KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

PROGRAMŲ INŽINERIJOS KATEDRA

Eugenijus Kriščiūnas

ATVIROJO KODO VAIZDO KODAVIMO H.264 REALIZACIJŲ, APRAŠYTŲ APARATŪROS APRAŠYMO KALBOMIS, TYRIMAS

Magistro darbas

Darbo vadovas:

prof. Vacius Jusas

KAUNAS, 2012

KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

PROGRAMŲ INŽINERIJOS KATEDRA

ATVIROJO KODO VAIZDO KODAVIMO H.264 REALIZACIJŲ, APRAŠYTŲ APARATŪROS APRAŠYMO KALBOMIS, TYRIMAS

Magistro darbas

Recenzentas:

lek. Darius Matulis
2012 m. gegužės d.

Darbo vadovas:

prof. V.Jusas
2012 m. gegužės d

Atliko:
IFM-0/5 grupės studentas
Eugenijus Kriščiūnas
2012 m. gegužės 23 d.

KAUNAS, 2012

SUMMARY

Video coding is the entire process of compressing and decompressing of a digital video signal. Then the mainstream video compression tools developed in the past several years by both Video Coding Experts Group (VCEG) and Moving Picture Experts Group (MPEG) are briefly introduced. Since its invention from early 1990 modern digital video compression techniques have played an important role in the world of telecommunication and multimedia systems where bandwidth is still a valuable commodity. Evolution from the early MPEG-1/H.261 to the current H.264/AVC video codec gradually improves the coding efficiency at the cost of design complexity. Compared with its prior standards the H.264/AVC is able to achieve nearly doubled coding gain, while the encoder's and decoder's complexity increase 5 – 10 and 2 – 3 times respectively.

SVARBIŲ TERMINŲ ŽODYNĖLIS

FPGA (Field Programmable Gate Array) - viena iš įterptinių sistemų rūšių,suteikianti galimybė patikrinti lustą prieš paleidžiant jį į gamybą.

HDL (Hardware Description Language) - aparatūros aprašymo kalbos.

VHDL (VHSIC Hardware Description Language) - viena iš aparatūros aprašymo kalbų.

Verilog - viena iš aparatūros aprašymo kalbų.

RTL (Register-Transfer Level) - abstrakcijos lygis naudojamas aprašant sinchroninės skaitmeninės grandinės veikimą.

RAM (Random Access Memory) - operatyvioji atmintis.

AVC (Advanced Video Codec) – Pirmaujantis standartas, skirtas besikeičiantiems vaizdams glaudinti.

CABAC (Context-Adaptive Binary Arithmetic Coding) – Aritmetinis dvejetainis glaudinimo būdas, prisitaikantis prie konteksto.

ITU-T (International Telecommunication Union) – Tarptautinė telekomunikacijų sąjunga.

H.264 - Pirmaujantis standartas, skirtas besikeičiantiems vaizdams glaudinti.

ISO (International Organization for Standardization) – Tarptautinė standartizavimo organizacija.

MPEG (Moving Picture Experts Group) – standartas, skirtas besikeičiantiems vaizdams glaudinti.

PSNR (Peak Signal to Noise Ratio) – Pikinis signalo ir triukšmo santykis

TURINYS

PAVEIKSLŲ SĄRAŠAS

LENTELIŲ SĄRAŠAS

1. ĮVADAS

H.264 yra šiuo metu vienas iš naujausių ir populiariausių vaizdo kodavimo standartų. Palyginus su ankstesniais koduojančiais standartais, jis sugeba pateikti aukštesnės kokybės vaizdą duotam suspaudimo lygiui, ir geresnį suspaudimo lygį tai pačiai vaizdo kokybei. Dėl tos priežasties, H.264 yra panaudotas daugelyje įrenginių, įskaitant HD-DVD, Blu-ray, iPod video, HDTV transliacijos ir įvairiose kompiuterių aplikacijose. Vienas iš populiariausio H.264 panaudojimo būdu yra aukšto raiškumo vaizdo turinio kodavimas. HD vaizdas taip pat aplikacija, kuriai reikalingas didelis dekoderio našumas. Standartiniai HD formatai, apima 720p (1280x720) ir 1080p (1920x1080) rezoliucijas, su kadrų skaičiumi tarp 24 ir 60 kadrų per sekundę.

H.264/AVC, išvystytas ITU-U Video, Video Coding Experts Group (VCEG) ir ISO/IEC Moving Picture Experts Group (MPEG), yra pažangiausias vaizdo kodavimo standartas, pasiekiamas šiandien. H.264/AVC tikslas yra aprūpinti gerą video kokybę esant mažesnei perdavimo spartai negu ankstesni standartai tokie kaip MPEG-2 ir H.263, nenaudojant sudėtingo projekto įgyvendinimo. Tuo metu, kai kodavimas ir procesų iššifravimas yra panašūs į ankstesnius standartus, kai kurios ypatybės, kurios įgalina pranašumą buvo sustiprintos. Šiuo metu, mobilūs vaizdo produktai, kurie priima šį standartą valdo rinką.

Tyrimo tikslas yra išanalizuoti jau esamus H.264 metodus, išsiaiškinti jų pagrindinius bruožus bei galimybes.

2. ANALITINĖ DALIS

2.1. Vaizdo suspaudimo sąvokos

2.1.1. Vaizdo suspaudimas

Nesuspausti skaitmeniniai vaizdo, garso ir foto duomenys užima didelę dalį informacijos saugojimo įrenginių vietos bei sunaudoja didžiąją informacijos srautų dalį internete. Nors nuolatos sparčiai didėja pastoviosios atminties įrenginių apimtys, procesorių skaičiavimo greičiai ir duomenų perdavimo sparta internetu, reikalavimai saugojamų duomenų dydžiui ir perdavimo greičiui viršija esamų technologijų galimybes. Šiuo metu vis populiarėjančios internetinės programos bei svetainės, naudojančios didelius kiekius skaitmeninės informacijos, dar labiau sustiprina efektyvių algoritmų ir metodų poreikį, skirtą tokio pobūdžio duomenims suspausti ir tuo pačiu išspręsti didelių duomenų

kiekio saugojimo, apdorojimo ir platinimo internetu problemas. Galima išskirti dvi pagrindines duomenų suspaudimo algoritmų klases – suspaudimo algoritmai be informacijos praradimo (angl. lossless) ir algoritmai su duomenų praradimu (angl. lossy) [1]. Naudojant algoritmus be informacijos praradimo, rekonstruotas vaizdas yra skaitiškai identiškas originaliam vaizdui. Tačiau, naudojant šio tipo algoritmus pasiekiamas tik nežymus duomenų suspaudimo lygis. Todėl suspaudimo algoritmai be duomenų praradimo naudojami tose srityse, kur duomenų dydžiai yra aktuali problema, tačiau bet koks duomenų praradimas yra nepageidaujamas – kaip pavyzdžiai gali būti medicinos ir astronomijos mokslai, kur atkurtų vaizdų kokybė yra ypač svarbi. Kai naudojami algoritmai su duomenų praradimu, atkurtas vaizdas netiksliai atkartoja originalų vaizdą. Informacijos praradimas įvyksta todėl, kad suspaudimo metu yra visiškai atmetama perteklinė (mažiausiai svarbi) informacija. Šis būdas leidžia daug didesnius duomenų suspaudimo lygius. Net ir esant skaitiniams duomenų praradimams, dažnai rekonstruotas ir pradinis vaizdas yra vizualiai identiški (angl. visually lossless).

2.1.2. Duomenų suspaudimo metodų poreikis

Vis labiau didėjanti kompiuterinės technikos ir internetinių technologijų sparta leidžia vis labiau naudoti vaizdo, garso ir foto informacijų. Nepaisant dabartinių kompiuterių pastoviosios atminties dydžių bei interneto ryšio spartos, daugeliu atveju neįmanoma saugoti ar tuo labiau internetu platinti nesuspaustų foto ar vaizdo medžiagų.

Filmuota medžiaga saugoma kaip vienas po kito einantys kadrai (nuotraukos). Panagrinėkime pavyzdį, kuris gerai parodo duomenų suspaudimo metodų poreikį. Tarkime, kad vartotojas naudojasi interneto ryšiu, kurio maksimali sparta yra 56 Kbit/s (kilobitai per sekundę). Klausimas – kiek laiko užtruktų vaizdo, išsaugoto nesuspaustu formatu ir trunkančio 90 minučių, siuntimas? Tarkime, kad vaizdo dydis yra 240x320 taškų. Tada vienas vaizdo kadras užimtų apie 1.8 Mbit (megabitų). Kadangi per sekundę parodoma 30 kadrų, tai 1 sekundės filmuota medžiaga užima apie 52.7 Mbit. Tada 90 minučių vaizdo medžiaga užimtų 278.1 Gbit (gigabitų). Esant pastoviam siuntimo greičiui, lygiam 56 Kbit/s, vartotojas vaizdą siųstų maždaug 60 dienų! Reiktų atkreipti dėmesį į tai, kad garsinė informacija nebuvo įskaičiuota, be to, retai siuntimas vyksta pastoviu maksimaliu greičiu. Tad realiai siuntimo ilgis padidėtų dar bent du kartus.

1 lentelė. Skirtingų duomenų dydžiai ir siuntimo laikai [1]

Duomenys	Dydis / trukmė	Bitai / pikseliui	Nesuspaustų duomenų dydis	Siuntimo laikas, naudojant 28.8 Kbit/s modemą
Puslapis teksto	A4 formatas	–	apie 50 Kbit	1-2 sek.
Telefoninės kokybės pokalbis	10 sekundžių	8	640 Kbit	22 sek.
Pilkos skalės nuotrauka	512×512 taškų	8	2 Mbit	1 min 13 sek.
Spalvota nuotrauka	512×512 taškų	24	6 Mbit	3 min. 39 sek.
Medicininė nuotrauka	2048×1680 taškų	12	41 Mbit	23 min 54 sek.
Filmuota medžiaga	2048×1680 taškų, 1 minutės trukmės	24	221 Mbit	5 dienos 8 val.

1 lentelė aiškiai parodo, koks didelis yra saugojimo vietos, interneto spartos ir laiko poreikis garso bei vaizdo duomenims. Esant šiuolaikiniams techniniams pasiekimams, duomenų suspaudimas prieš saugojimą ar siuntimą yra vienintelė įmanoma išeitis.

2.1.3. Duomenų suspaudimo principai

Bendra didžiosios dalies vaizdų savybė yra ta, kad gretimi vaizdo taškai koreliuoja tarpusavyje, vadinasi yra perteklinės informacijos. Tokiu atveju, pagrindinė užduotis yra surasti mažiausiai besikoreliuojantį pradinio vaizdo atvaizdą. Dveji fundamentalūs suspaudimo principai yra pertekliškumo (angl. redundancy) ir nereikšmingumo (angl. irrelevancy) mažinimas [1].

Mažinant pertekliškumą, siekiama pašalinti besidubliuojančias duomenų signalo (vaizdo) reikšmes. Nereikšmingumo mažinimo metu iš duomenų signalo yra pašalinamos tos reikšmės, kurios nebus pastebėtos signalo gavėjo. Kalbant apie vaizdus, tą įmanoma padaryti įvertinus žmogaus regos sistemos (angl. human visual system) savybes. Galima išskirti tokius pertekliškumo tipus:

1) statistinis pertekliškumas
2) erdvinis pertekliškumas (angl. spatial redundancy)
3) kodavimo pertekliškumas
4) vizualinis pertekliškumas (angl. psychovisual redundancy)
5) spalvinis pertekliškumas

6) dažninis pertekliškumas

Erdvinis pertekliškumas atsiranda todėl, kad pikselių reikšmės nėra tarpusavyje nepriklausomos. Paprastai yra labai didelė gretimų taškų koreliacija.

