The Battle Continues**, http://seekingalpha.com/article/85478-altera-and-xilinx-report-the-battle-continues, 17. srpnja 2008., datum pristupa: 10. svibnja 2010.

[17] Snow, J.F., **Digital Video Test Pattern Generators**, Application Note for Microblaze and Multimedia Development Board, 7. siječnja 2002.

[18] Maxim, **Video Basics**, http://www.maxim-ic.com/app-notes/index.mvp/id/734, 8. svibnja 2002., datum pristupa: 2. svibnja 2010.

[19] Xilinx Inc, **Virtex-6 FPGA Product Table**, http://www.xilinx.com/publications/prod_mktg/Virtex6_Product_Table.pdf , datum pristupa: 29. svibnja 2010.

[20] Mlinarić, H. **Ugradbeni računalni sustavi**, predavanja, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu

[21] Ribarić, S. **Računalni vid**, predavanja, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu

[22] Ribarić, S. **Projektiranje digitalnih sustava,** predavanja, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu

[23] Vučić, M. **Projektiranje ugradbenih računalnih sustava**, predavanja, Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu

[24] Neoh, H.S., Hazanchuk, A., **Adaptive Edge Detection for Real-Time Video Processing using FPGAs**, Altera Corporation, 2005.

[25] Vega-Rodriguez, M.A., Gomez-Iglesias, A., Gomez-Pulido, J., Sanchez-Perez, J.M., **Reconfigurable computing system for image processing via the internet**, Microprocessors and Microsystems 31 (2007), str. 498-515.

[26] International Telecomunication Union, **BT.656 : Interface for digital component video signals in 525-line and 625-line television systems operating at the 4:2:2 level of Recommendation ITU-R BT.601,** http://www.itu.int/rec/R-REC-BT.656/en, datum pristupa: 13. travnja 2010.

Izvedba algoritama računalnog vida za ugradbene sustave

Sažetak

Ovaj rad bavi se problematikom implementacije osnovnih algoritama računalnog vida u sklopovlju. Opisani su FPGA sustavi i procesori s mekom jezgrom te njihove prednosti i mane kod njihovog korištenja u svrhu obrade slike. Izloženi su principi dohvaćanja i obrade video signala u realnom vremenu. Nakon toga prikazani su neki osnovni algoritmi računalnog vida izvedeni u sklopovlju, kao što su binarizacija, filtriranje i detekcija rubova. Opisuju se problemi brzine rada i sinkronizacije signala, te se izlažu mogućnosti paralelizacije operacija u svrhu povećanja brzine.

Ključne riječi: Računalni vid, obrada slike, filtriranje, detekcija rubova, video, PAL, ugradbeni sustavi, VHDL, FPGA, procesori s mekom jezgrom, Microblaze

Implementation of computer vision algorithms for embedded systems

Summary

This paper deals with aspects of implementation of basic computer vision algorithms in hardware. FPGA systems and soft processors are described, as well as their advantages and disadvantages when used for image processing tasks. Real-time video signal retrieval and real-time processing principles are discussed. They are followed by a description of some basic computer vision algorithms implemented in hardware, such as binarization, filtering and edge detection. We discuss problems related to processing speed and signal synchronization. Finally, we show some possibilities in using the inherent parallelism of FPGA devices to increase the processing power of hardware.

Keywords: Computer vision, image processing, filtering, edge detection, real-time, video, PAL, embedded systems, VHDL, FPGA, soft processors, Microblaze

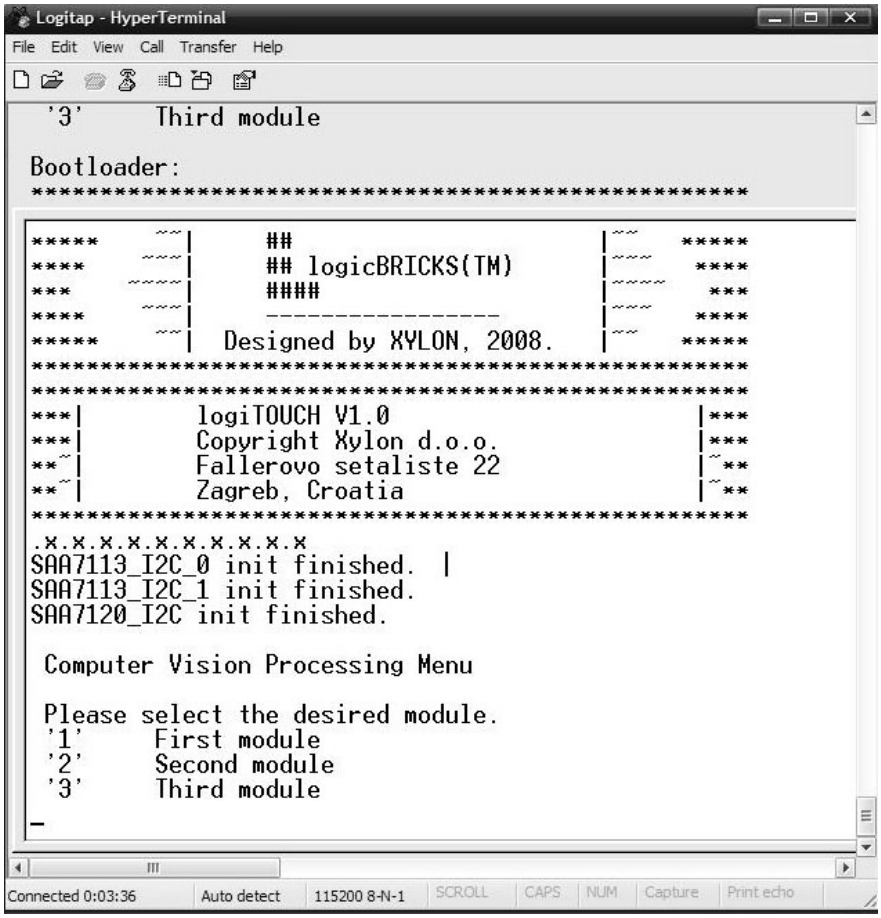
Dodatak A: Programska i sklopovska potpora

