UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA
Odsek za elektrotehniku i računarstvo
Institut za računarstvo i automatiku
Katedra za računarsku tehniku i računarske komunikacije

Jedno rešenje video dekodera po H.264 preporuci na TMS320C64x DSP-u

- Diplomski rad iz predmeta-
- Logičko Projektovanje Računarskih Sistema -

Mentor:
doc. dr. Nikola Teslić

Kandidat:
Zoran Marčeta, E7825

Novi Sad, Decembar 2003.

UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA
KLJUČNA DOKUMENTACIJSKA INFORMACIJA

Redni broj: RBR	
Identifikacioni broj: IBR	
Tip dokumentacije: TD	*Monografski rad*
Tip zapisa: TZ	*Štampa*
Autor: AU	*Zoran Marčeta*
Mentor/Komentor MN	*doc. dr. Nikola Teslić*
Naslov rada: NR	*Jedno rešenje video dekodera po H.264 preporuci na TMS320C64x DSP-u*
Jezik publikacije: JP	*srpski*
Jezik izvoda: JI	*sprski*
Zemlja publikovanja: ZP	*Srbija i Crna Gora*
Uže geografsko područje: UGP	*Vojvodina*
Godina: GO	*2003.*
Izdavač: IZ	*Fakultet Tehničkih Nauka*
Mesto i adresa: MA	*21000 Novi Sad, Trg Dositeja Obradovića 5*
Fizički opis rada: FO	
Naučna oblast: NO	*Elektrotehnika*

Naučna disciplina: ND	*Računarska tehnika*
Predmetna odrednica/ključne reči: PO UDK	*Računarstvo, televizija, digitalne video komunikacije*
Čuva se: ČU	*U biblioteci Fakulteta Tehničkih Nauka*
Važna napomena: VN	
Izvod: IA	*U radu je dato jedno rešenje video dekodera po H.264 preporuci na TMS320C64x DSP-u*
Datum prihvatanja teme Od strane NN veća: DP	
Datum odbrane: DO	*30.12.2003.*
Članovi komisije: KO	
Predsednik: Član: Član:	*prof. dr. Vladimir Kovačević* *prof. dr. Miroslav Popović* doc. *dr. Nikola Teslić*

UNIVERSITY OF NOVI SAD
FACULTY OF TECHNICAL SCIENCIES
KEY WORDS DOCUMENTATION

Accession number: ANO	
Identification number: INO	
Document type: DT	*Monographic*
Type of record: TR	*Printed*
Author: AU	*Zoran Marčeta*
Menthor/Comenthor MN	*Nikola Teslić, Ph D*
Title: TI	*One solution of video decoder according to H.264 recommendation on the TMS320C64x DSP*
Language of text: LT	*Serbian*
Language of abstract: LA	*Serbian*
Country of publication: CP	*Serbia and Montenegro*
Locality of publication: LP	*Vojvodina*
Publication year: PY	*2003.*
Publisher: PU	*Faculty of Technical Sciences*
Publication place: PP	*21000 Novi Sad, Trg Dositeja Obradovića 5*
Physical description: PD	
Scientific field: SF	*Electrical Engineering*

Scientific discipline: SD	*Computer engineering*
Subject/Key words: SKW	*Computer engineering, television, digital video communications*
UC	
Holding data: HD	*the Library of the Faculty of Technical Sciencies*
Note: N	
Abstract: A	*In this paper is given one solution of video decoder according to H.264 recommendation on the TMS320C64x DSP*
Accepted by the Scientific Boatd on: ASB	
Defended on: DE	*30.12.2003.*
Thesis defend board: DB	
President:	*Vladimir Kovačević, Ph D*
Member:	*Miroslav Popović, Ph D*
Member:	*Nikola Teslić, Ph D*

SKRAĆENICE

DSP - *Digital Signal Processor*, Procesor za obradu digitalnih signala
DVD - *Digital Versatile Disk*, Digitalni višestrani disk
CPU - *Central Processor Unit*, Centralni procesor
FIR - *Finite Impulse Response*, Konačni impulsni odziv
GND - Oznaka za signal na nultom potencijalu

SADRŽAJ

1. Uvod

U zadnjih 10-15 godina, kodovanje video signala je napredovalo od relativno ezoteričnih istraživačkih tema sa malo primena u praksi do ključne tehnologije za širok raspon primena, od personalnih računara da televizije. Kao i mnogi drugi skorašnji tehnološki razvoji, masovna primena video kodiranja usledila je zbog konvergencije u mnogim drugim područjima. Jeftini i moćni procesori, brzi mrežni pristup, svuda prisutni Internet, istraživanja velikih razmera i napori u standardizaciji su doprineli razvoju tehnologije kodiranja video signala. Kodiranje je omogućilo pojavu novih "multimedijalnih" primena uključujući digitalnu televiziju, filmove na DVD-u, prenos video signala posredstvom Interneta i video konferencije.

Kodovanje sa kompresijom premošćava krucijalni jaz u svakoj od ovih primena: jaz između zahteva korisnika (video visokog kvaliteta, koji se može brzo isporučiti po razumnoj ceni) i ograničenih mogućnosti mreža za prenos i uređaja za skladištenje. Na primer, digitalni video signal televizijskog kvaliteta zahteva 216Mb skladišnog prostora ili prenosnog kapaciteta za svaku sekundu video signala. Prenos ovakvog tipa signala u realnom vremenu se nalazi izvan mogućnosti danas prisutnih komunikacionih mreža. Film koji traje 2 časa (nekompresovan) zahteva 194GB skladišnog prostora što je ekvivalentno 42 DVD ili 304 CD-ROM diska. Da bi digitalni video postao prihvatljiva alternativa svojim analognim prethodnicima (analogni televizori ili VHS video trake), neophodno je bilo razviti metode za redukovanje ili kompresiju ovog signala vrlo velike bitske brzine.

Rešavanje ovog problema trajalo je nekoliko decenija i zahtevalo je velike napore u istraživanju, razvoju i standardizaciji (ovaj posao se nastavlja zbog unapređivanja postojećih metoda i razvoja novih paradigmi kodiranja). Međutim, efikasne metode za kompresiju su sada čvrsto utvrđene komponente novih tehnologija digitalnih medija kao što su digitalna televizija i DVD-video. Poželjan sporedni efekat ovog razvoja je taj da je kompresija video signala omogućila mnogo novih primena vizuelnih komunikacija koje ranije nisu bile moguće. Neke od ovih novih oblasti primena su razviju brže od drugih (na primer, dugo predviđan bum u video konferenciranju se još nije desio), ali nema sumnje da će video kompresija biti tu. Svaki novi PC ima brojne ugrađene dodatke koji su da podrže i ubrzaju algoritme za video kompresiju. Najrazvijenije države su postavile rokove za prestanak emitovanja analogne televizije , posle koga će svim televizijskim prijemnicima biti potrebna tehnologija dekompresije da bi dekodovali i prikazali TV sliku. VHS video trake će konačno biti zamenjene DVD-om koji će moći da se reprodukuju na DVD uređajima za reprodukciju ili na PC-u. Srce u svim ovim primenama je video kompresor i dekompresor, ili video KOder/DEKer - video KODEK

Poslednji u nizu video kodeka H.264 ili MPEG-4 AVC (*Advanced Video Coding)* je razvijen kao odgovor na rastuće potrebe za visokom faktorima kompresije pokretnih slika u raznim primenama kao što su video konferenciranje, digitalni mediji za skladištanje, televizijski prenos, prenos preko Internet i komunikacije. Takođe je osmišljen da omogući upotrebu kodovanog video prikaza na fleksibilan način u raznolikim mrežnim okruženjima. Upotrebom ove Preporuke (Međunarodnog Standarda) omogućeno je da se digitalni video signal tretira kao

oblik kompjuterskih podataka, da se skladišti na mnoštvu različitih tipova medija, šalje i prima preko postojećih i budućih mreža i distribuira na postojećim i budućim prenosnim kanalima. Ovaj razvoj je omogućen zbog pada u ceni procesorske moći i memorije i napretka u tehnologiji video kodiranja.

Poseban doprinos implementaciji kodeka u praksi dao je razvoj posebne vrste procesora opšte namene, a to su procesori za digitalnu obradu signala. U njihovom projektovanju dat je akcenat na izvršavanje velikog broj matematičkih operacija u jedinci vremena, što se postiže upotrebom malog skupa instrukcija koje se izvršavaju u jednom ciklusu, razdvajanjem programske memorije i memorije za podatke i uvođenjem protočne strukture (pipeline) koja uvodi paralelizam u izvršavanje instrukcija. Takvi procesori su u mogućnosti da izvrše složene matematičke transformacije (kakve se javljaju u video koderima i dekoderima) u kratkom vremenskom intervalu.

Zadatak ovog projekta je realizacija jednog rešenja programske podrške video dekodera po H.264 preporuci na Texas Instruments 6415 DSP-u. Takođe je potrebno prikazati dobijenu video sekvencu. Kao polazna osnova za ovaj projekat korišćen je referentni model JM6.1e programske podrške H.264 video dekodera na PC računaru

Pošto je data programska podrška prilagođena PC računaru, potrebno izvršiti njeno prilagođavanje Texas Instruments platformi, kao i radu u realnom vremenu. Takođe je potrebno realizovati mehanizam prikaza dobijene video sekvence.

Programska podrška je realizovana korišćenjem *Microsoft Visual C++ 6.0* i *Texas Instruments Code Composer Studio 2.21* programskih paketa, a testirana je na *ANDROMEDA* razvojnoj platformi.

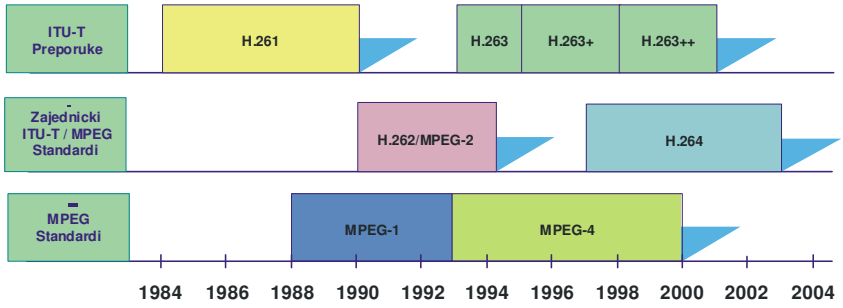
2. H.264/MPEG-4 AVC[1]

Digitalni video se koristi u mnogobrojnim primenama, čiji je broj u stalnom porastu, od video telefonije i video konferencija do DVD-a i digitalne televizije. Ovo se desilo zahvaljujući razvoju mnogih standarda za video kodiranje, koji su nastajali ciljajući pri tom razne oblasti primene. Ovi standardi obezbeđuju mehanizme potrebne za postizanje interoperablinosti između sistema koji su proizvedeni od različitih proizvođača, što potpomaže rastu video tržišta. ITU-T (*The International Telecommunications Union, Telecommunication Standardization Sector)* je jedna od dve zvanične organizacije koja razvija standarde za video kodiranje, druga je ISO/IEC JTC1 *(International Standardization Organization and the International Electrotechnical Commission, Joint Technical Committee number 1)*. ITU-T standardi za kodiranje se nazivaju preporukama i označavaju se sa H.26x (npr. H.261, H.262, H.263 i H.264). ISO/IEC standardi se označavaju sa MPEGx (npr. MPEG-1, MPEG-2 i MPEG-4).

ITU-T preporuke su napravljene uglavnom za video komunikacije u realnom vremenu, kao što su video konferencije i video telefonija. Na drugu stranu, MPEG standardi su napravljeni uglavnom za primenu u skladištenju video signala (DVD), emitovanju video signala (radio-difuzna TV) i prenosu video signala preko računarskih mreža. U većini slučajeva ove dve organizacije rade nezavisno na različitim standardima. Jedini izuzetak je bio H.262/MPEG-2 standard, koji je razvijan zajednički, od strane obe organizacije.

MPEG-2 standard za video kodiranje koji je razvijen pre oko 10 godina, je bio taj koji je obezbedio tehnologiju za sve sisteme digitalne televizije širom sveta. On omogućava efikasno slanje TV signala preko satelita (DVB-S), kablovski sistema (DVB-C) i zemaljskih radio-difuznih (DVB-T) platformi. Međutim ostali mediji za transmisiju kao što je xDSL ili UMTS nude male brzine prenosa podataka. Tako da je broj programa koji se mogu prenositi ograničen, što je zahtevalo dalja poboljšanja u metodama video kompresije.

Nedavno, ITU-T i ISO/IEC JTC1 su se složili da ujedine napore u razvoju nastajućeg H.264 standarda, koji je iniciran od strane ITU-T organizacije. H.264 (tzv. MPEG-4 Part 10 ili MPEG-4 AVC) je usvojen kod obe organizacije jer predstavlja prekretnicu,u smislu performansi,od svih postojećih standarda za video kodiranje. Slika 2.1 sumira evoluciju ITU-T preporuka i ISO/IEC MPEG standarda.



Slika 2.1 Razvoj ITU-T Preporuka i MPEG Standarda

2.1 Opšti pregled H.264/AVC standarda

ITU-T grupa eksperata za video kodiranje (*Video Coding Experts Group - VCEG - ITU-T SG16 Q.6*) je 1998 godine započela posao na projektu pod imenom H.26L sa ciljem da se udvostruči efikasnost kodiranja u poređenju sa bilo kojim postojećim standardom za video kodiranje. U Decembru 2001 godine, zbog osvedočene superiornosti u video kvalitetu koji je nudio H.264 u odnosu na kvalitet koji je postignut sa postojećim MPEG-4 standardom, ISO/IEC grupa eksperata za pokretnu sliku (*Moving Pictures Expert Group - MPEG - ISO/IEC JTC 1/SC 29/WG 11*) se pridružila ITU-T grupi formirajući pri tome zajednički video tim (*Joint Video Team - JVT*) koji je preuzeo H.264 projekat od ITU-T. Cilj JVT-a je da stvori jedan standard za video kodiranje koji će istovremeno biti novi deo (Part 10) MPEG-4 familije standarda i nova ITU-T (H.264) Preporuka. Razvoj H.264 je aktivnost koja još traje, sa tim da se prva zvanična verzija standarda očekuje pre kraja 2003 godine.

H.264 standard ima brojne prednosti koje ga izdvajaju, dok u isto vreme, deli zajedničke odlike drugih postojećih standarda. Ključne prednosti H.264 su:

1. **Do 50% manja bitska brzina**: U poređenju sa H.263v2 (H.263+) ili MPEG-4 Simple Profile, H.264 dozvoljava do 50% redukciju bitske brzine.
2. **Video visokog kvaliteta**: H.264 nudi dosledno dobar video kvalitet na viskom i niskim bitskim brzinama.
3. **Otpornost na greške** *(Error resilience):* H.264 obezbeđuje alate neophodne za izlaženje na kraj sa gubitkom paketa u paketnim mrežama i greškama na nivou bita u bežičnim mrežama koje su sklone greškama.
4. **Naklonjenost ka mrežama** *(Network friendliness):* Kroz sloj za apstrakciju mreže (Network Adaptation Layer), video zapis kodovan sa H.264 može sa lako transportovati preko različitih tipova mreža.

Glavni cilj H.264 standarda je da obezbedi metode za postizanje znatno višeg video kvaliteta u poređenju sa onim što se može ostvariti upotrebom bilo kojeg postojećeg standarda za video kodiranje. Međutim, osnovni pristup koji se koristi u H.264 je sličan onom koji prihvaćen u prethodnim standardima kao što su H.263 i MPEG-4, i koji se sastoji iz sledeće četiri glavne faze:

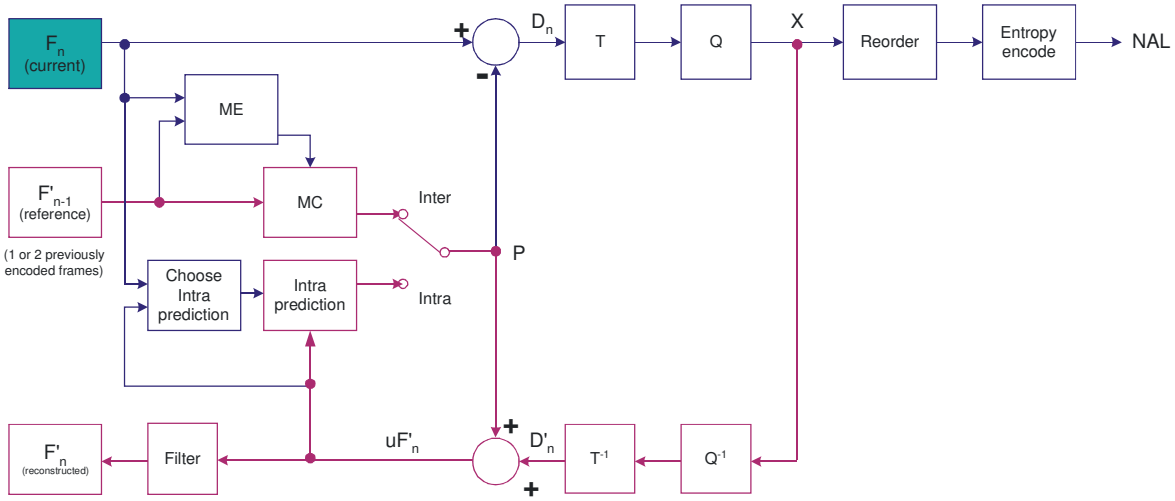
1. Podela svake slike u blokove piksela tako da se obrada slike sprovodi na nivou blokova
2. Eksploatacija prostornih (*spatial*) redundansi koje postoje unutar slike, kodiranjem nekih od originalnih blokova kroz transformaciju, kvantizaciju i entropijsko kodiranje.
3. Eksploatacija vremenskih (*temporal*) zavisnosti koje postoje između blokova uzastopnih slika, tako da se samo promene između sukcesivnih slika kodiraju. Ovo se postiže upotrebom estimacije i kompenzacije pokreta. Za svaki dati blok, pretraga se izvršava u prethodno kodovanim slikama (jedna ili više) kako bi se odredili vektori pomeraja koji se zatim koriste od strane kodera i dekodera za predikciju datog bloka.
4. Eksploatacija preostalih prostornih redundansi koje postoje unutar slike, kodiranjem rezidualnih blokova t.j. razlika između originalnih blokova i odgovarajućih blokova dobijenih predikcijom, ponovo kroz transformaciju, kvantizaciju i entropijsko kodiranje.

2.2 Osnovna struktura i funkcije kodeka

Kao i prethodni standardi (kao što su MPEG1, MPEG2 i MPEG4), nacrt H.264 standarda ne definiše eksplicitno KODEK (KODer / DEKoder par). Tačnije standard definiše sintaksu kodiranog video zapisa zajedno sa metodom za dekodiranje ovog zapisa. Međutim u praksi koder i dekoder koji su saglasni sa standardom će verovatno u sebi uključivati funkcionalne elemente prikazane na slici 2.2 i slici 2.3. Dok će funkcije prikazane na ovim slikama najverovatnije biti neophodne zbog saglasnosti, postoji značajna oblast varijacija u strukturi KODEK-a. Osnovni struktura i elementi (predikcija, transformacija, kvantizacija, entropijsko kodiranje) se malo razlikuju od prethodnih standarda (MPEG1,MPEG2,MPEG4,H.261,H.263); ono što čini da se H.264 razlikuje od njih je u detaljima svakog funkcionalnog elementa.

2.2.1 Koder

Koder (Slika 2-2) u sebi sadrži dve putanje toka podataka, direktna (*forward*) putanja (s leva na desno, označena plavom bojom) i rekonstukciona (*reconstruction*) putanja (s desna na levo, označena ljubičasto).



Slika 2.2 Blok šema H.264 kodera

2.2.1.1 Direktna putanja

Ulazna slika n u oznaci F_n je slika koji je potrebno kodirati. Slika se procesira u makroblokovima (oni odgovaraju blokovima od po 16 X 16 tačaka originalne slike). Svaki makroblok je kodiran u intra ili inter modu. U oba slučaja predikcioni makroblok P se formira na osnovu rekonstruisane slike. U intra modu, P se formira od tačaka iz tekuće slike n koje su prethodno kodirane, dekodirane i rekonstrisane ($uF`_n$ na slici; primetimo da se **nefiltrirane** tačke se koriste za formiranje P). U inter modu, P se formira predikcijom sa kompenzacijom pokreta iz jedne ili više referentnih slika. Na slici referentna slika je označena kao prethodno kodirana slika $F`_{n-1}$; međutim predikcija za svaki makroblok može da se formira od jedne ili više prošlih ili budućih slika (po vremenskom poretku) koje su već kodirane i rekonstruisane.

Predikcioni makroblok P se oduzima od trenutnog makrobloka da bi se dobila razlika makroblokova D_n. Ova razlika se transformiše (upotrebom blok transformacije) (T) i kvantizuje (Q) da bi se dobio skup X kvantizovanih koeficijenta transformacije. Ovi koeficijenti se pregrupišu i entropijski kodiraju. Entropijsko kodirani koeficijenti, zajedno sa dodatnim informacijama neophodnim da se dekoduje makroblok (kao što su vrsta predikcije makrobloka, veličina kvantizacionog koraka, informacije o vektoru pomeraja, itd) formiraju kompresovani bitski zapis. On se prosleđuje sloju za apstrakciju mreže (NAL) zbog dalje transmisije ili skladištenja.

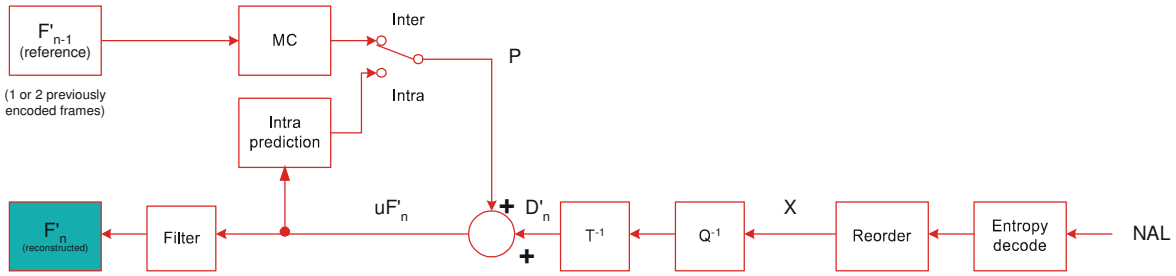
2.2.1.2 Rekonstrukciona putanja

Kvantizovani koeficijenti (X) se dekoduju zbog rekonstrukcije slike koja se koristi za kodiranje narednih makroblokova. Koeficijenti X se reskaliraju (Q^{-1}) i inverzno transformišu (T^{-1}) da bi se dobila razlika makroblokova $D_n`$. Ona nije identična originalnoj razlici makroblokova $D_n$; postupak kvantizacije unosi gubitak informacija tako da je $D_n`$ izobličena verzija od D_n.

Predikcioni makroblok P se dodaje na $D_n`$ da bi se dobio rekonstruisani nefiltrirani makroblok $uF`_n$ (izobličena verzija originalnog makrobloka). Na njega se primenjuje filter da bi se smanjili efekti izobličenja. Od filtriranih rekonstruisanih makroblokova formira se referentana slika $F`_n$.

2.2.2 Dekoder

Putanja podataka u dekoderu (slika 2.3) je prikazana sa leva na desno da bi se ilustrovala sličnost između kodera i dekodera.



Slika 2.3 Blok šema H.264 dekodera

Dekoder prima kompresovani bitski zapis od NAL-a. Podaci se entropijski dekodiraju i regrupišu da bi se dobio skup kvantizovanih koeficijenata X. Oni se reskaliraju i inverzno transformilšu da bi se dobio $D_n`$ (on je identičan $D_n`$ prikazanom u Koderu). Upotrebljavajući informacije iz zaglavlja dekodiranog iz bitskog zapisa, dekoder kreira predikcioni makroblok P, indentičan originalnom predikcionom makrobloku P formiranom u koderu. P se dodaje $D_n`$ da bi se dobio $uF`_n$ koji se filtrira da bi se dobio dekodovani makroblok $F`_n$.

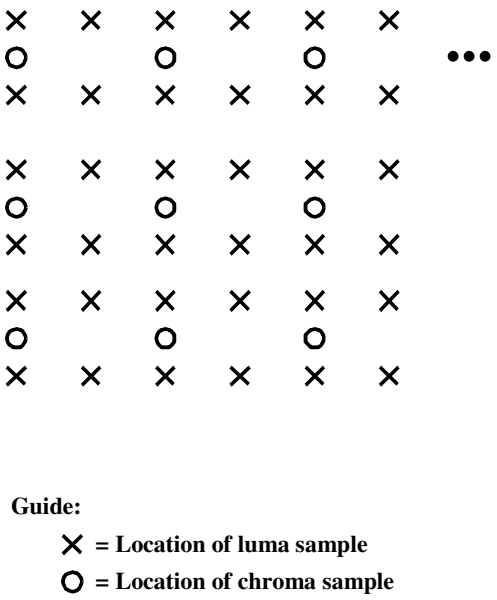
Iz prethodne diskusije i slika, vidi se da "reconstruction" putanja u koderu služi da osigura da i koder i dekoder koriste istu referentnu sliku da naprave predikciju P. Ako to ne bi bio slučaj tada predikcija P u koderu u dekoderu ne bi bila identična, što bi vodilo da povećanja greške i razmimoilaženja između kodera i dekodera.

Sa tačke gledišta kodiranja, glavna razlika između H.264 i drugih standarda je: U postupku estimacije i kompenzacije pokreta H.264 upošljava blokove različitih veličina i oblika, veću rezoluciji pri estimaciji pokreta (1/4 piksela), višestruke referente slike i višestruke bidirekcione režime. U postupku transformacije, H.264 upotrebljava transformaciju baziranu na celobrojnoj aritmetici koja je gruba aproksimacija DCT transformacije, koja se koristi u prethodnim standardima, ali bez problema razmimoilaženja u inverznoj transformaciji. U H.264, entropijsko kodiranje može da se izvrši upotrebom bilo kombinacijom jedne univerzalne tabele kodova sa promenjivom dužinom (UVLC) i konteksno adaptivnim kodovima promenjive dužine (CAVLC) za koeficijente transformacije, bilo upotrebom konteksno baziranim adaptivnim binarnim aritmetičkim kodovima (CABAC). Ove i ostale osobine biće opisane detaljnije u narednim delovima.

2.3 Opis slike i njen format

Svaka slika se sastoji iz konačnog broja elementa, od kojih svaki ima svoju sjajnost i boju. Jedan takav element zove se piksel. Kada se gustina slike, odnosno broj piksela po jedinici površine, poveća iznad određene granice vizuelni sistem prestaje da ih vidi kao odvojene elemente. Broj piksela u jednoj slici je određen horizontalnom i vertikalnom rezolucijom slike t.j brojem piksela po horizontali i vertikalni. Pokazuje se da je potrebno raspolagati sa tri vrednosti za svaki element slike da bi se informacija o sjajnosti i boji prenela do gledaoca. Za crno-belu sliku (monohromatsku) dovoljna je samo informacija o sjajnosti slike. Video signal, koji treba da se kodira, se postupkom A/D konverzije, digitalizuje t.j. sve tri komponente slike se predstavljaju sa tri niza odbiraka. Jedan niz vrednosti odbiraka predstavlja informaciju o sjajnosti (luminentnosti) svake tačke, on predstavlja luminentnu Y komponentu slike. Druga dva niza vrednosti odbiraka nose informaciju o boji svake tačke. Ove dva niza predstavljaju hrominentnu Cb (U) i Cr (V) komponentu slike. U slučaju monohromatske slike postoji sam jedan niz vrednosti. Formati digitalizacije kolor slike se razlikuju prema horizontalnoj i vertikalnoj učestanosti odbiranja vrednosti luminentne i hrominente komponente slike.Tako postoji YUV 4:4:4 format gde je svaka od tri kompnente odabiraju u punoj vertikanoj i horizontalnoj rezoluciji, zatim YUV 4:2:2 format gde je horizontalna učestanost odabiranja hrominentne komponente prepolovljena i YUV 4:2:0 format gde je i vertikalna učestanost odabiranja prepolovljena.

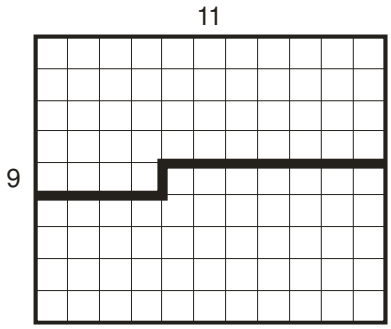
Digitalni format kolor slike koji se koristi u H.264 standardu je 4:2:0 format. Međusobni položaj odbiraka luminentnih i hrominentnih komponenti slike je prikazan na slici 2.4



Slika 2.4 Nominalni položaji luma i hroma vrednosti uYUV 4:2:0 formatu slike

2.3.1 Podela slike u makroblokove

Kao što je ranije pomenuto data slika se deli na blokove piksela kvadratnog oblika fiksne veličine koji se zovu makroblokovi. Svaki makroblok se sastoji od jednog 16x16 bloka luma vrednosti i dva 8x8 bloka hroma vrednosti. Na primer. Slika u QCIF (*Quarter Common Intermediate Format*) rezoluciji (176x144) se deli na 11x9 makroblokova kao što je prikazano na slici 2.5. Slična podela na makroblokove se upotrebljava i na slike drugih veličina, pri tome i horizontalna i vertikalna rezolucija luma komponente slike mora biti deljiva sa 16 (u tom slučaju je hrominenta komponenta slike deljiva sa 8). Slika dalje može biti particionisana na delove (*slices*), deo (*slice*) mora da sadrži celobrojan broj makroblokova. Poredak makroblokova unutra jednog dela pri kodiranju, zavisi od tzv. šeme rasporeda makroblokova (*Macroblock Allocation Map*) i ona nije nužno rasterska (*raster-scan*). Slika se particioniše na delove zbog opravka toka procesa dekodirnja u slučaju gubitka podataka, jer se svaki deo *(slice)* slike kodira nezavisno od drugih delova.



Slika 2.5 Podela slika u QCIF rezoluciji na makroblokove

Da bi se obezbedio efikasan način za prikrivanje greška nastalih u prenosu, metoda pod nazivom fleksiblino ređanje makroblokova (*Flexible Macroblock Ordering - FMO*) je podržana standardom H.264. FMO specifira obrazac za dodelu makoroblokva delovim slike. Svaki deo slike se prenosi odvojeno. Ako se deo slike izgubi u prenosu, vrednosti unutar susednih makroblokova, koji pripadaju ostalim ispravno prenetim delovima slike, se mogu upotrebiti za efikasno prikrivanje nastalih grešaka u prenosu. Makroblokovi u FMO obrascima mogu biti raspoređeni pravilno, zatim pravilno rasuti, kao što je šahovsko polje, ili slučajno rasuti.

2.4 Intra režim predikcije i kodiranja

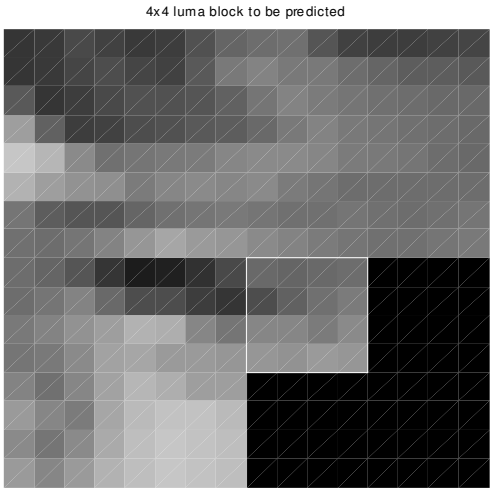
Kodiranje slika u Intra režimu se odnosi na slučajeve gde se samo prostorne redudanse unutar slike eksploatišu. Tako kodirana slika se naziva I slikom. I slika se uobičajno kodira direktnom primenom transformcije na makroblokove u slici. Konsekvence toga je da su kodirane I slike velike, s obzirom da je velika količina informacija obično prisutna u slici. Da bi se povećala efikasnost postupka kodiranja slika u Intra režimu po H.264 standardu, prostorne korelacije između susednih makrobolokova u slici se eksploatišu. Ideja je zasnovana na opservaciji da susedni makroblokovi teže da imaju slične odlike, pa je prema tome prvi korak u postupku kodiranja makrobloka određivanje predikcije tog makrobloka na osnovu susednih makroblokova (tačnije makrobloka koji je iznad i makrobloka koji je levo, jer su ti makroblokovi vec kodirani). Razlika između stvarnog makrobloka i njegove predikcije (formirane na osnovu susednih makroblokova) se kodira što dovodi do toga da je potrebno manje podataka za reprezentaciju makrobloka nego u slučaju da je transformacija primenjena na sam makroblok.

Da bi se izvršilo kodiranje makrobloka u Intra režimu, H.264 nudi 9 režima za predikicju 4x4 lumininentnih blokova. Za oblasti slike sa malo prostornih detalja (npr. uniformne površine), H.264 nudi 16x16 intra kodiranje, u kojem se jedan od 4 predikciona režima bira za predikciju čitave luminente komponente makrobloka. Za predikciju hrominatnih 8x8 blokova makrobloka H.264 raspolaže sa 4 predikciona režima.

2.4.1 4x4 luma Intra predikcioni režimi

Na slici 2.6a je uokviren makroblok u luminentnoj komponenti slike, dok je na slici 2.6b uokviren 4x4 luma blok za koji se traži predikcija.u intra režimu kodovanja.

Slika 2.6a



Slika 2.6b

. Tačke gore i na levo od trenutnog bloka su prethodno bile kodovane i rekonstruisane; one stoga stoje na raspolaganju koderu i dekoderu za formiranje reference za predikciju. Predikcioni blok P (tačke a-p) se računa na osnovu tačaka označenih sa A-M u tabeli 2.1.

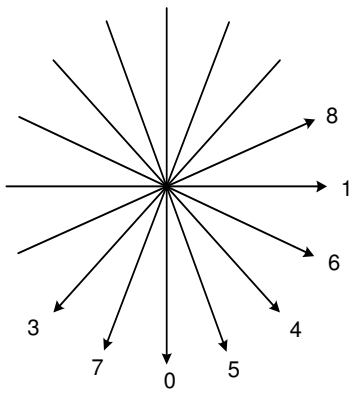
M	A	B	C	D	E	F	G	H
I	a	b	c	d				
J	e	f	g	h				
K	i	j	k	l				
L	m	n	o	p				

Tabela 2.1 Označavanje predikcionih tačaka A - M

Treba primetiti da u nekim slučajevima, nisu sve tačke od A-M dostupne u okviru trenutnog dela slike *(slice)*.U nameri da se sačuva nezavisno dekodovanje delova *(slices)* slike, samo tačke u okviru trenutnog dela slike su na raspolaganju za predikciju. DC predikcija (režim 2 se

modifikuje u zavisnosti od toga koje su tačke od A-M na raspolaganju; ostali režimi (0,1,3-8) se mogu upotrebiti samo ako su sve zahtevane tačke (u zavisnosti od režima) na raspolaganju. 4x4 luma Intra predikcioni režimi su:

- Režim 0 (Vertikalni): Gornje tačke A,B,C,D su ekstrapolirane vertikalno.
- Režim 1 (Horizontalni): Leve tačke I,J,K,L su ekstrapolirane horizontalno.
- Režim 2 (DC): Sve tačke u P su formirane na osnovu srednje vrednosti tačaka A-D i I-L
- Režim 3 (Dijagonalni dole-levo): Tačke se dobijaju interpolacijom po pravcu koji je se pod uglom od 45° proteže između gornjeg desnog i donjeg levog ugla bloka.
- Režim 4 (Dijagonalni dole-desno): Tačke se dobijaju interpolacijom po pravcu koji je se pod uglom od 45° proteže između gornjeg levog i donjeg desnog ugla bloka.
- Režim 5 (Vertikalno-levo): Ekstrapolacija pod uglom od približno 26.6° na levo od vertikale.
- Režim 6 (Horizontalno-dole): Ekstrapolacija (ili interpolacija) pod uglom od približno 26.6° ispod horizontale.
- Režim 7 (Verikalno-desno) Ekstrapolacija (ili interpolacija) pod uglom od približno 26.6° na desno od vertikale.
- Režim 8 (Horizontalno-gore) Interpolacija pod uglom od približno 26.6° iznad horizontale.