Spalvinį (intensyvumo) pertekliškumą nusako Vėberio dėsnis (angl. Weber‘s law): dirgiklis turi didėti tam tikra proporcija, kad sukeltų tiesinį atsako padidėjimą [2]. Šiuo atveju, koduojant yra išnaudojama žmogaus regos savybė, kad žmogus į šviesos pokytį reaguoja ne tiesiškai, bet logaritmiškai.

Dažninis pertekliškumas atsiranda taip pat dėl žmogaus regos savybių. Žmogaus akis veikia kaip žemo dažnio filtras, todėl spaudžiant vaizdą galima panaikinti aukšto dažnio harmonikas ir tai liks nepastebima žmogaus akimi. Tai bene labiausiai išnaudojama savybė vaizdų suspaudimo algoritmuose. Kaip pavyzdys gali būti hiperbolinio filtravimo algoritmai, kurie kodavimo metu visiškai pašalina aukščiausius dažnius ir taip pasiekiamas nemažas suspaudimo lygis, nesant dideliems kokybės pokyčiams.

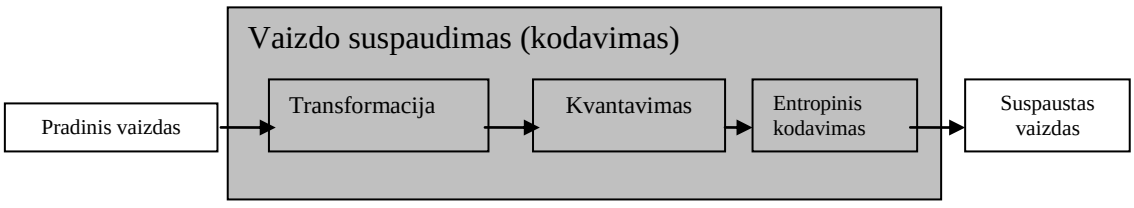
2.1.4. Tipinė vaizdo suspaudimo schema

Tipinė vaizdo suspaudimo su informacijos praradimu schema pavaizduota 1 paveiksle. Schema sudaryta iš trijų tarpusavyje susijusių komponentų – vaizdo kodavimo komponento, kvantizacijos komponento ir entropinio kodavimo komponento. Suspaudimo efektas pasiekiamas tokiu būdu:

1) Pradinis vaizdas tiesiškai transformuojamas, kad sumažinti duomenų tarpusavio koreliaciją.

2) Gauti tranformacijos koeficientai kvantuojami.

3) Pritaikomas entropinis kodavimas.



1 pav. Tipinė vaizdo suspaudimo schema

Vaizdo kodavimo (arba tiesinio transformavimo) algoritmų sukurta daug ir įvairių, kiekvienas turintis savų privalumų bei trūkumų. Galima paminėti tik kelis labiausiai paplitusius transformacijų algoritmus: DFT (angl. Discrete Fourier Transform) [3], DCT (angl. Discrete Cosine Transform) [4], DWT (angl. Discrete Wavelet Transform) [5].

Kvantavimo metu yra sumažinamas bitų skaičius, reikalingas transformuotiems koeficientams išsaugoti, mažinant saugojamų reikšmių tikslumą. Kadangi tai yra atvaizdavimas daug-su-vienu (angl. many-to-one mapping), tai šio proceso metu yra prarandama informacija. Tačiau šio procesu metu yra pasiekiama didžioji vaizdo kompresijos dalis. Kvantizacija gali būti atliekama kiekvienai reikšmei atskirai (skaliarinė kvantizacija) arba atliekama tam tikrai koeficientų grupei (vektorinė kvantizacija).

Entropinio kodavimo metu toliau spaudžiamos kvantizuotos reikšmės, naudojant suspaudimo algoritmus be informacijos praradimo, siekiant dar labiau padidinti vaizdo suspaudimo lygį. Labiausiai paplitę entropinio kodavimo algoritmai yra: Hufmano kodavimas, aritmetinis kodavimas, bitinis kodavimas (angl. run-length coding), žodyno kodavimas [6].

2.2. Populiarių vaizdo suspaudimo standartų apžvalga

2.2.1. Senesnių kodavimo standartų apžvalga

Prieš pristatydamas H.264/AVC trumpai praminėsiu senesnius vaizdo kodavimo strandartus tokius kaip MPEG-1, 2, 4 bei H.261, H.262, H.263.

Pirmasis standartas, kuri pristatė MPEG yra žinomas kaip MPEG-1, buvo panaudotas kompaktinių disku grotuvuose vaizdo bei garso suspaudimui ir atkūrimui. MPEG-1 pasiekimas buvo suspausti vaizdą bei garsą iki 1.4Mbps tai kokybė, atitinkantis VHS video įrašą.

MPEG-2 naudojamas skaitmeninės televizijos transliavimui ir standartinės raiškos bei aukštos raiškos. Šis standartas buvo suprojektuotas, kad apimtų MPEG-1 ir taip pat suteiktų aukštos kokybės vaizdo šaltinius, kuriu kokybė svyruoja nuo 4Mbit/s iki 30Mbit/s. H.262 yra tas pats kaip ir MPEG-2 tiktais tai yra oficialus jungtinis projektas, išvystytas ISO ir ITU-T organizacijų.

MPEG-4 standartas buvo sukurtas, kad praplėstu prieš tai buvusiu standartų galimybes. Jis palaiko nedidelės spartos aplikacijas, kur MPEG-1 ir MPEG-2 yra efektyvūs. Jis taip pat palaiko objektinį kodavimą, kur vaizdo scena gali būti traktuojama kaip priekinio plano ir foninių objektų komplektas, o ne tik reguliariais blokais paremtomis kadrų serijomis. Kaip palyginimui, MPEG-1 turi labai apribotą lankstumo laipsnį; MPEG-2 įvedė "toolkit" sąvoką, kur skirtingi profiliai ir lygmenys gali būti sukombinuoti įvairioms aplikacijoms; MPEG-4 suteikia lanksčius kodavimo įrankius, kurie leidžia panaudoti standartizuotą struktūrą bei įterpti naujus įrankius palaipsniui.

H.261 buvo sukurtas tuo metu, kai aparatinės ir programinės įrangos galimybės buvo apribotos, todėl yra pranašesnis, nes nėra labai sudėtingas. Perdavimo sparta svyruoja nuo 64 kbps iki 384 kbps. Tačiau, jo trūkumai yra bloga suspaudimo kokybė ir lankstumo trūkumas. Tai

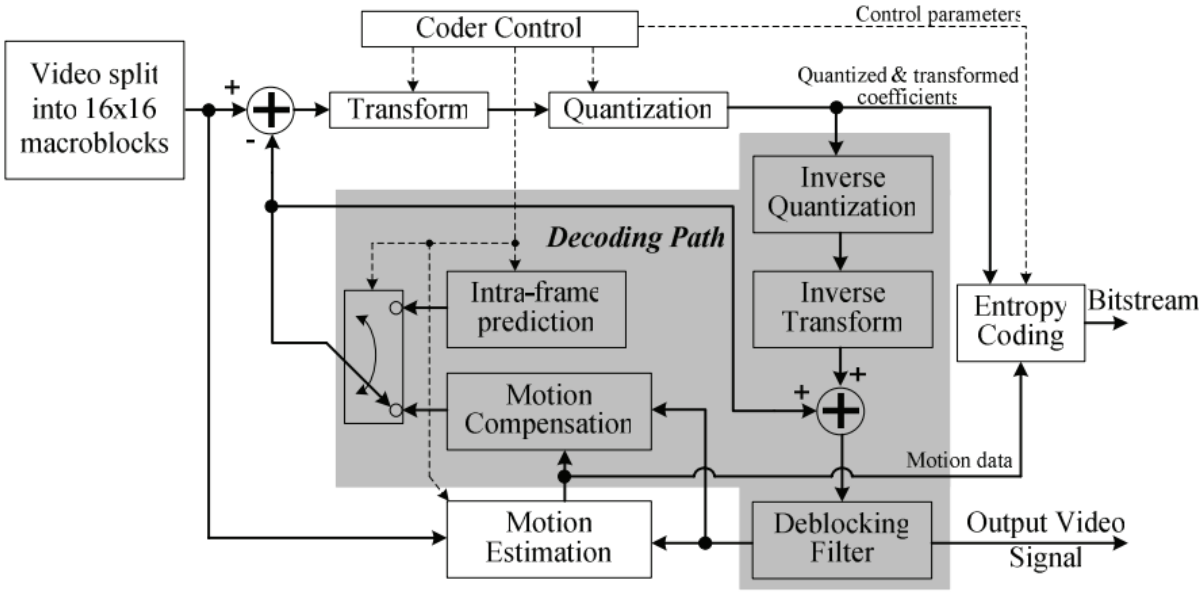
patobulino jo paveldėtojas, H.263, kuris turi geresnį suspaudimo efektyvumą ir didesnį lankstumą, bet yra vis dar naudojamas vaizdo konferencijų sistemose.

Palyginus H.261 ir H.263 pastarasis turi daug pranašumų. Suteikiamas geresnis suspaudimas, panaudojant geresnį kodavimo algoritmą. Taip pat pateikią geresnį lankstumą. H.263 tikslas yra žemos perdavimo spartos, mažo uždelsimo dvipusis vaizdo ryšys. H.263 gali palaikyti žemesnį negu 20 kbps perdavimo spartos vaizdo ryšį, net ir dabar yra plačiai naudojamas vaizdo telefonijoje bei konferencijose, taip pat vaizdo apdorojimo programose, kurios skirtos patalpinti vaizdo medžiagą internete.

2.2.2. H.264/AVC standartas

H.264/AVC yra naujausias tarptautinis vaizdo kodavimo standartas [7]. Jis apima Video Coding Layer (VCL) ir Network Adaptation Layer(NAL). Nors bendra VCL struktūra yra panaši į ankstesnius koduojančius video standartus kaip MPEG-2, H.263, MPEG-4 ji pateikia kelias naujas ypatybes, kurios įgalina tai pasiekti reikšmingą naudą kodavime palyginus su pirmtakais.

H.264/AVC kodekas naudoja tradicinį bloku paremtą mišrų vaizdo kodavimo metodą, kuris buvo jau panaudotas anksčiau minėtuose standartuose. Pagrindinis kodavimo algoritmas yra inter vaizdo apskaičiavimas ir transformacijos kodavimo likučių apskaičiavimo mišinys. H.264/AVC nepateikia nei vieno kodavimo elemento, kuris ženkliai būtu patobulintas palyginus su ankstesniais vaizdo kodavimo standartais. Tai daugiau yra maži patobulinimai, kurie suteikia esminį patobulinimą.



2 pav. Makrobloko mišri vaizdo kodavimo struktūra [7]

Įvesties vaizdo signalas yra padalytas į makroblokus. Kiekvienas makroblokas yra numatytas kaip intra ar inter apskaičiavimo (judėjimo kompensacija). Likučiai yra pertvarkomi ir kvantuojami. Visiems kontrolės parametrams, koeficientams, taip pat judėjimo vektoriams yra panaudojamas entropijos kodavimas Egzistuoja du duomenų srauto keliai, priekinis kelias (iš kairės į dešinę) ir iššifravimo kelias. Užšifruotojas netik užkoduoja ir perduodantis kiekvieną bloką į makrobloką, bet taip pat atstato, kad pateiktų informaciją tolimesniems skaičiavimams. Tai yra iššifruota originalaus bloko versija turinti tam tikrus mažus skirtumus palyginti su neapdirbtais vaizdo elementais. Galiausiai, filtras yra pritaikytas, kad sumažinti blokinio iškraipymo padarinius.