U sklopu ovog rada razvijen je programsko/sklopovski sustav pod nazivom *Aerilon.* Sklopovski dio sadržava VHDL dizajn komponente za obradu video signala u realnom vremenu, u sinergiji s Microblaze procesorskim sustavom. Programski sustav napisan je u programskom jeziku C++, a izvršava se na Microblaze procesoru. Sustav je razvijen za LogiTAP razvojnu platformu tvrtke Xylon, baziranu na Xilinx Spartan-3E FPGA čipu.

Sustav koristi IP module tvrtke Xylon (LogiMEM, LogiFLASH, Clock Module) te je za sintezu projekta potrebna važeća licenca.

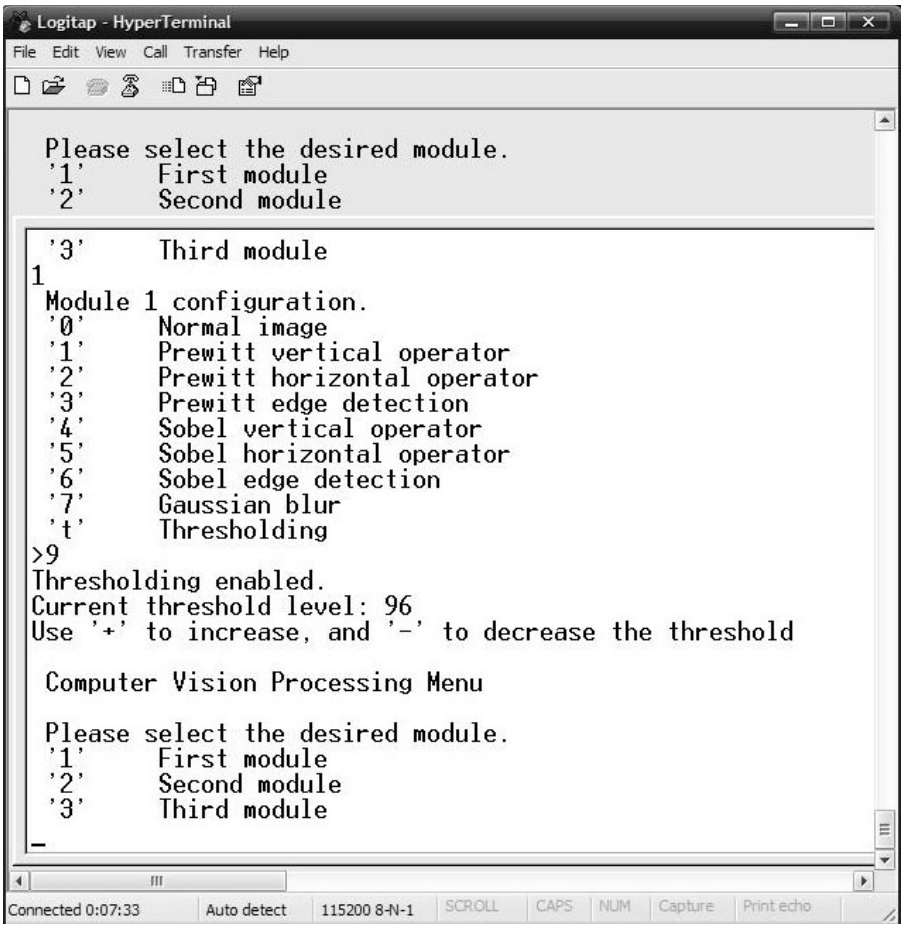
Opis sučelja programa

Budući da se programska potpora izvodi na Microblaze procesoru, standardni ulaz i izlaz preusmjereni su na RS232 vezu. Korisnik se može spojiti na sustav koristeći serijski kabel i terminal (npr. *Hyperterminal*). Slika 33 prikazuje tekstualno sučelje sustava.



Slika 33. Sučelje sustava na RS232 portu

Program se vrti u petlji i omogućava korisniku da adresira jedan od implementiranih modula računalnog vida. Nakon što je korisnik napravio odabir, prikazuje se podizbornik s mogućnostima podešavanja izabranog modula. Dostupni su svi algoritmi opisani u radu. Algoritam binarizacije ima dodatnu opciju za podešavanje vrijednosti praga (slika 34).



Slika 34. Podizbornik za odabir operacije

Program se izvodi u realnom vremenu, a promjene se izvode trenutno i vidljive su na video izlazu. Moduli su u sustavu spojeni u kaskadu tako da je za implementaciju cjevovoda dovoljno podesiti odgovarajuće module. Ukoliko je potrebna obrada u jednom koraku, ostali moduli mogu se podesiti na propuštanje originalne slike.