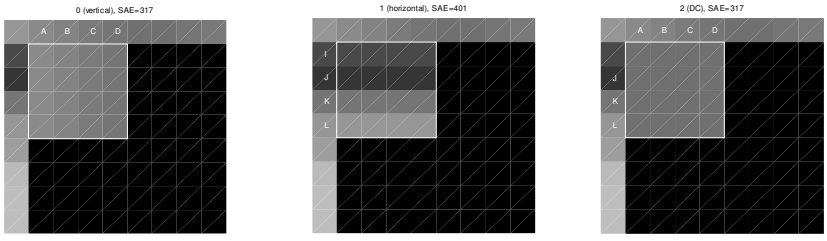


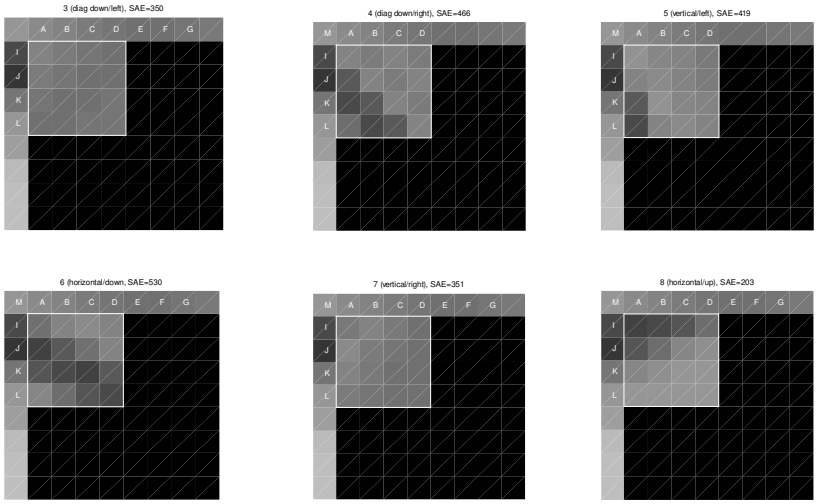
Slika 2.7 Smerovi za predikcije u 4x4 Intra režimima

Strelice na slikama označavaju smer predikciju u svakom modu. Za modove 3-8, tačke u P su formirane iz težinskog usrednjavanja predikcionih tačaka A-M. Na primer ako se odabere režim 4, gornja desna tačka od P (označena sa "d" u tabeli 2-1) se formira sa: round(B/4 + C/2 + D/4).

Koder bira predikcioni režim tako da minimizira razliku između P i bloka koji se kodira.

Primer: 9 predikcionih režima (0-8) se računa za 4x4 blok prikazan na slici 2.6b. Slika 2.8 prikazuje predikcioni blok P kreiran od svakog predikcionog režima. Suma apsolutnih grešaka (SAE) za svaku predikciju pokazuje magnitudu greške predikcije . U ovom primeru najbolje poklapanje daje režim 8 (horizontalno-gore) zato što ovaj režim daje najmanju SAE; vizuelna komparacija pokazuje da je P blok dobijen režimom 8 prilično sličan originalnom 4x4 bloku.



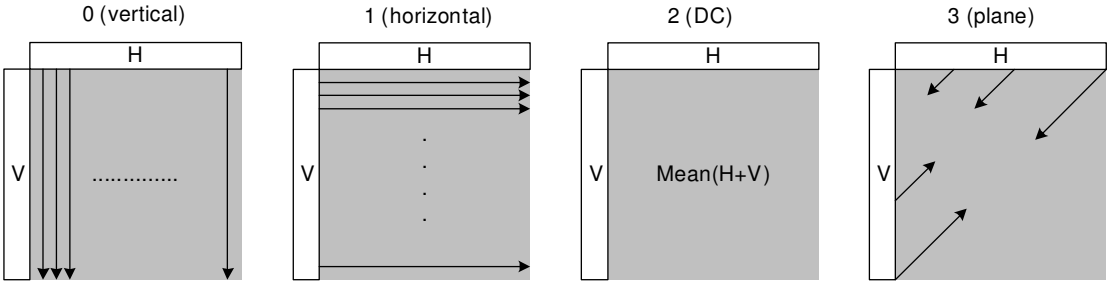


Slika 2.8 Predikcioni 4x4 blokovi

2.4.2 16x16 luma predikcioni režimi

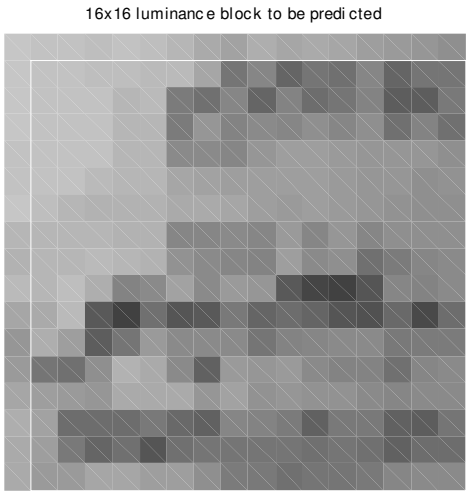
Kao alternativa za 4x4 luma predikcione režime kompletna 16x16 luminentna komponenta makrobloka može se predicirati. Četiri režima su na raspolaganju, prikazana na slici 2.9:

- Režim 0 (Vertikalni): Ekstrapolacija iz gornjih tačaka (H).
- Režim 1 (Horizontalni): Ekstrapolacija iz levih tačaka (V).
- Režim 2 (DC): srednja vrednost gornjih i levih tačaka (H+V).
- Režim 3 (Ravanski): linearna "ravanska" funkcija je fitovana između gornjih i levih tačaka H i V. Ovaj režim daje najbolje rezultate u oblastima gde luminentna komponenta blago varira.



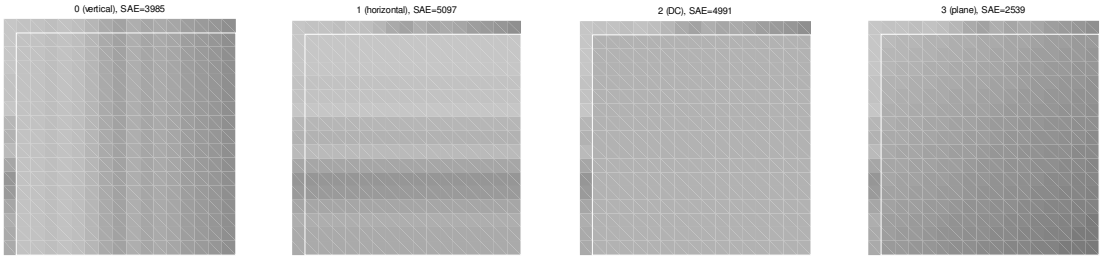
Slika 2.9 16x16 luma predikcioni režimi

Primer: Slika 2.10 prikazuje luminentnu komponentu makrobloka zajedno sa prethodno kodiranim i rekonstruisanim tačkama na gornjoj i levoj ivici.



Slika 2.10 16x16 luma makroblok

Rezultat predikcije za sva četiri režima (Slika 2.11) pokazuje da je najbolje poklapanje dato sa režimom 3. U ovom slučaju, režim 3 pravi površinu sa gradijentom od svetlog (gore-levo) ka tamnom (dole-desno). Intra 16x16 režimi daju najbolje rezultate u homogenim područjima slike.



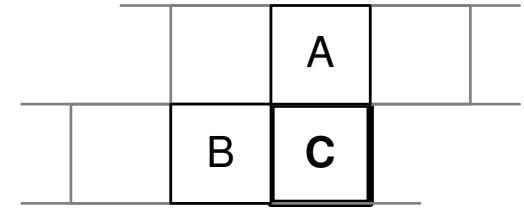
Slika 2.11 Intra 16x16 predikcije

2.4.3 8x8 hroma predikcioni režimi

Svaka 8x8 hroma komponenta makrobloka je predicirana pomoću hroma tačaka iznad i/ili levo, koje su prethodno bile kodovane i rekonstruisane. 4 predikciona režima su vrlo slična 16x16 luma režimima koji su opisani u prethodnom odeljku i ilustrovani slikom 2.2, a to su vertikalni (režim 0), horizontalni (režim 1), DC (režim 2) i površinski (režim 3). Ako je bilo koji 8x8 luma blok kodiran u Intra režimu, oba hroma bloka su Intra kodirana.

2.4.4 Kodiranje intra prediktivnih režima

Izbor Intra predikcionog režima za svaki 4x4 blok mora se signalizirati dekoderu i ovo može potencijalno da zahteva veliki broj bita. Međutim, Intra režimi za susedne 4x4 blokove su vrlo korelisani. Na primer, ako su prethodno kodirani 4x4 blokovi A i B na slici 2.12 predicirani upotrebom režima 2, vrlo je verovatno da će najbolji režim za predikciju bloka C (trenutnog bloka) biti takođe režim 2.



Slika 2.12 Susedni 4x4 intra kodirani blokovi

Intra predikcioni režim za svaki blok se efikasno kodira dodelom simbola sa kraćom kodnom reči više verovatnim režimima, gde se verovatnoća za svaki režim određuje na osnovu režima kodovanja susednih blokova. Za trenutni blok C, koder i dekoder računaju najverovatniji režim (*most_probable_mode*). Ako su A i B oba kodirana u 4x4 intra režimu i oba se nalaze u okviru trenutnog dela slike (*slice*), najverovatniji režim je minimum predikcionih režima od A i B; inače najverovatniji režim se postavlja na 2 (DC predikcija).

Predikcioni režim za makroblok kodiran u Intra-16x16 režimu ili hroma blok kodiran u Intra režimu se nalazi kodiran u zaglavlju makrobloka (najverovatniji režim se ne upotrebljava u ovom slučaju).

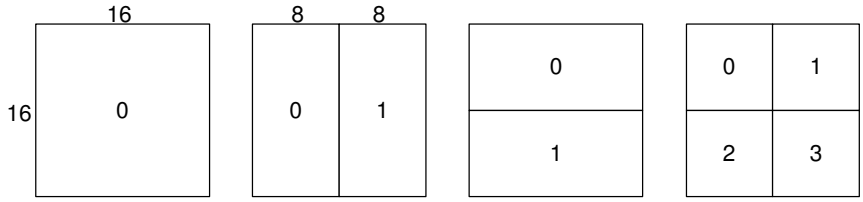
2.5 Inter režimi predikcije i kodiranja

Kodiranje slika u Intra režimu je bazirano na iskorišćavanju vremenslih redudansi koje postoje između sukcesvnih slika upotrebom estimacije i kompenzacije pokreta, te stoga obezbeđuje efikasno kodovanje video sekvenci. Kada prethodno kodovana slika postane refrentana slika u postupku estimacije i kompenzacije pokreta za sliku koja se trenutno koduje, tada trentnu sliku nazivamo P slikom. Kada su za referente slike izabrane prethodna kodirana i buduća slika, tada se slika koju trenutno kodujemo naziva B slika. Estimacija pokreta u H.264 standardu poseduje većinu ključnih osobina koje su prihvaćene u prethodnim standardima, ali je njena efikasnost unapređena kroz dodatnu fleksibilnost i funkcionalnost.

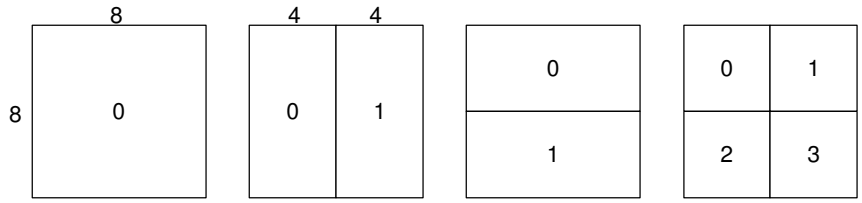
Naredne sekcije detaljnije opisuju četiri glavne osobine H.264 estimacije pokreta, a to su: upotreba blokova različitih veličina i oblika , upotreba visoko preciznih vektora pomeraja, upotreba višestrukih referentnih slika i upotreba filtera za uklanjane izobličenja u petlji predikcije.

2.5.1 P režimi kodiranja delova slike (P-slices)

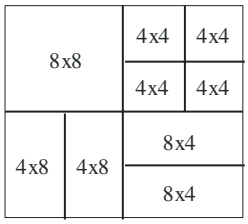
H.264 podržava kompenzaciju pokreta pomoću blokova veličina od 16x16 do 4x4 luminentnih vrednosti sa mnogo različitih međuvarijanti. Luminentna komponenta za svaki makroblok (16x16 vrednosti) može biti podeljena na 4 načina prikazana na slici 2.13: 16x16, 16x8, 8x16 ili 8x8. Svaki deo predstavlja makroblok particiju. Ako je 8x8 način izabran, svaka od četiri 8x8 makroblok particije može se dalje podeliti na još četiri načina prikazana na slici 2.14: 8x8, 8x4, 4x8 ili 4x4 (poznate kao makroblok pod-particije). Ove particije i pod-particije daju veliki broj mogućih kombinacija u okviru svakog makrobloka. Ova metoda particionisanja makrobloka u pod-blokove različite veličine, koji su kompenzovani na pokret, je poznata kao kompenzacija pokreta sa strukturom tipa stabla. Slika 2.15 prikazuje primer jednog načina particionisanja makrobloka.



Slika 2.23 Makroblok particije: 16x16, 16x8, 8x16, 8x8



Slika 2.14 Makroblok pod-particije: 8x8, 8x4, 4x8 4x4



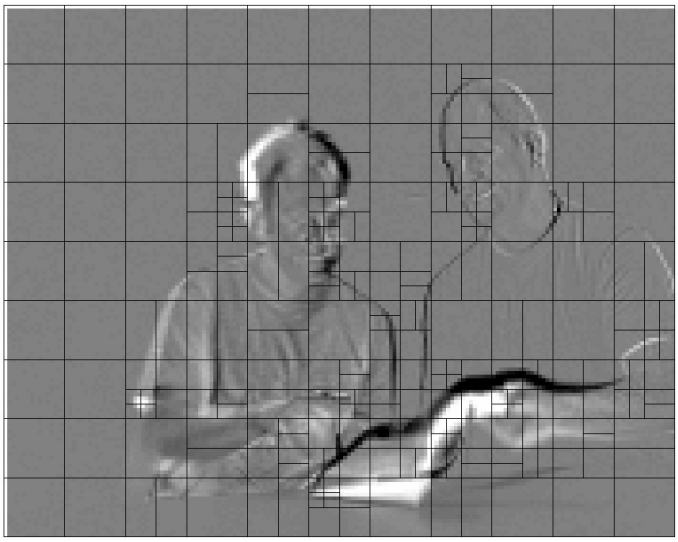
Slika 2.15 Primer particionisanja makrobloka

Uvođenje blokova manjih veličina u postupak kompenzacije pokreta u celosti unapređuje predikciju. Mali blokovi unapređuju mogućnost rukovanja sa finim pokretnim detaljima, rezultat toga je da se dobija bolji vizuelni subjektivani kvalitet zbog toga što oni ne proizvode velika izobličenja (*blocking*).

Poseban vektor pomeraja je potreban za svaku particiju i pod-particiju. Svaki vektor pomeraja mora biti kodiran i poslan; pored toga, izbor načina particionisanja mora biti kodiran u kodiranom bitskom zapisu. Odabirom krupne podele makrobloka (npr. 16x16, 16x8, 8x16) dobijamo da je mali broj bitova potreban za predstavljanje vektora pomeraja i načina particionisanja; međutim, ostatak nakon kompenzacije pokreta može da sadrži značajnu količinu energije u područjima slike sa mnogo detalja. Odabir sitnije podele makrobloka (npr. 8x4, 4x8, 4x4) može dati ostatak nakon kompenzacije pokreta sa malom količinom energije, ali takva podela zahteva veći broj bita za predstavljanje vektora pomeraja i načina particionisanja. Odabir načina podele makrobloka stoga ima značajan uticaj na performanse kompresije. Uopšteno govoreći krupna podela makrobloka odgovara homogenim područjima slike, dok sitna podela može biti korisna za područja sa velikim brojem detalja.

Rezolucija svake hroma komponente u makrobloku (Cr i Cb) je polovina rezolucije luminentne (luma) komponente. Svaki od hroma blokova je particionisan na isti način kao i luma komponenta, uzimajući u obzir da je veličina particije polovina horizontalne i vertikalne rezolucije (8x16 particija u lumi odgovara 4x8 particiji u hromi; 8x4 particija u lumi odgovara 4x2 u hromi; itd). Horizontalna i vertikalna komponenta svakog vektora pomeraja se polovi kada se primenjuje na hroma blok.

Primer: Slika 2.16 prikazuje rezidualnu sliku (bez kompenzacije pokreta). H.264 referentni koder odabira najbolju veličinu particija za svaki deo slike, t.j. takvu veličinu particija da se minimizuje ostatak i vektori pomeraja. Makroblok particije, odabrane za svako područje, prikazane su na rezidualnoj slici. U područjima gde postoji mala promena između dve slike (ostatak se pojavljuje u sivoj boji), bira se 16x16 particija; u područjima sa gde je promena velika (ostatak se pojavljuje u crnoj ili beloj boji), bira se sitnije particionisanje jer je efikasnije za kodiranje.

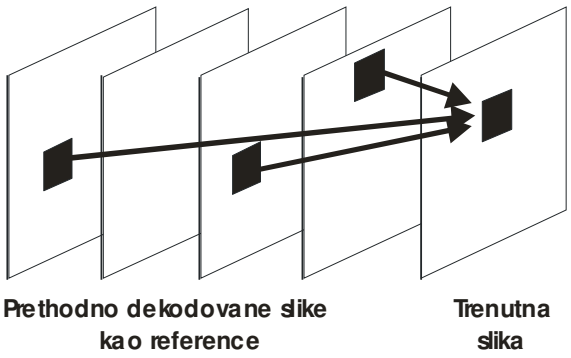


Slika 2.16 Podela rezidualne slike na particije sa optimalnom veličinom

2.5.2 Višestruke referentne slike

H.264 standard nudi mogućnost korišćena višestrukih referentnih slika u Inter režimu kodiranja slike, t.j. kada se više od jedne slike koristi kao referenca za predikciju kompenzacijom pokreta. Slika 2.17 ilustruje taj koncept. Koder i dekoder moraju da uskladište referentne slike u bafer koja može da primi više slika (*multi-picture buffer*). Dekoder replicira bafer koji se nalazi na strani kodera, u skladu sa režimom sladištanja referentne slike i kontrolih operacija za upravljanje baferom (*Memory Menagment Control Operation - MMCO*) , koje su specificirane u kodiranom zapisu. Osim u slučaju kada je veličina bafera postavljena za jednu sliku, pozicija (indeks) referente slike unutar bafera se mora signalizirati. Taj indeks se šalje za svaki 16x16,16x8,8x16 ili 8x8 blok, kodiran sa kompenzacijom pokreta.

Upotreba višestrukih slika dovodi do boljeg vizuelno subjektivnog kvaliteta slike i njenog efikasnijeg kodiranja. Pored toga, upotrebom višestrukih slika može se postići da je video zapis kodovan H.264 standardom otporniji na greške. Međutim sa tačke gledišta implementacije, doći će do dodatnih kašnjenja i većih memorijskih zahteva od strane kodera i dekodera.



Slika 2.17 Kompenazija pokreta sa višestrukim referentnim slikama

Pored režima kodovanja makrobloka koji određuju način particionsianja makrobloka, makroblok može biti kodiran u tzv. SKIP režimu. U ovom režimu ne prenose se ni kvantizovana razlika dobijena u postupku predikcije kompenzacijom pokreta, ni vektori pomeraja ili indeks referentne slike. Rekonstrukcija makrobloka se vrši samo na osnovu makrobloka referentne slike sa indeksom 0 u memoriji.

2.5.3 B režimi kodiranja delova slike (B-slices)

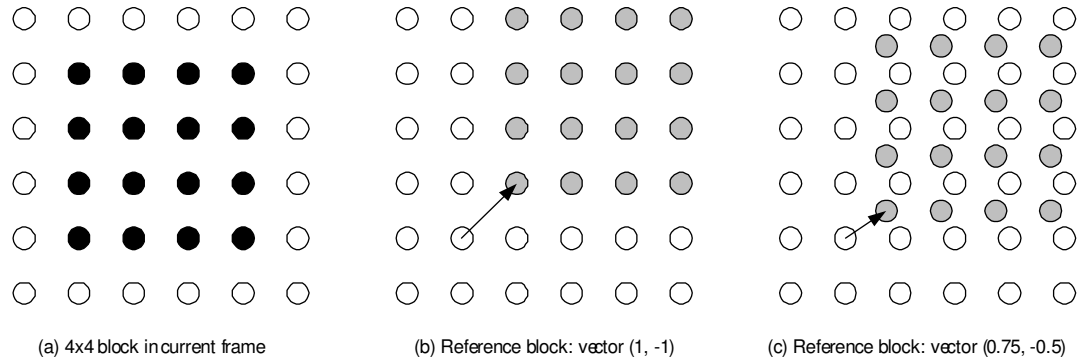
Prethodne sekcije su se odnosile na kodiranje delova slike u P režimu. U poređenju sa prethodnim standardima za video kodiranje, koncept B delova slike je generalizovan u H.264. Na prime, druge slike mogu da referenciraju B slike u procesu predikcije sa kompenzacijom pokreta, u zavisnosti od kontrolih operacija za upravljanje baferom za sladištenje slika. Znatna razlika između B i P delova slike je ta da se B delovi slike kodiraju na način u kojem neki od makroblokova ili blokova mogu da upotrebe težinsko usrednjavanje dve različite vrednosti predikcije (dobijene kompenzacijom pokreta) za dobijanje krajnje predikcije. B delovi slike koriste dva različita bafera za referentne slike, koje se nazivaju prvi i drugi bafer. Koja će se slika naći u svakom od bafera je zadatak kontrole za upravljanje baferima

U B delovima slika, četiri različita režima predikcije su podržana: lista 0, lista 1, bipredikcija i direktna predikcija. Režimi liste 0 označavju da se vrednost predikcije formira upotrebom kompenzacije pokreta na referentnim slikama koje se nalaze u prvom baferu, dok se režimi liste 1 odnose na upotrebu referentnih slika u drugom baferu. U biprediktivnom režimu, vrednost predikcije se formira na bazi težinskog usrednjavanja predikcija dobijenih na osnovu liste 0 i liste 1. Režim direktne predikcije označava da se koristi režim prethodno poslatog bloka.

B delovi slike koriste sličan metod particionisanja makrobloka kao i P delovi slike. Pored Inter-16x16, Inter 16x8, Inter-8x16, Inter-8x8 i Intra režima, postoji i režim makrobloka koji označava korišćenje direktne predikcije. Pored toga, za svaku 16x16,16x8, 8x16 i 8x8 particiju, predikcioni režimi se biraju odovojeno. 8x8 particija makrobloka u B delu slike takođe može biti kodirana u direktnom režimu predikcije. Ako se u direktonm režimu predikcije ne prenose kvantizovana razlika dobijena u postupku predikcije kompenzacijom pokreta tada se taj slučaj naziva SKIP režim B dela slike. Kodiranje vektora pomeraja je slično kodiranju u P delovima slike, sa dodatnim modifikacijama zbog toga što susedni blokovi mogu biti kodirani upotrebom različitih režima predikcije.

2.5.4 Pod-piksel vektori pomeraja

Svaka particija u Inter režimu kodovanja nakrobloka se estimira iz područja iste veličine koje se nalazi u referentnoj slici. Pomeraj između ova dva područja (vektor pomeraja) je u ¼-piksel rezoluciji (za luma komponentu). Luma i hroma vrednosti na pod-piksel poziciji ne postoje u referentnoj slici, te ih je stoga neophodno dobiti interpolacijom iz obližnjih tačaka slike. Slika 2.18 nam to ilustruje. 4x4 pod-particija u trenutnoj slici (a) (crne tačke) treba da se estimira iz okolnog područija referentne slike. Ako su horizontalna i vertiklana komponenta vektora pomeraja celobrojne vrednosti (b), relevantne vrednosti u referentnom bloku stvarno postoje (sive tačke). Ako jedna ili obe komponente nisu celobrojne vrednosti (c), estimirane vrednosti (sive tačke) se generišu interpolacijom između susednih tačaka u referentonoj slici (bele tačke).



Slika 2.18 Primer celobrojnih i decimalnih pod-piksel pomeraja

Pod-piksel kompenzacija pokreta može da obezbedi značajno bolje performanse kompresije nego celobrojna, po cenu povećane kompleksnosti.

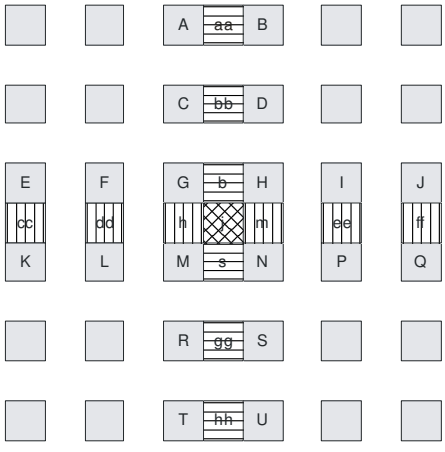
H.264 standard generalno dozvoljava neograničene vektore pomeraja, t.j. vektore pomeraja koji pokazuju na područje izvan slike. U tom slučaju, referentna slika se proširuje izvan granica slike ponavljanjem ivice slike. Nijedna komponenta vektora pomeraja ne može da pokazuje na mesto koje se nalazi preko granice delova slike (*slice boundary*).

2.5.4.1 Generisanje interpoliranih vrednosti na pod-pixel pozicijama

Interpolirane pod-piksel vrednosti se formiraju na sledeći način. Za luminentnu komponentu referentne slike, prvo se generišu vrednosti na ½-piksel pozicijama t.j vrednosti koje se nalaze između dva piksela (*half-pel*). Takve ½-piksel vrednosti su na slici 2.19 označene šrafirano, dok stvarne vrednosti su označene sivom bojom. Svaka generisana vrednost koja se nalazi između dve stvarne vrednosti (npr. **b,h,m,s**) se interpolira iz stvarnih vrednosti upotrebom FIR filtra u 6 tačaka. Težinski koeficijenti filtera su (1/32, -5/32, 5/8, 5/8, -5/32, 1/32). Na primer, vrednost **b** se računa pomoću 6 stvarnih vrednosti označenih sa E, F, G, H, I i J na sledeći način:

b = round((E-5F+20G+20H-5I+J) /32)

Na sličan način se interpolira vrednost **h**, upotrebom filtra nad vrednostima označenim sa A, C, G, M i T. Kada se izračunaju sve vrednosti koje se nalaze između dve stvarne vrednosti, preostale ½-piksel pozicije (npr. **j** na slici) se računaju interpolacijom 6 horizontalnih ili vertikalnih vrednosti koje su prethodno izračunate. Na primer, **j** se generiše filtriranjem cc,dd,h,m,ee i ff vrednosti (rezultat će biti isti bilo da se **j** interpolira horizontalno ili vertikalno). Interpolacioni filter u 6 tačaka je relativno kompleksan (u poređenju, na primer, sa bilineranom interpolacijom), ali on proizvodi tačnije vrednosti na osnovu stvarnih vrednosti, te stoga i bolje performanse u kompenzaciji pokreta.

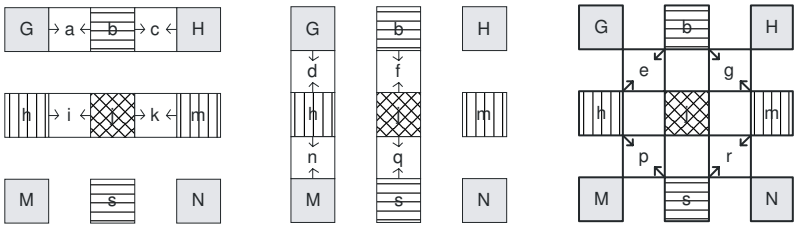


Slika 2.19 Inerpolacija luma vrednosti na ½-piksel pozicijama

Kada su sve vrednosti na pozicijama između dve stvarne vrednosti dostupne, vrednosti na ¼-piksel pozicijama se dobijaju linearnom interpolacijom (Slika 2.20). ¼-piksel vrednosti sa dve horizontalno ili vertikalno susedne ½-piksel ili stvarne vrednosti (npr. **a, c, i, k** i **d, f, n, q**) se linearno interpoliraju između tih susednih vrednosti. Na primer:

a = round((G+b) / 2)

Vrednosti za preostale ¼-piksel pozicije (**e, g, p** i **r**) se dobijaju linearnom interpolacijom između para ½-piksel vrednosti koje se nalaze na dijagonalno suprotnim stranama. Na primer, **e** se interpolira između **b** i **h**.



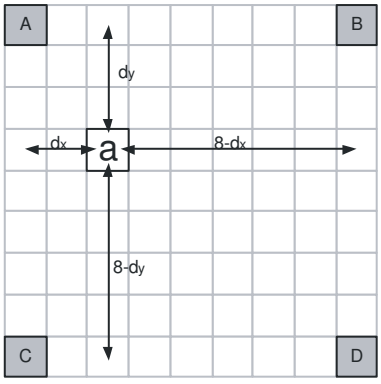
Slika 2.20 Inerpolacija luma vrednosti na ¼-piksel pozicijama

U hroma komponentama (podrazumeva se 4:2:0 format). Interpolirane vrednosti se generišu na 1/8-piksel pozicijama između stvarnih vrednosti sa svaku hroma komponentu. U ovom slučaju, upotrebljava se linearna interpolacija za dobijanje 1/8-piksel hroma vrednosti (Slika 2.21). Svaka pod-piksel pozicija **a** se dobija kao linearna kombinacija obližnjih stvarnih vrednosti A, B, C i D na sledeći način:

a = round([(8-dx)·(8-dy)·A + dx·(8-dy)·B + (8-dx)·dyC + dx·dy·D]/64)

Na slici 2.21, dx je 2 i dy je 3, tako da je:

a = round([30A + 10B + 18C + 6D] / 64)

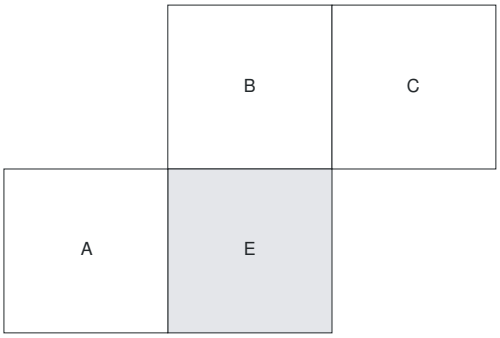


Slika 2.21 Interpolacija hroma vrednosti na 1/8-piksel pozicijama

2.5.5 Kodiranje vektora pomeraja

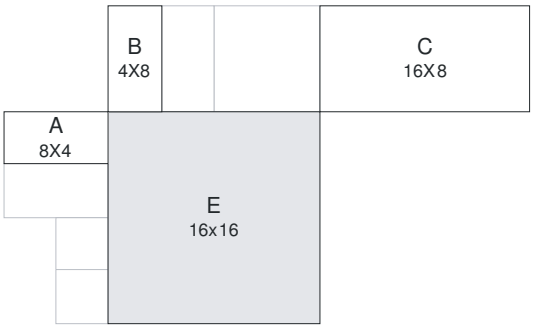
Kodovanje vektora pomeraja za svaku particiju može da uzme značajan broj bita, naročito ako se upotrebljavaju particije malih veličina. Vektori pomeraja obližnjih paticija su često vrlo korelisani i zato se svaki vektor pomeraja estimira iz vektora prethodno kodovanih particija. Estimirani vektor pomeraja, MVp, se formira na osnovu prethodno izračunatih vektora pomeraja. Razlika između stvarnog i estimiranog vektora pomeraja, MVD, se koduje i šalje. Metoda formiranja estimirang vektora pomeraja MVp zavisi od veličine particije za koju tražimo vektor pomeraja i od dostupnosti susednih vektora. Za makroblokove koji se nalaze u P-slikama, ulaz u metodu su MVp koji pripadaju susednim particijama. Susedne particije se određuju na sledeći način.

Neka je E tekući makroblok, makroblok particija ili pod-particija; neka je A particija ili pod-particija neposredno levo od E; neka je B particija ili pod-particija neposredno iznad E; neka je C particija ili pod-particija neposredno iznad i na desno od E. Ako postoji više od jedne particije neposredno iznad E, krajnje leva od njih se uzima za B. Slika 2.22 ilustruje izbor susednih particija kada su sve particije iste veličine (16x16 u ovom slučaju).



Slika 2.22 Tekuća i susedne particije (iste veličine particija)

Na slici 2.23 prikazan je primer izbora susednih particija u slučaju kada su one drugačije veličine od tekuće particije E.



Slika 2.23 Tekuća i susedne particije (različite veličine particija)

Metoda za formiranje estimiranog vektora pomeraja MVp za tekuću particiju E je sledeća:

1. Za sve particije svih veličina, osim za particje tipa 16x8 i 8x16: MVp je median vektora pomeraja za particije A, B i C
2. Za 16x8 particije: MVp za gornju 16x8 particiju se formira iz B, MVp za donju 16x8 particiju se formira iz A
3. Za 8x16 particije: MVp za levu 8x16 particiju se formira iz A, MVp za desnu 8x16 particiju se formira iz C
4. Za preskočene makroblokove: MVp se formira kao i slučaju navedenom pod tačkom 1 (t.j. kao da je blok kodovan u 16x16 Inter režimu)

Ako jedan ili više prethodno kodovanih i poslanih blokova, prikazanih na slikama, nisu dostupni (npr. ako se nalaze izvan slike ili dela slike (slice), tada se metoda za formiranje MVp menja u skladu sa tim.

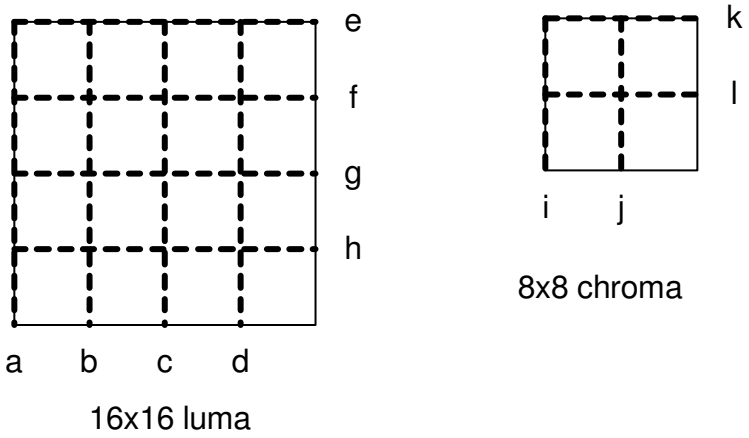
Na strani dekodera, estimirani vektor MVp se formira na isti načina, a zatim se dodaje dekodovanom vektoru razlike MVD. U slučaju preskočenih makroblokova, ne postoji dekodovani vektor razlike MVD. U tom slučaju vektor pomeraja postaje MVp (estimirani vektor pomeraja).

2.6 Filtar za rekonstrukciju (in-loop deblocking filter)

Ovaj filter se primenjuje na svaki dekodovani makroblok u cilju smanjenja izobličenja koja nastaju zbog obrade slike u blokovima. On se primenjuje posle inverzne transformacije u (pre skladištenja makrobloka u referentnu sliku) koderu i dekoderu (pre prikazivanja makrobloka). Filter ima dve korisne osobine: (1) prelazi na ivicima blokova se ublažavaju što poboljšava izgled dekodirane slike (ovo je naročito izražena pri viskom faktorima kompresije); (2) filtrirani makroblokovi se koriste kao referenca u kompenzaciji pokreta za sledeće slike, zbog toga što daju manji ostatak koje se na kraju koduje. (Napomena: kod predikcije u Intra režimu koriste se filtrirani makroblokovi, dok se kod za formiranje predikcije u Intra režimu koriste nefiltrirani rekonstruisani makroblokovi). Ivice slike se ne filtriraju.

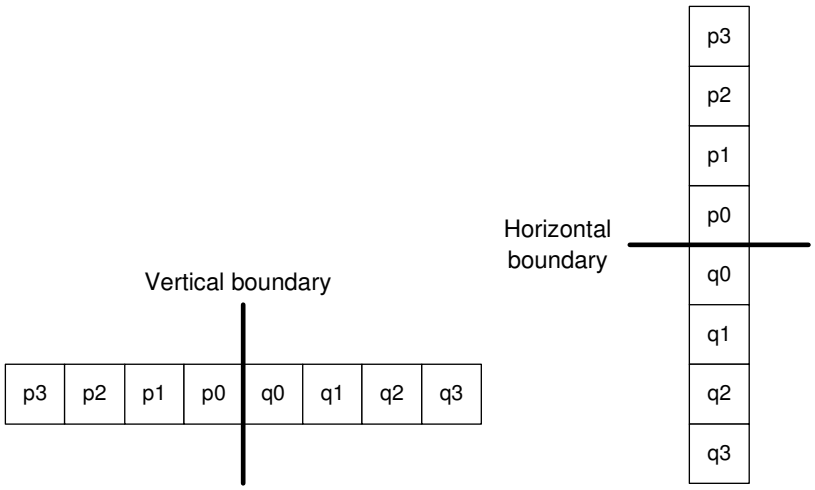
Filtriranje se primenjuje na vertikalne i horizontalne ivice 4x4 blokova u makrobloku, u sledećem redosledu:

1. Filtriraju se 4 vertikalne granice luma komponente (u redosledu a,b,c,d)
2. Filtriraju se 4 horizontalne granice luma komponente (u redosledu e,f,g,h)
3. Filtriraju se 2 vertikalne granice svake od hroma komponenata (u redosledu i,j)
4. Filtriraju se 2 horizontalne granice svake od hroma komponenata (u redosledu k,l)



Slika 2.24 Redosled filtriranja ivica u makrobloku

Svaka operacija filtriranja utiče na maksimalno tri tačke sa obe strane granice. Slika 2.25 prikazuje 4 tačke sa obe strane vertikalne ili horizontalne granice susednih blokova p i q (p0,p1,p2,p3 i q0,q1,q2,q3). U zavisnosti od tekućeg parametra kvantizacije, modova kodovanja susednih blokova i gradijenta vrednosti tačaka preko granice, nekoliko ishoda filtriranja je moguće, počevši od (a) da se nijedna tačka se ne filtrira, pa do (b) gde se p0,p1,p2, q0,q1 i q2 filtriraju da bi se dobilo kao rezultat tačke P0,P1,P2, Q0,Q1 i Q2.