H.264/AVC panaudoja 4x4 blokus skirtus transformaciniam kodavimui, taip gaunamas makroblokas, susidedantis iš 16 luma ir 8 chromą 4x4 blokų. Makroblokas gali būti koduojamas viename iš kelių INTRA rėžimų, kurie yra žymimi kaip Intra_4x4 ir Intra_16x16. Be intra rėžimų, H.264/AVC taip pat pateikia gausų pasirinkimą judesio kompensacijos rėžimų, skirtų makroblokams P/B pjūviuose. Kiekvienas judesio rėžimas atitinka fiksuoto dydžio blokus, skirtus judėjimui apibūdinti. Makrobloko dydis iš luma komponentų, padalytas į 16x16, dvi dalys 16x8 arba 8x16, ir keturios dalys 8x8, yra palaikomos. Tie blokai, kurie padalyti į 8x8, kiekvienas 8x8 blokas gali būti toliau padalytas į mažesnius blokus 8x8, dvi dalys 4x8 arba 8x4,ar keturios 4x4 dalys. Kalbant apie chroma komponentus, jeigu 4:2:0 diskretizavimo dažnis yra pritaikomas, kiekvienas chroma blokas yra tiktais vienas savo luma partnerio ketvirtis, pavyzdžiui 8x4 luma blokas atitinka 4x2 chroma bloko, 4x8 luma blokas atitinka 2x4 chroma bloko ir taip toliau.

Judėjimo kompensavimas yra įvykdytas su judėjimo vektoriais. Judėjimo vektoriaus komponentai yra skirtingai užkoduoti naudojant ar vidurinį arba kryptinį prediction nuo kaimyninių blokų.

H.264/AVC yra iš esmės panašus į ankstesnius standartus, kadangi jis taip pat naudoja, transformacijos kodavimą klaidos signalo nustatymui. Tačiau, H.264/AVC transformacija yra įvykdyta kiekviename 4x4 bloke. Mažesnis bloko dydis lemia reikšminga artefaktų sumažėjimą. Be to, vietoj tradicinės 8x8 diskrečiosios kosinusu transformacijos (DCT), jis naudoja atskiriamąją sveikąjų skaičių tranformaciją su tiksliomis sveikūjų skaičių operacijomis. Todėl, atvirkštinės transformacijos neatitikimai yra visiškai išvengti. Papildomas 4x4 transformacija yra pritaikyta 16 DC koeficientų kiekvienam Intra_16x16 metodui, taip pat kaip 2x2 transformacija 4 DC koeficientams kiekvienam chroma 8x8 blokui.

Transformuotų koeficientų kvantavimui H.264/AVC naudoja skaliarinį kvantavimą vietoj vektoriaus kvantavimo. Vienas iš 52 kvantavimo parametrų yra atrenkamas kiekvienam makroblokui. Šis platus diapazonas užšifruotojui leidžia tiksliai ir lanksčiai rasti kompromisą tarp perdavimo spartos ir kokybės. Luma ir chroma komponentai gali parinkti skirtingus parametrus kai

to reikia. Priekinis ir atvirkštinis kvantavimo mechanizmas yra sudėtingi reikalavimai, kad išvengti padalijimo ir slankaus kablelio aritmetikos.

H.264/AVC palaiko du entropijos kodavimo metodus. Sintaksės elementai be likutinių duomenų yra užkoduojam. Transformacijos likutinių koeficientu nuskaitymui panaudotas sudėtingesnis metodas, vadinamas Adaptyviu kontekstu Užkoduojančiu Kintamu Ilgiu (CAVLC). Ši schema iš esmės panaudoja ta pačią idėją, kuri sutinkama ankstesniuose MPEG-2, MPEG-4, ir H.263 standartuose. Skirtumas yra H.264/AVC VLC lentelėse įvairiems sintaksės elementams, kurie yra adaptyviai perjungiami priklausomai nuo anksčiau perduotų sintaksės elementų verčių. Kadangi visos VLC lentelės yra gerai suprojektuotos, kad atitiktų atitinkamą sąlyginę statistiką, iš esmės entropijos yra pagerintas kodavimo efektyvumas. Entropijos kodavimo efektyvumas gali būti pagerintas, jeigu yra panaudotas CABAC.

H.264/AVC naudoja ciklinį deblocking filtrą, kad sumažintų blokavimo artefaktus, pateiktus paveiksle. Filtruoti paveikslai yra panaudoti, kad apskaičiuotu judėjimą kitiems paveikslams. Deblocking filtras yra adaptyvus, kuris pats reguliuoja stiprumą, priklausantį nuo makrobloko suspaudimo būdo (Intra, ar Inter), kvantizacijos parametrus, judėjimo vektorius, ir pikselių reikšmes. Kai kvantizacijos dydis yra sumažintas, filtro efektyvumas irgi yra sumažintas, ir kai kvantizacijos dydis yra labai mažas, filtras yra išjungiamas. Filtras taip pat gali būti išjungtas arba pritaikytas pjūvių lygmens šifravime.

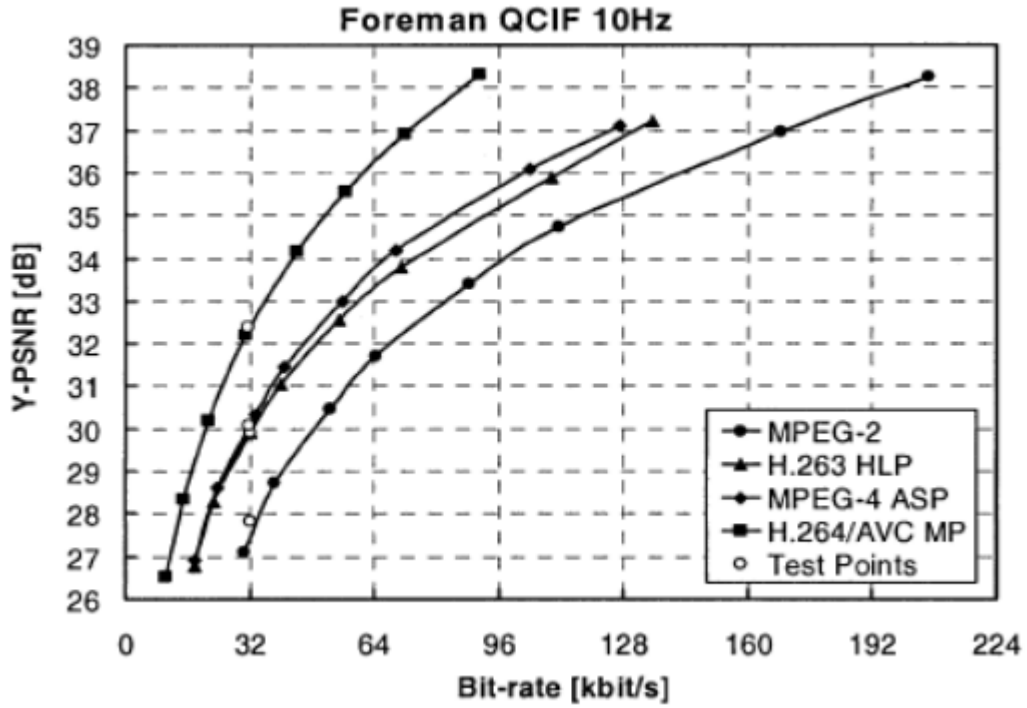
Apskritai, H.264/AVC yra pagrįstas tradicinėmis vaizdo kodavimo sąvokomis, bet palyginti su ankstesniais standartais, su svarbiais tam tikrais skirtumais. Reikšmingiausi skirtumai yra padidintas judėjimo apskaičiavimo pajėgumas, mažas bloko dydis su tikslia transformacija, adaptyvus deblocking filtras ir patobulinti entropijos kodavimo metodai. Skirtingų vaizdo standartų palyginimas yra pateiktas 2 lentelėje.

2 lentelė. Vaizdo kodavimo standartų palyginimas

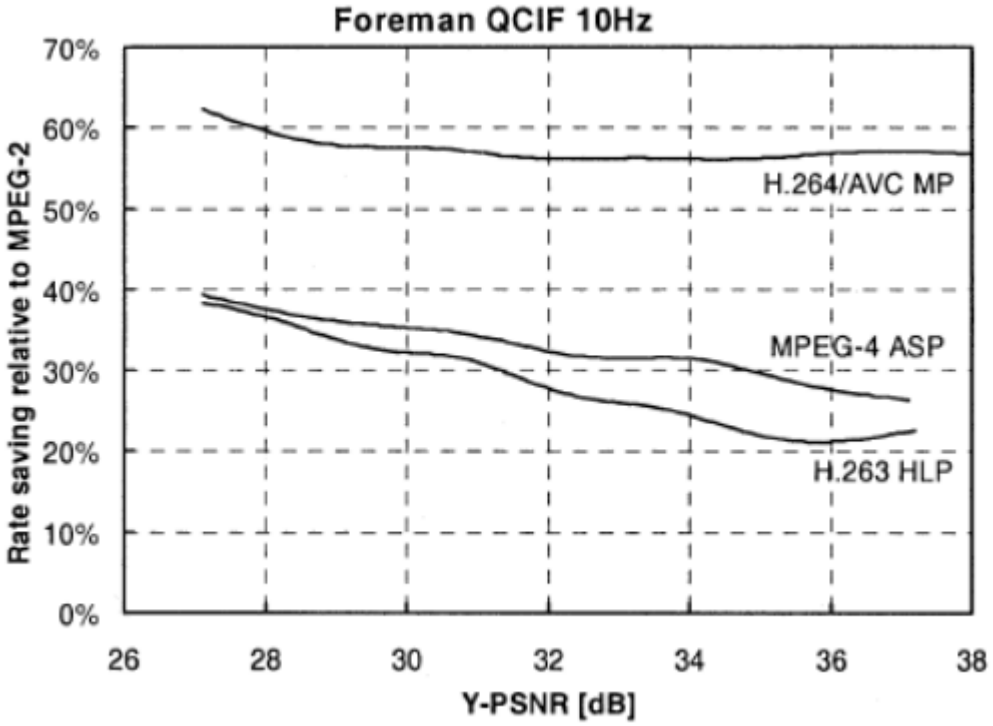
	MPEG-1	MPEG-2	MPEG-4	H.261	H.263	H.264/AVC
Macroblock size	16x16	16x16	16x16	16x16	16x16	16x16
Possible block sizes	16x16	16x16 8x8	16x16 8x8	16x16	16x16	16x16, 16x8, 8x16, 8x8, 4x8, 8x4, 4x4
Transform	8x8 DCT	8x8 DCT	8x8 DCT	8x8 DCT	8x8 DCT	4x4/8x8 integer DCT
Motion compensation accuracy	1, 1/2 pixels	1, 1/2 pixel	1, 1/2, 1/4 pixel	1 pixel	1, 1/2 pixel	1, 1/2, 1/4 pixel
Prediction	Bi-direction	Bi-direction	Bi-direction	One direction	Bi-direction	Bi-direction
In-loop deblocking filter	No	No	No	No	No	Yes
Entropy coding	VLC	VLC	VLC	VLC	VLC	CAVLC, CABAC

2.2.3. Kodavimo naudos įvertinimas

Paveiksle 3 pateikta tos pačios vaizdo sekos, užkoduotos skirtingais standartais, kokybės iškraipymo kreivės[8]. PSNR (Peak Signal-to-Noise-Ratio) yra objektyvus vaizdo kokybės matavimas. Kuo didesnis PSNR, tuo mažesnis skirtumas tarp užkoduoto ir originalaus vaizdo, ir tuo geresnė kokybė. Kaip pastebima, H.264/AVC įgyja didesnį PSNR , palyginus su kitais standartais Esant tam pačiam PSNR lygmenyje, užtenka mažesnės užkodavimo spartos, kuria pateikia H.264/AVC.



3 pav. Kokybės iškraipymo palyginimas [8]



4 pav. Perdavimo spartos sutaupymas [8]

Paveikslas 4 pateikia santykinius kirtingų standartų, spartos sutaupymus palygintų su MPEG-2 [8]. H.264/AVC pasiekia daugiau kaip 50 %, kodavimo prieaugį palyginus su MPEG-2 visuose PSNR situacijose.