Slika 2.2.35 Tačke koje se nalaze uz vertikalnu ili horizontalnu granicu

2.6.1 Određivanje jačine granice (*Boundary strength*)

Izbor tipa filtriranja zavisi od jačine granice (*boundary strength*) i od gradijenta vrednosti tačaka preko granice. Parametar jačine granice **Bs** se odabira u skladu sa sledećim pravilima:

p ili q su intra kodovani **i** granica blokova je granica između makroblokova	Bs = 4 (najjače filtriranje)
p ili q su intra kodovani i granica blokova **nije** granica između makroblokova	Bs = 3
ni p ni q nisu intra kodovani; p ili q sadrže kodovane koeficijente	Bs = 2
ni p ni q nisu intra kodovani; ni p ni q ne sadrže kodovane koeficijente; p i q imaju različite referentne slike **ili** različit broj referentnih slika **ili** različite vrednosti vektora pomeraja	Bs = 1
ni p ni q nisu intra kodovanin; ni p ni q ne sadrže kodovane koeficijente; p i q imaju iste referentne slike i identične vrednosti vektora pomeraja.	Bs = 0 (bez filtriranja)

Filter je "jači" na mestima gde je vrlo verovatno da dođe do značajnijeg izobličenja, kao što su granice makroblokova kodiranih u intra režimu ili granice između blokova koji sadrže kodovane koeficijente.

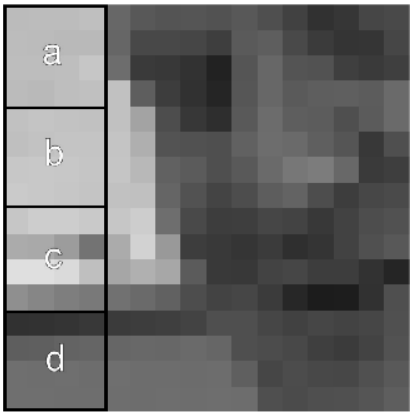
2.6.2 Izbor tipa filtriranja

Grupa tačaka iz skupa (p2,p1,p0,q0,q1,q2) se filtrira samo ako je:

(a) Bs > 0
(b) svaka od apsolutnih vrednosti |p0-q0|, |p1-p0| i |q1-q0| manja od praga α ili β (α i β su definisani standardom)

Pragovi α ili β se povećavaju sa srednjom vrednošću parametra kvantizacije QP blokova p i q. Svrha izbora tipa filtriranja je da "isključi" filter kada postoji značajna promena (gradijent) preko granice u originalnoj slici. Definicija značajne promene zavisi od QP. Kada je QP mali, sve drugo osim malog gradijenta preko granice verovatno potiče od detalja u slici (pre nego izobličenja) koje treba sačuvati i zato su pragovi α i β mali. Kada je QP veći, izobličenja su znatnija, te se α, β veći tako da dolazi do više filtriranja.

Primer: Slika 2.26 prikazuje 16x16 luma komponentu makrobloka (bez izobličenja) sa četiri uokvirena 4x4 bloka a,b,c i d. Pretpostavljajući da je vrednost QP u gornjem opsegu (srednjih do velikih), granica između a i b će verovatno biti filtrirana zato što je gradijent preko granice mali. Nema značajnih detalja u slici koje treba sačuvati, a izobličenje usled blokova će biti prilično očigledno na ovoj granici. Međutim, postoji značajna promena luminentnosti preko granice između c i d, usled horizontalnog detalja u slici; filtriranje bi zamutilo originalni detalj tako da sa u ovom slučaju filter "isključuje".



Slika 2.26 16x16 luma makroblok na kome su prikazane granice blokova

2.6.3 Implementacija filtra

(a) $0 < Bs < 4$:

4-tap linearni filtar je primenjen na p1, p0, q0 i q1, što kao rezultat daje filtrirane vrednosti P0 i Q0. Ako je vrednost |p2-p0| manja od praga β, 4-tap linearni filtar je primenjen na p2, p1, p0 i q0, dajući filtrirani P1. Ako je vrednost |q2-q0| manja od praga β, 4-tap linearni filtar je primenjen na q2, q1, q0 i p0, dajući filtrirani Q1. (p1 i q1 se nikad ne filtriraju za hromu, nego samo za lumu)

(b) $Bs = 4$:

Ako je |p2-p0| manja od β i |p0-q0| manja od round(α/4):
P0 se dobija 5-tap filtriranjem p2, p1, p0, q0 i q1
P1 se dobija 4-tap filtriranjem p2, p1, p0 i q0
(samo luma) P2 se dobija 5-tap filtriranjem p3, p2, p1, p0 i q0.
inače:
P0 se dobija 3-tap filtriranjem p1, p0 i q1.

Ako je |q2-q0| manja od β i |p0-q0| manja od round(α/4):
Q0 se dobija 5-tap filtriranjem q2, q1, q0, p0 i p1
Q1 se dobija 4-tap filtriranjem q2, q1, q0 i p0
(samo luma) Q2 se dobija 5-tap filtriranjem q3, q2, q1, q0 i p0.
inače:
Q0 se dobija 3-tap filtriranjem q1, q0 i p1.

2.6.4 Primer filtriranja

Video sekvenca je kodirana H.264 referentnim programskom podrškom sa fiksnim kvantizacionom parametrom 36 (relativno visok stepen kvantizacije). Slika 2.27 prikazuje originalnu sliku iz sekvence.

Slika 2.27 Originalna slika

Slika 2.28a prikazuje istu sliku nakon kodiranja u Intra režimu i rekonstrukcije, sa isključenim filtriranjem. Jasno se vide blokovski artifakti; vide se i efekti varirajuće veličine blokova za kompenzaciju pokreta. (na primer 16x16 blokovi u pozadini na levoj strani slike; 4x4 blokovi oko ruke).

Slika 2.28a Rekonstruisana slika (bez filtra) Slika 2.28b Rekonstruisana slika (sa filtrom)

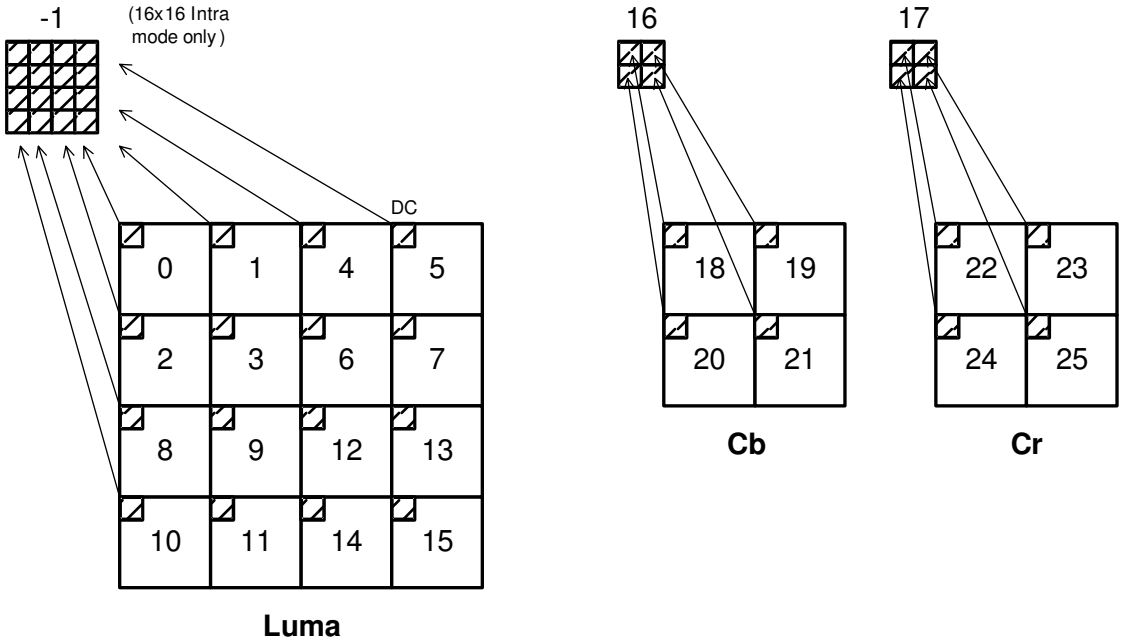
Sa uključenim filtriranjem (Slika 2.28b) izgled slike je znatno poboljšan; i dalje su tu neka očigledna izobličenja, ali većina ivica blokova je nestala ili ublažena. Na slici se vidi da su granice sa oštrim kontrastom (kao što je linija ruke ispred tamnog klavira) očuvane posle filtriranja, dok su ivice blokova u "ravnim" delovima slike (kao što je pozadina na levoj strani slike) blaža.

U ovom primeru filter pravi samo mali doprinos u efikasnosti kompresije: kodirani bitski zapis je oko 1.5% manji, a PSNR je oko 1% veći u slučaju kada je filtar aktiviran. Međutim, subjektivni kvalitet sekvence sa filtriranjem je značajno bolji. Dobitak u performansi kodovanja koji potiču od filtera zavisi od bitske brzine i sadržaja koji se kodira

2.7 Transformacija i kvantizacija

Slično prethodnim standardima za video kodiranje, H.264 takođe koristi transformaciju ostatka dobijenog nakon predikcije. H.264 je jedinstven u tome jer koristi čisto prostornu transformaciju koja se računa upotrebom celobrojen artmetike (grubu aproksimaciju od DCT) koja je 4x4 oblika, nasuprot uobičajenoj u ranijim standardima (MPEG-1, MPEG-2, MPEG-4 i H.263) 8x8 diskretnoj kosinusnoj transformaciji (DCT) koja se računa u pokretnom zarezu i za koju se specificiraju tolerancije za greške zaokruživanja. Mala prostorna veličina bloka nad kojom se vrši transformacija smanjuje artefakte nastale usled obrade slike u blokovima, dok precizna specifikacija operacija koje se izvode u celobrojnoj aritmetici eliminiše bilo kakav problem razmimoilaženja između kodera i dekodera u postupku inverzne transformacije. Dodatna 2x2 transformacija se primenjuje na 4 DC koeficijenta svake hroma komponente. Ako je makroblok kodirana u Intra-16x16 modu, slična 4x4 transformacija se izvodi za 4x4 DC koeficijente luma komponente.

Podaci u okviru makrobloka se šalju u poretku prikazanom na slici 2.29. Ako je makroblok kodiran u 16x16 Intra modu, tada se blok označen sa "-1" šalje prvi, on sadrži DC koeficijente od svakog 4x4 luma bloka. Sledeći, luma rezidualni blokovi označeni sa 0-15 se šalju u prikazanom poretku (sa DC koeficijentima postavljenim na nulu u makrobloku tipa 16x16 Intra). Blokovi 16 i 17 sadrže 2x2 DC koeficijente iz Cb i Cr hroma komponenti respektivno. na kraju, hroma rezidualni blokovi 18-25 (sa nultim DC koeficijentima) se šalju.



Slika 2.29 Poredak rezidualnih blokova u okviru makrobloka

2.7.1 4x4 transformacija i kvantizacija rezidualnih blokova 0-15, 18-25

Ova transformacija operiše nad 4x4 blokovima rezidualnih podataka (označenih sa 0-15 i 18-25 na slici 3.3-1) posle predikcije se kompenzacijom pokreta ili Intra predikcije. Transformacija je bazirana na DCT, ali sa nekoliko fundamentalnih razlika:

1. Sve operacije se mogu izvesti sa celobrojnom aritmetikom, bez gubitka na preciznosti.
2. Inverzna transformacija je potpuno specificirana u H.264 standardu i ako se ova specifikacija korektno implementira, ne bi trebalo da dođe do razmimoilaženja između kodera i dekodera.
3. Jezgro transformacije ne sadrži množenje, jedino se zahteva sabiranje i bitsko pomeranje.
4. Skalarno množenje (deo kompletne transformacije) je integrisano u kvantizator (smanjujući time ukupan broj množenja).

Kompletan proces transformacije i kvantizacije se može izvesti upotrebom 16-bitne celobrojne aritmetike i jednim množenjem po koeficijentu bez gubitka na preciznosti.

2.7.1.1 Izvođenje iz 4x4 DCT

4x4 DCT za ulaznu matricu **X** se dobija sa:

$$\mathbf{Y} = \mathbf{AXA^T} = \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix} \begin{bmatrix} a & b & a & c \\ a & c & -a & -b \\ a & -c & -a & b \\ a & -b & a & -c \end{bmatrix}$$

gde je: $a = \frac{1}{2}$, $b = \sqrt{\frac{1}{2}}\cos\left(\frac{\pi}{8}\right)$, $c = \sqrt{\frac{1}{2}}\cos\left(\frac{3\pi}{8}\right)$

Ovo matrično množenje se može faktorisati u sledeći ekvivalentni oblik:

$$\mathbf{Y} = \left(\mathbf{CXC^T}\right) \otimes \mathbf{E} = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & d & -d & -1 \\ 1 & -1 & -1 & 1 \\ d & -1 & 1 & -d \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & d \\ 1 & d & -1 & -1 \\ 1 & -d & -1 & 1 \\ 1 & -1 & 1 & -d \end{bmatrix} \right) \otimes \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix}$$

$\mathbf{CXC^T}$ je glavna 2-D transformacija. **E** je matrica skalirajućih faktora i simbol $\otimes$ označava da se svaki element od $\mathbf{CXC^T}$ množi skalirajućim faktorom koji se nalaz na istoj poziciji u matrici E (skalarno množenje umesto matričnog množenja). Konstante *a* i *b* su kao i ranije, dok je $d = c/b$ (približno 0.414).

Da bi se uprostila implementacija transformacije, *d* se aproksimira sa 0.5. Da bi se osiguralo da transformacija ostaje ortogonalna, *b* se takođe mora modifikovati tako da je :

$$a = \frac{1}{2}, \quad b = \sqrt{\frac{2}{5}}, \quad d = \frac{1}{2}$$

Da bi se izbeglo množenje sa 1/2 u glavnoj transformaciji $\mathbf{CXC^T}$ (što bi dovelo do gubitka preciznosti upotrebljavajući celobrojnu aritmetiku), drugi i četvrti red matrice **C** i druga i četvrta kolona matrice $\mathbf{C^T}$ su skalirane za faktor 2, to se kompenzuje skaliranjem za faktor 1/2 istih redova i kolona matrice **E**. Konačna direktna transformacija postaje:

$$\mathbf{Y} = \left(\mathbf{C_f X C_f^T}\right) \otimes \mathbf{E_f} = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \right) \otimes \begin{bmatrix} a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \\ a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \end{bmatrix}$$

Ova transformacija je aproksimacija 4x4 DCT. Zbog promene faktora *d* i *b*, rezultat nove transformacije neće biti identičan sa 4x4 DCT.

Primer:

4x4 blok **X** je transformisan sa 4x4 DCT i sa 4x4 aproksimativnom transformacijom.

	j = 0	1	2	3
i = 0	5	11	8	10
1	9	8	4	12
2	1	10	11	4
3	19	6	15	7

Rezultat DCT transformacije:

$$\mathbf{Y} = \mathbf{AXA^T} = \begin{bmatrix} 35.0 & -0.079 & -1.5 & -1.115 \\ -3.299 & -4.768 & -0.443 & -9.010 \\ 5.5 & 3.029 & 2.0 & 4.699 \\ -4.045 & -3.010 & -9.384 & -1.232 \end{bmatrix}$$

Rezultat aproksimativne transformacije:

$$\mathbf{Y'} = \left(\mathbf{CXC^T}\right) \otimes \mathbf{E} = \begin{bmatrix} 35.0 & -0.158 & -1.5 & 1.107 \\ -3.004 & -3.900 & 1.107 & -9.200 \\ 5.5 & 2.688 & 2.0 & 4.901 \\ -4.269 & -3.200 & -9.329 & -2.100 \end{bmatrix}$$

Razlika između DCT i aproksimativne transformacije:

$$\mathbf{Y} - \mathbf{Y'} = \begin{bmatrix} 0 & 0.079 & 0 & 0.008 \\ -0.295 & -0.868 & -0.664 & 0.190 \\ 0 & 0.341 & 0 & -0.203 \\ 0.224 & 0.190 & -0.055 & -0.868 \end{bmatrix}$$

Postoji očigledna razlika u koeficijentima koji zavise od b ili d. U kontekstu H.264, aproksimativna transformacija ima skoro identične kompresione performanse DCT-a i poseduje nekoliko važnih prednosti. Jezgro transformacije $\mathbf{CXC^T}$ može se izvesti upotrebom celobrojne aritmetike upotrebljavajući samo sabiranje, oduzimanje i pomeranje (da bi se implementiralo pomeranje za 2). Dinamički opseg operacija u transformaciji je takav da 16-bitna aritmetika može biti upotrebljena bez rizika od prekoračenja, sve dok su ulazni podaci unutar opsega od +/- 255. Skalirajuća operacija sa matricom **E** zahteva jedno množenje za svaki koeficijent, a ono može biti "apsorbovano" u kvantizacioni proces.

Inverzna transformacija je data sa:

$$\mathbf{X'} = \mathbf{C_i^T}\left(\mathbf{Y} \otimes \mathbf{E_i}\right)\mathbf{C}_i = \begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{bmatrix} \left(\begin{bmatrix} y_{11} & y_{12} & y_{13} & y_{14} \\ y_{21} & y_{22} & y_{23} & y_{24} \\ y_{31} & y_{32} & y_{33} & y_{34} \\ y_{41} & y_{42} & y_{43} & y_{44} \end{bmatrix} \otimes \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix} \right) \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix}$$

U ovom slučaju **Y** je prvo skalirana množenjem svakog koeficijenta odgovarajućim težinskim faktorom iz matrice $\mathbf{E_i}$. Treba primetiti da se faktori +/- 1/2 u matricama **C** i $\mathbf{C^T}$ mogu implementirati pomeranjem u desno bez značajnijeg gubitka na preciznosti zbog toga što su koeficijenti Y prethodno skalirani.

Direktna i inverzna transformacija su ortogonalne t.j. $\mathbf{T^{-1}(T(X)) = X.}$

2.7.2 Kvantizacija

Kvantizacije je postupak u kojem se obavlja značajniji deo kompresije podataka. U H.264, vrednosti dobijene nakon transformacije se kvantizuju upotrebom skalaranog kvantizatora široke mrtve zone. 52 različite veličine kvantizacionih koraka se mogu odabrati na nivou kodiranja makrobloka. (H.263 podržava 31 veličinu). Pored toga, u H.264 veličina koraka se uvećava sa stopom od 12.5%, a ne za konstantni iznos. Vernost hrominatne komponente je unapređena upotrebom finijih veličina kvantizaciong koraka u poređenju sa onim koje su upotrebljene za luminentnu koeficijente, pogotovu kada su luminentniu koeficijenti grubo kvantizovani.

Definicija i implementacija kvantizacije su komplikovani zbog zahteva da se izbegne aritmetika u pokretnom zarezu i da se inkorporira matrica $\mathbf{E_f}$ u proces kvantizacije.

Osnovni direktni kvantizator opisuje operacija definisana sa:

$$Z_{ij} = round\left(\frac{Y_{ij}}{Q_{step}}\right)$$

gde je:

- Y_{ij} koeficijenti dobijeni nakon direktne transformacije.
- Q_{step} veličina kvantizacionog koraka.
- Z_{ij} kvantizovani koeficijenti.

Ukupno 52 vrednosti za Q_{step} su podržani standardom i one su indeksirane kvantizacionim parametrom, QP. Vrednosti za Q_{step} koji odgovaraju svakom QP su prikazani u tabeli 2.2. Q_{step} se duplira kada se QP poveća za 6; Q_{step} se poveća za 12.5% kada se QP poveća za 1. Širok opseg kvantizaciong koraka omogućava da koder precizno i fleksibilno kontroliše balans između bitske brzine i kvaliteta.

QP	0	1	2	3	4	5	6	7	8
Q_{step}	0.625	0.687	0.8125	0.875	1	1.125	1.25	1.375	1.625
QP	9	10	11	12	...	18	...	24	...
Q_{step}	1.75	2	2.25	2.5	...	5	...	10	...
QP	30	...	36	...	42	...	48	...	51
Q_{step}	20	...	40	...	80	...	160	...	224

Tabela 2.2

Skalirajući fakori a^2, $ab/2$ ili $b^2/4$ su inkorporirani u direktni kvanitzator. Prvo se ulazni blok **X** transformiše do bi se dobio blok neskaliranih koeficijenata $\mathbf{W=CXC^T}$, a onda se svaki koeficijent W_{ij} kvantizije i skalira u jednoj operaciji definisanoj sa:

$$Z_{ij} = round\left(W_{ij} \cdot \frac{PF}{Q_{step}}\right)$$

PF je a^2, $ab/2$ ili $b^2/4$ u zavisnosti od pozicije (i,j) u matrici $\mathbf{E_f}$:

Pozicija (i,j)	PF
(0,0), (2,0), (0,2) or (2,2)	a^2
(1,1), (1,3), (3,1) or (3,3)	$b^2/4$
Sve preostale	$ab/2$

Faktor PF/Q_{step} je implementiran u referentnom modelu H.264 kao množenje sa MF (multiplikacioni faktor) i sa pomeranjem u desno, na taj način izbegava se operacija deljenja. Ova implementacija je definisana sa:

$Z_{ij} = round\left(W_{ij} \cdot \frac{MF}{2^{qbits}} \right)$, gde je: $\frac{MF}{2^{qbits}} = \frac{PF}{Q_{step}}$ i qbits = 15 + floor(QP/6)

U celobrojnoj aritmetici prethodna jednačina se može implementirati sa:

$Z_{ij} = \left(W_{ij} \cdot MF + f \right) >> qbits$, gde >> označava binarno pomeranje u desno.

U referentnom modelu f je $2^{qbits}/3$ za Intra ili $2^{qbits}/6$ za Inter makroblokove.

Primer:
Neka je kvantizacioni parametar QP = 4, tada je Qstep = 1.0, a qbits = 15 (jer je floor(4/6) = 0), pa je 2^{qbits} = 32768, a pozicija koeficijenta je (i,j) = (0,0), tada je PF = a^2 = 0.25. Koristeći da je:

$\frac{MF}{2^{qbits}} = \frac{PF}{Q_{step}}$, dobijamo da je MF = (32768x0.25)/1 = 8192

Prvih 6 vrednosti MF-a, mogu se izračunati na ovaj način u zavisnosti od QP i pozicije koeficijenta (i,j). One su date u tabeli

QP	Pozicija (0,0), (2,0), (0,2) ili (2,2)	Pozicija (1,1), (1,3), (3,1) ili (3,3)	Ostale pozicije
0	13107	5243	8066
1	11916	4660	7490
2	10082	4194	6554
3	9362	3647	5825
4	8192	3355	5243
5	7282	2893	4559

Tabela 2.3 Multiplikacioni faktori MF za QP=0,1,..,6

Za QP>5, faktori MF ostaju nepromenjeni ali delilac 2^{qbits} se povećava za faktor 2 kada se QP poveća za 6. Na primer qbits=16 za 6<QP<11; qbits = 17 za 12<QP<17; i tako dalje.

2.7.3 Reskaliranje (Dekvantizacija)

Osnovni dekvantizator (inverzni kvantizator) je definisan sa:

$Y'_{ij} = Z_{ij} \cdot Q_{step}$

Skalirajući faktor za inverznu transformaciju (matrica $\mathbf{E_i}$, koja sadrži vrednosti a^2, *ab/2* ili $b^2/4$ u zavisnosti od pozicije koeficijenta) je inkorporirana u ovu operaciju, zajedno sa skaliranjem za konstantan faktor 64 radi izbegavanja greška zaokruživanja.

:

$W'_{ij} = Z_{ij} \cdot Q_{step} \cdot PF \cdot 64$

W'_{ij} je skalirani koeficijent koji se onda transformiše jezgrom inverzne transformacije $\mathbf{C_i^T W C_i}$ (jednačina 3-2). Dobijeni rezultat se deli sa 64 da bi se uklonilo prethodno skaliranje za faktor 64. Ovo se sve može implementirati upotrebljavajući samo sabiranje i pomeranje u desno.

H.264 standard ne specificira Q_{step} ili PF direktno. Umesto njih parametar V=($Q_{step} \cdot PF \cdot 64$) se definiše za 0<QP<5 i svaku poziciju koeficijenta. Tada se inverzna transformacija definiše sa:

$W'_{ij} = Z_{ij} \cdot V_{ij} \cdot 2^{floor(QP/6)}$

Primer:

Neka je kvantizacioni parametar QP = 3, tada je Qstep = 0.875, a $2^{floor(QP/6)} = 1$, pozicija koeficijenta je (i,j) = (1,2), tada je PF = *ab* = 0.3162. Tada je V = ($Q_{step} \cdot PF \cdot 64$) = $0{,}875 \cdot 0{,}3162 \cdot 64 = 17{,}7072 \approx 18$, pa se dobija da je $W'_{ij} = Z_{ij} \cdot 18 \cdot 1$.

Vrednosti V za 0<QP<5 su definisane standardom i date su sledećom tabelom:

QP	Pozicija (0,0), (2,0), (0,2) ili (2,2)	Pozicija (1,1), (1,3), (3,1) ili (3,3)	Ostale pozicije
0	10	16	13
1	11	18	14
2	13	20	16
3	14	23	18
4	16	25	20
5	18	29	23

Tabela 2.4 Reskalirajući faktori V

Faktor $2^{floor(QP/6)}$ u jednačini čini da se rezultat reskaliranja uvećava za faktor 2 kada se QP poveća za 6.

2.7.4 Transformacija i kvantizacija 4x4 luma DC koeficijenata (samo 16x16 Intra tip)

Ako je makroblok kodiran u 16x16 Intra predikcijom (gde se kompletna 16x16 luminentna komponenta formira na osnovu susednih tačaka), svaki od 4x4 rezidualnih blokova se prvo transformiše upotrebom jezgra direktne transformacije ($\mathbf{C_f X C_f^T}$). A zatim se DC koeficijenti svakog 4x4 bloka transformišu upotrebom 4x4 Hadamard transformacije:

$$\mathbf{Y}_D = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} w_{D11} & w_{D12} & w_{D13} & w_{D14} \\ w_{D21} & w_{D22} & w_{D23} & w_{D24} \\ w_{D31} & w_{D32} & w_{D33} & w_{D34} \\ w_{D41} & w_{D42} & w_{D43} & w_{D44} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \right)$$

$\mathbf{W_D}$ je blok 4x4 DC koeficijenata,a $\mathbf{Y_D}$ je blok posle transformacije. Koeficijenti $Y_{D(i,j)}$ se dele sa 2 (sa zaokruživanjem). Oni se tada kvantizuju da bi se dobio blok kvantizovanih DC koeficijenata:

$$Z_{D(i,j)} = \left(Y_{D(i,j)} \cdot MF + 2f \right) >> \left(qbits + 1 \right)$$

gde su MF, f i qbits definisani kao i ranije, MF zavisi od pozicije (i,j) u okviru 4x4 bloka DC koeficijenata kao i ranije.

Na strani dekodera inverzna Hadamard transformacija se primenjuje pre dekvantizacije (ovo je u suprotnosti sa redosledom koji bi se očekivao).

$$\mathbf{W_{QD}} = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} z_{D11} & z_{D12} & z_{D13} & z_{D14} \\ z_{D21} & z_{D22} & z_{D23} & z_{D24} \\ z_{D31} & z_{D32} & z_{D33} & z_{D34} \\ z_{D41} & z_{D42} & z_{D43} & z_{D44} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \right)$$

Ako je QP veće ili jednako od 12, dekvantizacija se izvodi sa:

$$W'_{D(i,j)} = W_{QD(i,j)} \cdot V_{(0,0)} \cdot 2^{floor(QP/6)-2}$$

Ako je QP manje od 12, dekvantizacija se izvodi sa:

$$W'_{D(i,j)} = \left[W_{QD(i,j)} \cdot V_{(0,0)} + 2^{1-floor(QP/6)}\right] >> \left(2 - floor(QP/6)\right)$$

V je definisano kao i pre. Dekvantizovani koeficijenti $\mathbf{W'_D}$ se onda umeću u njihove odgovarajuće 4x4 blokove koeficijenata. Zatim se svaki blok inverzno transformiše upotrebom jezgra,DCT bazirane,inverzne transformacije ($\mathbf{C_i^T W C_i}$).

U makroblokovima kodiranim u intra modu, glavnina energije je koncentrisana u DC koeficijentima i ova dodatna transformacija pomaže u dekorelaciji 4x4 luma DC koeficijenata (t.j. da se iskoristi korelacija između koeficijenata).

2.7.5 Transformacija i kvantizacija 2x2 hroma DC koeficijenata

Svaka hroma komponenta u makrobloku je sastavljena od četiri 4x4 bloka tačaka. Svaki 4x4 blok se transformiše na način koji je ranije opisan. DC koeificijenti svakog 4x4 bloka koeficijenata se grupišu u 2x2 blok $\mathbf{W_D}$.Oni se još jednom transformišu pre kvantizacije:

$$\mathbf{Y_D} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} w_{D11} & w_{D12} \\ w_{D21} & w_{D22} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Kvantizacija 2x2 bloka $\mathbf{Y_D}$ se izvodi sa:

$$Z_{D(i,j)} = \left(Y_{D(i,j)} \cdot MF + 2f\right) >> \left(qbits + 1\right)$$

gde su MF, f i qbits definisani kao i ranije.

Tokom dekodiranja, inverzna transformacija se primenjuje pre dekvantizacije:

$$\mathbf{W_{QD}} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} z_{D11} & z_{D12} \\ z_{D21} & z_{D22} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Ako je QP veće ili jednako od 12, dekvantizacija se izvodi sa:

$$W'_{D(i,j)} = W_{QD(i,j)} \cdot V_{(0,0)} \cdot 2^{floor(QP/6)-2}$$

Ako je QP manje od 12, dekvantizacija se izvodi sa:

$$W'_{D(i,j)} = \left[W_{QD(i,j)} \cdot V_{(0,0)}\right] >> 1$$

Dekvantizovani koeficijenti se onda umeću u njihove odgovarajuće 4x4 hroma blokove koeficijenata.Koji se zatim inverzno transformišu kao i ranije ($\mathbf{C_i^T W C_i}$). Kao i kod luma DC koeficijenata ova dodatna transformacija pomaže u dekorelaciji 2x2 hroma DC koeficijenata te stoga poboljšava performanse kompresije.

2.7.6 Kompletan proces transformacije , kvantizacije, dekvantizacije i inverzne transformacije

Kompletan postupak od ulaznog rezidualnog bloka **X** do izlaznog rezidualnog bloka **X'** je opisan ispod i ilustrovan je na slici 2.

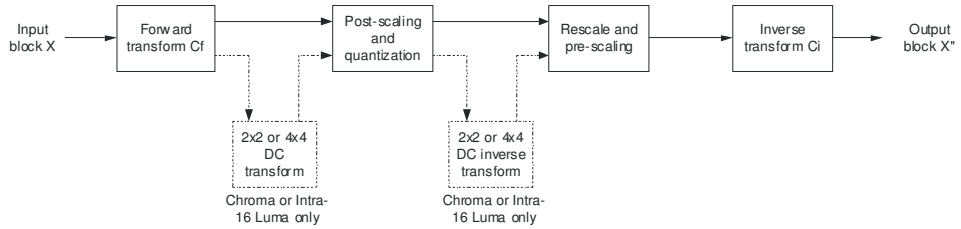
Kodiranje:

1. Ulaz: 4x4 rezidualne vrednosti: **X**
2. Direktna "glavna" transformacija: $\mathbf{W = C_f X C_f^T}$,
   (praćena sa direktnom transformacijom za hroma DC ili Intra-16 Luma DC koeficijente)
3. Skaliranje i kvantizacija: $\mathbf{Z = W \cdot \frac{PF}{Q_{step} \cdot 2^{qbits}}}$

(modifikovano za hroma DC ili Intra-16 Luma DC koeficijente)

Dekodiranje:
(inverzna transformacija za hroma DC ili Intra-16 Luma DC koeficijente)

4. Dekvantizacija (sa uključenim pred skaliranjem inverzne transformacije):
   $$\mathbf{W' = Z \cdot Q_{step} \cdot PF \cdot 64}$$
   (modifikovana za hroma DC ili Intra-16 Luma DC koeficijente)
5. Inverzna "glavna" transformacija $\mathbf{X' = C_i^T W C_i}$
6. Skaliranje: $\mathbf{X'' = round(X'/64)}$
7. Izlaz: 4x4 rezidualne vrednosti: **X''**



Slika 2.30 Tok transformacije, kvantizacije, reskaliranja i inverzne transformacije

Primer (luma 4x4 rezidualni blok, Inter mod):

Ulazni blok **X:**

	j = 0	1	2	3
i = 0	5	11	8	10
1	9	8	4	12
2	1	10	11	4
3	19	6	15	7

Rezultat "glavne" transformacije **W**:

	j = 0	1	2	3
i = 0	140	-1	-6	7
1	-19	-39	7	-92
2	22	17	8	31
3	-27	-32	-59	-21

MF = 8192, 3355 ili 5243 (zavisno od pozicije) i qbits=16. Rezulat kvantizacije **Z**:

	j = 0	1	2	3
i = 0	17	0	-1	0
1	-1	-2	0	-5
2	3	1	1	2
3	-2	-1	-5	-1

V = 16, 25 ili 20 (zavisno od pozicije) i $2^{floor(QP/6)} = 2^1 = 2$. Rezultat dekvantizacije **W'**:

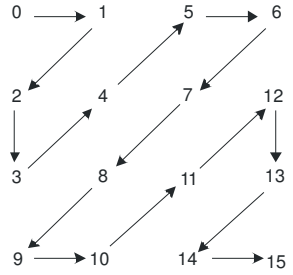
	j = 0	1	2	3
i = 0	544	0	-32	0
1	-40	-100	0	-250
2	96	40	32	80
3	-80	-50	-200	-50

Rezultat "glavne" inverzne transformacije (posle finalnog skaliranja) **X''**:

	j = 0	1	2	3
i=0	4	13	8	10
1	8	8	4	12
2	1	10	10	3
3	18	5	14	7

2.7.7 Cik-cak pregrupisavanje koeficijenta transformacije

Za tipični blok slike najveći broj koeficijenta dobijenih nakon trasformacije i kvantizacije će po vrednosti biti nula. Većina koeficijenata koji imaju vrednost veću od nule se obično grupiše u gornjem levom uglu bloka, t.j. u oblasti oko DC koeficijenta. Da bi se iskoristila ta osobina u kodovanju, koeficijenti bloka se cik-cak metodom pregrupišu. Na taj način dobijaju se nizovi vrednosti koeficijenata u kojima se javljaju duži podnizovi uzastopnih nultih vrednosti. Takvi nizovi se efikasno kodiraju entropijskim kodiranjem. Cik-cak pregrupisavanje je prikazano na slici 2.31.



Slika 2.31 Cik-cak pregrupisavanje

2.8 Entropijsko kodiranje

2.8.1 Kodovani elementi

Parametri, koje traba kodavati i poslati, su navedeni u narednoj tabeli.

Parametar	Opis
Sintaksni elementi sloja sekvence, slike i dela slike.	
Tip makrobloka	Predikcioni mod za svaki kodovani makroblok
Šema kodovanih blokova	Pokazuje koji blokovi unutar makrobloka sadrže kodovane koeficijente
Kvantizacioni parametar	Šalje se kao delta vrednost u odnosu na prethodnu vrednost QP
Indeks referentne slike	Indentifikje referentnu sliku za intra predikciju
Vektor pomeraja	Šalje se kao razlika (MVD) od estimiranog vektora pomeraja
Rezidualni podaci	Koeficijenti za svaki 4x4 ili 2x2 blok

Table 2.5 Parametri koji se koduju

Iznad sloja dela slike, sintaksni elementi se kodiraju kao binarni kodovi fiksne ili promenjive dužine. U sloju dela slike i ispod njega, elementi se koduju upotrebom kodova promenljive dužine (VLCs) ili konteksno-baziranim adaptivnim binarno aritmetičkim kodovima (CABAC) u zavisnosti od moda entropijskog kodiranja.

2.8.2 Kodovi sa promenjivom dužinom (VLCs)

Kada je mod entropijskog kodovanja (entropy_coding_mode) postavljen na 0, rezidualni blok podataka se koduje upotrebom konteksno-adaptivnog koda sa promenjivom dužinom (*Context-Adaptive Variable Length Coding)* CAVLC, dok se ostali podaci koduju upotrebom Exp-Golomb kodova.

2.8.2.1 Exp-Golomb entropijsko kodiranje

Exp-Golomb kodovi (Eksponencijalni Golomb kodovi) su kodovi promenljive dužine sa pravilnom konstrukcijom.

code_num	Kodna reč
0	1
1	010
2	011
3	00100
4	00101
5	00110
6	00111
7	0001000
8	0001001
…	…

Tabela 2.6 Exp-Golomb kodne reči

Tabela 2.6 pokazuje prvih 9 kodnih reči; iz ove tabele je jasno se kodne reči razvijaju u logičnom poretku. Svaka kodna reč se konstruiše na sledeći način:

M [nula][1][INFO]

gde je sa INFO označeno polje od M bitova koje nosi informaciju. Prva kodna reč nema vodećih niti pratećih nula; kodne reči 1 i 2 imaju INFO 1-bitno polje; kodne reči 3 do 6 imaju 2-bitno polje; i tako dalje. Dužina svake kodne reči je (2M+1) bitova.

Svaka Exp-Golomb kodna reč može da se konstruiše u koderu na osnovu njenog indeksa **code_num**:

$M = \log_2(code\_num+1)$
$INFO = code\_num + 1 - 2M$

Kodna reč može biti dekodovna sledećim postupkom:

1. Pročita se M vodećih nula t.j. odredi se broj nula pre prve jedinice.
2. Pročita se M-bitno INFO polje.
3. $code\_num = 2^M + INFO - 1$ (Za kodnu reč 0, INFO i M su nula).

Parametar **v**, koji treba kodovati, može da se predstavi pomoću code_num na jedan od sledeća 3 načina:

ue(**v**): code_num = **v**. (mapiranje neoznačenih vrednosti). Upotrebljava se za kodovanje tip makrobloka, indeksa referentne slike i QP i druge parametre

se(**v**): **v** se predstavlja sa code_num na sledeći način: (mapiranje označenih vrednosti).

$code\_num = 2|v| \quad ,(v < 0)$
$code\_num = 2|v| - 1 \quad ,(v > 0)$

v	code_num
0	0
1	1
-1	2
2	3
-2	4
3	5
…	…

Tabela 2.7 Mapiranje označenih vrednosti se(v)

Upotrebljava se za kodovanje razlike vektora pomeraja, promenu QP i druge parametre.

me(**v**): **v** se predstavlja sa code_num pomoću tabele koja je specificirana standardom (mapiranje simbola). Ovaj način se upotrebljava za kodovanje parametra koji određuje šemu kodovanih blokova (coded_block_pattren) unutar makrobloka. Table 2.8 je mali deo tabele za makroblokove kodovane u Intra režimu: šema kodovanih blokova pokazuje koji od 8x8 blokova u makrobloku sadrži koeficijente različite od nule.

šema kodovanih blokova (Inter režim)	code_num
0 (no non-zero blocks)	0
16 (chroma DC block non-zero)	1
1 (top-left 8x8 luma block non-zero)	2
2 (top-right 8x8 luma block non-zero)	3
4 (lower-left 8x8 luma block non-zero)	4
8 (lower-right 8x8 luma block non-zero)	5
32 (chroma DC and AC blocks non-zero)	6
3 (top-left and top-right 8x8 luma blocks non-zero)	7
…	…

Tabela 2.8 Deo tabele za mapiranje šeme kodovanih blokova (Intra režim)

Svaki od ovih načina mapiranja (ue, se i me) je osmišljen da proizvede kratku kodnu reč za vrednosti parametara koje se često pojavljuju,a dugačku kodnu reč za vrednosti parametara koje se pojavljuju ređe. Na primer, makrobloku tipa Pred_L0_16x16 (t16x16 predikcija iz prethodne slike) je dodeljen code_num 0 zato što se pojavljuje često, dok je makrobloku tipa Pred_8x8 (8x8 predikcija iz prethodne slike) dodeljen code_num 3 zato što se pojavljuje ređe. Vrednost razlike vektora pomeraja (MVD) koja se često pojavljuje je 0 i ona se predstavlja se code_num 0, dok se vrednost MVD = -3 predstavlja sa code_num 6, jer se ona pojavljuje ređe.

2.8.2.2 Konteksno-adaptivni kod sa promenjivom dužinom (CAVLC)

Ova metoda kodiranje se koristi za dobijanje, cik-cak poređanih 4x4 (i 2x2) blokova koeficijenata transformacije. CAVLC je dizajniran da iskoristi nekoliko karakteristika kvantizovanog 4x4 bloka:

1. Posle predikcije, transformacija i kvantizacije, blokovi su obično retki (sadrže uglavnom nule). CAVLC upotrebljava run-level kodiranje za kompaktno predstavljanje nizova nula koeficijenata.
2. Najveći koeficijenti različiti od nule posle cik-cak skaniranja su često sekvence od +/-1. CAVLC signalizira broj visoko frekventnih +/-1 koeficijenata ("Prateće jedince (Trailing 1s)" ili "T1s") na kompaktan način.
3. Broj koeficijenata različitih od nule u susednim blokovima je korelisan. Taj broj se koduje upotrebom look-up tabela; izbor look-up tabele zavisi od broja koeficijenata različitih od nule u susednim blokovima.
4. Nivo (magnituda) koeficijenata različitih od nule teži da bude veća na početku niza koeficijenata, koji se dobija cik-cak skaniranjem, (bliže DC koeficijentu) i da se smanjuje kako idemo prema većim frekvencijama. CAVLC koristi ovo za izbor VLC look-up tabela za parametar "nivo" (level) u zavisnosti od nivoa koeficijenata koji prethodno kodovani.

CAVLC kodovanje bloka koeficijenata transformacije sprovodi se na sledeći način:

2.8.2.2.1 Kodiranje broja koeficijenata i pratećih jednica (coeff_token).

Prvi kod sa promenljivom dužinom, coeff_token, kodira ukupan broj koeficijenata različitih od nule (TotalCoeffs) i broj pratećih +/-1 vrednosti (TrailingOnes). TotalCoeffs može biti bilo koja vrednost između 0 (nema koeficijenata u 4x4 bloku) i 16 (16 koeficijenata različitih od

nule) (Napomena: šema kodovanih blokova (prethodno opisana) pokazuje koji od 8x8 blokova u makrobloku sadrži koficijente različite od nule, međutim, u okviru 8x8 bloka, mogu da postoje 4x4 podblokovi koji ne sadrže nijedan od tih koeficijenata, stoga TotalCoeffs može da bude 0 u bilo kom 4x4 podbloku. Činjenica je da se ova vrednost TotalCoeffs pojavljuje najčešće, pa joj je dodeljen najkraći kod promenljive dužine). TrailingOnes može biti bilo koja vrednost od 0 do 3; ako postoji više od 3 prateće jedinice (+/-1), samo se tri poslednja tretiraju kao "specijalni slučaj", dok se druge kodiraju kao normalni koeficijenti.

Postoji izbor između 4 look-up tabele koje se upotrebljavaju za kodiranje coeff_token i koje se nazivaju Num-VLC0, Num-VLC1, Num-VLC2 and Num-FLC (3 tabele sa promenljivom dužinom koda i jedna sa fiksnom dužinom koda). Izbor tabele zavisi od broja koeficijenata različitih od nule u blokovima, koji su prethodno kodirani, koji se nalaze iznad (blok U) i na levo (blok L) od tekućeg bloka. Označimo sa N_u broj koeficijenata različitih od nule u bloku iznad, a sa N_l broj koeficijenata različitih od nule u bloku levo. Parameter N se računa na sledeći način: Ako su blokovi U i L dostupni, $N = (N_u + N_l)/2$. Ako je samo blok U dostupan, $N=N_u$; ako je samo blok L dostupan , $N=N_l$, ako nijedan nije dostupan, $N = 0$.

Parametar N služi za odabir look-up tabele (Tabela 2.9), na ovaj način se kodiranje, kodom promenjivoe dužine, prilagođava u zavisnosti od broja kodovanih koeficijenata u susednim blokovima (konteksna adaptivnost). Num-VLC0 tabela je "balansirana" prema malom broju koeficijenata (različitih od nule); niskim vrednostima TotalCoeffs (0 i 1) su dodeljeni kratki kodovi, dok su višljim vrednostima TotalCoeffs dodeljeni duži kodovi. Num-VLC1 je "balansirana" prema srednjem broju koeficijenata (vrednostima TotalCoeffs oko 2 do 4 su dodeljeni relativno kratki kodovi), Num-VLC2 je "balansirana" prema visokom broju koeficijenata. Num-FLC dodeljuje 6-bitni fiksni kod svakoj vrednosti TotalCoeff.

N	Tabela za coeff_token
0, 1	Num-VLC0
2, 3	Num-VLC1
4, 5, 6, 7	Num-VLC2
8 ili više	FLC

Tabela 2.9 Izbor look-up tabele za coeff_token

2.8.2.2.2 Kodiranje znaka svake prateće jedince (TrailingOnes)

Za svaku TrailingOnes (prateća +/-1) kodovanu sa coeff_token, koristi se po jedan bit za kodovanje znaka (0 = +, 1 = -). Oni se koduju u kontra redosledu, počevši od prateće +/-1 sa najvećom frekvencijom.

2.8.2.2.3 Kodiranje nivoa preostalih koeficijenata različitih od nule

Nivo (znak i magnituda) svakog preostalog koeficijenata različitog od nule u bloku, se koduje u kontra redosledu, počevši od koeficijenata sa najvišljom frekvencijom pa do DC koeficijenta. Izbor look-up tabele, sa kodovima promenljive dužine, za kodovanje svakog nivoa, se menja u zavisnosti od magnitude svakog prethodno kodovanog nivoa (konteksna adaptivnost). Postoji ukupno 7 takvih tabela, od tabele Level_VLC0 pa do tabele Level_VLC6; Level_VLC0 je balansirana prema malim nivoima; Level_VLC1 je balansirana prema malo većim nivoima i tako dalje. Izbor tabele se vrši na sledeći način:

(a) Kodovanje se počinje sa tabelom Level_VLC0 (sem ako postoji više od 10 koeficijenata različitih od nule i manje od 3 prateće jedinice, u tom slučaju počinje se sa tabelom Level_VLC1).
(b) Kodira se koeficijent različit od nule se najvećom frekvencijom.
(c) Ako je magnituda ovog koeficijenta veća od unapred definisanog praga, prelazi se na sledeću Level_VLC tabelu.

Na ovaj način izbor tabele se poklapa sa magnitudom prethodno kodovanih koeficijenata. Definisani pragovi se nalaze u tabeli 2.10; prvi prag je nula što znači da je se uvek posle kodovanja prvog koeficijenta tabelom Level_VLC0 prelazi na tabelu Level_VLC1.

Trenutna VLC tabela	Prag za promenu tabele
VLC0	0
VLC1	3
VLC2	6
VLC3	12
VLC4	24
VLC5	48
VLC6	N/A (poslednja tabela)

Tabela 2.10 Pragovi za određivanje promene trenutne Level_VLC tabele

2.8.2.2.4 Kodiranje ukupnog broja nula pre poslednjeg koeficijenta.

TotalZeros je zbir svih nula koja prethode poslednjem koeficijentu različitom od nule (u cik-cak preuređenom nizu). Ovaj parametar se kodira sa kodom promenljive dužine. Razlog za slanje posebnog koda promenljive dužine TotalZeros je u tome što većina blokova sadrži koeficijente različite od nule na početku niza, ovaj pristup znači da nizovi nula na početku ne moraju da se kodiraju

2.8.2.2.5 Kodiranje nizova nula

Broj nula koji prethodi svakom koeficijentu različitom od nule (run_before) se kodira u obrnutom redosledu. Parametar (run_before) se koduje za svaki koeficijent različit od nule, počevši od koeficijenta najviše frekvencije, sa dva izuzetka:

(a) Ako nema više nula sa se koduje (t.j. [run_before] = TotalZeros), tada više nije više potrebno kodirati ni jednu run_before vrednost.
(b) Nije potrebno kodovati run_before za poslednji (najniže frekvencije) koeficijent različit od nule.

Kod promenjive dužine za svaki niz nula se bira u zavisnosti od:

(a) broja nula koji jos nije kodiran (ZerosLeft)
(b) run_before

Na primer, ako je samo još 2 nule ostalo da se kodira run_before može da uzme samo 3 vrednosti (0,1 ili 2), tako da kod promenljive dužine ne treba da bude duži od 2 bita; ako je ostalo još 6 nula da se koduje tada run_before može da uzme 7 vrednosti (0 do 6) tako da kod promenljive dužine mora biti značajno veći.

2.8.3 Konteksno-bazirano adaptivno binarno aritmetičko kodiranje (CABAC)

Kada je mod entropijskog kodovanja (entropy_coding_mode) postavljen na 1, aritmetički sistem kodiranja se upotebljava za kodiranje i dekodiranje sintaksnih elemenata. Aritmetička vrsta kodiranja u H.264, konteksno-bazirana adaptivna binarno aritmetičko kodiranje ili CABAC, postiže dobre performans kompresije kroz (a) odabir modela verovatnoće za svaki sintaksni element u skladu sa njegovim sadržajem, (b) prilagođavanje procena verovatnoća, oslanjanjem na lokalnu statistiku i (c) upotrebom aritmetičkog kodiranja. Postupak kodiranje uključuje sledeće etape:

1. Binarizacija: CABAC koristi binarno aritmetičko kodiranje što znači da su samo binarni simboli (1 ili 0) kodovani. Simbol koji nije binarni (npr. koeficijent transformacije ili vektor pomeraja) se pretvara u binarni kod pre aritmetičkog kodiranja. Ovaj postupak je sličan postupku konverzije u kod sa promenjivom dužinom, ali se binarni kod dalje kodira (sa aritmetičkim koderom) pre slanja.

Etape 2, 3 i 4 se ponavljaju za svaki bit (ili "bin") binarizovanog simbola.

2. Izbor konteksnog modela: "kontekstni model" je model verovatnoće za jedan ili više "bin"-ova "binarizovanog" simbola. Ovaj model može biti izabran iz asortimana dostupnih modela u zavisnosti od statistike prethodno kodovanih simbola. Kontekstni model čuva verovatnoću svakog "bin"-a o tome da li 1 ili 0.

3. Aritmetičko kodiranje: Aritmetički koder kodira svaki bin u skladu sa odabranim modelom verovatnoće. Postoje samo dva pod opsega za svaki bin (koji odgovaraju "0" i "1").

4. Ažuriranje verovatnoća: Odabrani kontekstni model se ažurira na osnovu stvarno kodovane vrednosti (npr. ako je vrednost "bin"-a bila "1", frekvencija pojave "1" se povećava).

2.8.3.1 Postupak kodiranja

Ovaj postupak biće ilustrovan na primeru kodovanja MVD_x (razlika vektora pomeraja po x pravcu)

1. Binarizacija vrednosti MVD_x. Binarizacija se sprovodi u skladu sa tabelom za $|MVD_x| < 9$ (veće vrednosti MVD_x se binarizuju upotrebom Exp-Golomb kodnih reči). Prvi bit binarizovanog simbola je bin 1; drugi bit je bin 2; i tako dalje.

$\|MVD_x\|$	Binarizacija
0	0
1	10
2	110
3	1110
4	11110
5	111110
6	1111110
7	11111110
8	111111110

2. Izbor konteksnog modela za svaki bin. Jedan od 3 modela se bira za bin 1, na osnovu prethodno kodovanih MVD vrednosti. L1 norma dve prethodno kodovane vrednosti, e_k, se računa kao:

   $e_k = |MVD_A| + |MVD_B|$ gde su A i B blokovi neposredno levo od i iznad tekućeg bloka (respektivno)

e_k	Kontekstni model za bin 1
$0 \leq e_k < 3$	Model 0
$3 \leq e_k < 33$	Model 1
$33 \leq e_k$	Model 2

   Ako je e_k malo, tada postoji velika verovatnoća da će tekući MVD imati malu magnitiudu; nasuprot tome, ako je e_k veliko tada je verovatnije da će tekući MVD imati veliku magnitudu. Prema e_k odabira se tabela verovatnoća (kontekstni model).

   Preostali bin-ovi se kodiraju upotrebom naredna 4 kontekstna modela:

Bin	Kontektsni model
1	0, 1 ili 2 zavisno od e_k
2	3
3	4
4	5
5	6
6 i veći	6

3. Kodovanje svakog bin-a. Odabrani kontekstni model obezbeđuje dve procenjene verovatnoće: verovatnoću da bin sadrži "1" i verovatnoću da bin sadrži "0". Ove procene određuju dva pod opsega, koje aritmetički koder koristi za kodovanje bin-a.
4. Ažuriranje kontekstnog modela. Na primer, ako je kontekstni model 2 izabran za bin 1 i vrednost bin 1 je "0", frekvencija pojave "0" se uvećava. Ovo znači da će sledeći put, kada ovaj model bude bio izabran, verovatnoća za "0" biti nešto malo veća. Kada ukupan broj pojavljivanja modela pređe vrednost praga, frekvencija pojave za "0" i "1" se skalira na dole, što kao efekat ima davanje većeg prioriteta skorijim opservacijama.

2.8.3.2 Kontekstni modeli

Kontekstni modeli i pravila binarizacije za svaki sintaksni element su definisani standardom. Postoji ukupno 267 odvojenih kontekstnih modela, od 0 do 266 za razne sintaksne elemente. Neki modeli imaju različitu upotrebu u zavisnosti od tipa slike: na primer, preskočeni makroblokovi nisu dozvoljeni u I delovima slike, tako da se kontekstni modeli od 0 do 2 upotrebljavaju za kodovanje binova od mb_skip ili mb_type u zavisnosti dali je tekući deo slike kodovan u intra režimu.

Na početku svakog kodovanog dela slike, kontekstni modeli se inicijalizuju u zavisnosti od inicijalne vrednosti kvantizacionog parametra QP (pošto on ima značajan efekat na verovatnoću pojavljivanja raznih simbola).

2.8.3.3 Aritmetičko kodiranje

Aritmetički dekoder je u opisan u standardu. On ima tri upečatljive osobine:

1. Estimacija verovatnoće se izvodi prelaznim procesom između 64 odvojena verovatnosna stanja za "Najmanje Verovatan Simbol".
2. Opseg R koji predstavlja trenutno stanje aritmetičkog kodera je kvantizovan na male opsege unapred postavljenih vrednosti pre računanja novog opsega u svakom koraku, na ovaj način je moguće računanje novog opsega upotrebom look-up tabela (t.j. nema množenja).
3. Pojednostavljeni postupak kodiranja i dekodiranja je definisan za simbole sa skoro uniformnom raspodelom verovatnoće.

Definicija postupka dekodiranja je osmišljena tako da olakša implementaciju sa malom kompleksnošću aritmetičkog kodiranja i dekodiranja. CABAC obezbeđuje poboljšanu efikasnost kodiranja u poređenu sa VLC na račun veće računske kompleksnosti.

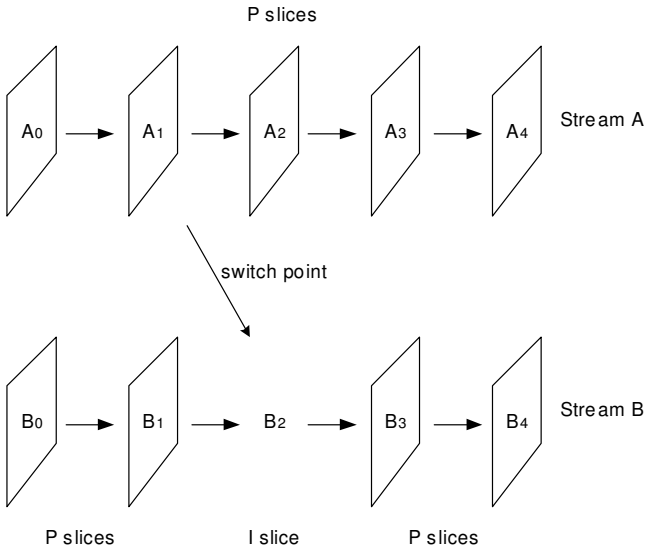
2.9 Preključivajuće P i I slike

2.9.1 Slike kodovane u SP i SI režimu

Režimi kodovanja SP i SI pripadju proširenom (Extended) profilu standarda H.264. Slike kodovne u SP i SI režime su slike, kodovane na specifičan način, koje omogućavaju, između ostalog, efikasno preključivanje između video sekvenci i efikasno slučajno pozicioniranje unutar sekvence na strani video dekodera. Ovo je čest zahtev koji se postavlja video dekoderu. Na primer, isti video materijal je kodovan na više bitskih brzina za prenos preko Internet-a; dekoder će pokušati da dekodira bitsku sekvencu sa najvećom bitskom brzinom koju može da primi, ali može da se automatski prebaci na sekvencu sa manjom bitskom brzinom ako propusna moć kanala, koji prenosi podatke, opadne.

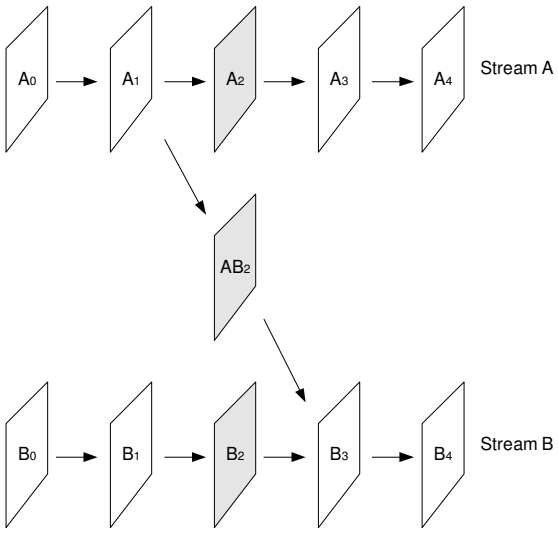
Primer: Dekoder dekodira Sekvencu A i želi da se prebaci na dekodovanje Sekvence B (Slika 2.32). Zbog jednostavnosti, pretpostavimo da je svaka slika kodovana u celini i da su predicirane na osnovu jedno referentne slike (prethodno dekodovane slike). Posle dekodiranja P slika A_0 i A_1 dekoder želi da se prebaci na Sekvencu B i da dekoduje B_2, B_3 i tako dalje. Ako su sve slike u Sekvenci B kodovane u P režimu, tada dekoder neće imati ispravno dekodovanu sliku potrebnu za rekonstrukciju slike B_2 (pošto je B_2 predicirana iz B_1,a ona ne postoji u Sekvenci A). Jedno rešenje je da se slika B_2 kodira u I režimu. Pošto se kodira bez referenciranja na druge

slike, može se dekodovati nezavisno od prethodnih slika u Sekvenci B. Sada dekoder može da se prekljućuje između Sekvence A i Sekvence B kao što je prikazano na slici 2.32. Prekljućivanje se može izvesti ubacivanjem I slika u pravilnim intervalima u kodovanoj sekvenci, kao bi se napravile "tačke za prekljućivanje". Međutim, slika kovana u I režimu će vrlo verovatno sadržavati mnogo više podataka od slike kodovane u P režimu što kao rezultat daje nagli porast bitske brzine u svakoj tački prekljućivanja.



Slika 2.32 Prekljućivanje sekvenci upotrebom slika kodovanih u I režimu

SP slike su osmišljene da podrže prekljućivanje bez porasta bitske brzine koja je prouzrokovana slikama kodovanim u I režimu. Slika 2.33 pokazuje kako se mogu upotrebiti. U tački prekljućivanja (slika broj 2 u svakoj sekvenci), sada se nalaze 3 slike kodovane u SP režimu. One su kodovane predikcijom sa kompenzacijom pokreta (što ih čini efikasnijim od slika kodovanih u I režimu). Slika u SP režimu može da se dekoduje upotrebom referentne slike A_1; SP slika B_2 može da se dekoduje upotrebom referentne slike B_2. Ključ postupka prekljućivanja je SP slika AB_2 (prekljućivačka SP slika): ona je napravljena na taj način da se njenim dekodovanjem na osnovu referentne slike A_1 (upotrebom kompenzacije pokreta) dobija slika B_2 (t.j. slika B_2 je ista bilo da smo je dobili posle B1 ili iz A1 (preko AB2)). Ova dodatana SP slika je potrebna u svakoj tački prekljućivanja (potrebna je još jedna SP slika, BA_2 ako je potrebno prekljućivanje u drugom smeru). Ovo je efikasnije od kodovanja A_2 i B_2 u I režimu. Tabela 2.10 navodi korake koji su prisutni kad se dekoder prekljućuje sa Sekvence A na Sekvencu B.

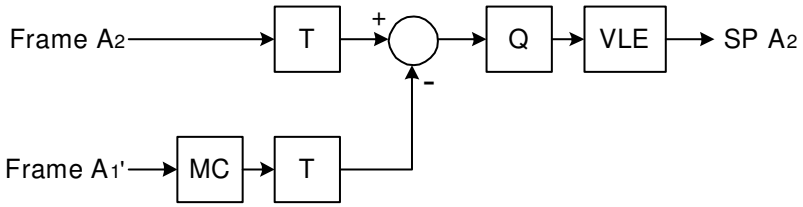


Slika 2.33 Prekljućivanje sekvenci upotrebom slika kodovanih u SP režimu

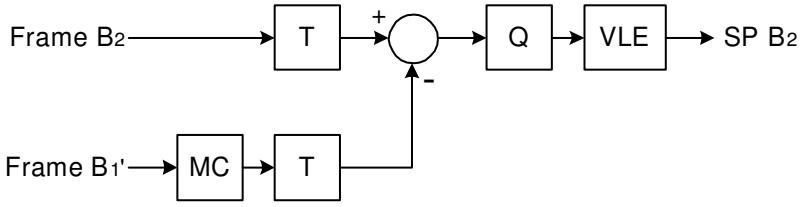
Ulaz u dekoder	Referenta slika	Izlaz iz dekodera
P-slika A_0	[prethodno dekodovana slika]	Dekodovna slika A_0
P-slika A_1	Dekodovna slika A_0	Dekodovna slika A_1
SP-slika AB_2	Dekodovna slika A_1	Dekodovna slika B_2
P-slika B_3	Dekodovna slika B_2	Dekodovna slika B_3
….	….	….

Tabela 2.10 Prekljućivanje iz Sekvnce A u Sekvencu B upotrebom SP slika

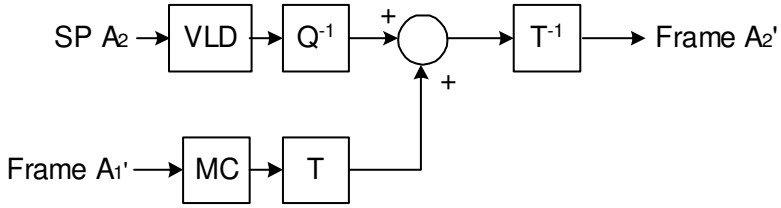
Uprošćeni dijagram postupka kodiranja slike A_2 u SP režimu je prikazan na Slici 2.34. A_2 se formira oduzimanjem verzije slike A_1' koja je kompenzovana na pokret (dekodovana slika A_1) od slike A_2 i kodovanjem ostatka. Za razliku od "normalne" P slike, oduzimanje se dešava u domenu transformacije (t.j. posle direktne transformacije blokova). SP slika B_2 se kodira na isti način (Slika 2.35). Dekoder koji poseduje prethodno dekodovanu sliku A_1 može da dekoduje SP sliku A_2 na način prikazan na slici 2.36 (zbog jasnoće prikazan je pojednostavljeni dijagram)



Slika 2.34 Kodovanje slike A_2 u SP režimu (uprošćen postupak)

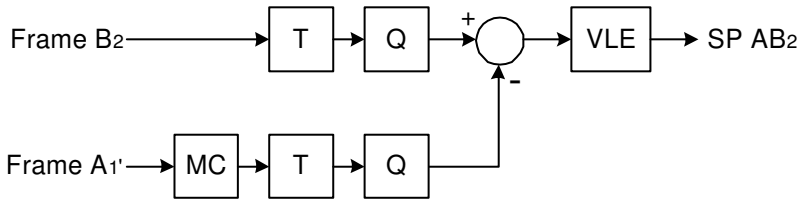


Slika 2.35 Kodovanje slike B_2 u SP režimu (uprošćen postupak)

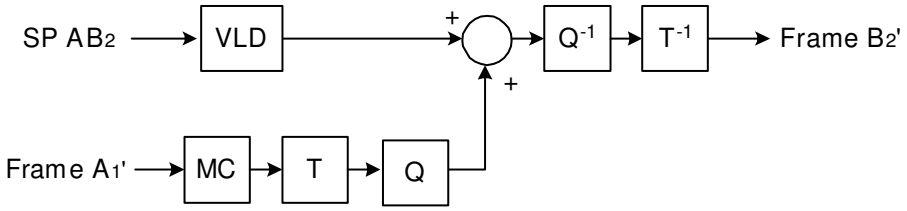


Slika 2.36 Dekodovanje A_2 slike u SP režimu (uprošćen postupak)

Slika AB_2 se kodira u SP režimu na način prikazan na slici 2.37. Slika B_2 (slika na koju se prekljućujemo) je transformisana i kvantizovana. Predikcija koja je kompenzovana na pokret se formira iz A_1' (slika sa koje se prekljućujemo). Blok označen sa "MC" u dijagramu pokušava da nađe najbolje poklapanje za svaki makroblok MB slike B_2 upotrebljavajući dekodovanu sliku A_1 kao referencu. Predikcija koja je kompenzovana na pokret se transformiše, kvantizuje i oduzima od transformisane i kvantizovane slike B_2, ostatak koji je dobijen na ovaj način se koduje. Dekoder koji poseduje prethodno dekodovanu A_1' sliku može da dekodira SP sliku AB_2 da bi proizveo sliku B_2' (Slika 2.38). Slika A_1' se kompenzuje na pokret (upotrebom vektora pomeraja kodovanih u AB_2 slici), transformiše, kvantizuje i dodaje na dekodavni ostatak, rezulat se dekvantizije i inverzno transformiše kako bi se proizvela slika B_2'.

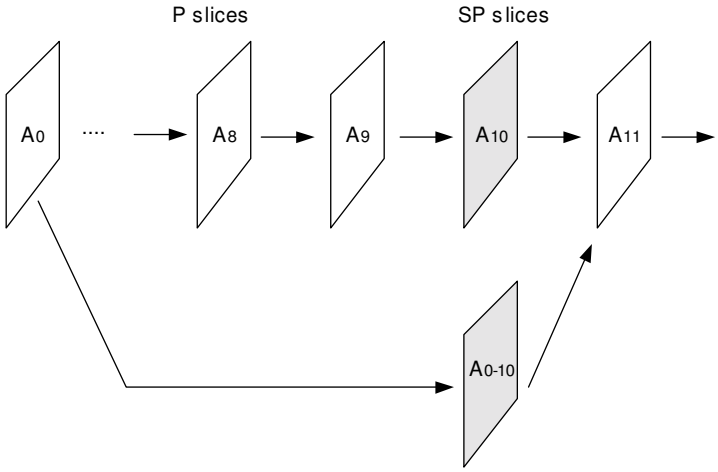


Slika 2.37 Kodovanje slike AB_2 u SP režimu (uprošćen postupak)



Slika 2.38 Dekodovanje slike AB_2 u SP režimu (uprošćen postupak)

Ako su sekvnce A i B različite verzije od iste originalne sekvence, kodovane sa različitim bitskim brzinama, tada predikcija koja je kompenzovana na pokret slike B_2 iz A_1' (SP slika AB_2) treba da bude prilično efikasna. Rezulati pokazuju da je preključuvanje između različitih verzija iste sekvence upotrebom SP slika značajno efikasnije nego umetanje I slika u tačke preključivanja. Druga primena SP slika je da obezbedi slučajno pozicioniranje i funkcionalost kao kod klasičnih video uređaja. Na primer SP slika i preključivačka SP slika su postavljene na poziciju slike 10 (Slika 2.39). Dekoder može brzo da se pozicionira unapred sa slike A_0 diretktno na sliku A_{10} na taj način što prvo dekodira sliku A_0, a zatim preključujuću SP sliku A_{0-10}, koja kada se dekodira proizvodi sliku A_{10} predikcijom iz A_0.



Slika 2.39 Brzo pozicioniranje unapred upotrebom slika kodovanih u SP režimu

Drugi tip slika za preključivanje, su slike kodovane u SI režimu.One su takođe podržane u proširenom profiliu standarda. Ovaj režim se upotrebljava na sličan način kao i SP slike, sa tom razlikom što se predikcija formira upotrebom 4x4 Intra predikcije nad prethodno dekodovanim vrednostima tačaka rekonstruisane slike. Ovaj režim se koristi (na primer) za prebacivanje sa jedne video sekvence na totalno drugačiju video sekvencu (u tom slučaju ne bi bilo efikasno koristiti predikciju sa kompenzacijom pokreta, zato što nema korelacije između ovakve dve sekvence).

2.10 Profili i Nivoi (Profiles and Levels)

Profili i nivoi specificiraju tačke usaglašavanja različitih implemtacija dekodera. Ove tačke usaglašavanja su namenjene da potpomognu mogućnost međusobnog funkcionisanja između raznih primena H.264 standarda koje imaju slične funkcionalne zahteve. *Profil* definiše skup alata za kodirnje ili algoritma koji se mogu upotrebiti za generisanje kodiranog bitskog zapisa (izvornog video materijala) koji je u saglasnosti sa standardom, dok *nivo* postavlja izvesna ograničenja na određene ključne parametre kodiranja.

Svi dekoderi koji su usaglašeni sa specificiranim profilom, moraju da podrže sve karakteristike koje se tim profilom definišu. U H.264 tri profila su definisana: Osnovni (Baseline), Glavni (Main) i Prošireni (Extended).

2.10.1 Osnovni profil (Baseline)

Osnovni profil se upotrebljava u primenama koje zahtevaju malo vreme kašnjenja kao što su video konferencije i bežične (Wireless) komunikacije. Dekoder koji je usaglašen sa ovim profilom mora da podržava:

- slike kodovane u I i P režimu
- filter za uklanjanje izobličenja nastalih usled obrade slike u blokovima
- cik-cak pregrupisavanje koeficijenata
- interpolaciju sa rezolucijom od 1/4 piksela
- kompenzaciju pokreta sa veličinom blokova od 16x16 do 4x4
- entropijsko kodiranje na bazi kodova sa promenljivom dužinom
- proizvoljan poredak kodiranih delova slike (Arbitrary Slice Order - ASO)
- fleksibilno ređanje makroblokova (FMO)
- 4:2:0 progresivni format slike.

2.10.2 Glavni profil (Main)

Glavni profil se upotrebljava u radio-difuziji. Glavnim profilom su podržane sve karakteristike Osnovnog profila (bez ASO i FMO) sa dodatnom podrškom za:

- isprepletani (interlaced) format video materijala i metode za njegovo efikasno kodiranje.
- slike kodovane u B režimu
- CABAC entropijsko kodiranje
- predikcija sa težinskim usrednjavanjem

2.10.3 Prošireni profil (Extended)

Poršireni profilom su podržane sve karakteristike Glavnog profila sa dodatnom podrškom za:

- slike kodovane u SP i SI režimu
- podelu kodiranih podatka delova slike na više particija

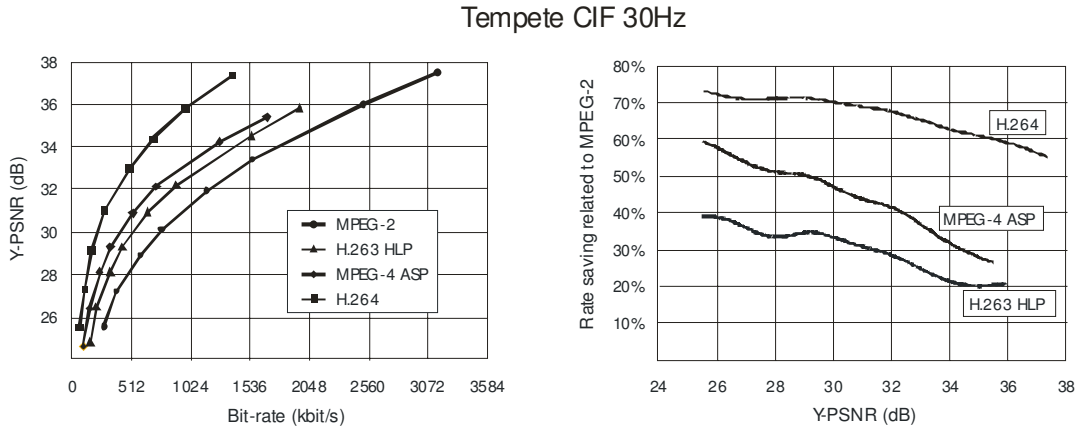
U H.264 isti skup definicija nivoa se upotrebljava sa svim profilima, ali pojednine implementacije podržavaju različite nivoe za svaki podržani profil. Standarnom je definisano 14 nivoa koji specificiraju gornje granice za veličinu slike (u makroblokovima), brzinu dekodiranja (u makroblokovima u sekundi), veličinu memorije za referentne slike, itd.