PSNR yra patogus vaizdo iškraipymui nustatyti, tačiau neapgrendžia žmogaus regimosios sistemos sudėtingumo. Palyginimo rezultatai tarp MPEG-2 ir H.264/AVC pateikia tai, kad perdavimo spartos sutaupymas yra didesnis, kai yra panaudotas subjektyvus testas , o ne PSNR matmenys. Net ir sumažinus 70% perdavimo spartą H.264/AVC sugeba pateikti tokios pat kokybės vaizdą kaip MPEG-2, esant didelei perdavimo spartai, H.264/AVC gali būti pasiekti iki 40 % sutaupyti perdavimo.

Išvada, kad H.264/AVC standartas dirba žymiai geriau už visus ankstesnius standartus, dėl padidėjusio kodavimo lankstumo ir sudėtingumo.

2.2.4. Sudėtingumo analizė

H.264/AVC standarto tikslas yra dvigubinti kodavimo efektyvumą palyginus su bet kokiais kitais egzistuojančiais vaizdo kodavimo standartais. Šis pagerintas kodavimo efektyvumas pasiekiamas su žymiai padidėjusiu kodeko sudėtingumu. Aparatinės įrangos ir energijos sąnaudų reikalavimai kodavimui ir iššifravimui, taip pat žymiai padidėjo.

3 lentelė. QCIF iššifravimo įvertinimas atliktas ARM platformoje [9]

Video sequence	Bit rate	Software only	Software + hardware
Container	13.15 kbps	6.8 fps	8.6 fps
Akiyo	8.67 kbps	8.1 fps	10.3 fps
Coastguard	58.69 kbps	2.3 fps	3.6 fps
Mother daughter	14.43 kbps	4.5 fps	6.8 fps
foreman	42.47 kbps	2.4 fps	3.7 fps
Average	27.48 kbps	4.8 fps	6.6 fps

3 Lentelė pateikia H.264/AVC atliktą iššifravimą ant dekoderio platformos, kuri turi ARM966 centrinį procesorių (pasiekiantį 130 MHZ), integruotus SRAM, daugialypius paskirtus akceleratorius, išorinę atminties sąsają, taip pat kaip 32 bitų pločio AHB magistralę, jungiančią visus funkcijos blokus [9]. Kaip galima pamatyti iš 3 lentelės, aparatinės įrangos pagreitis palyginus su programinės įrangos yra 37.5 % (6.6/4.8 kadrų per sekundę), ir net po paspartinimo, iššifravimo platforma negali iššifruoti QCIF vaizdo sekos, didesne negu 10 kadrų per sekundę sparta.

CIF 352x288 raiška esant 25 kadrų per sekundę

4 lentelė. CIF atlikto iššifravimo analizė [9]

Video sequence	Percentage of CPU usage		
	125 kbps	500 kbps	1000 kbps
	4.4% CPU	6.5% CPU	8.7% CPU
	4.5% CPU	7.8% CPU	10.2% CPU
	6.7% CPU	10.2% CPU	13.3% CPU
	6.2% CPU	9.8% CPU	12.6% CPU
Average	5.45% CPU	8.58% CPU	11.2% CPU

Lentelė 4 pateikia procesoriaus Pentium4 3GHz apkrovos matavimus paleidus kelias CIF vaizdo kokybės iššifravimo sekas su skirtingais apribojimais. Pentium4 panaudojo 5 % ~ 15 % savo apkrovos atliekant CIF 25fps iššifravimą. Galima teigti, kad iššifravimo sudėtingumas yra proporcingas vaizdo dydžiui ir atvaizdavimo dažniui. Jei vaizdo taikymas reikalauja HDTV raiškos esant 50/60Hz, net Pentium4 nepajėgia iššifruoti vaizdo sekų realiu laiku.

Palyginti su MPEG-4, šifratoriaus sudėtingumas padidėjo nuo 5 iki 10 kartų tuo metu, kai dekoderio sudėtingumas padidėjo 2–3 kartais. Kai vaizdo raiška pakyla iki HDTV lygmens, pavyzdžiui, HDTV 1024p (2048 x 1024, 30fps), H.264/AVC iššifravimo procesui, reikalingos 83 GIPS (Giga Instrukcijos Per Sekundę) ir 70 GBPS (Giga Baitas Per Sekundę) atminties prieiga. Beveik neįmanoma vienam mikroprocesoriui taip greitai veikti, kad sugebėtu aprūpinti pakankamai realaus laiko skaičiavimo pajėgumo ir atminties kreipimosi.

5 lentelė. H.263 ir H.264/AVC palyginimas [10]

Video Sequence	H.263 baseline			H.264/AVC baseline			H.263/H.264 speed ratio
	QP	Bit rate (kbps)	Decoding speed (fps)	QP	Bit rate (kbps)	Decoding speed (fps)	
Container QCIF 10Hz	6	61	710	11	65	300	2.37
	9	30	900	15	32	390	2.31
Foreman QCIF 10Hz	13	62	580	18	66	230	2.52
	22	34	670	24	32	280	2.39
Silent QCIF 10Hz	9	61	750	16	64	330	2.27
	15	32	880	22	29	390	2.26
Paris CIF 15Hz	5	672	140	11	677	67	2.09
	10	298	183	17	313	82	2.23
	21	111	220	24	116	100	2.20
Foreman CIF 30Hz	9	652	149	14	680	56	2.66
	14	342	192	19	326	67	2.87
	31	149	225	28	120	85	2.65
Mobile & Calendar CIF 30Hz	12	1842	96	16	1723	44	2.18
	23	699	130	21	704	55	2.36
	31	443	155	26	307	65	2.38
Average							2.4

Lentelėje 5 yra palyginami iššifravimo greičių santykiai tarp H.263 ir H.264/AVC, išmatuoti su Pentium3 mikroprocesorium [10]. Visoms šešioms užkoduotoms vaizdų sekoms,iššifravimo greičio santykis panašaus ilgio duomenų srauto tarp H.263 ir H.264/AVC yra diapazone nuo 2.1 iki 2.9 vidurkis 2.4. Tai įrodo, kad bazinio H.264/AVC iššifravimo sudėtingumas yra nuo dviejų iki trejų kartų didesnis negu H.263 duotai sekai, užkoduotai tam tikra perdavimo sparta. Reikia atkreipti dėmesį, kad esant tai pačiai perdavimo spartai, vaizdai užšifruoti su H.264/AVC, turi esminius regimuosius kokybės pagerinimus palyginti su H.263. Kitaip tariant, palyginti su H.263, kad H.264/AVC pasiektų beveik tą pačią vaizdo kokybę, perdavimo sparta galima sumažinti nuo 35 % iki 50 %.

2.3. Išvados

H.264/AVC yra pagrįstas tradicinėmis vaizdo kodavimo sąvokomis, bet palyginti su ankstesniais standartais, su svarbiais tam tikrais skirtumais. Reikšmingiausi skirtumai yra padidintas judėjimo apskaičiavimo pajėgumas, mažas bloko dydis su tikslia transformacija, adaptyvus deblocking filtras ir patobulinti entropijos kodavimo metodai. palygintų su MPEG-2. H.264/AVC pasiekia daugiau kaip 50 %, kodavimo prieaugį palyginus su MPEG-2 visuose PSNR situacijose. Išvada, kad H.264/AVC standartas dirba žymiai geriau už visus ankstesnius standartus, dėl padidėjusio kodavimo lankstumo ir sudėtingumo.Galima teigti, kad iššifravimo sudėtingumas yra proporcingas vaizdo dydžiui ir atvaizdavimo dažniui.

3. PROJEKTINĖ DALIS

3.1. Vaizdo dekoderis

Vaizdo dekoderis yra sudėtinga skaitmeninė sistema, kuriai reikalingi didžiuliai aritmetiniai skaičiavimai ir milžiniškos atminties prieigos. Dėl greitos šiuolaikinės VLSI technologijos pažangos, sudėtingi užšifravimo/iššifravimo algoritmai, seniau yra įvertinti kaip nepraktiški, dabar yra įmanomi. Yra daug pasiekiamų sprendimų, kad įgyvendinti tokią sistemą. Projekto gyvybingumas priklauso nuo prieštaringu veiksnių, tokių kaip našumas greičio atžvilgiu, energijos sąnaudos, kainos, lankstumo, ir gamybos apimties. Pasirenkant tinkama vykdymo strategiją, optimizuojant skirtinguose projekto lygmenyse ir tinkamai dalinant iššifravimo užduotis tarp skirtingų blokų yra vienas iš pagrindinių projekto žingsnių.

3.2. Projekto metodologija

Kad pasiekti efektyvų vaizdo iššifravimo atlikimą, projektuotojai turi turėti įžvalgumo savybių, kad suprastu apie vaizdo duomenis , užkodavimo algoritmą, taip pat bendrą efektyvią projekto metodologiją. Tada įvairios optimizavimo technikos gali būti pritaikytos skirtinguose projekto lygmenyse, kad gauti beveik optimalų projektą su lygiu duomenų srautu, aukštu aparatinės įrangos panaudojimu, ir žemomis energijos sąnaudomis. Projekto metodologija, šiame darbe, gali būti padalyta į kelis abstrakcijos lygmenis:

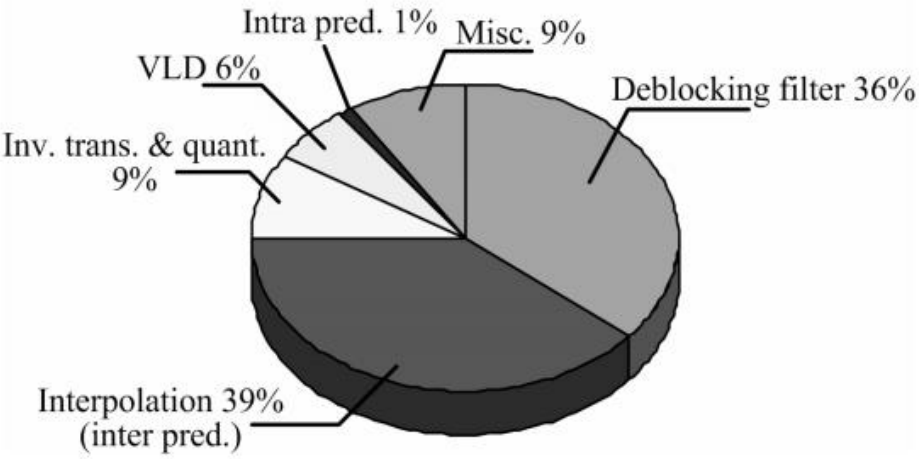
1) Sistemos apsvarstymas;
2) Algoritmo optimizavimas;
3) Architektūros tyrimas;
4) Grandinės optimizavimas;

5) Fizinis įgyvendinimas.

3.2.1. Sistemos apsvarstymas

Sistemos apsvarstymas yra pirmas žingsnis, kuris reikalingas nustatyti kritinėms projektuojamos sistemos klaidoms. Kadangi vaizdo kodavimo algoritmai tipiškai sudaro kelios intensyvių skaičiavimų užduotys, projekto dėmesys pirmiausiai sutelkiamas šių užduočių optimizavimui. Tačiau, efektyvus kitų užduočių vykdymas yra būtinas bendram sistemos veikimui ir negali būti ignoruotas. Kai kurie vaizdo iššifravimo sistemos moduliai, kaip duomenų srauto apdorojimas yra intensyviai valdomi.

Nėra sudėtingų matematinių operacijų, bet tikslūs ir subtilūs kintamo greičio bei ilgio duomenų apdorojimas. Kiti, kaip nuspėjimo apskaičiavimas ar deblocking filtras, reikalauja didžiulių skaičiavimo su didelę atminties prieiga. Bendrai kalbant, pagrindinės vaizdo iššifravimo sistemos kliūtys yra skaičiavimas ir atminties prieiga.