2.11 Poređenje H.264/AVC efikasnosti kodiranja sa prethodnim standardima.

Performanse kodiranja H.264/AVC se porede sa performansama prethodnim uspešnim standardima za video kodiranje MPEG-2 Visual , H.263++ i MPEG-4 Visual na skupu poznatih QCIF i CIF sekvenci za različitim pokretnim i prostornim detaljima. QCIF sekvence su: *Foreman*, *News*, *Container Ship* i *Tempete*. CIF sekvence su: *Bus*, *Flower Garden*, *Mobile,Calendar* i *Tempete*.. Da bi se omogućilo što objektivnije i pravednije poređenje performansi, dozvoljeno je korišćenje jedinstvenih osobina svakog od kodera koje utiču na efikasnost kodirnja. Tako da je MPEG-2 Visual koder generisao bitski zapis koji je saglasan sa

dobro poznatim MP@ML (*Main Profile at Main Level*) zahtevima, H.263++ koder je koristio metode dozvoljene profilom sa visokim kašnjenjem (*High Latency Profile - HLP*). U slučaju MPEG-4 Visual koder koristi metode naprednog prostog profila (*Advanced Simple Profile - ASP*) sa tačnošću kompenzacija pokreta od četvrtine piksela i omogućenom globalnom kompenzacijom pokreta. Nakon dekodovanja slike upotrebljeni su filteri za uklanjanje efekata nastalih usled obrade slike u blokovima (*deblocking/deringing fliter*).

U slučaju H.264/AVC kodera, koriste se mogućnosti koje su saglasne sa glavnim (*Main*) profilom, uz korišćenje 5 referentnih slika. Kod svih kodera, samo je prva slika svake sekvence kodirana u I režimu, za kojom slede dve slike kodirane u B režimu pri čemu su one umetnute između dve uzastopne slike kodirane u P režimu. U slučaju H.264/AVC kodera, slike kodirane u B režimu se ne koriste kao referentne slike u procesu estimacije i kompenzacije pokreta. Metoda estimacije pokreta sa potpunom pretragom, sa opsegom pretrage od 32 piksela se upotrebljava od strane svih kodera. Bitske brzine se podešavaju upotrebom fiksnog parametra kvantizacije.



Slika 2.40 Bitska brzina-nivo izobličenja (*rate-distortion*) krive i dijagrami uštede bitske brzine

Na levoj strani slike 2.40 prikazana je kriva koja povezuje kvalitet slike posle kodiranja i dekodiranja (y-osa) na datoj bitskoj brzini (x-osa) za sva četiri kodeka, za sekvencu Tempete u CIF rezoluciji. Kvalitet slike se meru pomoću vršnog odnosa signal-šum (*Peak Signal To Noise Ratio - PSNR*), to je mera koliko neki postupak obrade signala (u ovom slučaju kodiranje - dekodiranje) unosi izobličenja, odnosno koliko degradira početni signal, što je taj odnos veći to je manji stepen degradacije. Na desnoj strani slike 2.40, prikazane su uštede u bitskoj brzini u poređenju sa na najlošijim standardom, MPEG-2, u odnosu na dati PSNR luminentne komponente.

Koder	MPEG-4 ASP	H.263 HLP	MPEG-2
H.264/AVC	38.62%	48.80%	64.46%
MPEG-4 ASP	-	16.65%	42.95%
H.263 HLP	-	-	30.61%

Tabela 2.11 Prosečna ušteda bitske brzine u poređenju sa prethodnim standardima za kodiranje

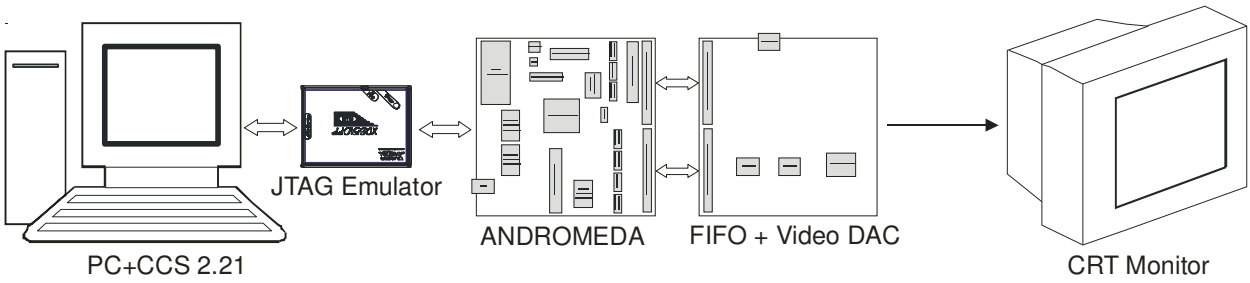
Prosečna ušteda u bitskoj brzini koji obezbeđuje svaki koder u odnosu na sve druge kodere nad celim skupom sekvenci i svim datim bitskim brzinma, su prikazane u tabeli 2.11. Može se videti da H.264 značajno nadmašuje sve druge standarde. Vrlo fleksibilno modelovanje pokreta i vrlo efikasna koneteksno bazirana šema aritmetičkog kodiranja su dva primarna faktora koja omogućavaju superioran odnos između bitske brzine i kvaliteta slike.

3. Opis okruženja za razvoj i testiranje

Za realizaciju zadatog problema u ovom projektu korišćene su sledeće komponente

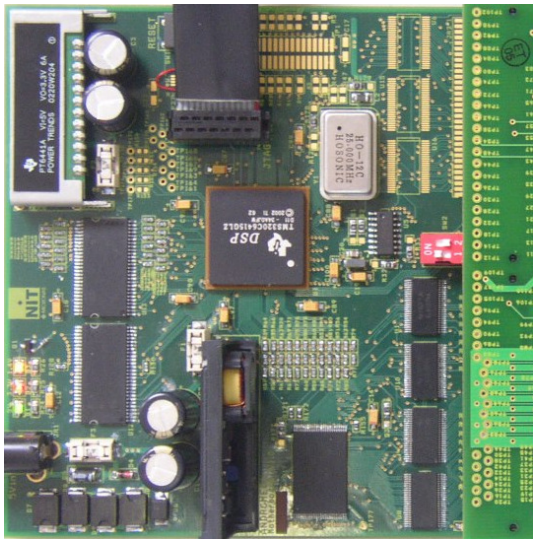
- ANDROMEDA razvojna platforma za TMS320C6415 DSP
- PC kompatibilan računara sa Code Composer Studio 2.21 razvojnim okruženjem
- Spectrum Digital XDS510PP PLUS JTAG emulator
- ANDROMEDA Video Enkoder dodatna ploča.
- prikazna jedinica sa ulazom za kompozitni video signal

Na sledećoj slici prikazan je blok dijagram okruženja za razvoj i testiranje



Slika 3.1 Blok dijagram okruženja za razvoj i testiranje

3.1 Razvojna platforma ANDROMEDA

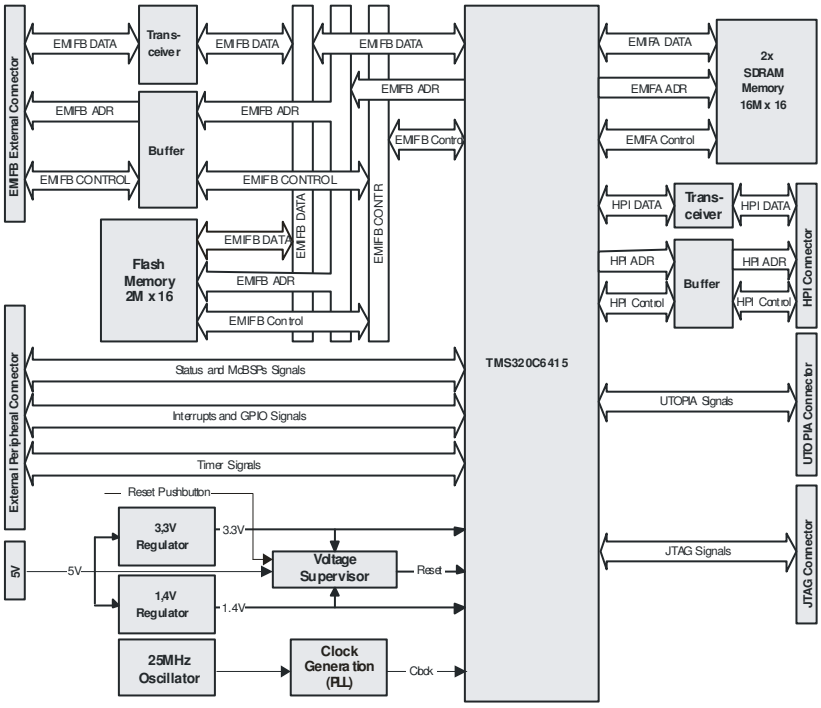


ANDROMEDA je samostalna razvojna platforma koja omogućava korisniku da razvija programsku podršku za TI C64xx DSP procesore. Razvojna platforma je projektovana za mnoštvo različitih primena Struktura fizičke arhitekture je vrlo prilagodljiva sa mogućnošću proširenja.

Platforma se sastoji iz sledećih funkcionalnih jedinca:

- DSP TMS320C6415
- 16M x 32 sinhrone DRAM memorija.
- 2M x 16 Flash Memorija
- Regulatori napajanja za 3,3V i 1,4V
- Kolo za generisanje takta (Oscilator i PLL)
- Kolo za generisanje RESET signala
- konektora za JTAG spregu
- konektora za proširenja.
- konektora za napajanje od 5V

Osnovni funkcionalni blok dijagram ANDROMEDA platforme prikazana je na slici 3.1.



Slika 3.1 Osnovni funkcionalni blok dijagram ANDROMEDA platforme

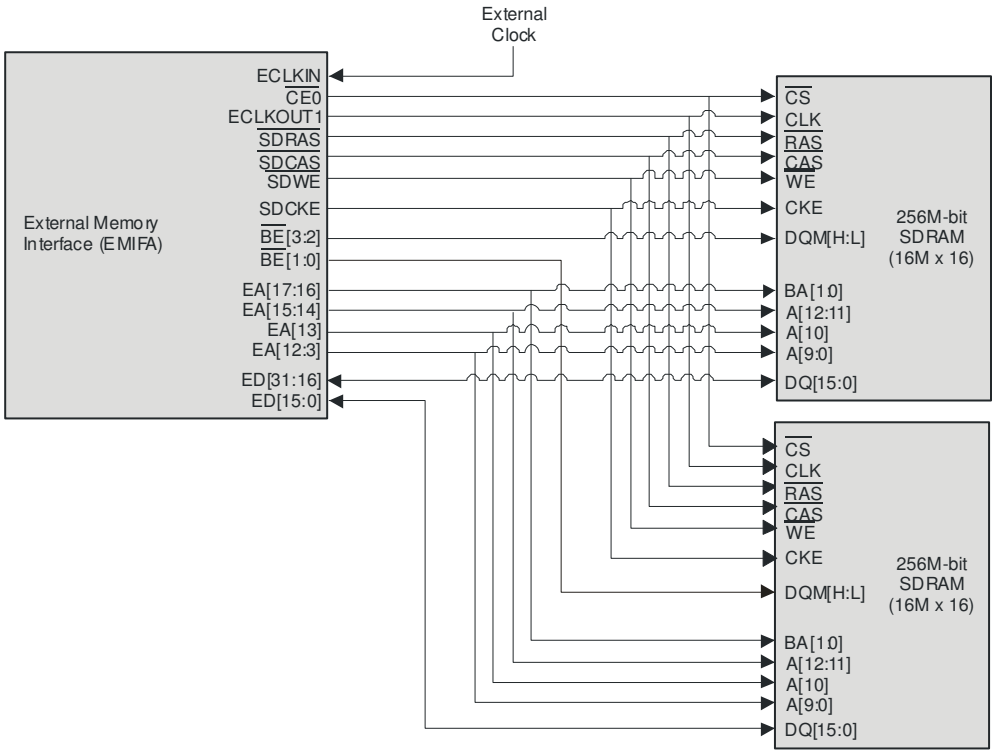
3.1.1 TMS320C6415 DSP[4]

Integrisano kolo sa oznakom TMS320C6415GLZ-6E3 (*Texas Instruments)* je procesor namenjen digitalnoj obradi signala. On je izrađen u 0,12µm CMOS tehnologiji. i smešten u 532-pinsko BGA (*Ball Grid Array*) kućište sa 0,8mm rasterom. U mogućnosti je izvši masimalno 4800 miliona instrukcija u sekundi (4800 MIPS) pri maksimanoj frekvenciji takta od 600MHz (tada je perioda takta 1,67ns). Za njegov rad potrebna su dva napona: 3,3V za napajanje U/I logike i 1,4V za napajanje jezgra. Pri maksimalnom opterećenu potrošnja iznosi oko 1, 1W. Detaljniji opis ovog procesra nalazi se u glavi ARHITEKTURA C64x.

3.1.2 Sinhorni DRAM[5][6]

ANDEOMEDA koristi par standardnih 256Mb SDRAM-ova u CE0 adresnom prostoru EMIFA. Ova dva integrisana kola MT48LC16M16A2-TG75 (*Micron Technology*) su SDRAM memorije kapaciteta 256Mb u organizaciji 4M x 16 x 4 banke, sa maksimalnom radnom učestanosti takta od 133MHz. One formiraju spoljnu radnu memoriju kapaciteta 64MB. SDRAM-ovi su povezani u paralelu sa procesorom TMS320C6415 preko EMIFA sprege (Slika

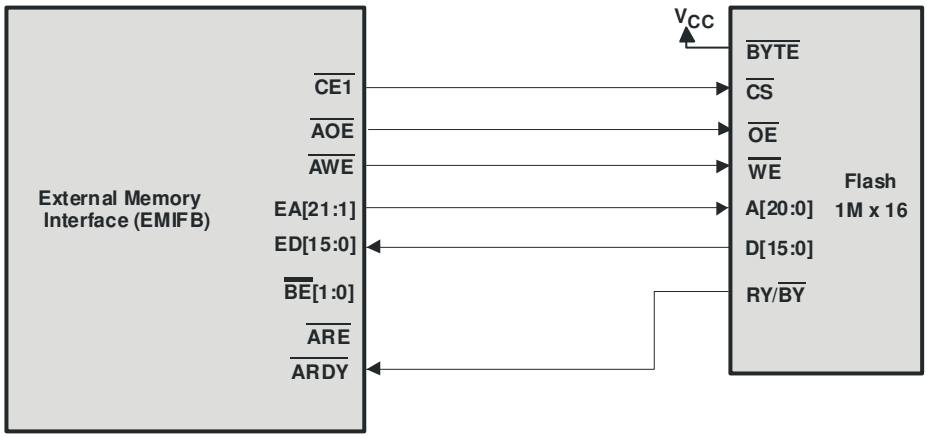
3.2), kako bi formirali sabirnicu podataka širine 32-bita. Memorija koristi takt od 100MHz koji generiše EMIFA. Ugrađeni SDRAM kontroler se pokreće konfigurisanjem EMIFA u programskoj podršci, na osnovu poznatih vremenskih parametra SDRAM-a.



Slika 3.2 Sprega C64x EMIFA i dva 256Mb SDRAM-a širine 16-bita

3.1.3 Flash memorija[7][8]

Flash memorija Am29DL322DB-90 (*Advanced Micro Devices*) kapaciteta 32Mb je povezana na EMIFB 16-spregu uz korošćenje ARDY signalai nalazi se u EMIFB CE1 adresnom prostoru (Slika 3.3). Ova memorija je kapaciteta 2M x 16, ali zbog ograničenja koja nemeće broj adresnih linija EMIFB sprege u adresnom prostoru dostupna je polovina kapaciteta t.j. 1M x 16. Flash memorija je tip memorije koji ne gubi sadržaj po nestanku napajanja. Kada se čita one se ponaša kao obična asinhrona ROM memorija. Flash može de se briše u velikim blokovima, koji se često nazivaju sektorima ili stranicama. Kada se blok izbriše u svaku se lokaciju se može jedanput upisati upotrebom specijalne sekvence komandi. Flash memorija ima vreme pristupa od 90ns, te se na osnovu toga i drugih vremenskih parametara podešava EMIFB CE1 asinhroni memorijski kontroler. Ova memorija se koristi za skladištenje izvršnog koda, koji treba da se učita u DSP pri njegovoj inicijalizaciji.



Slika 3.3 EMIFB 16-bitni Flash sprega sa ARDY

3.1.4 Regulatori napajanja za 3.3V i 1.4V[9][10]

Ova dva regulatora su moduli iz serije integrisanih prekidačkih regulatora (*Integrated Switching Regulator)* visokih performansi. Oba poseduju mogućnost podešavanja nivoa izlaznih napona. Modul PT6441 za ulazni napon od 5V daje na svom izlazu napon od 3.3V sa maksimalnom strujom od 6A. Modul PT6527 je podešen tako da za ulazni napom od 5V daje na svom izlazu 1.4V sa maksimalom strujom od 8A. Napon od 3.3V koristi se za napajanje U/I logike DSP-a i svih ostalih integrisanih kola, dok se napon od 1.4V koristi samo za napajanje jezgra DSP-a.

3.1.5 Kolo za generisanje takta[11][12]

Kolo za generisanje takta se sastoji iz kvarcnog osciltora učestanosti 25 MHz i IDT2308-4DC množača učestonasti. Sa izlaza oscilatora signal učestonsti 25MHz ulazi u množač koji duplira frekvenciju signala. Takt učestanosti 50MHz koji se dobija na izlazu množača se vodi na CLKIN priključak DSP. Ovaj prilključak predstavlja ulaz u DSP-ov unutrašnji PLL. On može da propusti da radi u režimu x1,x6 i x12. Režim se bira u zavisnosti od nivoa na prilkjučcima CLKMODE0 i CLKMODE1. Za pouzdan rad PLL potrebno je preko PLLV priključaka dovesti filtrirano napajanje za PLL, a to se postiže pomoću EMI filtra.

3.1.6 Kolo za generisanje RESET signala[13] [14]

Da bi se obezbedio pouzdan rad DSP-a nepohodno ga je pravilno inicijalizovati. Za ovo je potrebno obezbedti signal odogovarajućih karakteristika na RESET ulaznom priljučku DSP-a. Ovu ulogu obavlja TPS3307-33 integrisano kolo zajedno sa namenskim RESET tasterom. Ovo kolo pored mogućnosti generisanja RESET signala na osnovu stanja tastera, poseduje mogućnost generisanja na osnovu nivoa napona napajnja. Ako iz bilo kog razloga bilo koji od napona t.j. 5V, 3.3V i 1.4V odstupi od svoje referentne vrednosti, automatski se generiše RESET signal. Ovim se DSP štiti od ulaska u nedefinisano stanje koje je često karakterisano prekomernom potrošnjom struje, a samim tim i zagrevanjem.

3.1.7 Konektor za JTAG spregu[15]

JTAG sprega na DSP-u je saglasna sa standardom IEEE 1149.1. Priključci JTAG sprege DSP-a se vode direktno na 14-pinski konektor. Ovo je standardni konektor (po rasporedu i tipu signal i mehaničkim karakteristikama) koja se upotrebljava od strane JTAG emulatora za povezivanje sa Texas Instruments DSP-ovima. U kombinaciji sa Code Compsoer Studio programskim paketom, kod generisan prevodiocem može se napuniti u DSP, a potom testirati. Imena i raspored signal na ovom konektoru dati su u sledećoj tabeli.

Pin	Signal
1	TMS
2	TRST-
3	TDI
4	GND
5	PD
6	no pin
7	TDO
8	GND
9	TCK-RET
10	GND
11	TCK
12	GND
13	EMU0
14	EMU1

Tabela 3.1 JTAG sprega

3.1.8 Konektori za proširenja

ANDROMEDA platforma je opštenamenska platforma koja ne poseduje A/D i D/A pretvarače, te stoga obezbeđuje dva konektora koji se upotrebljavaju za dodavanje dodatnih mogućnosti. Ovo omogućava da korisnik upotrebi ovu platformu u različitim primenama. Jedan od ova dva konektora je namenjen za memoriska proširenja dok je drugi namenjen za periferije.

Memorijski konektor obezbeđuje pristup EMIFB signalima radi sprezanja sa memorijama i memorijski mapiranim uređajima. Periferijski konektor stavlja na raspolaganje signale DSP periferija kao što su McBSP-ovi, tajmeri i taktove. Oba konektora obezbeđuju napajanje i masu

Signali na memorijskom konektoru su baferisani tako da dodatna elektronska kola ne mogu da utiču na DSP. Upotrebom Texas Instruments nisko naponskih i tolerantih na 5V bafera, elektronska kola koja koriste signale na ovom konektoru mogu da budu bilo 5V ili 3.3V napajani uređaji. Ova dva konektora su 80-pinski 0.050" x 0.050" konektori koje proizvodi firma ERNI. Oni su dizajnirani za rad na visokim učestanostima zato što imaju malo vreme kašnjenja, kapacitivnost i preslušavanje.

3.1.8.1 EMIFB konektor za proširenje (JH1)

Pin	Signal	Tip	Opis	Pin	Signal	Tip	Opis
1	5V	Vcc	5V voltage supply pin	41	5V	Vcc	5V voltage supply pin
2	GND	Vss	System ground	42	GND	Vss	System ground
3	3.3V	Vcc	3.3V voltage supply pin	43	3.3V	Vcc	3.3V voltage supply pin
4	GND	Vss	System ground	44	GND	Vss	System ground
5	BEA1	O	EMIFB address pin 1	45	BEA2	O	EMIFB address pin 2
6	BEA3	O	EMIFB address pin 3	46	BEA4	O	EMIFB address pin 4
7	BEA5	O	EMIFB address pin 5	47	BEA6	O	EMIFB address pin 6
8	BEA7	O	EMIFB address pin 7	48	BEA8	O	EMIFB address pin 8
9	GND	Vss	System ground	49	GND	Vss	System ground
10	BEA9	O	EMIFB address pin 9	50	BEA10	O	EMIFB address pin 10
11	BEA11	O	EMIFB address pin 11	51	BEA12	O	EMIFB address pin 12
12	BEA13	O	EMIFB address pin 13	52	BEA14	O	EMIFB address pin 14
13	BEA15	O	EMIFB address pin 15	53	BEA16	O	EMIFB address pin 16
14	GND	Vss	System ground	54	GND	Vss	System ground
15	BEA17	O	EMIFB address pin 17	55	BEA18	O	EMIFB address pin 18
16	BEA19	O	EMIFB address pin 19	56	BEA20	O	EMIFB address pin 20
17	D0	I/O	EMIFB data pin 0	57	D1	I/O	EMIFB data pin 1
18	D2	I/O	EMIFB data pin 2	58	D3	I/O	EMIFB data pin 3
19	GND	Vss	System ground	59	GND	Vss	System ground
20	D4	I/O	EMIFB data pin 4	60	D5	I/O	EMIFB data pin 5
21	D6	I/O	EMIFB data pin 6	61	D7	I/O	EMIFB data pin 7
22	D8	I/O	EMIFB data pin 8	62	D9	I/O	EMIFB data pin 9
23	D10	I/O	EMIFB data pin 10	63	D11	I/O	EMIFB data pin 11
24	GND	Vss	System ground	64	GND	Vss	System ground
25	D12	I/O	EMIFB data pin 12	65	D13	I/O	EMIFB data pin 13
26	D14	I/O	EMIFB data pin 14	66	D15	I/O	EMIFB data pin 15
27	nBBE0	O	EMIFB byte enable 0	67	nBBE1	O	EMIFB byte enable 1
28	nBCE0	O	EMIFB chip enable 0	68	nBCE1	O	EMIFB chip enable 1
29	GND	Vss	System ground	69	GND	Vss	System ground
30	nBCE2	O	EMIFB chip enable 2	70	nBCE3	O	EMIFB chip enable 3
31	nBPDT	O	EMIFB peripheral data transfer	71	nBHOLDA	O	EMIFB hold-request-acknowledge to the host
32	nBHOLD	O	EMIFB hold request from the host	72	nBBUSREQ	O	EMIFB bus request output
33	BARDY	I	EMIFB asynchronous ready	73	NC	-	Not connencted
34	GND	Vss	System ground	74	GND	Vss	System ground
35	nRESET	O	RESET signal	75	nBARE	O	EMIFB async read enable

36	nBAOE	O	EMIFB async output enable	76	nBAWE	O	EMIFB async write enable
37	BECLKIN	I	EMIFB external clock	77	BSOE3	O	EMIFB sync output enable
38	BECLKOUT1	O	EMIFB clock 1 output	78	BECLKOUT2	O	EMIFB clock 2 output
39	GND	Vss	System ground	79	GND	Vss	System ground
40	5V	Vcc	5V voltage supply pin	80	3.3V	Vcc	3.3V voltage supply pin

Tabela 3.2 Opis i raspored signala ne EMIFB konektoru

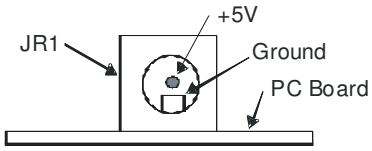
3.1.8.2 Periferijski konektor za proširenje (JH2)

Pin	**Signal**	**Tip**	**Opis**	**Pin**	**Signal**	**Tip**	**Opis**
1	5V	Vcc	5V voltage supply pin	41	5V	Vcc	5V voltage supply pin
2	GND	Vss	System ground	42	GND	Vss	System ground
3	3.3V	Vcc	3.3V voltage supply pin	43	3.3V	Vcc	3.3V voltage supply pin
4	GND	Vss	System ground	44	GND	Vss	System ground
5	CLKX2	I/O	McBSP2 transmit clock	45	CLKS2	I	McBSP2 clock source
6	DR2	I	McBSP2 receive data	46	FSR2	I/O	McBSP2 receive frame sync
7	GND	Vss	System ground	47	GND	Vss	System ground
8	CLKR2	I/O	McBSP2 receive clock	48	CLKR1	I/O	McBSP1 receive clock
9	FSX2	I/O	McBSP2 transmit frame sync	49	DR1	O	McBSP1 receive data
10	GND	Vss	System ground	50	GND	Vss	System ground
11	DX2	O	McBSP2 transmit data	51	CLKX1	I/O	McBSP1 transmit clock
12	CLKS1	I	McBSP1 clock source	52	DX1	O	McBSP1 transmit data
13	GND	Vss	System ground	53	GND	Vss	System ground
14	FSX1	I/O	McBSP1 transmit frame sync	54	FSR1	I/O	McBSP1 receive frame sync
15	CLKR0	I/O	McBSP0 receive clock	55	CLKS0	I	McBSP0 clock source
16	GND	Vss	System ground	56	GND	Vss	System ground
17	DR0	I	McBSP0 receive data	57	CLKX0	I/O	McBSP0 transmit clock
18	FSR0	I/O	McBSP0 receive frame sync	58	FSX0	I/O	McBSP0 transmit frame sync
19	GND	Vss	System ground	59	GND	Vss	System ground
20	TOUT2	O	Timer 2 output	60	DX0	O	McBSP0 transmit data
21	TOUT1	O	Timer 1 output	61	TINP2	I	Timer 2 input
22	GND	Vss	System ground	62	GND	Vss	System ground
23	TOUT0	O	Timer 0 output	63	TINP1	I	Timer 1 input
24	GP0	I/O	GPIO pin 0	64	TINP0	I	Timer 0 input
25	GND	Vss	System ground	65	GND	Vss	System ground
26	CLKOUT6/GP2	I/O	CPUCLK/6 or GPIO pin 2	66	CLKOUT4/GP1	I/O	CPUCLK/4 or GPIO pin 1
27	GP4/EXTINT4	I/O	GPIO pin 4 or Extrenal interrupt 4	67	GP3	I/O	GPIO pin 3
28	GND	Vss	System ground	68	GND	Vss	System ground
29	GP6/EXTINT6	I/O	GPIO pin 4 or Extrenal interrupt 4	69	GP5/EXTINT5	I/O	GPIO pin 4 or Extrenal interrupt 4
30	GP10	I/O	GPIO pin 10	70	GP7/EXTINT7	I/O	GPIO pin 4 or Extrenal interrupt 4
31	GND	Vss	System ground	71	GND	Vss	System ground
32	GP12	I/O	GPIO pin 12	72	GP9	I/O	GPIO pin 9
33	GP14	I/O	GPIO pin 14	73	GP11	I/O	GPIO pin 11
34	GND	Vss	System ground	74	GND	Vss	System ground
35	NMI	I	Non-maskable interrupt	75	GP13	I/O	GPIO pin 13
36	NC	-	Not connencted	76	GP15	I/O	GPIO pin 15
37	GND	Vss	System ground	77	NC	-	Not connencted
38	5V	Vcc	5V voltage supply pin	78	5V	Vcc	5V voltage supply pin
39	GND	Vss	System ground	79	GND	Vss	System ground
40	5V	Vcc	5V voltage supply pin	80	3.3V	Vcc	3.3V voltage supply pin

Tabela 3.3 Opis i raspored signala na perfrijskom konektoru

3.1.9 Konektor za napajanje

Napajanje od 5V se dovodi na ANDROMEDA platformu preko konektora JR1. Konektor ima spoljašnji prečnik od 5.5mm, i unutrašnji prečnik od 2.5mm. Konektor JR1 je prikazan na sledećoj slici.

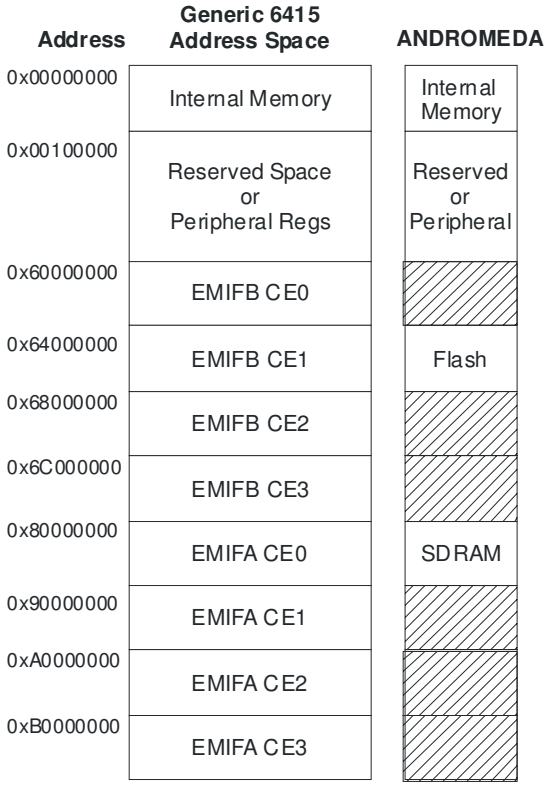


Slika 3.4 ANDROMEDA konektor za napajanje

3.1.10 Memorijska mapa

C64xx DSP familija ima veliki bajt adresibilni adresni prostor. Programski kod i podaci mogu biti smešteni bilo gde u unificiranom adresnom prostoru. Adrese su uvek širine 32 bita. Memorijska mapa prikazuje adresni prostor 6415 procesora na levoj strani i kako se svaki od regiona upotrebljava na desnoj strani. Svaka spoljašnja memorijska sprega - EMIF ima 4 odvojena adresabilna regiona koji se nazivaju CE0-CE3 prostori (*chip enable spaces*). SDRAM okupira CE0 od EMIFA, dok je Flash mapiran u CE1 od EMIFB. Proširenja mogu da koriste CE0, CE2 i CE3 koji pripadaju EMIFB sprezi

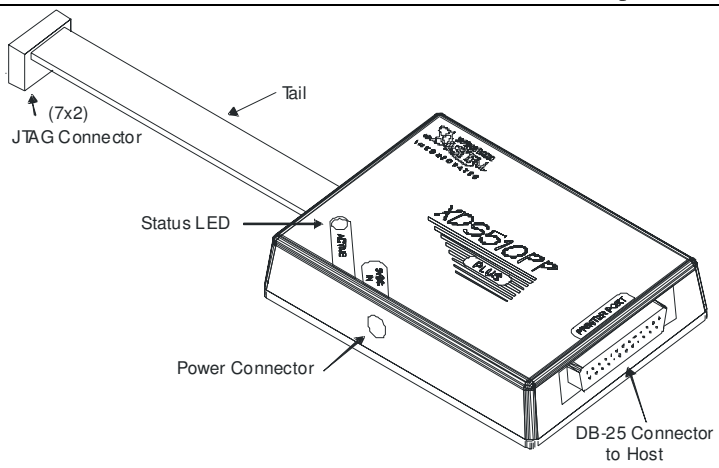
.



Slika 3.5 ANDROMEDA memorijska mapa

3.2 Spectrum Digital XDS510PP PLUS JTAG emulator[16]

XDS510PP PLUS JTAG Emulator je projektovan za upotrebu sa procesorima za digitalnu obradu signala i mikroprocesorima koji rade sa naponskim nivoima od +3.3 V do +5 V. Ovaj emulator može biti napajan preko PD priključka na JTAG konektoru ili preko spoljašnjog naponskog izvora koji se dobija uz emulator. XDS510PP PLUS je projektovan da bude kompatibilan sa Texas Instruments XDS510 emulatorom i da radi sa razvojnim okruženjem koje obezbeđuje Texas Instruments. Na slici 3. 4 prikazan je emulator

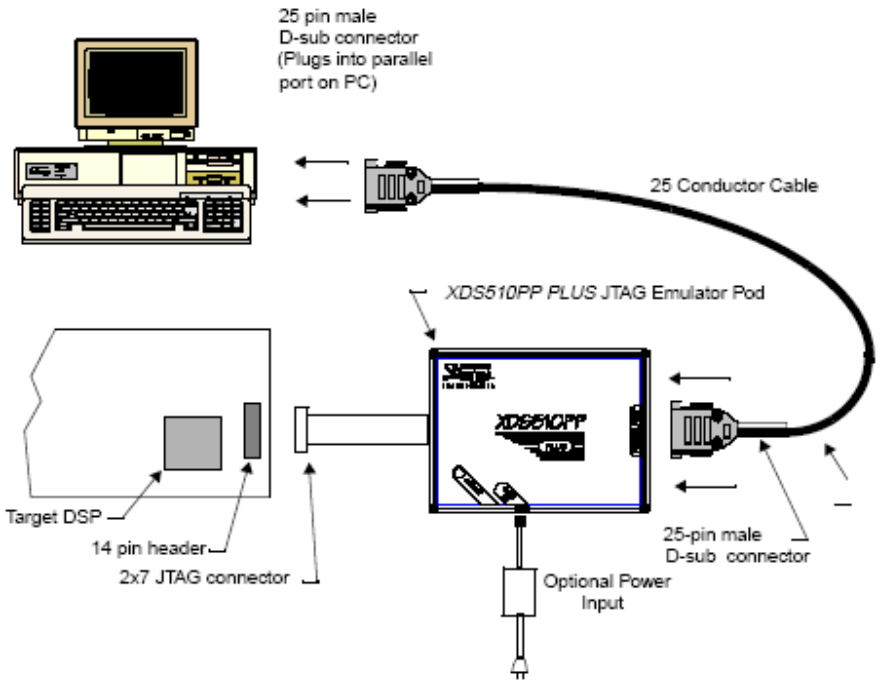


Slika 3.4 XDS510PP PLUS JTAG Emulator

XDS510PP PLUS JTAG ima sledeće karakteristike:

- Podržava Texas Instruments procesore za digitalnu obradu signala i mikrokontrolere sa JTAG spregom (IEEE 1149.1)
- Kompatibilnost sa Texas Instruments XDS510 emulatorom
- Podržava standardnu paralelnu komunikacionu spregu sa PC računrom (SPP8, EPP i ECP). Nisu potrebni nikakvi dodatni adapteri.
- Poržava JTAG sprege sa naponskim nivoima od 3.3V do 5V
- Može da se napaja sa platforme na koju je priključen
- Kompatibilan je sa Texas Instruments Code Composer Studio razvojnim alatom.

Na slici 3.5 prikazuje povezivanje XDS510PP PLUS u tipičnoj postavci sistema sa PC računarom i ciljnom DSP platformom.



Slika 3.5 Povezivanje XDS510PP PLUS emulatora

Za povezivanje sa ciljnom DSP platformom emulator poseduje 14-pinski konektor čiji je raspored prikazan na slici 3.6, dok je opis signala dat u tabeli 3.4

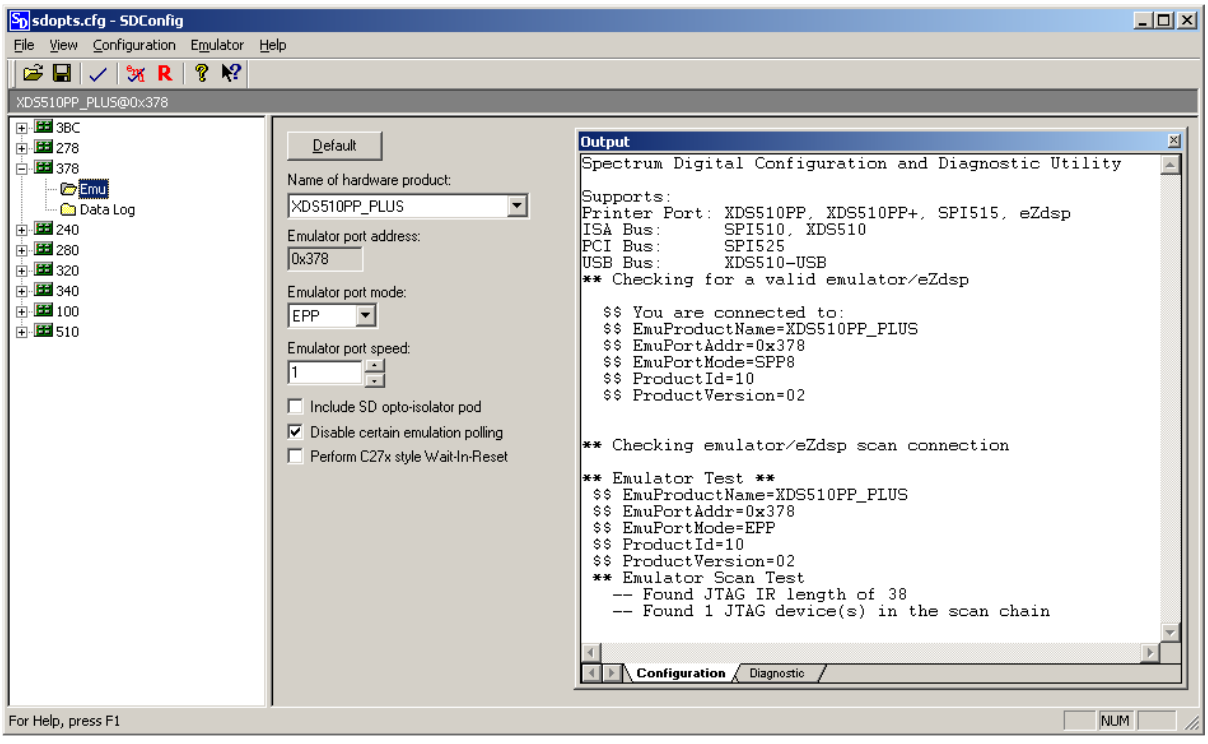
TMS	1	2	TRST-
TDI	3	4	GND
PD	5	6	**no pin (key)**
TDO	7	8	GND
TCK-RET	9	10	GND
TCK	11	12	GND
EMU0	13	14	EMU1

Slika 3.6 Raspored signala na 14-pinskom konektoru

Signal	Opis	Tip	
		Emulator	Platforma
TMS	JTAG test izbor režima.	Izlaz	Ulaz
TDI	JTAG test ulaz podataka.	Izlaz	Ulaz
TDO	JTAG test izlaz podataka.	Ulaz	Izlaz
TCK	JTAG test takt. TCK je izvor takta od 10MHz od strane emulatora.	Izlaz	Ulaz
TRST-	JTAG test reset.	Izlaz	Ulaz
EMUO	Emulacioni priključak 0.	U/I	U/I
EMU1	Emulacioni priključak 1.	U/I	U/I
PD	Detekcija prisustva. Indikator da je kabel emulatora povezan i da je ciljna platforma pod napajanjem. PD treb da se poveže na naponski nivo koji odgovara nivou kojom se napaja U/I logika procesora.	Ulaz	Izlaz
TCK_RET	JTAG test povratni takt.	Ulaz	Izlaz

Tabela 3.4 Opis signala na 14-pinskom konektoru

Da bi se XDS510PP PLUS mogao koristiti potrebno je na PC računar postaviti odgovarajuću programsku podršku. Programska podrška se sastoji od programa *SDConfig* i veznih programa za Code Composer Studio. Program *SDConfig* služi za postavljanje parametra paralelne komunikacije, inicijalizaciju, testiranje i dijagnostiku emulatora i JTAG sprege ciljne platforme. Izgled ovog programa prikazan je na slici 3.7.



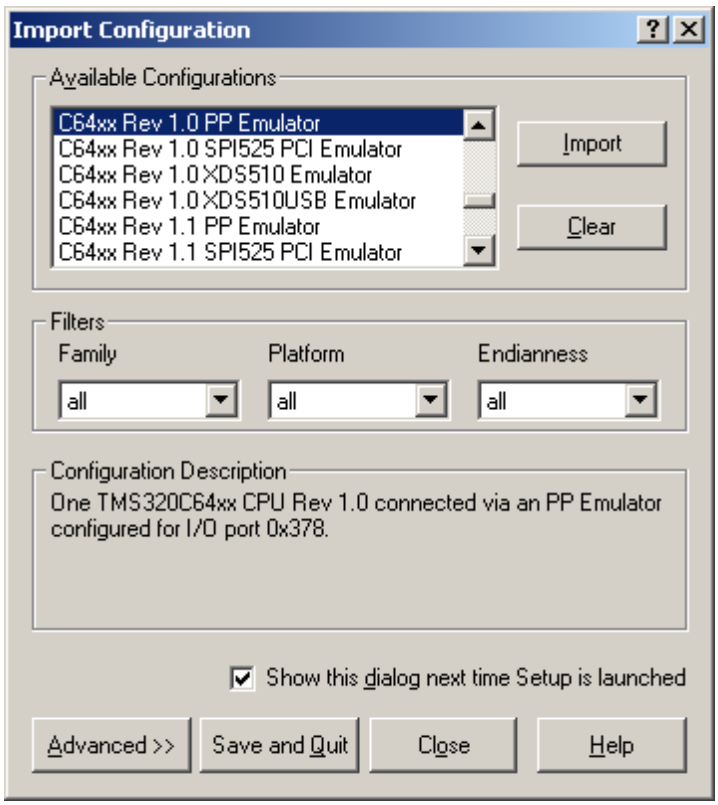
Slika 3.7 Program SDConfig

3.3 Integrisano razvojno okruženje Code Composer Studio[20]

Za prevođenje, analiziranje i testiranje programske podrške realizovane u okviru ovog projekta korišćen je programski paket Code Composer Studio 2.21 (U daljem tekstu CCS). Ovaj programski paket predstavlja integrisano razvojno okruženje (*Integrated Development Enviroment*) namenjeno Texas Instruments DSP procesorima iz familije C6000. Pomoću ovog okruženja moguće je realizovati sve faze razvoja programske podrške, počev od pisanja izvornog koda u asembleru ili C/C++-u, preko prevođenja i povezivanja programa do analiziranja i testiranja u realnom vremenu.

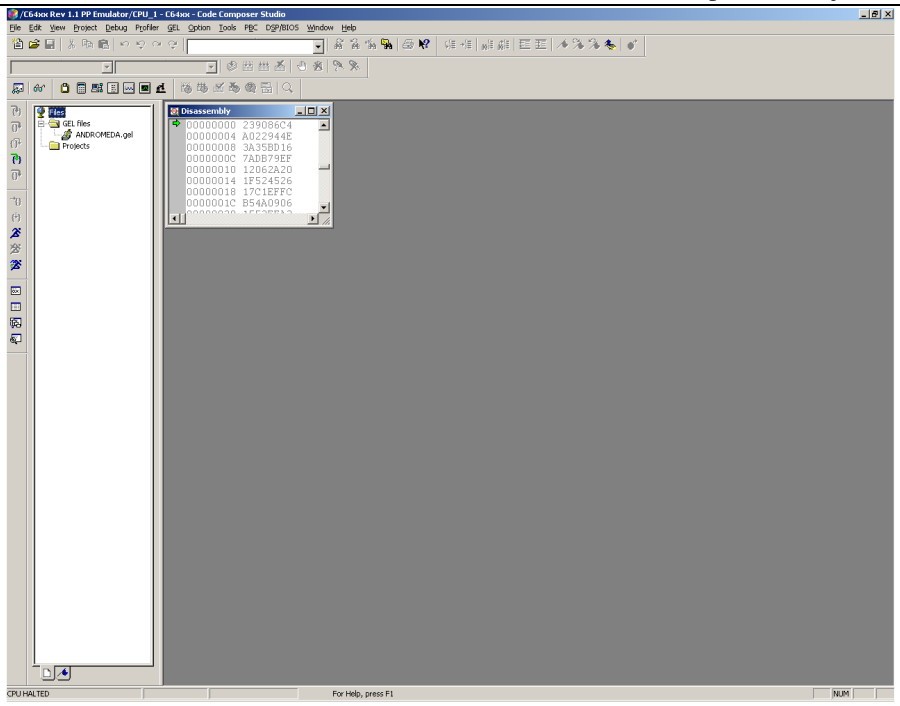
3.3.1 Konfigurisanje CCS-a

Pre korišćenja CCS-a, potrebno je izvršiti njegovo konfigurisanje programom CCS Setup. Po startovanju ovog programa, otvara se prozor koji nudi izbor konfiguracije sa kojom će se raditi (Slika 3.8).



Slika 3.8: Prozor za izbor konfiguracije

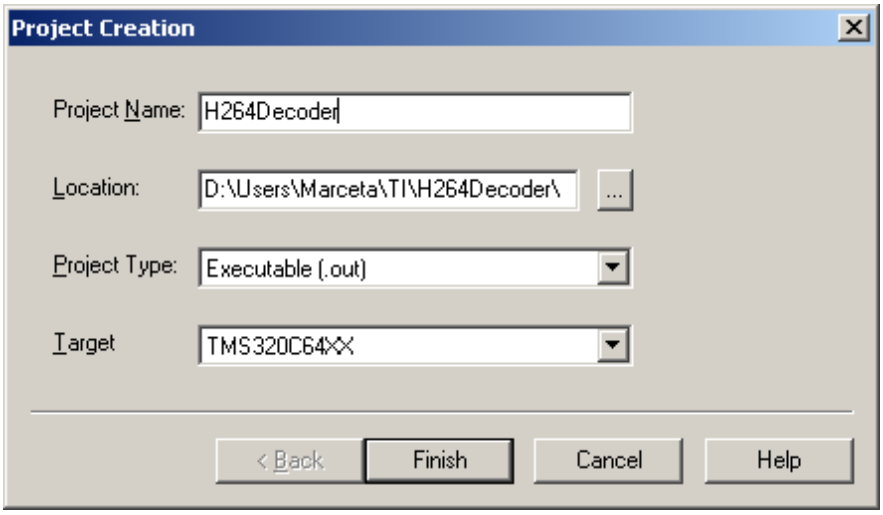
Pre izbora konfiguracije, potrebno je obrisati prethodnu konfiguraciju klikom na dugme *Clear*, a zatim potvrditi brisanje klikom na dugme *Yes*. Nakon toga treba odabrati željenu konfiguraciju. U ovom slučaju to je konfiguracija C64xx Rev 1.1 PP Emulator namenjena DSP familiji TMS320C64xx (Revizija 1.1). Po izboru konfiguracije potrebno je kliknuti na dugme *Import*, a zatim na dugme *Save and Quit*, čime se snima izabrana konfiguracija i izlazi iz programa CCS Setup. Posle toga se klikom na dugme *Yes* automatski ulazi u CCS. Ekran CCS-a sastoji se iz palete alatki i radne površine na kojoj se nalazi prozor sa spiskom otvorenih projektat i spiskom datoteka uključenih u svaki projekat (Slika 3.9).



Slika 3.9: Radni ekran Code Composer-a

3.3.2 Formiranje projekta u CCS-u

Prvi korak u razvoju programske podrške je kreiranje novog projekta, izborom opcije *New* iz *Project* menija. Po izboru ove opcije pojavljuje se čarobnjak za kreiranje projekta *(The Project Creation wizard*), Slika 3.10. Naziv projekta koji sadrži programske podrške video dekodera je H264Decoder. Tip projekta je *Executable* (.out), a odredišna platforma je TMS320C64xx.



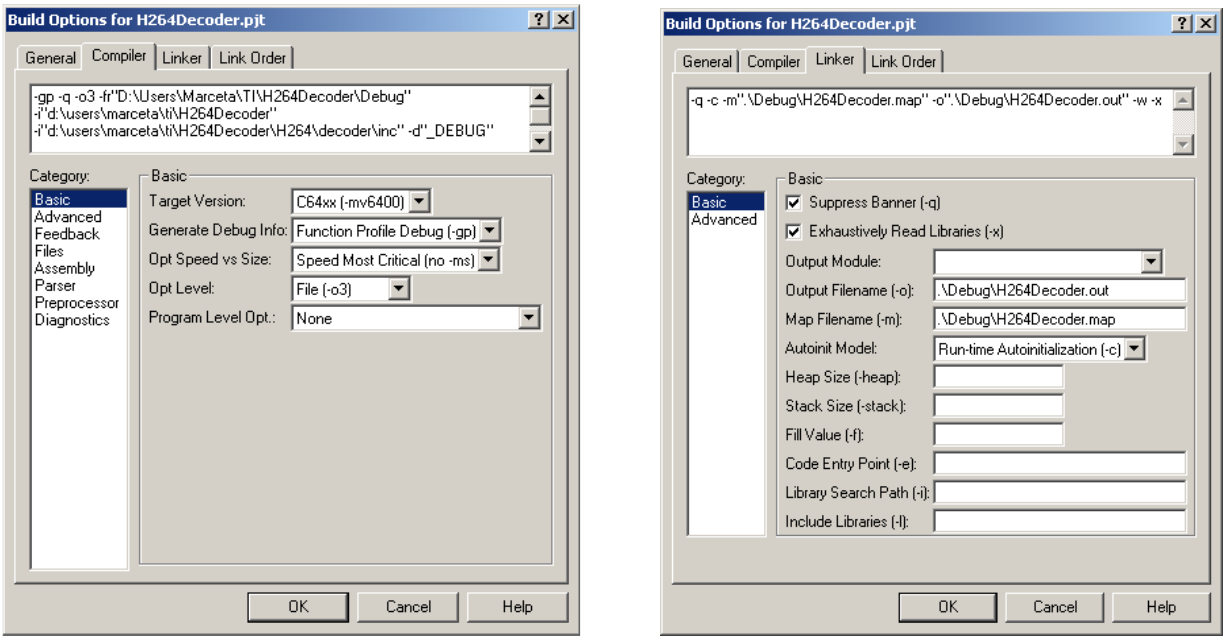
Slika 3.10: Prozor čarobnjaka za kreiranje projekta

Posle kreiranja projekta, potrebno je u njega uključiti datoteke sa izvornim kodom, izborom opcije *Add Files to Project* iz *Project* menija. Potrebno je uključiti samo datoteke sa izvornim kodom u C/C++-u i asembleru (datoteke sa ekstenzijama .c, .cpp i .s62), dok će datoteke sa zaglavljima modula u C/C++-u odnosno asembleru (datoteke sa ekstenzijama .h i .h62) biti automatski uključene u projekat prilikom prevođenja programa.

3.3.3 Prevođenje i povezivanje programa

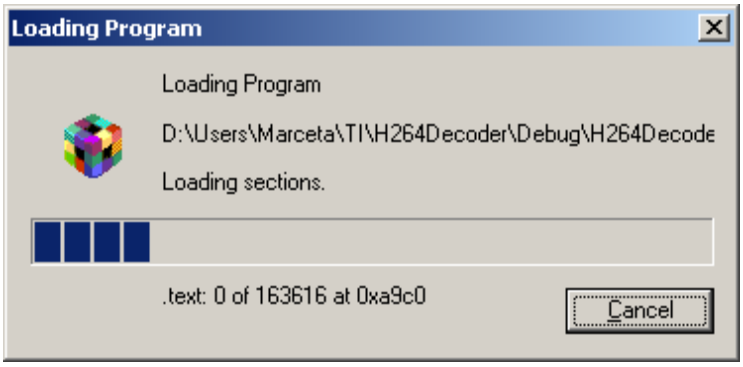
Sledeći korak u razvoju programske podrške je njeno prevođenje i povezivanje. Prevođenje programa u CCS-u vrši se izborom opcije *Rebuild All* iz *Project* menija. Ukoliko se posle prevođenja izvrši izmena neke od datoteka uključenih u projekat, nije potrebno ponovo prevoditi ceo program, već je dovoljno prevesti samo tu datoteku i povezati je sa programom izborom

opcije *Incremental Build* iz *Project* menija. Pre prevođenja programa, potrebno je podesiti parametre prevodioca i povezioca, izborom opcije *Build Options* iz *Project* menija. Parametre koje je potrebno podesiti prikazuju Slika 3.11.



Slika 3.11: Osnovni parametri prevodioca i povezioca

Ukoliko nije bilo grešaka pri prevođenju, prevodilac će u Debug poddirektorijumu direktorijuma u kome se nalazi projekat kreirati datoteku sa ekstenzijom .out u kojoj se nalazi relokatibilni izvršni kod prevedenog programa. Ovakav izvršni kod se prebacuje u memoriju DSP procesora opcijom *Load Program* iz *File* menija (Slika.3.12).



Slika 3.12 Prebacivanje izvršnog koda u memoriju DSP-a

3.3.4 DSP/BIOS

DSP/BIOS je skalabilno jezgro operativnog sistema namenjeno radu u realnom vremenu. Ovo jezgro je namenjeno aplikacijama koje zahtevaju raspoređivanje procesa u realnom vremenu, pristup resursima PC računara, kao i instrumentaciju u realnom vremenu. DSP/BIOS podržava *preemptive multi-threading*, apstrakciju hardvera i analizu u realnom vremenu. DSP/BIOS i njegov alat za analizu u okviru programskog paketa Code Composer Studio projektovani su tako da minimizuju zahteve za resursima procesora. Ovo je postignuto na sledeće načine:

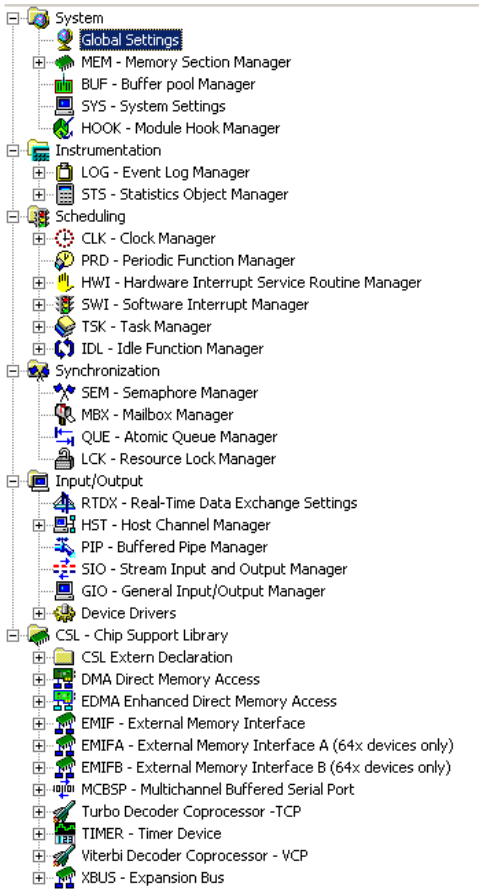
- Svi DSP/BIOS objekti mogu se kreirati statički, pomoću alata za konfiguraciju DSP/BIOS-a i kasnije povezati sa izvršnim kodom programa. Ovim se optimizuje veličina koda, kao i veličina internih struktura podataka.
- Svi instrumentacioni podaci se formatiraju na PC računaru, čime se rasterećuje DSP procesor.

- Programski interfejs (API) DSP/BIOS-a je modularan, što omogućava da se sa izvršnim programom povezuju samo oni moduli koje program koristi.
- Biblioteka sa funkcijama DSP/BIOS-a je većim delom napisana u asembleru i optimizovana tako da zahteva najmanji mogući broj instrukcijskih ciklusa procesora.
- Komunikacija između DSP procesora i alata za analizu na PC računaru realizovana je u okviru procesa najnižeg prioriteta, tako da ne utiče na rad ostalih procesa.

Pored ovoga, DSP/BIOS pruža veliki broj mogućnosti pri razvoju programa:

- Programi mogu da koriste DSP/BIOS objekte kreirane statički pomoću alata za konfiguraciju, ali po potrebi mogu i da dinamički kreiraju i uništavaju druge DSP/BIOS objekte.
- Podržano je više vrsta procesa, u zavisnosti od njihove namene: funkcije za obradu hardverskih i softverskih prekida, zadaci (*tasks*), besposlene (*idle*) funkcije i periodične funkcije.
- Podržane su strukture koje omogućavaju komunikaciju i sinhronizaciju između procesa, kao što su semafori, poštanski sandučići i zaključavanje resursa.
- Podržana su dva modela ulazno-izlaznih uređaja: model niskog nivoa koji se koristi kada postoji jedan proces koji čita podatke iz bafera i jedan proces koji ih upisuje, i model visokog nivoa koji se koristi u složenijim situacijama i koji podržava rukovaoce ulazno-izlaznim uređajima.

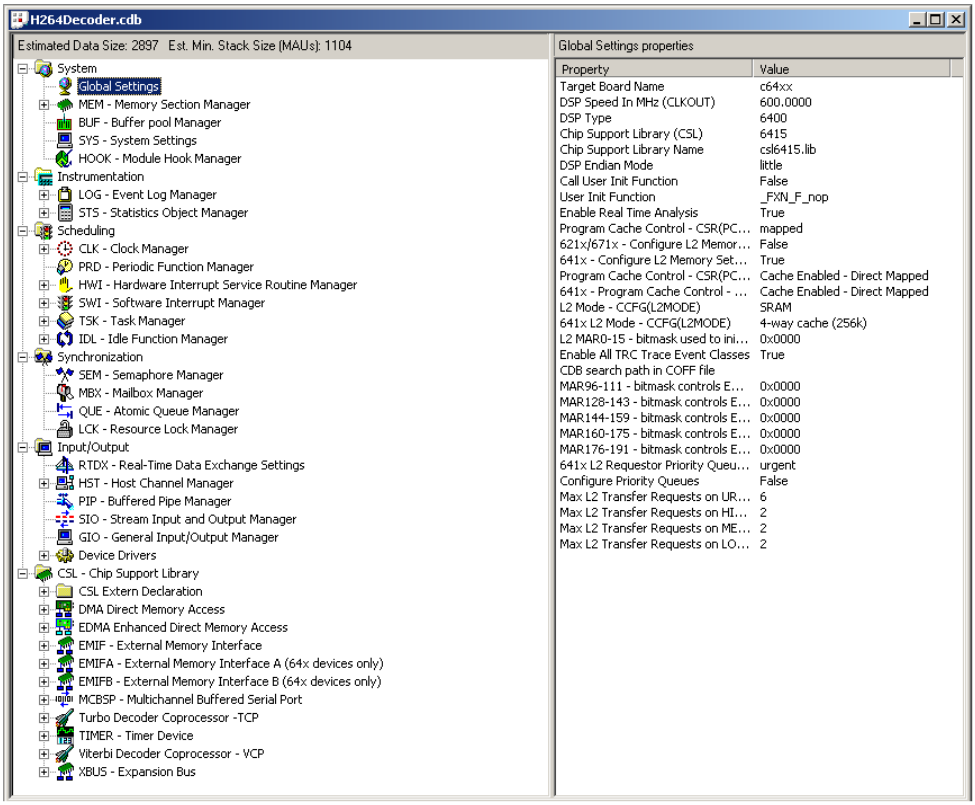
Najlakši način za kreiranje i korišćenje DSP/BIOS-a i njegovih objekata (modula) je kreiranje konfiguracione datoteke (datoteka sa nastavkom .CDB) pomoću alata za konfigurisanje koji se nalazi u okviru Code Composer Studio programskog paketa. Ovaj alat omogućava da se na jednostavan način statički kreiraju i podese DSP/BIOS objekti.DSP/BIOS takođe automatski generiše kod potreban za deklarisanje, inicijalizaciju i korišćenje ovih objekata. Korisnička sprega alata za konfiguraciju prikazuje slika 3.13



Slika 3.2: Korisnička sprega alata za konfiguraciju DSP/BIOS-a

3.3.5 Konfigurisanje DSP/BIOS-a

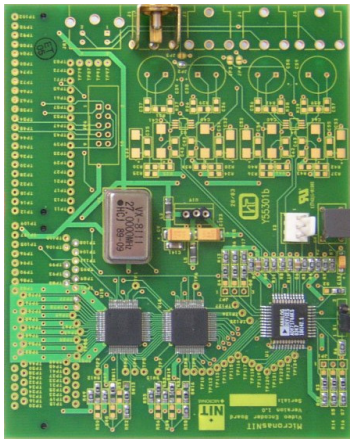
Sledeći korak je kreiranje DSP/BIOS konfiguracione datoteke, izborom opcije *New->DSP/BIOS Configuration* iz *File* menija. Po izboru ove opcije na ekranu se otvara prozor sa alatom za kreiranje DSP/BIOS konfiguracije. U levom delu ovog prozora nalazi se lista DSP/BIOS objekata, a sa desne strane se nalaze parametri trenutno izabranog objekta (Slika 3.3).



Slika 3.3: Prozor alata za konfiguraciju DSP/BIOS-a

Parametri svakog od objekata podešavaju se desnim klikom na ime objekta i izborom opcije *Properties* iz iskačućeg menija. U nastavku su dati parametri svakog od korišćenih objekata.

3.4 ANDROMEDA Video Enkoder dodatna ploča[17]



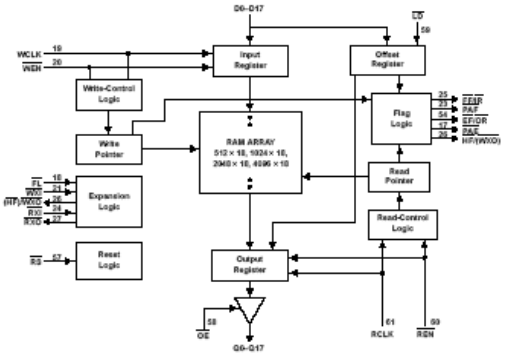
Slika 3.13 FIFO+VideoDAC proširenje

Da bi proizveo kompozitni PAL video signal (CVBS) potrebno je ANDROMEDA platformu proširiti sa Video Enkoder pločom. Na njoj se nalaze sledeće komponete:

- SN74V245 Texas Instruments FIFO memorije
- ADV7176 Analog Device PAL/NTSC Video enkoder
- Oscilator učestanosti 27MHz

3.4.1 Texas Instruments SN74V245 FIFO memorija[18]

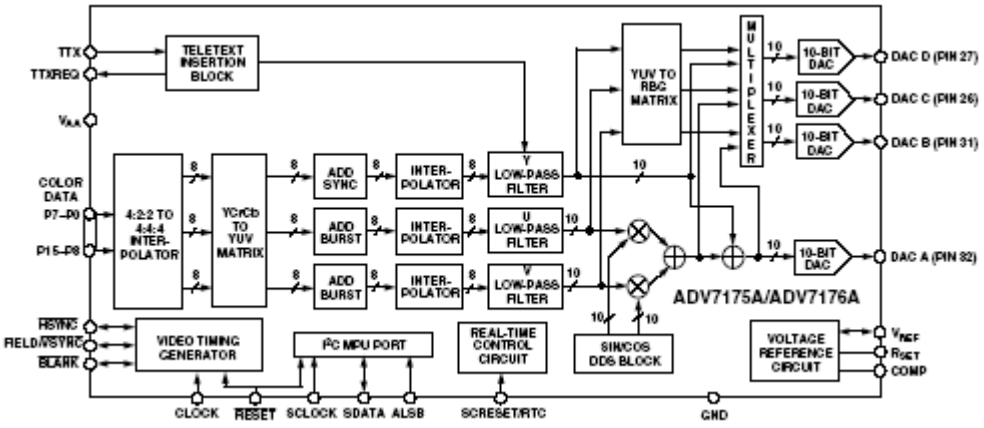
TI SN74V245 ju vrlo brza sinhrona (taktovana) FIFO memorija sa malom potrošnjom (CMOS tehnologija proizvodnje). Podržava rad na učestanostina do 133MHz i brzine pristupa do 5ns. Ova FIFO memorija ima kontrole za čitanje i upis prilagođene primenama kao što su A/D-DSP ili DSP-D/A baferovanje u video ili mrežnim uređajima. Kapacitet ove memorije je 4096 reči, dok je maksimalna dužina reči 18 bita. Povezivanjem nekoliko memorija ovog tipa na odogovarajući način mogu se dobiti FIFO memorije sa većom širinom reči kao i sa većim brojem reči. SN74V245 raspolaže sa indikatorima na osnovu kojih možemo utvrditi stanje u kome se memorija nalazi. Funkcionalni blok dijagram SN74V245 je prikazan na slici 3.14.



Slika 3.14 SN74V245 FIFO memorija

3.4.2 Analog Device ADV7176A PAL/NTSC Video enkoder[19]

ADV7176A je integrisani digitalni video enkoder koji konvertuje digitalne video podatke u formatu ITU-R BT.601/656 u standardni analogni PAL/NTSC televizijski signal. Ovaj enkoder može da primi digitalne video podatke sa vremenskim referencama koje su diktirane specijalnim kodovima unutar podataka ili HSYNC,VSYNC i BLANK signalima Za promenu režima i parametara rada ADV7176A raspolaže sa I^2C protećom (*slave*) spregom. Na slici 3.15 prikazan je blok dijagram ovog video enkodera.

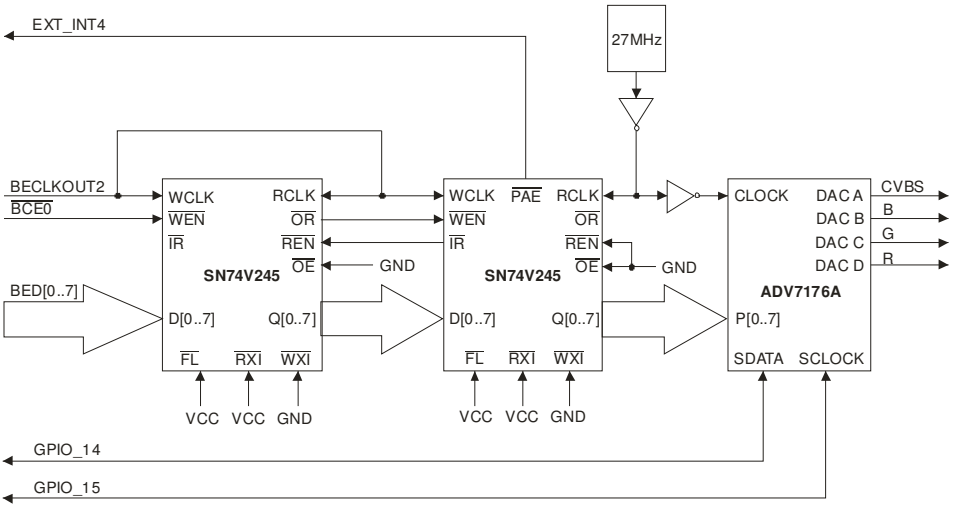


Slika 3.15 ADV7176 Video enkoder

3.4.3 Opis funkcionisanja ANDROMEDA Video Enkoder dodatne ploče

U početnom stanju obe FIFO memorije su prazne. Kada DSP otpočne sekvencu upisa u CE0 prostor EMIFB sprege, aktivira se WEN ulazni priključak prve FIFO memorije. Na svaku rastuću ivicu BECLKOUT2 takta upisuje se jedan bajt (BED[0..7]) u prvu FIFO memoriju. Taj isti takt se dovodi i na RCLK ulazni priključak prve FIFO memorije i WCLK druge memorije, čime se jedan bajt iz prve FIFO memorije prenosi u drugu FIFO memoriju. Ovo se dešava jer jer aktivan IR izlaz druge FIFO memorije (jer memorija nije napunjena) što aktivira REN ulaz prve memorije. Iz druge memorije se čita bajt na svaku rasuću ivicu takta učestanosti 27MHz. Ove

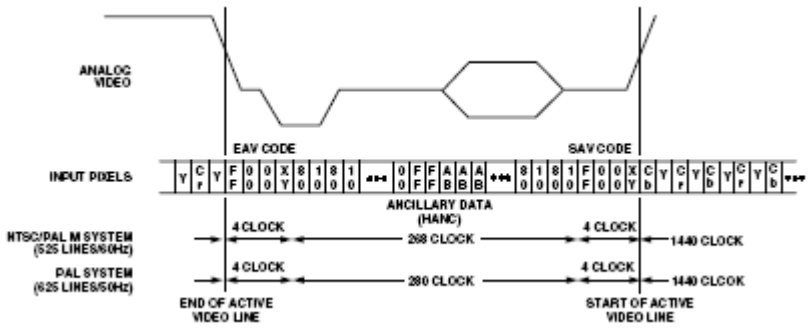
dve FIFO memorije su povezane na takav način da formiraju jednu FIFO memoriju duplo većeg kapaciteta (po broju reči).



Slika 3.15 Funkcionalni blok dijagram FIFO+VideoDAC ploče

Kako učestanost takta BECLKOUT2 iznosi 100MHz, u jednom trenutku obe FIFO memorije će se napuniti jer se podaci upisuju sa znatno većom brzinom nego što se čitaju. To će se deseti nakon upisa 10800 reči od strane DSP-a. Kada prestane upis od strane DSP-a, FIFO memorije se prazne, tako što se prvo ispazni prva FIFO memorija, a zatim druga. Kada broj reči u drugoj FIFO memoriji opadne ispod 128, signal PAE prelazi sa visokog nivoa na niski. Ovaj prelaz aktivira prekidni mehanizam u DSP-u preko EXT_INT4 priključka DSP-a, koji započinje novu sekvencu upisa u CE0 prostor. Kada broj reči u drugoj FIFO memoriji pređe 128 signal PAE se vraća na visok nivo. Na ovaj način rešen je problem kontinuiranog upisa podatka u ADV7176 taktom od 27MHz. Kontinitet upisa je neophodan zbog toga što proces bi prekidi u procesu prikazivanje slike doveli do vidnih smetnji u slici. Direktno povezivanje video enkodera na EMIFB na video enkoder sa tatktovanjem EMIFB sprege na 27MHz (BECLKIN priključak) bi blokiralo EMIFB spregu i EDMA kontroler DSP-a. Na prethodno opisan način postignut je u isto vreme kontinuitet i oslobađanje resursa DSP-a za vreme intervala dok se FIFO memorije prazne. Na slici 3.15 je prikazan blok dijagram video enkoder dodatne ploče.

Video enkoder ADV7176A treba da se nalazi u pratećem režimu 0 (*Slave Mode 0*), što znači da je ADV7176 kontrolisan SAV i EAV vremenskim kodovima koji se nalaze u video podacima. Sve vremenske refence se prenose upotrebom sinhronizacionih obrazaca dužine 4 bajta. Sinhronizacioni obrazac se šalje neposredno pre i posle svake linije tokom aktivnog dela slike i intervala povratka mlaza na početak slike. Režim 0 je ilustrovan slikom 3.16. HSYNC, VSYNC i BLNK priključci treba da se dovedu na viskom nivo tokom ovog moda.



Slika 3.16 Vremnski režim 0 (prateći režim) ADV7176A Video enkodera

Digitalni video podaci sadrže sliku je u PAL formatu, dok se ADV7176 u početnom stanju nalazi u NTSC režimu. Režim se menja upisom odgovarajućih vrednosti u njegove registre posredstvom I2C sprege. I2C sprega se ostvaruje upravljanjem GPIO14 i GPIO15 priključcima.

4. Opis programske podrške video dekodera

Realizovana programska podrške H.264 video dekodera se sastoji iz:

- bloka H.264 video dekodera
- bloka mehanizma za prikaz slike
- bloka za I^2C spregu
- DSP/BIOS konfiguracije

4.1.1 H.264 video dekoder

Kao polazna osnova za realizaciju korišćen je referentni model JM6.1e programske podrške H.264 video dekodera na PC računaru. Referentni model je ralizovan u programskom jeziku C i pored video dekodera sadrži i programsku podršku video enkodera. Ovaj video enkoder je iskorišćen za formiranje kodovanog video zapisa. Referentni model nije optimizovan, te u svojom inicijalnom obliku nije u mogućnosti da ispuni zahtev za rad u realnom vremenu, odnosno da omogući dekodovanje video sekvence u CIF rezoluciji sa 25 slika u sekundi.

Opis realizacije H.264 video dekoder je dat na kroz tok procesa dekodovanja jedne slike. Pored naziva funkcije u zagradama je data i C datateka u kojoj se funkcija nalazi.

4.1.1.1 Inicijalizacija dekodera

Incijalizacija dekodera vrši vrši funkcija `InitDecoder` (*H264.c*). U njoj se vrši alokacija memorije za strukturu podatka `ImageParameters` (*global.h*) koja sadrži kompletan kontekst dekodera u bilo kom vremenskom trenutku. U ovoj funkciji se postavljaju i početne vrednosti unutar ove strukture.

4.1.1.2 Dekodovanje jedne slike

Ovaj zadatak izvršava funkcija `decode_one_frame` (*H264.c*). Unutar ove funkcije postavljaju se inicijalne vrednosti strukture `currSlice` (*global.h*) čiji kontekst određuje stanje porocesa dekodovanja jednog dela slike (ova struktura pripada `ImageParameters` strukturi) i alokacija memorije za strukturu podatka NALU_t (*Nalu.h*) funkcijom `AllocNALU` (*Nalu.c*) koja sadrži bafer na koga pokazuje pokazivač `bsbuf`. U ovaj bafer treba da se smeste podaci koji se čitaju iz kodovanog video zapisa. Pored ovog pokazivača na ovaj bafer (alokacija memorije za ovaj bafer se vrši u funkciji `AllocNALU)`. Ova struktura sadrži dužinu i tip dela kodovanog video zapisa koji se nalazi u baferu `bsbuf`.