5 pav. H.264/AVC iššifravimas [11]

Paveiksle 5 yra grafiškai pavaizduotas veikiantis, H.264/AVC iššifravimas [11]. Nors analizės rezultatai gali skirtis, kai programa paleista skirtinguose platformose, analizės duomenys vis dar pateikia projektuotojams gerą pradinį tašką, kad suprasti visos sistemos ir kiekvieno modulio sudėtingumą. Veiksmai susideda iš deblocking filtro, inter apskaičiavimo, intra apskaičiavimo, transformacijos ir kvantavimo, kintamo ilgio iššifravimo (VLD), ir įvairių operacijų. Inter apskaičiavimas ir deblocking filtras yra dvi daugiausiai atimančios operacijos,todėl turi būti apdairiai optimizuotos. Jei perdavimo sparta padidėja, atitinkamai, proporcingai auga ir VLD sudėtingumas.

3.2.2. Algoritmo optimizavimas

Į aparatinę įrangą orientuoto algoritmo optimizavimas yra privalomas prieš aparatinės įrangos atlikimą. Optimizavimas aukštesniame lygmenyje paprastai turi didesnį poveikį visai sistemai. H.264/AVC standartas apibrėžia atitinkamą vaizdo dekoderį, palikdamas pakankamai įvairovės šifratoriaus projektavimui.

Judesio apskaičiavimas (ME) yra daugiausiai skaičiavimo ir galios reikalaujantis šifratoriaus modulis. Tačiau, jo paieškos algoritmas yra ne vaizdo kodavimo standartų apimtyje, kas reiškia, kad projektuotojai gali laisvai išsirinkti tarp skirtingų algoritmų. Todėl, ten yra daug vietos algoritmo optimizavimui. Keletas greitų paieškos algoritmų ir modelių yra pasiūlyti [12]. Greitos paieškos algoritmas ieško mažiau kandidatų negu pilna paieška, tokiu būdu jis mažina skaičiavimą ir atminties prieigą prastesnės kokybės sąskaita. Pilnos paieškos judesio apskaičiavimas, skaičiavimą taupantys planai tokie kaip PDE (Dalinis Iškraipymo Pašalinimas) ir SEA (Nuoseklus Pašalinimo Algoritmas) yra vis dar taikomi nereikalingų operacijų sumažinimui [13].

Projektas [14] pasiūlė keičiamo dydžio algoritmą ir architektūrą skirta IME (Sveikojo skaičiaus judesio Įvertinimas) ir FME (Trupmeninis Judesio Įvertinimas). Tai suteikia keturis keičiamos kokybės rėžimus (QS0 ~ QS3), kurie padeda sumažinti projekto sudėtingumą 53 % ~ 87 % už 0.13dB ~ 0.23dB PSNR netektį. Taip pat siūlo keturis kintamos kokybe rėžimus intra apskaičiavimui, kuris pasiekia 1.4 ~ 1.74 karto pralaidumo pagerinimą su mažesniu negu 0.2dB PSNR kritimu.

Palyginus su užšifruotoju, dekoderio projektavimas yra mažiau lankstus kadangi vaizdo standartai išsamiau apibrėžia kiekvienus iššifravimo žingsnius. Svarbiausios iššifravimo stadijos, tokios kaip intra apskaičiavimas, inter apskaičiavimas bei deblocking filtrs yra griežtai apibrėžti ir kiekvienas dekoderis turi laikytis šių iššifravimo žingsnių.

3.2.3. Architektūros tyrimas

Perkėlimas iš algoritmo į atitinkamą aparatinės įrangos architektūrą yra labai susietas su projektavimo specifikacijomis, tokiomis kaip funkcija, kaina, našumas, plotas, ir galia.

Dėl sudėtingų realaus laiko apribojimo kaip pateikia 3 lentelė, konvejeris ir lygiagretumas yra dvi plačiai pritaikytos vaizdo kodeko projektavimo metodai. Be to, dėl reikalaujamos atminties prieigos ir jungtinės didelės galios prieigos, atminties organizavimas ir prieigos struktūra turi būti protingai pritaikyti.

Aparatinės įrangos kaina taip pat yra svarbi problema vaizdo kodeko projektavime. Tiek lygiagretumas bei konvejeris reikštu didesnes išlaidas plotui. Paprastai kopijuojant kelis apdorojimo

elementus ar konvejerio registrus be apribojimų nėra optimaliausias kelias. Optimizuotas tikslas yra pakankamai lygiagretumas ir konvejeris su kaip įmanoma didesniu aparatinės įrangos panaudojimu.

3.2.4. Grandinės optimizavimas

Kai sistemos architektūra yra sukurta, įskaitant konvejerį, lygiagretumą bei atminties organizavimą, keli metodai gali būti pritaikyti vaizdo iššifravimo grandinės optimizavimui, kad būtu pasiektas aukštesnio detalumo lygmuo. Dėl vaizdo duomenų savybių, kurios neišvengiamai turi savyje įvairaus tipo perteklių ir pateikią tam tikrą statistinį pasiskirstymą, statistika paremtas optimizavimas yra išnaudojamas, kad sumažintų iššifravimo sudėtingumą. Apskritai šis metodas pagerina sistemos pralaidumą ir sumažina galios suvartojimą, nepatiriant papildomų aparatinės įrangos išlaidų.

Kitas svarbus optimizavimo metodas yra duomenų kelio perėjimų sumažinimas. Pavyzdžiui, iššifravimo metu, dauguma daugybos gali būti išskaidoma į papildymus ir poslinkius kadangi paprastai, vienas iš dviejų operandų yra pastovi reikšmė, kaip $\times 5 = \times 4 + \times 1$, kur daugybą pakeičia vienas postūmis ($\times 4$) ir vienas pridėjimas ($\times 4 + \times 1$). Palyginti su originaliu atlikimu, ne tiktai aparatinės įrangos kaina yra žymiai sumažinama, kad gauti galutinius rezultatus signalo perėjimai yra sumažinti su pagerintu našumu. [15]

3.3. Įgyvendinimo strategijos

Apskritai, yra daug praktinių būdų įgyvendinti skaitmeninę sistemą. Iš esmės, kokia optimali architektūra turi būti, priklauso pritaikymo ir bendro sistemos apsvarstymo. Bendro panaudojimo procesorius (GPP-General Purpose Processor) toks kaip Intel Pentium, su derama programine įranga yra lanksčiausias ir patogiausias būdas. Nors DSP procesorius, nėra toks lankstus kaip bendras bendro panaudojimo ir yra neefektyvus sistemos valdymui bei užduočių vykdymui jis yra suprojektuotas tam, kad paspartintu, kai kurių specifinių skaitmeninių signalų atlikimą. FPGA(Field Programmable Gate Array) yra kitas lankstus sprendimas, kuris yra ypač tinkamas sistemos prototipui. ASIC (Application Specific Integrated Circuit) yra pritaikytas tam tikram pritaikymui ir visos jo funkcijos po gamybos yra "užtvirtintos". Sistemoms, kur kelios skirtingos užduotys yra įvykdytos lygiagrečiai, GPP ar DSP yra geras pasirinkimas, tuo metu, kai FPGA ir ASIC yra ypač pageidaujami specialiam pritaikymui. Dėl mažesnio projektavimo ir žemesnių gamybos apimčių, FPGA gali būti ekonomiškai efektyvesnis negu ASIC projektas. Tačiau, projektas su milijonais loginių elementų, ASIC yra daug tinkamesnis.

Pageidaujamos funkcijos metodo pritaikymas silicio plokštei priklauso nuo pačios funkcijos. Projektuotojo iššūkis yra parinkti stilių, kuris atitiktu produkto specifikaciją ir apribojimus.

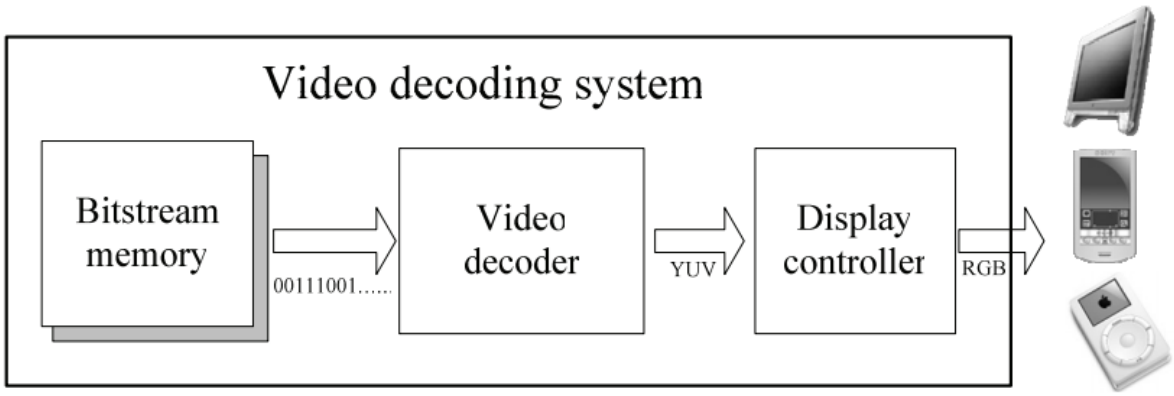
Skaičiavimo sudėtingumas yra problema su kuria pirmiausia susiduria vaizdo kodekas. Vaizdo kodekas turi atlikti duomenų apdorojimą esant aukštam duomenų srautui realiu laiku, paprastai didesniu negu 30 kadrų per sekundę. Bendro panaudojimo procesoriui, sudėtinga pasiekti realaus laiko atlikimą, net ir dirbant aukštais dažniais. Tai yra todėl, kad bendras duomenų kelias ir nuoseklus GPP apdirbimo srautas yra neefektyvūs vaizdo kodavimui. DSP arba daugiafunkcinis procesorius, yra optimizuotas signalo apdirbimui, vis dar reikia pridėti ypatingas instrukcijas su lygiagrečiu apdirbimo gebėjimu arba paskirti akceleratorius, kai kurių kritiškų operacijų pagreitinimui. Vaizdo apdirbimo funkcijų perkėlimas į FPGA reikalauja nemažai resursų. Pavyzdžiui, tiktai visai nesenai, Xilinx paskelbė vaizdo kodekų IP palaikymą ant savo aukšto našumo ir brangaus Virtex-4 įrenginio. Palygintu su anksčiau paminėtais trimis sprendimais, visiškai vaizdo kodavimui pritaikyta architektūra yra geras sprendimas gauti optimizuotą atlikimą, kadangi kiekvienam moduliui koduojančiame vaizdo algoritme, operacijų savybės yra atidžiai ištirtos. ASIC projektavimas yra linkęs pateikti lygiagrečią ir konvejerio architektūrą, kad reikalaujamas veikimo dažnis galėtų būti ženkliai sumažintas.

Jei energijos suvartojimas yra svarbus, tai ASIC yra ko gero labiausiai energiją taupantis sprendimas, kadangi jis suprojektuotas taip, kad aprūpintų "tik pakankamai" funkcijos pritaikymui. Standartiniame ASIC luste, kiekvienas funkcijos blokas vykdo aiškią iš anksto paskirtą užduotį. Be to, ASIC veikimo dažnis yra daug žemesnis negu kitų sprendimų, vadinasi maitinimo įtampa gali būti sumažinta taip sutaupant energijos suvartojimą. Pavyzdžiui, MPEG2 iššifravimui skirtas ASIC lustas suvartoja 0.5W, kai tuo metu bendram panaudojimo procesoriui reikia 30W bei standartinis DSP suvartoja 3W [16]. Vienintelė kliūtis vaizdo kodavimo projektavimui, skirtam ASIC yra lankstumo trūkumas, palyginti su procesoriumi ar FPGA. ASIC lusto lankstumas yra paaukotas dėl aukšto našumo ir žemų energijos sąnaudų. 6 lentelėje palyginamos šitos keturios strategijos.