Nakon alokacije memorije za deo kodovanog video zapisa koji opisuje sliku, ulazi se u petlju koja se izvršava sve dok se ne dekoduju svi delovi (*slices*) jedne slike ili dok se ne dostigne kraj celokupne sekvence. Unutar te petlje vrši se pribavljanje i analiza kodovanih podataka dela slike funkcijom `read_new_slice` (*H264.c*), da bi se zatim na osnovu tih podatka izvršilo dekodovanje dela slike funkcijom `decode_frame_slice` (*Image.c*). Unutar

funkcija `decode_frame_slice` prvo se vrši provera da li je u pitanju početna slika kodovane video sekvence. Ako je u pitanju prva slika poziva se funkcija `init_frame` (*Image.c*). Pozivom funkcije `reorder_mref` vrši se uređivanje bafera koji sadrži više referentnih slika u skladu sa rasporedom režima kodovanih slika i operacija za upravljanje baferom (koje se mogu nalaziti u kodovanoj video sekvenci). Pozivom fukcije `decode_one_slice` (*Image.c*) vrši se dekodovanje dela slike. Po završetku dekodovanja jedne slike funkcijom `FreeNALU` vrši se oslobađanje memorije koja je zauzeta od strane `AllocNALU` funkcije.

4.1.1.2.1 Pribavljanje i analiza kodovanog dela slike

Unutar ovog procesa koga vrši funkcija `read_new_slice` odvija se postupak pribavljanja kodovanih podataka mehanizmom RTP paketa funkcijom `GetRTPNALU` (*Rtp.c*) i analiza njegovog tipa radi utvrđivanja daljeg toka pribavljanja toka pribavljanja podatka. Na početku video sekvence nalazi se skup parametara vezanih za sekvencu (*Sequence Parameter Set*) koga prati skup parametara slike (*Picture Parameter Set*). Ovim u ovim skupovima se nalaze parametri koji su zajednički za sve kodovane slike jedne sekvence. Kada se u toku pribavljanja kodovanih podataka naiđe na ove skupove, tada se pozivaju funkcije `ProcessSPS` i `ProcessPPS` (*Parset.c*), koje analiziraju i skladište dobijene parametre. Kada se u pribavljanju i analizi naiđe na kodovane podatke vezane za deo slike, tada se prvo vrši priprema kodovanih podataka za analizu funkcijom `PrepareBitstream` (*H264.c*). Prvo se analizira zaglavlje dela slike (*slice header*). Ova analiza se vrši pomoću dve funkcije `FirstPartOfSliceHeader` i `RestOfSliceHeader` (*H264.c*). Ovo je urađeno na ovaj način iz razloga što se u prvom delu zaglavlja dela slike nalazi referenca koja pokazuje koji skup parametra treba upotrebiti za dekodovanje tog dela slike. Ta reference se prosleđuje funkciji `UseParameterSet` (*Parset.c*). Nakon ovog može se izvršiti analiza drugog dela zaglavlja dela slike funkcijom `FirstPartOfSliceHeader` (tok izvršenja ove funkcije zavisi od toga koji skup parametara slike treba koristiti). Nakon ovog vrši se postavljanje inicijalnih vrednosti indikatora i brojača vezanih se dalji proces dekodovanja slike. Pokazivač `bsbuf` pokazuje na prve podatke koji koduju sam deo slike

4.1.1.2.2 Dekodovanje dela slike

Dekodovanje dela slike vrši se po makroblokovima. U petlji koja se izvršava sve dok se ne dostigne kraj dela slike, vrši se procedura koju sačinjavaju sledeće funkcije koje se nalaze u datoteci *Makroblok.c*: `start_macroblock`, `read_one_macroblock`, `decode_one_ macroblock` i `exit_macroblock`. Funkcija `start_macroblock` izvodi inicijalizaciju stanje trenutnog makrobloka. Svaki makroblok slike je opisan strukturom tipa `Macroblok`. Ove strukture se nalaze u nizu `mb_data`, a indeks trenutnog makrobloka se čuva u promenjivoj `current_mb_nr`. U strukturi tipa `Macroblok` nalaze se svi neophodni podaci potrebni u postupke dekodovanja makrobloka. Funkcija `read_one_macroblock` analizira kodovane podatke. Na osnovu te analize dekoder dobija informaciju o tipu makrobloka, rasporedu koeficijenta, koeficijente, razliku vektora pomeraja, itd. Sada funkcija decode_one_macroblok može da izvrši sam proces dekodovanja, odnosno da rekonstriše makroblok na osnovu podataka dobijenih predhodnom funkcijom. Funkcija `exit_macroblock` proverava da li je u pitanju poslednji makroblok dela slike, ako jeste proces dekodovanja dela slike je završen.

4.1.1.2.3 Dekodovanje makrobloka

Jezgro dekodovanja makrobloka čine funkicje `read_one_macroblock` i `decode_one_ macroblock`. Proces dekodovanja ide sledećim redom prvo se funkcijom `CheckAvailabilityOfNeighbors` (*Macroblock.c*) proverava dostupnost susednih makroblokova t.j. da li se mogu upotrebiti u postupku dekodovanja makrobloka. Zatim se pribavlja i analizitra tip makrobloka. Ako je makroblok kodiran u Intra režimu potrebno je pribaviti podatke o režimima kodavanja svakog od 16 4x4 bloka. Ovo radi funkcija `read_ipred_modes` (*Macroblock.c*). Ako je makro blok kodiran u Inter režimu tada je

potrebno funkcijom `readMotionInfoFromNAL` (*Macroblock.c*). pribaviti podatke o vektorima pomeraja i indeksu referentne slike. Ova funkcija koristi `SetMotionVector Predictor` (*Macroblock.c*) funkciju da bi formirala vektor pomeraja trenutnog makrobloka na osnovu prethodnih vektora pomeraja. Bilo da je u pitanju Intra ili Inter režim kodovanja poziva se funkcija `readCBPandCoeffsFromNAL` (*Macroblock.c*) da bi se odredila šema kodovanih blokova t.j. koji od blokova unutar makrobloka sadrže koeficijente različite od nule i da bi se pribavili koeficijenti takvih blokova. Za pribavljanje koeficijenta koristi se funkcija `readCoeff4x4_CAVLC` (*Macroblock.c*). Postupak dekodovanja makrobloka se svodi u slučaju Intra makrobloka kodovanog u Inter_4x4 režimu na formiranje predikcije za svaki 4x4 blok fukcijom `intrapred` (*block.c*) U slučaju Inter_16x16 režima funkcijom `intrapred_16x16` (*block.c*) formira se predikcija za čitav makroblok. U Intra režimima kodovanja predikcija se formira na osnovu vektora pomeraja i indeksa referentne slike funkcijom `get_block` (*block.c*). Nakon dobijanja predikcije i vrši se sabirnje inverzno transformisanih koeficijenta funkcijom `itrans` (*block.c*) sa predikcijom kako bi se dobio dekodovani makroblok.

4.1.1.3 Završetak dekodiranja slike

Po završetku izvršavanja funkcije `decode_one_frame` vrši se proces filtriranja slike rekonstrukcionim filtrom. Ovaj zadatak obavlja funkcija `DeblockFrame` (*LoopFilter.c*). Nakon filtriranja slika može da se prebaci u bafer za referentne slike, kako bi bila upotrebljena pri dekodiranju slika kodovanih u Inter režimima predikcije. Slika se nalzi u memoriji na mestu na koje pokazuju pokazivači `imgY[0]` (Y komponenta), `imgUV[0][0]` (U komponenta) i `imgUV[1][0]` (V komponenta).

4.1.1.4 Ostale funkcije

U datoteci *memalloc.c* nalaze se funkcije koje se koriste potrebe za alokaciju memorije za višedimenzionalne nizove. *Mbuffer.c* sadrži funkcije za smeštaj referentnih slika i funkcije za rukovanje baferom referentnih slika na osnovu kontrolnih operacija zadatih kodovanom sekvencom.

4.1.1.5 Analiziranje bitske sekvence (Bitstream parsing)

H.264 bitska sekvenca je serijska, i ima dobro definisanu sintaksu specificiranu ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC. Funkcije za raščlanjavanje su funkcije za indentifikaciju zaglavlja, čitanje parametara za proces dekodiranja i stvarnog čitanja podatka slike kao što su vektori pomeraja i koeficijenti transformacije predstavljeni pomoću kodova sa promenljivom dužinom. Ove funkcije uključuju operacije na nivou bita (pomeranje, maskiranje, poređenje), kao i značajan broj operacija vezanih za kontrolu i donošenje odluka. Ovo poslednje nastaje usled postojanja mnoštva različitih odluka vezanih za režim (tip slike, tip makrobloka, tip kompenzacije pokreta, VLC tabela itd.) koji treba doneti tokom faza raščlanjavanja. Čak iako je raščlanjavanje najbolje izvedeno programskom podrškom, procesori sa dubokom protočnom strukturom neće ga efikasno izvršavati zbog čestih operacija grananja i izračunavanja uslova. Zbog posedovanja mogućnosti uslovnog izvršavanja instrukcija, na C64x arhitekturi je poželjno koristiti kratke CASE strukture i IF-THEN-ELSE strukture bez ugnježdenih struktura te vrste.

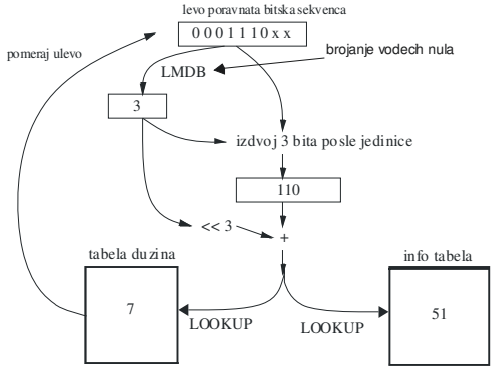
4.1.1.5.1 Dekodiranje kodova sa promenjivom dužinom

Kod osnovnog H.264 profila koriste se dva tipa kodova sa promenljivom dužinom i to dva tipa: Exp-Golomb i CAVLC. Proces dekodirnja uključuje poređenje ulaznih bitova sa korektnim kodom sa promenljivom dužinom i napredovanje u bitskoj sekvenci za dužinu koda. Činjenica da je dužina koda poznata nakon njegovog dekodovanja predstavlja ozbiljan problem za jer se onda proces dekodovanja ne može paralelizovati, te stoga potrebno na što efikasniji (mali broj instrukcija) način izvesti dekodovanje jedne kodne reči.

Exp-Golumb kodovi se mogu na dosta efikasan način (sa stanovišta potrebnog broja instrukcijskih ciklusa) dekodiraju jer se formiraju prema tačno utvrđenom pravilu. Potrebno je

odrediti broj nula koje prethode prvoj jedinici u bitskoj sekvenci da bi se odredila kodna reč i njena dužina. Ovo se na C64x arhitekturi vrši pomoću instrukcije LMBD, ona posle jednog ciklusa daje broj 0(1) koji prethode prvoj 1(0).

CAVLC tabele kodnih reči su formirane empirijskim putem te stoga ne postoji jasno pravilo koje povezuje sadržaj kodne reči sa njenom dužinom. To povlači za sobom korišćenje look-up tabela za dekodiranje. Metoda koja se koristi u referentom modelu H.264 je potpuno neupotrebljiva na C64x arhitekturi, jer se koristi metoda totalnog pretraživanja look-up tabela CAVLC kodova, što dovodi do velikog i promenljivog broja instrukcijskih ciklusa potrebnih za dekodovanje jedne kodne reči. Metoda koja se koristi u ovom rešenju je zasnovana na formiranju kombinovanih tabela koje sadrže informaciju koju nosi kodna reč i njenu dužinu. Na osnovnu broja vodećih nula (instrukcija LMBD) u trenutnoj kodnoj reči i bitova koji slede posle prve jedinice formira se indeks koji se zatim koristi u odgovarajućim tabelama, tada se iz tabela iščitava informacioni sadržaj i dužina koda. Ovaj metod obezbeđuje fiksan broj ciklusa po kodnoj reči t.j. efikasnost metode ne zavisi od dužine kodne reči. U praksi se ova metoda se pokazala vrlo efikasnom.



Slika 4.1 Dekodiranje VLC kodova

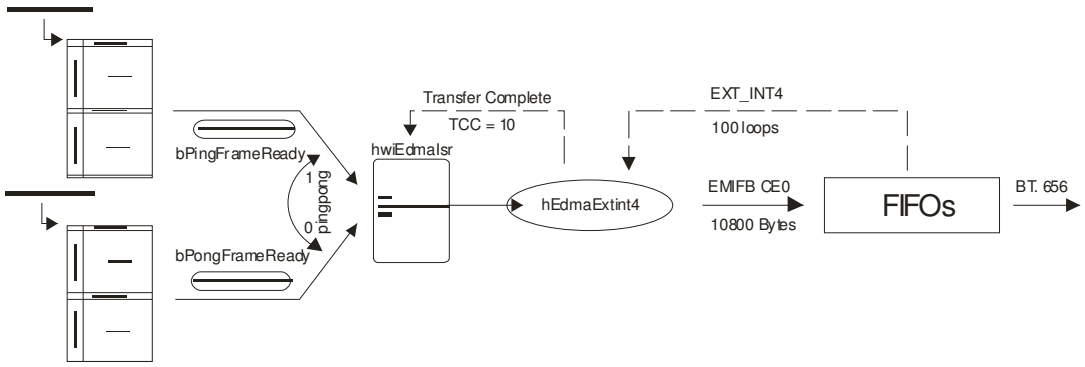
Funkcije koje realizuju ovaj proces se nalaze u datoteci *Vlc.c*. Funkcija `u_v` isčitava niz bitova čija se dužina broj unapred zna. Taj niz bitova se interpretira kao celobrojna neoznačena vrednost (unsigned_value) Funkcije koje dekoduju Exp-Golomb VLC kodove su `ue_v` i `se_v`. Prva služi za dekodovanje neoznačenih celobrojnih vrednost, dok druga dekoduje celobrojne označene vrednosti. Sledeće funkcije realizuju CAVLC mehanizam dekodovanja koeficijenta makrobloka. Funkcija `read_Intra4x4PredictionMode` dekoduje režime kodovanja svakog 4x4 bloka unutar makrobloka. Ona se koristi u funkciji `read_ipred_modes`. Funkcije `readSyntaxElement_NumCoeffTrailingOnes`, `ParseLevelInformation`, `readSyntaxElement_TotalZeros`, `ParseRunInformation` realizuju proces CAVLC dekodovanja koeficijenta opisan u drugom poglavlju. Ove funkcije se pozivaju od strane funkcije `readCoeff4x4_CAVLC`. Za potrebe dekodovanja hroma vrednosti makrobloka koristi se još i funkcije `readSyntaxElement_ NumCoeffTrailingOnesChromaDC` i `readSyntaxElement_TotalZerosChroma DC`.

4.1.2 Realizacija mehanizma za prikaz slike[20][21]

Slika koja se treba da se prikaže formira se ADV7176 video enkoderom na osnovu podataka koji dobijaju iz dve FIFO memorije koju su vezane redno na takav način da formiraju jednu FIFO memoriju dvostruko većeg kapaciteta. FIFO memorije su mapriane u CE0 prostor EMIFB sprege. Slika u 4:2:2 formatu u isprepletanom režimu zajedno sa BT.656 horizontalnim i veriklanim vremenskim referencama, se nalazi u baferu na koji ukazuje pokazivač *pPingFrameBuffer* (*pPongFrameBuffer).*

Kompletan mehanizam slanja podatka iz bafera je izveden upotrebom kanala br. 4 EDMA kontrolera i prekida generisanog EDMA kontrolerom. Prenos preko EDMA kanala 4 se aktivira pomoću događaja na DSP priključku EXT_INT4. Prenos se vrši upotrebom sinhronizacije na nivou niza podataka, to znači da se jedan kompletan niz dužine 10800 bajtova prenese po

jednom sinhronizacionom događaju. Sinhornizacioni događaj predstavlja opadajuća ivica signala na priključku EXT_INT4 koji je povezan sa priključkom PAE na FIFO memoriji koji služi za indikaciju da je FIFO memorija skoro prazna. Takvih nizova podataka ima 100 u jednom baferu. Po početku slanja poslednjeg niza EDMA kontroler generiše prekid koji je vezan za prekidnu rutinu *hwiEdmaIsr*. U prekidnoj rutini se donosi odluka o tome da li da se šalju podaci iz suprotnog bafera ili da se ponovi slanje podataka iz trenutnog bafera. Ova odluka se donosi na osnovu spremnosti suprotnog bafera za slanje. Spremnost bafera zavisi od integriteta podataka slike koje on sadrži (t.j. da li se podaci iz jedne kompletne slike nalaze u tom baferu i da li je ta slika sledeća na redu za prikazivanje). Pošto se prekidna rutina *hwiEdmaIsr* koristi i od strane drugih EDMA procesa, prenosu ovih bafera je dodeljen kod 10 završetka prenosa (TCC = 10). Promenjiva *pingpong* određuje koji se bafer trenutno šalje, ako je 1 tada je u toku slanje PongFrameBuffer-a, a ako je 0 tada se šalje PingFrameBuffer. Kada je suprotni bafer spreman za slanje, ova promenjiva menja vrednost (sa 1 na 0 ili sa 0 na 1) u prekidnoj rutini kada je u toku slanje poslednjeg niza iz trenutnog bafera, sa promenom vrednosti *pingpong* promenjive vrši se postavljanje parametra EDMA kanala 4, tako da kada se FIFO memorija skoro isprazni doći će do generisanja sinhornizaciong događaja koji će pokrenuti novi ciklus slanja nizova podataka EDMA kanalom 4, time se obezbeđuje neprekidnost slanja podataka prema FIFO memoriji. U slučaju da suprotni bafer nije spreman ponavlja se trenutni bafer zahvaljujući mehanizmu ponovnog punjenja (*reload*) parametara EDMA kanala 4 sa vrednostima koje su bile postavljne na početku slanja trenutnog bafera. Vreme potrebno za slanje jednog kompletnog bafera iznosi 40ms, što predstavlja 25 slika u sekundi. Od procesa koji postavlja podatke u jedan od ova dva bafera (kada se šalje *PongFrameBuffer*, tada se slobodno mogu menjati podaci u *PingFrameBuffer-u* i obrnuto) se zahteva da celokupnu obradu i postavljanje vrednosti bafera završi za manje od 40ms. Mehanizam slanja slike se jednom inicijalizuje, a potom se pokreće izazivanjem sinhronizacionog događaja od strane CPU. Nakon toga ovaj proces se neprekidno odvija sve dok se procesor ponovo ne inicijalizuje.

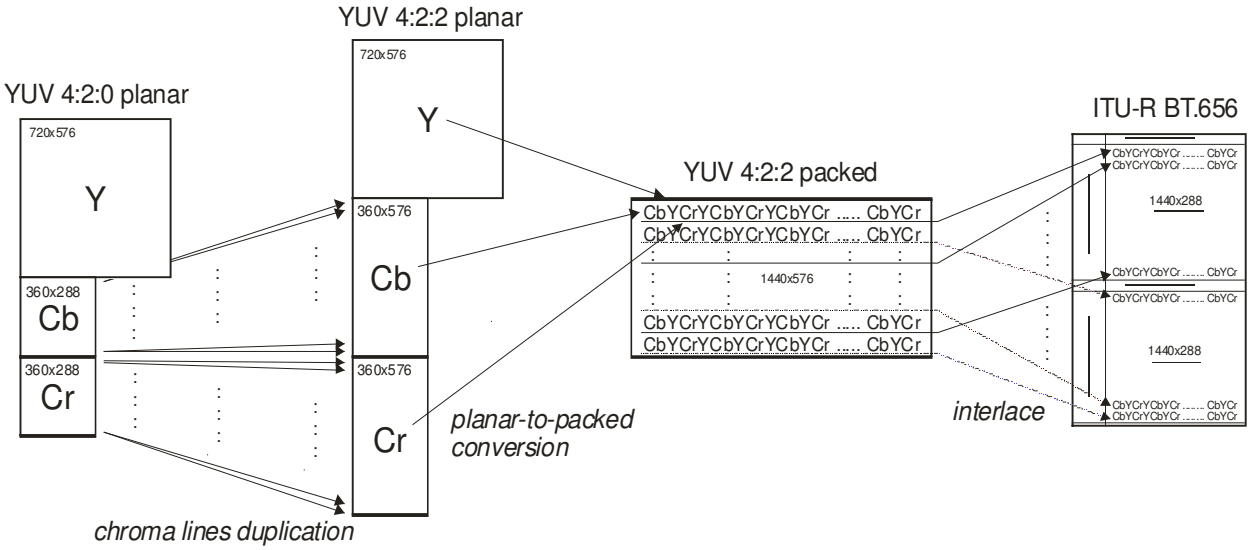


Slika 4.2 Mehanizam prikazivanja slike

4.1.2.1 Priprema slike za prikazivanje[20][21]

Y,Cb i Cr komponente slike u 4:2:0 formatu formiraju se u memoriji nakon procesa H.264 dekodiranja. Pokazivač *imgY[0]* ukazuje na memoriji gde se nalazi Y komponenta slike, dok pokazivači *imgUV[0][0]* i *imgUV[0][1]* pokazuju na Cb i Cr komponente, respektivno. Ovaj tip skladištenja slike se naziva planarni (*planar*),on se označava sa YCbCr (YUV). Za potrebe prikazivanja slike potreban nam je pakovni (*packed*) tip skladištenja, on se označava sa CbYCrY (UYVY). Međutim, CbYCrY skladištenje implicira 4:2:2 format slike, te je potrebno iz 4:2:0 formata preći u 4:2:2 format. Ovaj prelaz se izvodi dupliciranjem linija Cb i Cr komponenti. Slika uskladištena na CbYCrY predstavlja polaznu osnovu za slanje pomoću sprege opisane preporukom ITU-R BT.656.

BT.656 sprega podrazumeva prikazivanje slike u isprepletanom režimu, dok je slika koja je dobijana H.264 dekoderom u progresivnom režimu, te je potrebno razdvojiti na dve odvojene grupe parne i neparne linije. Grupisanje parne linije formiraju polusliku 1 (*field 1*), dok neparne linije formiraju polusliku 2 (*field 2*). Ovako formirane grupe (u pakovanom 4:2:2 formatu - CbYCrY) su spremne za umetanje u prethodno pripremljeni okvir za BT.656 sekvence.

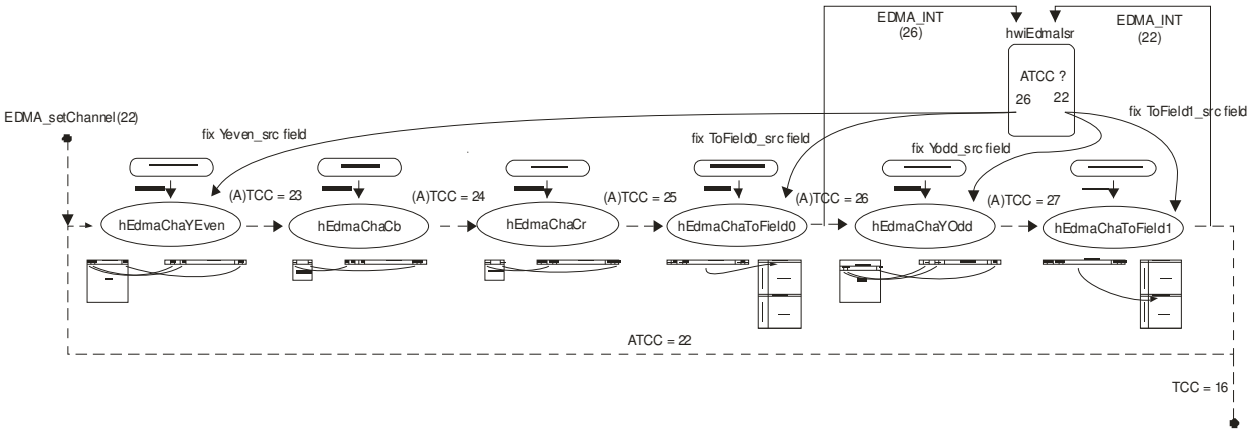


Slika 4.3 Priprema slike za prikazivanje

Postupak za formiranje BT.656 sekvenci na osnovu dekodirane slike u 4:2:0 progresivnom formatu je realizovan pomoću EDMA kontrolera i linijskog bafera koji se nalazi u internoj memoriji. Ovaj bafer, koga referencira pokazivač `pLineBuffer` je dužine 1440 bajtova i može da uskladišti jednu kompletnu liniju slike u pakovnom CbYCrY formatu. Za ovaj mehanizam upotrebljava se ukupno 6 kanala EDMA kontrolera počevši od kanala 22 do kanala 27. Ovih 6 kanala su sekvencijalno ulančani tako da po prenosu jednog okvira kanalom n dolazi do aktiviranja sledećeg n+1 kanala. Poslednji kanal u nizu (27) aktivira prvi (22), tako da se ciklus aktivacije zatvara. Ovo je moguće izvesti zahvaljujući mogućnosti ulančavanja kanala i postojanju ATC mehanizma koji omogućava da se po prenosu jednog okvira u jednom kanalu aktivira prenos u drugom kanalu. Ovaj ciklus se ponavlja onoliko puta koliko ima linija u Cb ili Cr komponenti (u svakom od kanala ova vrednost je postavljena kao brojač okvira). Kanal 22 prenosi jednu parnu liniju Y komponente na lokacije koje počinju od mesta `pLineBuffer+1` i pri tome se sukcesivne Y vrednosti postavljaju u linijskom baferu sa razmakom od 1 bajta. Kanal 23 prenosi jednu liniju Cb komponente u na lokacije sa početkom od mesta `pLineBuffer` i pri tome pri tome pravi razmak od 3 bajta između dve sukcesivne Cb vrednosti u liniji. Kanal 24 radi isto što i kanal 23 samo je u pitanju Cr komponenta i što je `pLineBuffer+2` početna lokacija za smeštanje u linijskom baferu. Kanal 25 prenosi linijski bafer u BT.656 sekvencu, na lokacije koje pripadaju aktivnoj video liniji u poluslici 1. Kanal 26 radi isto što i kanal 22 samo što su sada prenose Y vrednosti koje pripadaju neparnoj linije. Kanal 27 radi isto što i kanal 25, samo je sada u pitanju aktivna video linija u poluslici 2. Kanali 22,23,24 i 25 vrše pretvaranje iz planarnog YCbCr u pakovni CbYCrY režim skladištanja. Pretvaranje iz 4:2:0 u 4:2:2 format je ostvareno time što se linijski bafer koji sadrži parnu liniju slike puni (posle njegovog prenosa u polusliku 1 kanalom 25) neparnom linijom Y komponente, čuvajući pri tome Cb i Cr vrednosti koje su postavljene u linijskom baferu pri formiranju parne linije (zbog postojanja proreda u smeštanju Y vrednosti). Kada se sada neparana linija prenese kanalom 27 u polusliku 2, hroma vrednosti i u parnoj i u neparnoj liniju su iste t.j. ostavreno je dupliciranja hrominentnih linija. Kanali 25 i 27 vrše razvrstavanje parnih i neparnih linija u poluslike BT.656 sekvence, a samim tim i njihovo preplitanje. Zbog osobina korišćenih režima EDMA kontrolera (ne možemo ostvariti različito napredovanje SOURCE i DESTINATION parametera nakon prenosa jednog okvira) mora se u prekidnoj rutini *hwiEdmaIsr* vršiti korekcija parametara pojednih kanala u momentima kada je kanal neaktivan. SOURCE parametar kanala 22 (kanala 25) se mora uvećati za dužinu linije Y komponente, jer EDMA kontroler nakon završetka prenosa jednog okvira kanalom 22 (kanalom 25) u SOURCE polju kanala 22 (kanala 25) ima adresu prve lokacije sledeće linije, a to je neparna (parna) linija. Da bi EDMA kontroler prenosio parne linije kanalom 22, a neparne kanalom 25 mora se izvršiti korekcija SOURCE

parametra za dužinu Y komponente linije. Isto tako se SOURCE parametar uvek mora postavljati na vrednost `pLineBuffer` u kanalima 25 i 27 nakon prenosa parne,odnosno neparne linije, jer po prenosu jednog linijskog bafera SOURCE parametar pokazuje na kraj linijskog bafera, pa bi pri sledećem prenosu linijskog bafera u polusliku došlo do prenosa slučajnih vrednosti,a samim tim i pogrešnih.

Pripremu ovog mehanizma radi funkcija Prepare .Postupak se aktivira aktiviranjem kanala 22 od strane CPU. Završetak prenosa jedne kompletne slike se može proveriti ispitivanjem bita broj 16 u registru EDMA kontrolera koji sadrži informacije o završetku prenosa svakog kanala. Ovaj mehanizam je vrlo efikasan jer zahteva vrlo malo CPU aktivnosti (prekidna rutina) i može se izvršavati u isto vreme kada i dekodiranje naredne slike. Upotrebom LINK parametra, po završetku celokupnog procesa, svi kanali dobijaju početne vrednosti koje obezbeđuju da se ceo postupak može opet aktivirati.



Slika 4.4 Mehanizam pripreme slike

4.1.2.2 Formiranje inicijalnog okvira za BT.656 sekvencu[20][21]

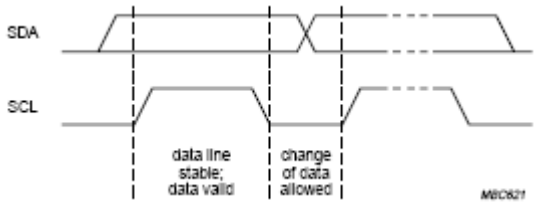
Inicijalna BT.656 sekvenca se formira funkcijom *InitBT656FrameBuffer*. Bafer veličine 1080000 (1728 X 625) bajta, koji se nalazi u delu spoljne SDRAM memorije koja nije obuhvaćena mehanizmom skrivene memorije (EXTERNAL_NONCACHED_HEAP), se puni vrednošću 0x80108010 što u pakovanom načinu skladištenja slike CbYCrY predstavlja odsustvo sve tri komponente slike Y,Cb i Cr. Potom se na odogovarjuće lokacije unutar bafera postavljaju specijalni kodovi vremenskih referenci . Lokacija i vrednost ovih kodova specificirana je proprukom ITU-R BT.656. Formiraju se dva bafera ovog tipa u spoljašnjoj memoriji *PingFrameBuffer* i *PongFrameBuffer*, zbog mehanizma dvostrukog baferovanja (*ping-pong buffering*) koji se primenjuje pri slanju podatka iz ovih bafera.

4.1.3 Realizacija I^{2}C sprege[22]

I^{2}C spega je realizovana zbog potrebe postavljanja ADV7176A u PAL režim prikaza slike, jer u incijalnom stanju ADV7176A prikazuje sliku u NTSC formatu. Kao i zbog uključivanja kompozitnog video izlaza (inicijalno DAC A je isključen). Ova programska podrška je realizovana u tri nivoa:

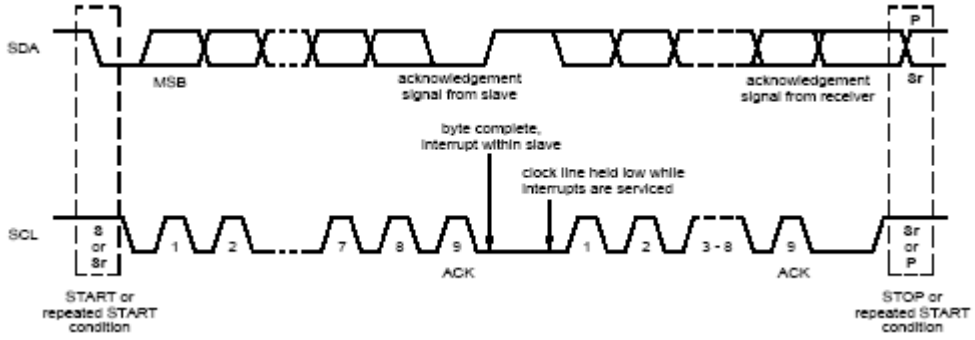
- nivo rukovanja sa SCL i SDA signalima
- nivo rukovnja slanja i prijema jednog bajta posredstvom I^{2}C sprege
- nivo upisa i čitanja registara ADV7176A video enkodera

Datoteke *gpio4i2c.c* i *gpio4i2c.h* sadrže funkcije niskog nivoa koje upravljaju stanjem na SCL i SDA priključcima I^{2}C sprege. Za SCL liniju koristi se GPIO14, a za SDA liniju GPIO15 priključak DSP-a. Pomoću funkcija iz CSL biblioteke kontrolišu se GPIO14 i GPIO15 priključci.



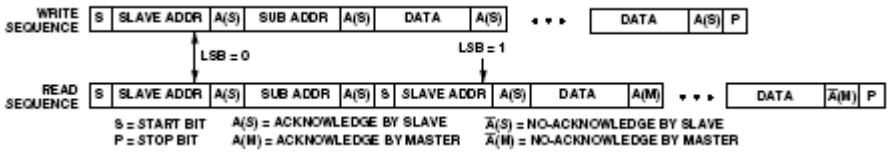
Slika 4.5

Datoteke *i2c.c* i *i2c.h* sadrže funkcije drugog nivoa. To su funkcije koje realizuju protokol za slanje i prijem bitova jednog bajta preko I^2C sprege, kao i za postavljenje START i STOP uslova na I^2C sabirnici. Ove funkcije se oslanjaju na I^2C funkcije niskog nivoa. U ovim funkcijama se koristi tajmer DSP-a, zbog potrebe generisanja signala na SCL i SDA linijama sa odgovarajućim vremenskim karakteristikama. Takođe je potrebno meriti vreme trajanja pojednih operacija zbog detekcije grešaka u toku slanja ili prijema. Za ove potrebe koriste se fukcije `TIMER_Delay` i `TIMER_Init`, koje se nalaze u datoteci *timer.c i timer.h*



Slika 4.6

Funkcije trećeg nivoa se nalaze u datotekama *ADV7176i2c.c* i *ADV7176i2c.h*. One su realizovane prema protoklu koji je specificiran dokumentacijom za ADV7176A. To su funkcije za koji služe za upis vrednosti u odgovarajući registar ili niz registara, kao i za čitanje vrednosti iz registra ili niza registara. One se oslanjaju funkcije drugog nivoa. U datoteci *ADV7176i2c.c* se nalazi niz `ADV7176_PAL_conf` koji sadrži postavku parametara za PAL režim ADV7176 video enkodera.



Slika 4.7

4.1.4 DSP/BIOS konfiguracija (H264Decoder.cdb)

U okviru ovog projekta korišćeni su sledeći DSP/BIOS objekti:

- MEM *(Memory Segment Manager*) – Modul koji upravlja alokacijom memorijskih segmenata
- HWI (*Hardware Interrupt Service Routine Manager*) – Modul koji upravlja funkcijama za obradu hardverskih prekida
- SWI (*Software Interrupt Manager*) – Modul koji upravlja funkcijama za obradu softverskih prekida
- CSL (*Chip Support Library*) – Modul koji upravlja raznim delovima procesora, kao što su EMIF sprega, EDMA kontroler i tajmeri

4.1.5 Objekat MEM

Ovaj objekat upravlja alokacijom memorijskih segmenata. Njega je potrebno konfigurisati da bi povezilac programa (*linker*) znao u koji segment memorije treba da smesti koji deo programske podrške. Pošto je kapacitet interne memorije dovoljan da u nju stane cela programska podrška, za smeštaj je korišćena ISRAM memorija. Da bi se mogli koristiti pojedini segmenti memorije, potrebno je i svaki od njih posebno konfigurisati. Dodavanje novog segmenta memorije vrši se desnim klikom na MEM objekat i izborom opcije *Insert MEM* iz iskačućeg menija. Memorijski segmenti ISRAM i CACHE_L2 koji odgovaraju internoj memoriji unutar samog procesora već su konfigurisani, pa je potrebno dodati segmente SDRAM_CACHED i SDRAM_NONCACHED. Njihove parametre ovi parametri obuhvataju početnu adresu i veličinu segmenta, veličinu onog dela segmenta u okviru kojeg se kreiraju dinamičke promenljive tzv. heap i indetifikator heap-a. Pomoću ovog indentifikatora moguće je određivati iz kog će se segmenta memorije alocirati dinamičke promenjive i baferi.

4.1.6 Objekat HWI

Ovaj objekat upravlja funkcijama za obradu hardverskih prekida. U okviru programske podrške koriste se prekid HWI_INT8 kojeg generiše EDMA kontroler. HWI_INT8 prekidom se signalizira da je završen transfer posredstvom EDMA kontrolera. Funkcija za obradu ovog prekida je _hwiEdmaIsr Pošto je ova funkcija napisane u C-u, za njeno pozivanje mora se koristiti sistemski raspoređivač procesa (*dispatcher*), koji pre poziva funkcije prebacuje sadržaj svih registara procesora na stek.

4.1.7 Objekat SWI

Objekat SWI upravlja softverskim prekidima. U okviru programske podrške korišćena su dva objekata ovog tipa: swiProcess pomoću kojeg je ralizovana funkcija obrade koja na osnovu podataka iz ulaznog bafera izračunava podatke koji se smeštaju u izlazni bafer. Novi SWI objekat dodaje se desnim klikom na objekat SWI i izborom opcije *Insert SWI* iz iskačućeg menija. Parametrima SWI objekta određuje se funkcija za obradu prekida, argumenti koji se prosleđuju funkciji, kao i inicijalna vrednost u poštanskom sandučetu procesa koji upravlja softverskim prekidom.