6 lentelė. Strategijų palyginimas

	GPP/DSP	FPGA	ASIC
Programmability	High	High	Low
Development cycle	HW + SW	HW	HW
Area efficiency	Medium	Low	High
Power efficiency	Medium	Low	High
Performance	Low	Medium	High

3.4. Sistemos padalijimas



6 pav. Vaizdo iššifravimo sistema

Visa iššifravimo sistema turi būti padalyta lengvam projektui ir atlikimui. Kaip pateikta 6 paveiksle, užbaigta vaizdo iššifravimo sistema apima tris dalis, tai užkoduoto duomenų srauto atminties kaupimas, vaizdo dekoderis ir atvaizdavimo valdiklis.

Užkoduotas vaizdas visada pateikiamas duomenų srauto (bitstream) formate, kuris susideda iš nuoseklių skaitmenų, tai "0" ir "1". pagal užkoduotą duomenų srauto struktūrą, kiekvienas vaizdo standartas turi savo savą specifinę duomenų srauto struktūrą, video dekoderis interpretuoja kiekvienus sintaksės elementus ir atstato vaizdą su iškraipymais. Atstatyti vaizdai paprastai yra pateikiami YCbCr (YUV) formatu. Iššifruoto YUV vaizdo seka siunčiama į atvaizdavimo valdiklį, kuris verčiama iš YUV į RGB formatą ir tik tada gali būti atvaizduotas.

3.5. Bluespec system verilog projekto apžvalga

Bluespec SystemVerilog - nauja aparatinės įrangos apibūdinimo kalba, kuri padeda greičiau ir lengviau sukurti, pašalinti, ir modifikuoti veiksmingus aparatinės įrangos projektus. Bluespec SystemVerilog taip pat turi savyje daug ypatybių,kurios padeda teisingai rašyti nuorodos kodus. Nors ši kalba šiuo metu yra naudojama plėtoti aparatūros įrangai, tačiau yra vilčių, kad ateityje peržiūrint jį bus galima naudoti kaip programinės įrangos programavimo kalboms kurti, ypač sistemose turinčiose kelis branduolius.

Bluespec SystemVerilog yra aukšto lygio aparatinės įrangos apibūdinimo kalba, pagrįsta "Hoe’s TRAC" kalba. Bluespec yra pagrįstas sąvokomis "Guarded Atomic Actions" ar taisyklėmis. Kiekviena taisyklė turi savyje "boolean predicate“ ir veiksmą nusakantį būsenos pakeitimui. Predicate tiksliai apibrėžia, kai taisyklė yra galiojanti. Bluespec programos įgyvendinimas gali būti apibūdinta kaip "a sequence of firings of the rules in the design", reikšdama, kad kiekviena taisyklė gali būti pagrįsta atskirai. Tačiau, diegimas leidžia atlikti daugialypes taisykles tuo pačiu metu, jei

jos atitinka Bluespec "one-at-a-time" semantiką. Dabartinis Bluespec kompiliatorius kuria veiksmingą kombinuotą tvarkaraštį, kuris bando pasirinkti "maksimalų" komplektą taisyklių "to fire each cycle".

Buvo įrodyta, kad dideli projektai gali būti efektyviai padaryti su Bluespec. Taip pat projektai, padaryti su Bluespec, yra tokios pačios kokybės kaip ir projektai atlikti su RTL dublikatais.

Bluespec turi galingą tikrinančią sistemą.Tai leidžia išvengti visą klasę klaidų, kurios paprastai būna padaromos tradicinėmis aparatinės įrangos apibūdinimo kalbomis.

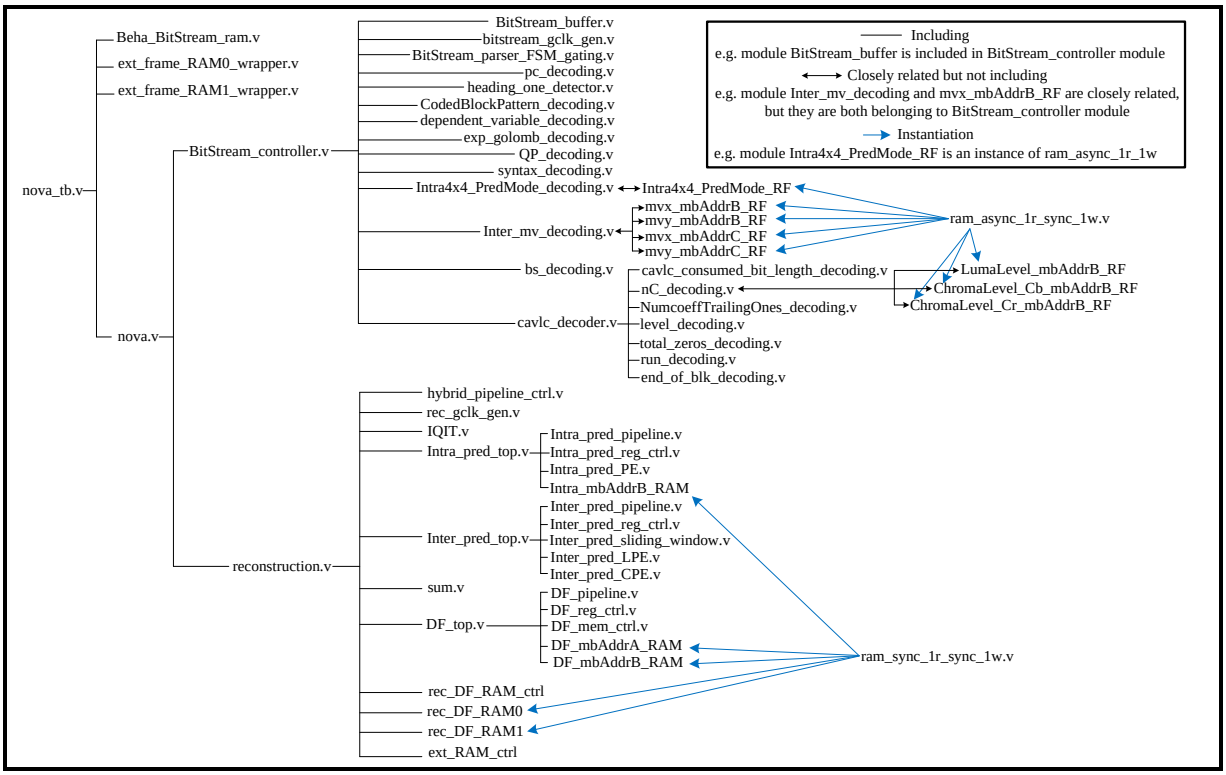
Kaip kitos aparatinės įrangos apibūdinimo kalba, Bluespec projektas gali būti suskirstytas į modulius. Tačiau, skirtingai nuo kitų aparatinės įrangos apibūdinimo kalbų, Bluespec modulio modulio sąsaja susideda iš metodų. Šitie metodai yra panašūs į metodus objektinio programavimo kalbose. Naudojant metodu pagrįstą sąsają padeda kelti lygmenį abstrakcijos ir trukdo klaidingam modulių naudojimui. Tai ypač naudinga dideliems projektams, kur kartu veikiantys moduliai yra dažnai rašomi įvairių žmonių.

Bluespec taip pat rodo apsaugos metodų funkcijas, kurios nurodo, kada metodas yra paruoštas. Pavyzdžiui, eilės (enqueues) metodas FIFO yra paruoštas, kai yra vietos FIFO metode, kad priimtų kitą elementą. Bluespec automatiškai įtraukia numanomas sąlygas metodų taisyklės naudojant jos tarinį. Tokiu būdu taisyklė, kuri naudojama, ir elementas nuo vieno FIFO į kitą bus paruoštas, kai bus elementas pirmame FIFO, ir vieta antrame.

Bluespec projektui, taisyklių sąrašų įvairovė yra galima su jų savomis našumo charakteristikomis. Bluespec kompiliatorius pasirenka vieną galimybę, bet duoda grįžtamąjį ryšį projektuotojui, kad priimtų geresnius sprendimus. Pavyzdžiui, kompiliatorius paskelbs, kad dvi taisyklės sistemoje negalės būti lygiagrečios, jeigu neįvestas kombinuotas kelias. Projektuotojas, norintis leisti šį kelią, turėtų nurodyti šį faktą kompiliatoriui.

3.6. Nova projektas

Nova - mažos galios realaus laiko H.264/AVC QCIF raiškos dekoderis, skirtas mobiliems įtaisams. Tai yra pilnai skirtas aparatinei įrangai ir turintis savyje ASIC dizainą, nepanaudojant jokių GPP/DSP branduolių. Tai buvo sėkmingai patikrinta su Xilinx Virtex-4 FPGA ir 0.18um ASIC mikroschemų. Nustačius 30 kadrų per sekundę bei QCIF iššifravimą, išmatuotos energijos sąnaudos yra 293 uW tekant 1V.



7 pav. Verilog-HDL kodo struktūra

Nova projekto ypatybės:

- ❖ RTL koduotas Verilog-HDL.
- ❖ Palaiko realaus laiko H.264/AVC QCIF raiškos iššifravimą. Gali būti išplėstas iki aukštesnės raiškos per smulkius pakeitimus.
- ❖ Platus konvejerio ir lygiagretumo panaudojimas, kad pagerintu našumą ir sumažintu galios suvartojimą.
- ❖ Mišri ir prisitaikanti konvejerio architektūra, kad išvengtų nereikalingų ciklų ir pagerintų našumą:
  - ✓ Adaptyvus konvejeris skirtas intra, ir inter apskaičiavimui.
  - ✓ 4×4/16×16 mišrus konvejeris.
  - ✓ 1×4 pikselių kolonos lygio lygiagretumas.
- ❖ Mažų išlaidų intra apskaičiavimo blokas:
  - ✓ Adaptyvus konvejeris.
  - ✓ Hierarchinė atminties organizacija, kad sumažinti išorinės atminties prieigą.

- ✓ "Sėklos" metodas plokštumos rėžimo skaičiavimui.
- ✓ Duomenų pakartotinio panaudojimo tyrimas tarp 1×4 skilčių.
- ✓ Daugiafunkciniai apdorojimo elementai visiem intra nuspėjimo būdų apdirbimui.

❖ Optimizuota judėjimo kompensacija (inter apskaičiavimas), blokas:

- ✓ Prisitaikantis konvejeris.
- ✓ Hierarchinė atminties organizacija, kad sumažintų išorinę atminties prieigą.
- ✓ "Kintamoji bloko forma", kad sumažintu likusią be darbo atminties prieigą ir pagerintų pralaidumą.
- ✓ Luste esantis nukreipiamasis pikseliu buferis reikalingas ištirti pakartotinį panaudojimą.
- ✓ Konvejerio ir lygiagretumo luma darantis interpoliavimas, sudarytas iš 9 horizontalių 6 filtrų, 4 vertikalių 6 filtrų, ir 4 dvitiesių filtrų.
- ✓ Novatoriškas chroma darantis interpoliavimas, panaudojantis mažiausiai sumatorių skaičių.

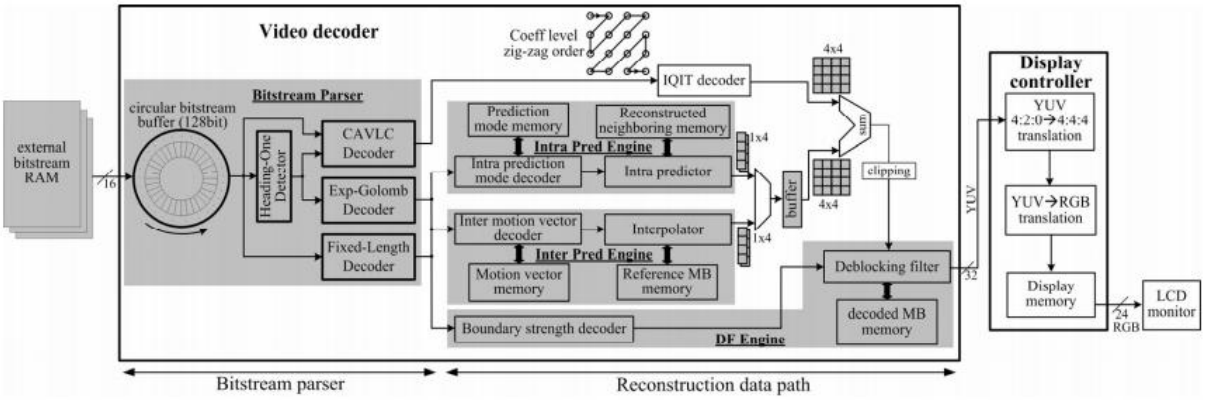
❖ Aukšto našumo atblokavimo(deblocking) filtras.

- ✓ Novatoriška 5 stadijų konvejerio architektūra susitvarkanti su duomenų/struktūros pavojais.
- ✓ Vieno porto paremtas SRAM, nereikalingas dviejų portų SRAM.
- ✓ 204 ciklu/MB pralaidumas su maksimaliu 200 MHZ dažniu (0.18µm procesas). Gali pasiūlyti iki 980kMB/s pralaidumą.

❖ Mažos galios, pigus projektas.

- ✓ Reikalauja tiktai ~1.5MHz QCIF 30fps realaus laiko iššifravimui.
- ✓ Tiktai 169 kb loginiai elementai.
- ✓ Apskaičiuotas galios suvartojimas yra žemas kaip 293µW.

3.6.1. Sistemos architektūra



8 pav. Vaizdo iššifravimo sistemos architektūra

Apskritai tai gali padalyti į tris dalis, duomenų srauto atmintis(RAM), video dekoderis ir atvaizdavimo valdiklis su LCD atvaizdavimu.

3.6.2. Veikimo procesas

Šis skyrius apibūdina branduolio operacijas:

1. Po reset "bitstream buferis" paima duomenų srautą iš "bistream_ram" per 16 bitų pločio magistralę (bistream buffer input).

2. Po 4 ciklų kai pusė 128bitu "BiStream" buferio yra užpildyta, jis siunčia duomenys sekančiam dekoderiui per 16bitų pločio magistralę (bistream_buferio_output). Duomenų srauto buferis yra kaip interfeisas tarp išorinės duomenų srauto atminties ir lusto dekoderio. Automatinio užpildymo mechanizmas yra įdarbinamas, kad užpildytu buferi jei puse talpinamo duomenų srauto (>=64bit) yra sunaudojama.

3. Duomenų srauto valdiklis (Bistream_controller) pradeda iššifruoti, kai "bistream_buffer" n= 1 b0. Taip sukuriami kontrolės parametrai ir valdymo signalų rekonstrukcijos kelias.

4. Rekonstrukcijos kelias susideda iš "intra prediction" ir "inter prediction" bei "debloking filtro". Jei kuris nors intra ar inter prediction blokas yra sužadinamas remiantis esamu makrobloko tipu. intra/inter bloko išėjimas yra siunčiamas į debloking filtrą jei reikia.

5. Debloking filtro išvestis yra siunčiama į išorine atminti. Yra dviejų tipų atmintys ext_frame_RAM0_wrapper ir ext_frame_RAM1_wrapper. jeigu viena veikia kaip nukreipiamoji atmintis(suteikia nukreipiamuosius pikselius dekodavimui), kita veikia kaip atvaizdavimo atmintis (dekoderis talpina dekoduotus pikselius i savo atminti). Po to kai 1 kadras dekoduotas, RAM0 ir RAM1 apsikeičia funkcijomis.

3.6.3. Duomenų srauto atmintis

Išorinė atmintis talpina duomenų srautą, kuris bus iššifruojamas. Ši atmintis yra priskiriama išorinei, tai suteikia pakankamai lankstumo. Užkoduotos vaizdo sekos dydis priklauso nuo dviejų faktorių: ilgio ir vaizdo pobūdžio. Savaime suprantama, kuo ilgesnė vaizdo seka, tuo didesnis užkoduoto duomenų srauto dydis. Be to, vaizdo pobūdis taip pat veikia užkoduoto duomenų srauto ilgį. Greita judanti vaizdo dalis savaime suprantama yra didesnė negu lėtai judanti arba išvis nejudančio. Greitą judančiame vaizde, einamasis paveikslas turi mažesnes galimybes surasti deramus atitikmens blokus, tai daro apskaičiavimą mažiau efektyvų. Todėl, užkoduotu duomenų srautų ilgis yra kintamas, jeigu atmintis yra integruota į lustą, tai ji turi būti gana didelė, kad sugebėtų prisitaikyti prie svarbiausių duomenų srautų. Tai yra beveik neįmanoma, kadangi negalime niekada numatyti, kokia bus taikoma iššifruota vaizdo seka , kai lustas yra pagamintas. Išorinė atmintis yra lankstesnė savo dydžiu ir gali būti patogiai sureguliuota per skirtingą sistemos integraciją.

3.6.4. Vaizdo dekoderis

Visos iššifravimo sistemos branduolys yra video dekoderis, kuris atstato apskaičiuotus atvaizdus, pagrįstus įvestu duomenų srautu ir nukreipiamaisiais kadrais.

Vaizdo dekoderis gali būti padalytas į dvi svarbiausias dalis, duomenų srauto analizatorius(bitstream parser) ir rekonstrukcijos duomenų kelias(reconstruction data path). Duomenų srauto analizatorius aptinka ir iššifruoja kiekvieną įeinantį sintaksės elementą ir generuoja iššifravimo parametrus bei valdymo signalus.

Duomenų srauto analizatoriuje viduje, duomenų srauto buferis naudojamas kaip sąsaja tarp išorinės duomenų srauto atminties(RAM) ir luste esančio dekoderio; detektorius identifikuoja, pirmo bito padėtį “1” dabartinės įvesties užšifruotą žodį; mišraus ilgio dekoderis, kuris yra susideda iš CAVLC (Content Adaptive Variable Length Coding) dekoderis, Exp-Golomb fiksuoto ilgio dekoderiai nagrinėja pastovaus ilgio, ir su kintamo ilgio slaptažodžius. Duomenų srauto analizatorius išėjimai perduodami į rekonstrukcijos duomenų kelią, kad nukreiptų tolimesnei paveikslo rekonstrukcijai.

Rekonstrukcijos duomenų kelias savyje turi kelis pagrindinius sudėtingus funkcinius blokus, IQIT (Inverse Quantization Inverse Transform) dekoderį, intra apskaičiavimo varikliuką, inter apskaičiavimo varikliuką, Deblocking Filtras(DF)varikliuką. IQIT dekoderis gauna likutinius koeficientų lygius iš CAVLC dekoderio, ir po kelių atvirkštinės kvantizacijos ir transformacijos

periodu yra sugeneruojami likutiniai blokai, kurie bus pridėti prie apskaičiavimo rezultatų. Pagal makrobloko apskaičiavimo tipą tai būtu ar intra, ar inter apskaičiavimo varikliukai yra pasitelkiamas, kad pagamintų originalų atvaizdo apskaičiavimo bloką.

Šis apskaičiuotas blokas yra susumuojamas su likutiniu bloku iš IQIT dekoderio, kad suformuotų atstatytą bloką. Adaptyviais parametrais pagrįstas rinkinys iš skirtingų sintaksės lygmenų, DF variklis gali būti aktyvintas, kad filtruotų kiekvieną makrobloką esanti linijoje, kad sumažintų blokavimo artefaktus. Paskutinis blokas po filtravimo yra YCbCr formato, kuris negali būti tiesiogiai atvaizduotas daugumos atvaizduotojų. Todėl, jis yra nusiunčiamas į atvaizdavimo valdiklį formato vertimui.

3.6.5. Atvaizdavimo valdiklis

Atvaizdavimo valdiklis gali būti įvertintas kaip blokas po apdirbimo skirto vaizdo dekoderiui. Pirmiausia, 4:2:0 YCbCr formatas yra paverčiamas į 4:4:4 formatą. Tada suminkštintas atvaizdas yra verčiamas į RGB formatą 24 bitų kiekvienam vaizdo elementui ir kaupiamas didelėje atvaizdavimo atmintyje.

Pagal atvaizdavimo reikalavimus, RGB atvaizdai yra iš eilės perkeliami iš atvaizdavimo atminties ir pateikiami vaizduoklyje.

3.6.6. Konvejerio projektavimas

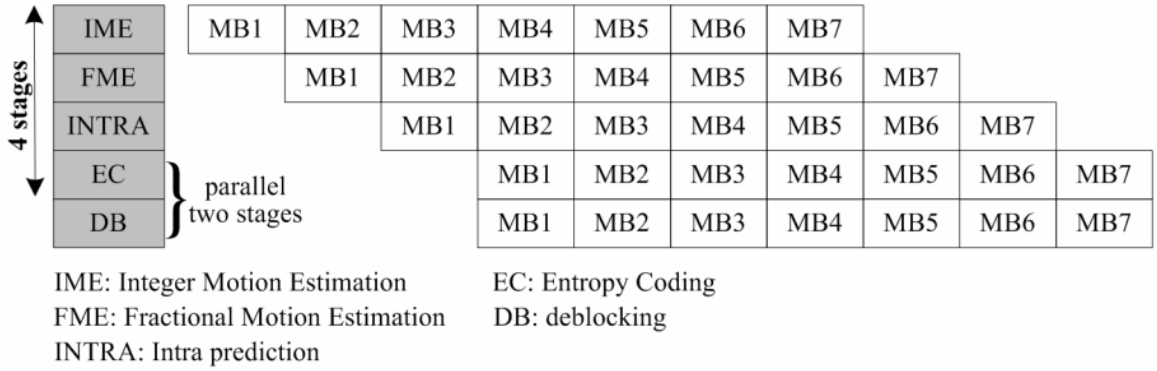
Kadangi video iššifravimo sudėtingumas be perstojo didėja, nuoseklus apdirbimas neabejotinai negali patenkinti realaus laiko iššifravimo reikalavimų. Todėl gerai žinomas konvejerio metodas paspartinantis sistemos darbą yra būtinas vaizdo iššifravimui. Kad išgauti geriausią našumą, konvejerio stadijos turi būti kruopščiai išlygintos. Idealiu atveju, kiekviena konvejerio stadija yra lygaus ilgio ir visas pralaidumo pagerinimas yra proporcingas konvejerio stadijoms. Pavyzdžiui, jei konvejerio konstrukcija turi penkias lygiai išlygintas stadijas, tai jo našumas teoriškai yra penki kartus didesnis už nenaudojančia konvejerio konstrukciją.

Be subalansuoto konvejerio, jungtinė logika taip pat turi būti sutvarkyta tokiu būdu, kad nebūti perdaug pridėtinių konvejerių. Šitos pridėtinės, dažnai vadinamos konvejerio pavojais, kadangi gadina konvejerio našumą, priversdamos konvejerį laikas nuo laiko sustoti. Iš viso yra trijų tipų konvejerio pavojai, tai yra valdymo, duomenų ir struktūros pavojai. Kad juos sumažinti, paprastai turi būti pateikta papildoma aparatinė įranga [17].

Apskritai šiame vaizdo kodavimo projekte yra dvi konvejerio architektūros. Makrobloko lygmens konvejeris, kuris yra pritaikytas 16x16 makroblokui ir 4x4 bloko lygmens konvejeris, kuris yra pritaikytas 4x4 blokui.