4.1.8 Objekat CSL

Objekat CSL (*Chip Support Library*) upravlja uređajima koji se nalaze unutar samog procesora – EDMA kontrolerom, višekanalnim serijskim portom (McBSP), spregom za eksterne memorije (EMIFA i EMIFB). Parametri svakog od ovih uređaja podeljeni su u dve grupe. Prva grupa parametara određuje koji resursi uređaja će biti korišćeni od strane programske podrške, dok druga grupa parametara definiše preinicijalnu konfiguraciju svakog od korišćenih resursa, odnosno vrednosti koje se upisuju u kontrolne registre uređaja pre njegovog korišćenja. U okviru programske podrške korišćeno je 8 kanala EDMA kontrolera. Kanali i njihovi indetifikatori su navedeni u delu dokumentacije koja objašnjava mehanizam slanja i pripreme slike. Svi parametri su postavljeni na osnovu CIF rezolucijom slike i ITU-R BT.656 prporuke.

4.1.9 Globalni parametri

Ovo su parametri čije je podešavanje neophodno za ispravno funkcionisanje DSP/BIOS-a u celini. Ovi parametri obuhvataju izbor familije i tipa procesora, učestanosti takta na kome radi procesor, i.t.d.

Posle podešavanja parametara svih DSP/BIOS objekata potrebno je snimiti datoteku sa DSP/BIOS konfiguracijom (datoteka sa ekstenzijom .CDB) izborom opcije *Save* iz *File* menija. Prilikom snimanja ove datoteke, alat za konfigurisanje automatski generiše i datoteke sa izvornim kodom u C-u i asembleru u okviru kojih su definisani i inicijalizovani svi DSP/BIOS objekti. Pored ovih datoteka, automatski se generiše i komandna datoteka povezioca (datoteka sa ekstenzijom .cmd), koja je neophodna za povezivanje programa u slučaju da se koristi tekstualni povezilac. Datoteke sa ekstenzijama .cmd i .cdb potrebno je uključiti u projekat izborom opcije *Add Files to Project* iz *Project* menija.

5. TMS320C64x

Procesori serije TMS320C64x pripadaju familiji TMS320C6000 procesora sa VelociTI arhitekturom. Ovo familija je predstavljena 1997 godine sa DSP jezgrima C62x i C67x, C6000 familija upotrebljava naprednu arhitekturu sa veoma dugačkom instrukcijskom reči (*Very Long Instruction Word - VLIW*). Ova arhitektura sadrži više izvršnih jedinica koje rade u paraleli, što ima omogućava da izvrše više instrukcija u jednom ciklusu takta. Paralelizam je ključ za ekstremno visoke performanse. Na taktu od 200 MHz i 1600 miliona instrukcija u sekundi (MIPS), C6201 je postigao deset puta veće performanse od ranijih DSP rešenja. Danas C62x serija može da postigne 2400 MIPS-a na taktu od 300MHz.

Najnoviji pripadnik familije C6000, C64x donosi najviši nivo performansi. Na taktu od 1.1GHz i većim, C64x može da obrađuje informacije brzinom od 8800+ MIPS-a ili blizu 9 milijardi instrukcija u sekundi. Početne verzije rade u opsegu takta od 600MHz do 800MHz, dajući pri tome nivo performansi od 4800 MIPS-a do 6400 MIPS-a. Pored većeg takta, više posla može biti obavljeno u svakom ciklusu, zbog VelociTI.2 proširenja VelociTi arhitkture. Ova proširenja uključuju nove instrukcije koje uvećavaju performanse u ključnim primenama i uvode dodatni paralelizam u arhitekturi.

Povećana brzina taka i povećana propusna moć CPU-a su samo deo rešenja. Obrada podataka sa ovako velikim brzinama uvećava potrebu za I/O propusnim opsegom. C64x imaju tri eksterne magistrale sa brzinom da 133MHz. Svaka magistrala ima svoju primarnu funkciju. Jedna obezbeđuje brzu spregu bez dodatne logike za sinhrone i asinhrone memorije sa propusnim opsegom od 1.1GB/s. Druga magistrala obezbeđuje spregu ka sporim perifernim uređajima dok treća podržava standardnu industrijsku spregu. Tri fleksibilna više kanalna serijska priključka sa međumemorijom (Multi–channel Buffered Serial Port - McBSP) mogu da obezbede 100Mb/s po priključku dodatnog propusnog opsega. Interni DMA mehanizam može da obezbedi preko 2GB/s I/O propusne moći sa 64 nezavisna kanala.

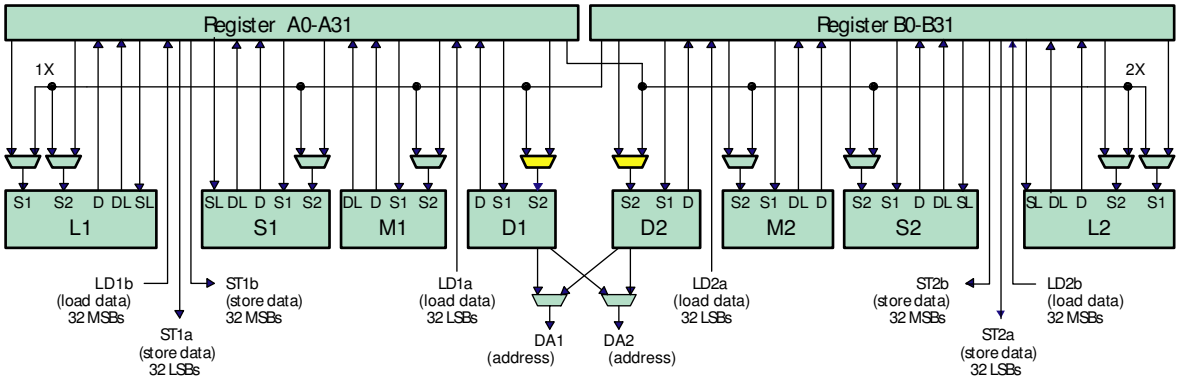
C64x ide dalje od jezgra i skupa periferija da bi obezbedio maksimalni nivo performansi za brzu obradu digitalnih podataka. Čvrsta veza između CPU arhitekture i prevodioca pomaže u maksimiziranju procesorske propusne moći. Instrukcijski skup koji liči na RISC i široka upotreba protočne obrade omogućava mnogim instrukcijama da budu izvršene u paraleli. Ključna proširenja načinjena u C62x arhitekturi koja omogućavaju C64x da uradi više posla u svakom taktu uključuju šire putanje podataka, veći skup registar, veću ortogonalonst i nove instrukcije koje podržavaju paketnu obradu podataka.

5.1 C64x centralna procesorska jedinica

C64x CPU se sastoji od:

- dva skupa registara opšte namene (*register files)* A i B
- osam funkcionalni jedinica (*functional units)* .L1, .L2, .S1, .S2, .M1, .M2, .D1, i .D2
- dve putanje za podatke koji se pribavljaju iz memorije (*load-from-memory data path)* LD1 i LD2.
- dve putanje za podatke koji se skladište u memoriju (*store-to-memory data paths)* ST1 i ST2.
- dve putanje za adrese podatka (*data address paths*) DA1 i DA2
- dve poprečne putanje podataka skupa registara *(register file data cross paths)* 1X i 2X

Slika 5.1 ilustruje VelociTI (sa VelociTI.2 proširenjima) CPU arhitekturu.



Slika 5.5.1 C64x VelociTI.2 arhitektura CPJ-e

5.1.1 Skup registara

Postoje dva skup registra opšte namene.Ovi registri opšte namene mogu da se upotrebljavaju za smeštaj podatka, pokazivače na podatke i kao za uslovne registre. Kod C64x svaki od ova dva skupa sadrži 32 32-bitna registra (A0-A31 za skup A i B0-B31 za skup B). Na C64x A0, A1, A2, B0, B1 i B2 mogu da se upotrebljavaju kao uslovni registri, dok registri A4-A7 i B4-B7 mogu sa se upotrebljavaju za cirkularno adresiranje. Skup registara C64x podržava podatke širine 8,16, 32, 40 i 64 bita. 40-bitni i 64-bitni se dobiju pomoću regitarskih parova, sa 32 bita najmanje težine postavljenim u registru sa parnim brojem, a sa preostalim 8 ili 32 bita najveće težine u sledećem višem registru po redu. C64x skup registara podržava paketni tip smeštaja podataka: 4 8-bitne ili 2 16-bitne vrednosti u jednom 32 bitnom registru ili 4 16-bitne vrednosti u 64-bitnom registarskom paru.

5.1.2 Funkcionalne jedinice

8 funkcionalnih jednica u C64x putanjama podataka mogu da se podele u dve grupe od po četiri jedince; svaka funkcionalna jednica i u jednoj putanji podatka je identična odgovarajućoj jedinici u drugoj putanji podatka. Svaka funkcionalna jedinica ima direktan pristup skupu registara unutar svoje putanje podataka. Tako da .L1, .S1, .D1, i .M1 jednice pristupaju skupu registara A, dok L2, .S2, .D2, and .M2 jednice pristupaju skupu B registara. Većina linija podataka unutar CPU podržava 32-bitne operande, a neke podržavaju 40-bitne operande. Svaka funkcionalan jedinica ima svoj 32-bitni priključak za upisivanje rezultata u skup registara. Svaka funkcionalna jedinica ima 32-bitne priključke za čitanje operanada. Četiri jedinice (.L1, .L2, .S1 i .S2) imaju dodatne 8-bitne priključke za 40-bitno upisivanje , kao i 8-bitne ulaze za 40-bitno čitanje. Zbog toga što svak jedinica ima svoj 32-bitni port za upisivanje, svih 8 jedinca mogu da se upotrebljavaju u paraleli u svakom ciklusu. U tabeli 5.1 navedene su neke od operacija koje su podržane pojedinim funkcionalnim jedinicama. Dve od ovih osam jedinca su množači (*multipliers*). C64x su sposobni da izvedu 2 16 x 16 bitna množenja, ukupno 4 16x16 bitna

množenja u jednom ciklusu. Štaviše, svaki množač je ustanju da izvede 4 8x8 bitna množenja, ukupno 8 8x8 bitna množenja u jednom ciklusu.

Funkcionalna jedinica	Operacije
.L jedinca (.L1,.L2)	32/40-bitne aritmetičke i operacije poređenja Logičke operacije
.S jedinca (.S1, .S2)	32-bitne aritmetičke operacije 32/40-bitna pomeranja i 32-bitne bit orjentisane operacije 32-bit logičke operacije
.M jedinca (.M1, .M2)	16 x 16 bitne operacije množenja
.D jedinca (.D1, .D2)	32-bitne linearne i cirkularne adresne kalkulacije

Tabela 5.5.1 Funkcionalne jedinice i operacije

5.1.3 Poprečne putanje podataka

Funkcionalne jedince koje pripadaju jednom skupu registra mogu da pristupe drugom skupu preko 1X i 2X poprečnih putanja. Ove poprečne putanje dozvoljavaju funkcionalnim jedinicama koje pripadaju jednom skupu registara da pristupe 32-bitnom operandu iz drugog skupa registra. .M1, .M2, .S1, .S2, .D1 i .D2 imaju jedinice mogućnost da im jedan od operanda bude iz suprotnog skupa registara. Kod L1 i .L2 jedinca oba operanda mogu biti iz suprotnog skupa registara. Pošto postoje samo dve poprečne putanje, moguće je izvesti jedno ili dva čitanja u jednom ciklusu suprotnog skupa registara.

5.1.4 Putanje za pribavljanje i skladištenje podataka

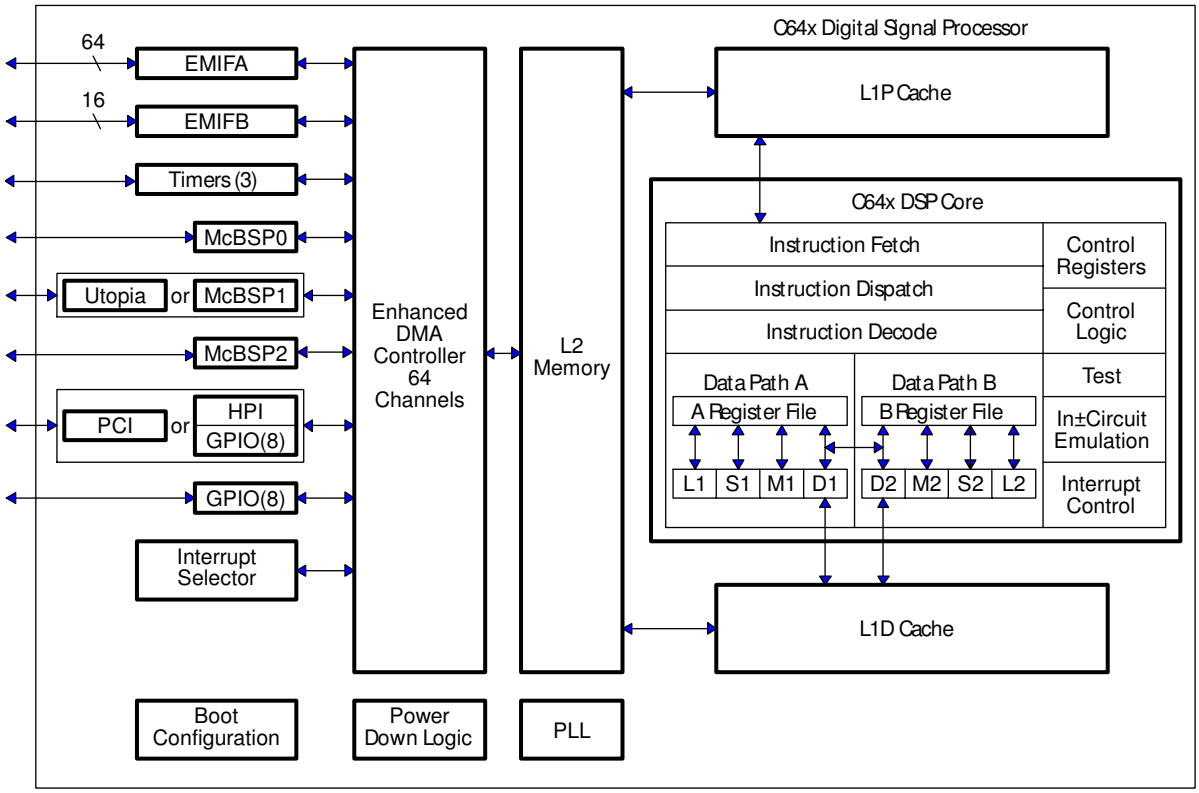
C64x podržava 64-bitno pribavljanje i skladištenje podataka. Postoje četiri 32-bitne putanje za učitavanje podataka iz memorije u skup registara i četiri 32-bitne putanje za smeštanje vrednosti registara u memoriju. Za stranu A, putanje LD1a i LD1b sliže za učitavanje 64-bitnog podatka, dok ST1a i ST1b služe za smeštanje 64-bitnih podataka. Na strani B putanje LD2a i LD2b služe učitavanje, ST2a i ST2b služe za smeštanje 64-bitnih podataka. Ovakav 64-bitni pristup memoriji je esencijalan u podršci procesnoj moći. C64x može da pristupi 32-bitnom i 64-bitnom podatku na svakoj memorijskoj lokaciji (nije bitno poravnanje (*aligment*)). Ovo značajno olakšava korišćenje procesne moći C64x.

5.1.5 Putanje adresa podataka

Putanje adrese podataka (DA1 i DA2) na slici 8.1-1 koje izlaze iz .D1 i .D2 jedinca omogućavaju da adresa generisana u jedom skupu registra bude upotrebljena za učitavanje i skladištenje podataka u drugom skupu registra.

5.2 TMS320C6415 DSP

Na slici 5.2 prikazan je blok dijagram TMS320C6415 DSP-a.



Slika 5.2 Blok dijagram TMS320C6415

Osobine od najvećeg značaja, koje poseduje TMS320C6415 su:

- C64x DSP jezgro (opisano u prethodnom odeljku).
- organizacija skrivene memorije u dva nivoa *(Two–Level Cache Architecture).*
- poboljšani DMA kontroler *(Enhaced Direct Memory Acces - EDMA).*
- tri eksterne magistrale.
- fleksibilnu serijsku komunikaciju.
- UTOPIA priključak.
- Ulaze/izlaze opšte namene (*General Purpose Input/Output*).

5.2.1 Organizacija skrivene memorije u dva nivoa

Kod ovog DSP-a, CPJ je direktno povezana na skrivenu memoriju prvog nivoa za instrukcije (L1P) i za podatke (L1D) svaka je kapaciteta 16KB. Ove skrivene memorije rade na brzini procesora. Drugi nivo skrivene memorije čini L2 memorija koja obezbeđuje smeštaj za instrukcije i podatke. Slika 8.2.1-1 opisuje primer L2 memorije od 1024 KB; veličina i segmentacija L2 skrivene memorije mogu se menjati. U jednoj postavci L2 je ceo mapiran kao unutršnja SRAM memorija. Druge postavke imaju i SRAM i skrivenu memoriju različitih veličina. Menjajući postavku L2 memorije, korisniku je dozvoljeno da kritične delove programa kao što su prekidne rutine ili često pozivane funkcije postavi u unutrašnju L2 SRAM memoriju, kao i kritične podatke kao što su stek i često korišćeni koeficijenti.

5.2.2 Poboljšani DMA kontroler (EDMA)

C64x EDMA može da obezbedi preko 2GB/s propusne moći. EDMA podržava do 64 kanala koji se aktiviraju nezavisnim događajima. Ukupno 85 skupova parametara stoje na raspolaganju za povezivanje *(linking)* i ulančavanje *(chaining)*. Povezivanje omogućava da niz prenosa podatak bude izvršen kada se desi jedinstven događaj. Ulančavanje omogućava da jedan EDMA kanal aktivira drugi po završetku prenosa podataka. Povezivanje i ulančavanje omogućavaju

kontinualne samo-inicijalizujuće DMA operacije gde je samo početna postavka urađena od stane CPU. Pomoću ovih osobina moguće je izvesti cirkularne bafere, ping-pong bafere i prenose kompleksnih struktura podataka. Prenosi mogu biti aktivirani na pojedinačne elemente ili na kompletne okvire, kao i da prenos može biti izveden element po element ili okvir po okvir. Svaki kanal podržava (1-D) jedno- i (2-D) dvo-dimenzionalne prenose. Korak se nezavisno programira za svaku dimenziju. Upotrebom 1-D i 2-D korisnik može da prenese deo slike, kao da automatski prepliće i raspliće vremenski raspodeljene (*time–division multiplexed - TDM*) digitalne podatke. Bajt, polu-reč, reč i dvostruka reč kao veličina elementa su podržani.

EDMA poseduje nenadmašnu konkuretnost operacija. Četiri nezavisna reda za zakazivanje prenosa (*transfer queue*) omogućuju operacije sa viskom efikasnošću. Kanali koji se nalaze u različitim radovima mogu da se međusobno prepliću od ciklusa do ciklusa. Na primer, u ciklusu 1 red 0 može da uslužuje promašaj u skrivenoj memoriji L2 prema EMIFA. U ciklusu 2, red 1 može da premešta podatke iz serijskog priključka u EMIFB. U ciklusu 3, HPI može da prenese podatke iz unutrašnje memorije kroz red 3. U ciklusu 4, EMIFA može da premešta podatke u serijski priključak. Ključna prednost ovog sistema je to što prostim sabiranjem potrebnih zahteva za propusnom moći možemo da vidimo da li EDMA može da podrži zahteve korisnika.

5.2.3 Tri spoljašnje sabirnice

TMS320C6415 poseduje 3 paralelne spoljne magistrale: dve spoljne memorijske sprege (*External Memory Interfaces - EMIFs*) i jednu spregu za povezivanje sa drugim procesorima (*Host Port Interface - HPI*). Jedna spoljašnja memorijska sprega (EMIFA) ima širinu od 64-bita i namenjena je za direktno povezivanje sa brzim sinhronim memorijama. Druga 16-bitna spoljašnja memorijska sprega (EMIFB) je namenjena za spoljašnje U/I periferije kao što su FIFO-e i paralelni A/D i D/A pretvarači Razdvajanje memorije od U/I uređaja pojednostavljuje projektovanje fizičke arhitekture i obezbeđuje konkurentost U/I operacija. Iako su namene ove dve memorijske sprege različite, one su funkcionalno identične osim po širine, što dozvoljava mnoštvo različitih rešenja u postupku projektovanja.

Ove spoljašnje memorijske sprege imaju maksimalnu brzinu magistrale od 133MHz, poseduju četiri signala za selekciju memorijsko prostora kojem se pristupa (*chip enable - CE*). EMIFA podržava operacije čitanja i upisa sa spoljašnjim uređajima čija je magistrala podataka širine od 64-,32-,16-, i 8-bita. EMIFB podržava 16- i 8-bitne spoljašnje uređaje. Ova promenjiva širina sabirnice podataka, dozvoljava međusobno funkcionisanje mnogih spoljašnjih U/I periferija i dozvoljava projektantima da čine kompromise između propusnog opsega, cene i potrošnje. Svaki EMIF poseduje tri različita memorijska kontrolera. SDRAM kontroler podržava SDRAM memorijske kapaciteta od 16Mb do 256Mb. Programabilni sinhroni kontroler sa mogućnošću podešavanja trajanja operacija čitanja/pisanja (*read/write latency*) nudi mogućnost direktnog spajanja sa sinhronim FIFO-ma i sinhronim SRAM memorijama. Programabilni asinhroni kontroler sa mogućnošću nezavisnog podešavanja vremena trajanja faze postavljanja (*setup*), faze čekanja (*strobe*), faze zadrške (*hold*) operacija čitanja, kao i operacija pisanja, omogućava jednostavnu spregu sa mnogim asihronim SRAM memorijam, FIFO-ma i perifernim uređajima. Spoljašnje memorijske sprege poseduju mogućnost razdvajanja radne frekvencije CPJ-e i frekvencije magistrale, uz pomoć ulaznih priključaka za spoljni takt. Svaki od kontrolera može da radi sa 1x, 1/2x, 1/4x takta magistrale. Sve ove mogućnosti mogu se nezavisno koristiti i podešavati u svakom od četiri adresna prostora u svakom EMIF-u.

32-bitna HPI sprega dopušta povezivanje sa mnoštvom različitih industrijski standardnih procesora i PCI mostova. HPI može da radi bilo na 32-bitnim (HPI32) ili 16-bitnim (HPI16) sabirnicama. Dodatna upotreba HPI sprege je u ulozi prolaza za podatke, koje nadređeni (*master*) periferni uređaj može da čita ili piše u podređeni (*slave*),u toj postavci,DSP.

U zavisnosti od konfiguracije uspostavljene pri inicijalizaciji DSP-a, HPI prolaz može da bude zamenjen sa PCI prolazom. PCI prolaz podržava povezivanje DSP-a na PCI sabirnicu preko ugrađene PCI 32-bitne sprege koja radi na frekvenciji od 33MHz. PCI prolaz potpuno je saglasan sa revizijom 2.2 PCI specifikacije. Posredstvom PCI sprege omogućeno je ostalim uređajima na PCI sabirnici da pristupaju unutrašnjoj i spoljašnjoj memoriji (preko EMIF sprega) DSP-a.

5.2.4 Prilagodljiva serijska veza

Tri više-kanalna serijska prolaza sa međumemorijim (McBSPs) podržavaju mnoštvo različitih standardnih serijskih sprega. Kao što su telekomunikacione ST-Bus, H.110/H.100, T1/E1, IOM2 spege, zatim digitalne audio I2S i AC97 spege, kao i SPI spregu.

5.2.5 UTOPIA prolaz

Jedan od McBSP prolaza može se konfigurisati pri inicijalizaciji kao UTOPIA (*Universal Test and Operations Interface for ATM)* prolaz. C64x UTOPIA sprežna logika je podređeni (*slave*) ATM kontroler (ATMC) koji se spreže na nadređeni (*master*) ATM kontroler. UTOPIA prolaz odgovara specifikaciji standarda ATM foruma. Ovaj prolaz podržava UTOPIA nivo 2 spregu što omogućava 8-bitnu vezu na radnoj frekvenciji od 50MHz za operacije slanja i primanja.

5.2.6 Ulazi/Izlazi opšte namene

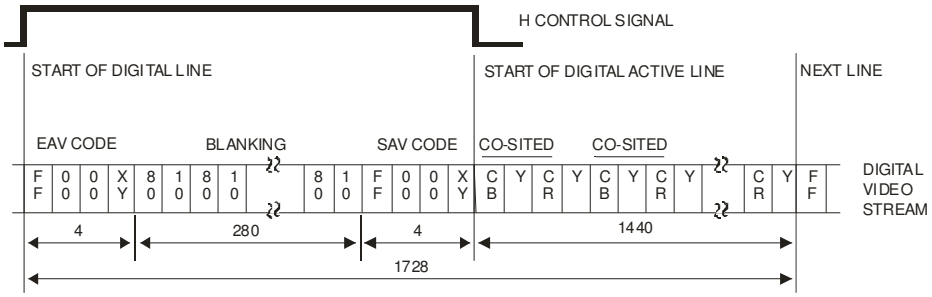
Ulazi/izlazi opšte namene (*General-Purpose Input/Output - GPIO*) pružaju posvećene priključke opšte namene koje se mogu konfigurisati bilo kao ulazi ili izlazi. Kada su konfiguriše kao izlaz, korisnik može da kontroliše stanje koje će se nalaziti na priključku. Kada se konfiguriše kao ulaz, korisnik može da detektuje stanje u kome sa nalazi priključak. Postoji ukupno 16 GPIO priključaka. Ovi priključci mogu da izazovu CPU prekid i EDMA događaj

6. ITU-R BT.656

ITU-R BT.656 preporuka definiše paralelnu i serijski spregu za transmisiju digitalnog video signala saglasnog sa 4:2:2 formatom ITU-R BT.601 preporuke. Rezolucija aktivnog video signala je 720x576 piksela za 625/50 video sisteme (npr. PAL sistem). BT.656 paralelna sprega upotrebljava 8-bitne ili 10-bitne vremenski prepletene YCbCr podatke i takt frekvencije 27MHz. Umesto da prenosi konvencionalne signale za vremenske reference (HSYNC, VSYNC i BLANK), BT.656 koristi jedinstvene vremenske kodove koji su ugrađeni u video sekvencu. Ovo redukuje broj potrebnih provodnika (u priključaka na integrisanim kolima) potrebnih za BT.656 video spregu. Pomoćne digitalne informacije (kao što su zvuk, titlovanje i teletekst) mogu se prenositi tokom intervala kada nema aktivnih video podataka (*blanking intervals*), a to su intervali kad se mlaz elektrona gasi i vraća na početak sledeće linije (*horizontal blanking interval*) ili u gornji levi ugao slike (*vertical blanking interval*). Ovo eliminiše potrebu za posebnom spregom za zvuk i dodatnim kontrolnim signalima.

6.1 YCbCr video sekvenca

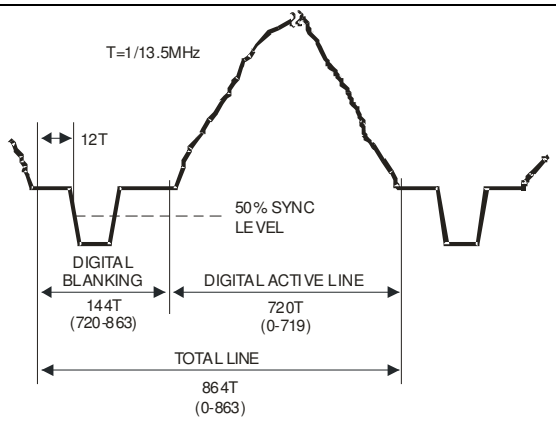
4:2:2 YCbCr podaci se preuređuju u 8-bitnu ili 10-bitnu sekvencu $Cb_0Y_0Cr_0Y_1Cb_2Y_2Cr_2$.... Slika 6.1 ilustruje raspored 8-bitnih YCbCr podataka za 625/50 video sisteme.



Slika 6.1 BT.656 8-bitni format podatka paralelne spege za 625/50 video sisteme

Posle svakog SAV koda, sekvenca aktivnih video podataka uvek počinje sa Cb vrednošću. U preuređenoj sekvenci, vrednosti koje odgovaraju istom pikselu (*co-sited*) na slici sa grupišu kao Cb,Y,Cr.

Svaka linija video signala se odabira sa učestanošću os 13.5MHz, generišući pri tome 720 aktivne 24-bitne vrednosti u formatu 4:4:4 YCbCr, kao što je prikazano na slici 6.2. Te vrednosti se konvertuju u 16-bitni 4:2:2 YCbCr format, čime se dobija 720 aktivnih Y vrednosti po liniji i 360 aktivnih vrednosti od svake Cb i Cr komponente po liniji.



Slika 6.2 Odnos BT.656 horizontlanih vremenskih referenci za 625/50 video sisteme

Y podaci i CbCr podaci se prepliću,a učestanost takta podatka se sa 13.5MHz povećava na 27MHz.

6.2 SAV i EAV vremenski kodovi

SAV (Start of Active Video) i EAV (End of Active Video) kodovi su ugrađeni unutar YCbCr video sekvence. Oni eliminišu potrebu za signalima HSYNC,VSYNC i BLANK vremenskih referenci, koji se inače upotrebljavaju u video sistemima. EAV i SAV kodne sekvence su prikazane u tabeli 1.

Statusna reč XY,koja pokazuje da li je u pitanju SAV ili EAV sekvenca, je definisana sa bitovima:

- F = 0 za polusliku; F = 1 za polusliku 2
- V = 1 tokom povratka mlaza u gornji levi ugao ekrana (VBI)
- H = 0 kada je SAV, H = 1 kada je EAV
- P3-P0 = bitovi za zaštitu

  P3 = V ⊕ H

  P2 = F ⊕ H

  P1 = F ⊕ V

  P0 = F ⊕ V ⊕ H

gde ⊕ predstavlja ekskluzivno ILI funkciju. Bitovi za zaštitu omogućavaju detekciju i korekciju jednobitnih grešaka i detekciju nekih višebitnih grešaka na strani prijemnika.

BT.656 upotrebljava intervale gašenja mlaza, koji su definisani BT.601 preporukom, kao što je prikazano na slici 3. Međutim, rezolucije manje od 720 X 576 tačaka se mogu dobiti podešavanjem mesta pojavljivanja EAV i SAV kodova i trenutka promene vrednosti bita V.

	8-BIT_DATA								10-BIT_DATA	
	D9 (MSB)	D8	D7	D6	D5	D4	D3	D2	D1	D0
Preamble	1	1	1	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
Status_Word	1	F	V	H	P3	P2	P1	P0	0	0

Tabela 6.1 BT.656 SAV i EAV sekvence

6.3 Digitalni ITU-R BT 601 4:2:2 format za analogni PAL video signal

6.3.1 Video Signal

- 3 komponente
    - luminantna Y
    - hrominentna Cb i Cr
- 25 slika u sekundi, što je 50 poluslika/s
- 625 linija po slici
- za luminentnu komponentu: 864 odbirka po liniji, što predstavlja učestanost odbiranja od 13.5 MHz.
- za hrominentnu komponentu: 432 odbirka po liniji, što predstavlja učestanost odbiranja od 6.75 MHz.
- struktura odabirnja: ortogonalna
- kodiranje odbiraka: uniformno (Pulse-Code Modulation - PCM) sa 8 bita ili 10 bita.
- kvantizacioni nivoi (za 8 bita po odbirku) i analogni signal:
    - 0 i 255 samo za sinhronizaciju
    - od 1 do 254 za video informacije.
    - luminansa: 16 = crno, 235 = belo
    - hrominansa: 128 ± 112

Rezultujući protok podataka je 216Mbps za 8-bitne odbirke ili 270Mbps za 10-bitne odbirke.

6.3.2 Aktivni video signal

Aktivni video signal je onaj deo video signala koji opisuje sliku, drugim rečima to je video signal bez intervala u kojima je mlaz ugašen (*Vertical or Horizontal Blanking Interval*)

- 576 linija po slici
- 720 odbirka po liniji za luminentnu i 360 odbiraka po liniji za hrominentnu komponentu

Rezultujući protok podatka je 165.9 Mbps za 8-bitne odbirke

7. Zaključak

U ovom radu realizovana je programska podrška video dekodera po H.264 preporuci na TMS320C6415 DSP-u. Kao rezultat dobijeno je 25 slika u sekundi u CIF rezoluciji (352 X 288). Kodirani video zapis je smešten u flash memoriji na ANDROMEDA platformi. Kompletna programska podrška realizovana je u C programskom jeziku uz upotrebu CSL funkcija uz pomoć kojih se upravlja resursima DSP-a

U okviru ove programske podrške realizovano je:

- video dekodiranje u skladu sa H.264 preporukom
- generisanje kompozitnog CVBS video signala na osnovu dekodirane slike

Video dekodiranje je realizovano na osnovu referentnog modela dekodera JM6.1e JVT-a. Izmenjen je deo koda koji su najviše uticali na performanse koda, a to je deo koda odgovoran za entropijsko dekodiranje koeficijenata transformacije t.j. dekodiranje CAVLC kodnih reči.

Generisanje kompozitnog CVBS video signala je realizovano pomoću EDMA kontrolera i dodatne ANDROMEDA Video Enkoder ploče. EDMA kontroler je upotrebljen za formiranje digitalnih video podataka koji su formirani prema ITU-R BT.656 preporuci. Video Enkoder ploča vrši formiranje analognog video signala.

Predmet daljeg proučavanja mogla bi biti realizacija programske podrške u asembleru, odnosno samo delova koda koji su kritični sa stanovišta brzine izvršavanja. Time bi se omogućila bolja optimizacija koda i bolje iskorišćenje resursa, a time i povećanje brzine izvršavanja koda za faktor 10 (optimalan slučaj). Jedana od mogućih izmena u postojećoj programskoj podršci mogla bi biti upotreba drugih sprega raspoloživih na DSP-u za prenos kodiranog video zapisa.

H.264 video dekoder je zasnovan na istim principima na kojima su zasnovani i njegovi prethodnici (H.261, H.263, MPEG-1 Visual, MPEG-2 Vsual i MPEG-4 Visual), tako da se na osnovu znanja, koja su stečena ovim projektom, može u znatno kraćem vremenskom roku izvršiti relizacija dekodera prethodno navedenih standarda. Očekuje sa da će se na osnovama ovog rada biti stvoreno jezgro za istraživanje u oblastima primene različitih algoritama u obradi video informacija.

Kao podrška ovom projektu realizovana je ANDROMEDA razvojna platforma za TMS320C6415 DSP. Ona je napravljena kao opšte namenska DSP platforma tako da ju je moguće koristiti i u drugim primema koje uključuju u sebe digitalnu obradu signala ili kompleksne matemetičke operacije koje treba izvesti u realnom vremenu.

8. Literatura

[1] *Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC),* Joint Video Team, 2003

[2] *TMS320C6201/6701 Evaluation Module User's Guide*, Texas Instruments, 1998.

[3] *TMS320C6201/6701 Evaluation Module Technical Reference*, Texas Instrumenst, 1999.

[4] *TMS320C6414, TMS320C6415, TMS320C6416 Fixed-Point Digital Signal Processors*, Texas Instruments, 2003.

[5] *256Mb: x4,x8,x16 Synchronous DRAM*, Micron Technology, 2003.

[6] *TMS320C6000 EMIF-to-External SDRAM Interface*, Texas Instruments, 2001.

[7] Am29DL322DB-90, Advanced Micro Devices, 2001

[8] *TMS320C6000 EMIF to External Flash Memory*, Texas Instruments, 2002.

[9] *PT6440 Series 6-A 5-V/3.3-V Input Adjustable ISR,* Texas Instruments, 2002.

[10] *PT6520 Series 8-A 5-V/3.3-V Input Adjustable ISR with Short-Circuit protection,* Texas Instruments, 2002.

[11] *IDT2308 3.3V Zero Delay Clock Multiplier*, Integrated Device Technology, 2003

[12] *TMS320C6000 System Clock Circuit Example,* Texas Instruments, 2001

[13] *TPS3307-33 Triple Processor Supervisor*, Texas Instruments, 2002

[14] *TMS320C6000 Board Design: Considerations for Debug*, Texas Instruments, 2002

[15] *TMS320C6000: Board Design for JTAG*, Texas Instruments, 2002

[16] *XDS510PP PLUS Parallel Port JTAG Emulator Installation Guide*, Spectrum Digital. Inc, 1999

[17] *ADV7176A High Quality, 10-Bit, Digital CCIR-601 to PAL/NTSC Video Encoder*, Analog Device, 2000

[18] *SN74V245 4096x18 DSP-SYNC FIRST-IN, FIRST-OUT MEMORY,* , Texas Instruments, 2002

[19] *Using TI FIFOs to Interface High-Speed Data Converters With TI TMS320E DSPs,* Texas Instruments, 2001

[20] *Code Composer Studio 2 Getting Started Guide*, Texas Instruments, 2001.

[21] *TMS320C6000 Chip Support Library API User's Guide,* Texas Instruments, 2001.

[22] *TMS320C6000 Peripherals Reference Guide*, Texas Instruments, 2001.

[23] *THE I 2 C-BUS SPECIFICATIO VERSION 2.1*, Philips Semiconductors, 2000