3.6.6.1. Makrobloko Lygmens Konvejeris

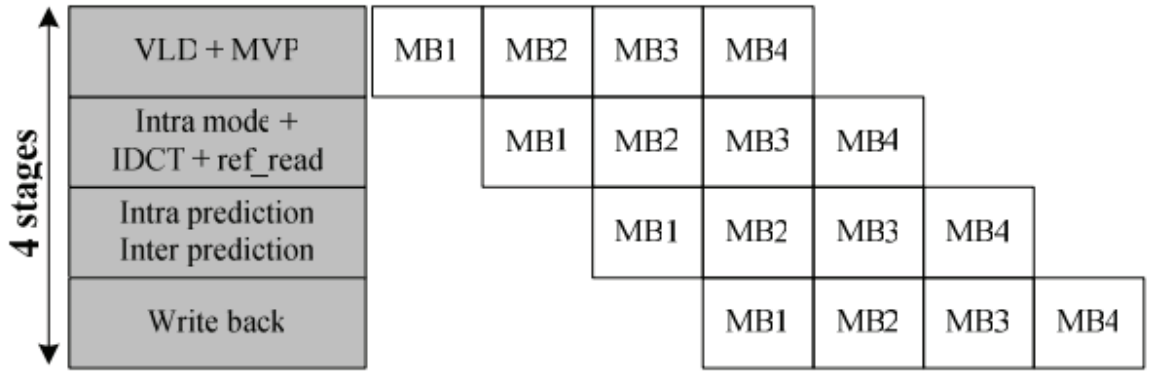
Makrobloko lygmens konvejeris yra pagrįstas visam 16x16 makroblokui. Įprastai, vaizdo kodavimo architektūrai yra pritaikomi makroblokai, kadangi tai yra pagrindinis apdirbimo objektas. Makrobloko konvejeris tradiciniam šifruotojui yra pavaizduotas 9 paveiksle. [18]



9 pav. Makroblokas paremtas 4 stadijų vaizdo kodavimas konvejeriu

Vaizdo šifravimo užduotis yra suskirstytos tarp keturių konvejerio stadijų Entropijos kodavimo (EC) ir deblocking (DB) stadijos yra lygiagrečios. Duomenų apskaičiavimo procedūra turi (IME), (FME) ir (INTRA) stadijas. IME yra aukščiausią skaičiavimo sudėtinguma turinti stadija kodavimo sistemoje. Ji turi būti labai išvystyta, kad palaikytų kintamus blokų dydžius ir daugialypę nukreipiamų kadrų funkciją. FME yra atskirtas nuo IME kaip viena konvejerio stadija, kad sumažintų sudėtingumą. Nors intra apskaičiavimo sudėtingumas nėra aukštas, duomenų priklausomybė tarp smulkesnių blokų yra neišvengiamo nuoseklaus srauto priežastis. Jei FME stadija bus perpildyta, tai kritinis šio makroblokų konvejerio stadijos kelias bus per ilgas. Todėl, intra apskaičiavimas, kartu su DCT/Q/IQ/IDCT, formuoja trečiąją stadiją.

Deblocking, kuris reikalauja didelio pralaidumo, panaudoja atstatytus duomenis po INTRA stadijos. Entropijos kodavimas, kuris reikalauja nuoseklių bitų lygmens operacijų, koduoja duomenis po pasirinkto rėžimo ir likučių generacijos. Dėl nuoseklios procedūros ir unikalių ypatybių, kurias yra sudėtinga pasiekti, ištekliai pasidalinami su kitais paspartintojais,kad gauti aukštesnį aparatinės įrangos panaudojimą, DB ir EC yra lygiagretūs ir priskiriami ketvirtai stadijai.



10 pav. Makroblokas su 4 stadijų vaizdo iššifravimo konvejeriu [19]

Panašus kaip kodavimas, vaizdo iššifravimas gali taip pat naudoti makroblokų konvejerį. Tipiškas vaizdo dekoderis su tokiu konvejeriu yra pateiktas 10 paveiksle.[19]

Konvejerio pirmoji stadija susideda iš VLD (Kintamas Ilgio Iššifravimas) ir MVP (Judėjimo Vektoriaus apskaičiavimas). Antroji stadija turi intra rėžimo apskaičiavimą, IDCT (Atvirkštė diskrečiojo kosinuso transformacija) ir "ref_read", kuris nuskaito duomenis iš nukreipiamūjų kadrų atminties, naudodamas judėjimo vektorius gautus ankstesnėje MVP stadijoje. Trečioji stadija turi intra ir inter apskaičiavimo blokus, kurie aktyvuojami pagal makrobloko tipą. Paskutinė konvejerio stadija susideda iš atgalinio įrašymo bloko, kuris saugoja apskaičiuotus duomenis išorinėje kadro atmintyje.

Makrobloko konvejerio esmė yra, kad kiekviena konvejerio stadija suderinta su makrobloko riba. Pavyzdžiui, 9 paveiksle vaizdo kodavime, kai visas MB1 užbaigia IME, jis eis į FME stadiją tuo pačiu metu, kai MB2 eis į IME stadiją.

3.6.6.2. 4x4 bloko konvejeris

Ankstesnės vaizdo kodavimo realizacijos, nesvarbu šifratorius ar dekoderis, naudojo tiktais makrobloko konvejerį, kadangi tai yra pagrindinis apdirbimo objektas. Pranašumas yra tas, kad konvejerio stadijos gali būti lengvai padalytos tarp makrobloko ribų ir konvejerio srautas gali būti lengvai valdomas. Kliūtis yra ta, kad tarpinis buferis tarp kiekvienų konvejerio stadijų turi būti gana didelis, kad kauptu tarpinius viso 16x16 dydžio rezultatus. Taip pat, nėra duomenų pakartotinio panaudojimo kiekviename makrobloke, nes visas makroblokas yra apdirbamas kaip visuma.

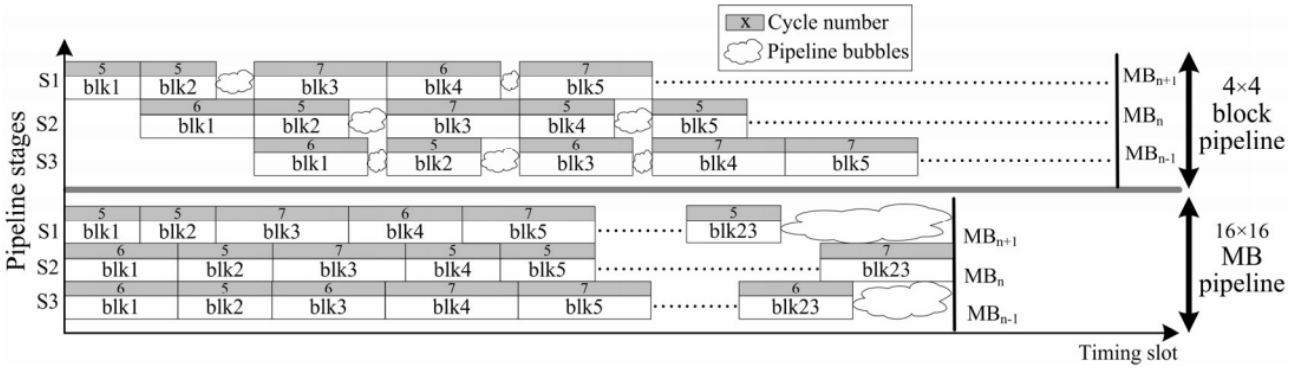
Skirtingai negu ankstesni standartai, mažiausias operacinis H.264/AVC blokas yra tiktais 4x4 luma komponentas, atitinkamai 2x2 chroma. Tai suteikia konvejeriui potencialią galimybę dirbti su mažesne blokų geometrija. 4x4 konvejeris pasiūlytas [20]. Palyginti su 16x16 makrobloko konvejeriu, jis turi tris pranašumus:

1) Tai atitinka mažiausią blokų dydį, apibrėžtą H.264/AVC standarte ir tokiu būdu turi naudos iš greito situacijos perėjimo;

2) Laikinoji atmintis, tarpinių duomenų palipinimui, tokia kaip registrai ar SRAM, gali būti iš esmės mažesnė todėl, kad reikia tiktai saugoti 4x4 bloką vietoj 16x16;

3) Duomenų pakartotinis panaudojimas tarp 4x4 blokų makrobloke yra tinkamas.

Kliūtis yra sistemos pralaidumo sumažėjimas todėl, kad apskaičiavimas turi būti sinchronizuotas kiekviename 4x4 bloke. Tačiau, tai galima iš dalies išlyginti pritaikius kelis kitus projektavimo metodus, tokius kaip saviadaptyvus konvejeris ir lygiagreti architektūra.

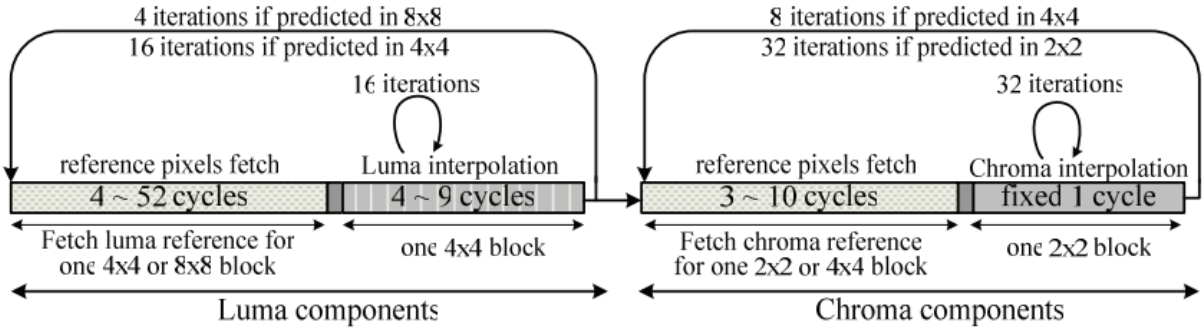


11 pav. Makrobloko konvejeris prieš 4x4 bloką

Paveikslas 11 pateikia dvi konvejerių struktūras. Burbulai 4x4 bloko konvejeryje, gali paskirstyti tarp kiekvienų kaimyninių konvejerio stadijų, dėl skirtingų konvejerio stadijų ilgių ant to paties parinkimo laiko plyšio (vertikalia kryptimi). Tačiau, kiekvienos stadijos pabaigoje,burbulai makrobloko konvejeryje yra sujungiami. Tai padeda pagerinti pralaidumą, kadangi pridėtų burbulų suma yra visada mažesnė negu arba lygi pasiskirsčiusiems burbulams.

3.6.6.3. Prisitaikantis konvejeris

Inter apskaičiavimo algoritmo analizė atpažįsta, kad skirtingos makrobloko dalys ir judėjimo vektoriai yra reikalingi skirtingoms atminties prieigoms bei pikselių skaičiavimams. Pastovaus konvejerio architektūra nėra efektyvi šitoje svyruojančio darbo krūvio aplinkoje. Inter apskaičiavimui skirtas prisitaikantis konvejeris yra pateiktas 12 paveiksle, kuris sugeba atpažinti visas galimas užduoties kombinacijas ir atitinkamai sureguliuoti konvejerio stadijas. Iš esmės, tai padalinama į dvi makro stadijas: nukreipiamų pikselių pernešimas ir įterpimas.



12 pav. Prisitaikantis inter apskaičiavimo konvejeris

Laiko ciklai yra reikalingi apskaičiuoti vieną makrobloką, kuris priklauso nuo makrobloko padalijimo ir judėjimo vektoriaus sprendimo kiekvienam padalijimui. Nėra jokios kontrolės ar duomenų priklausomybės tarp luma ir chroma komponentų apskaičiavimo. Idealiai, jie gali būti apdirbti ar nuosekliai arba lygiagrečiai. Kaip pateikia 12 paveikslas inter apskaičiavimo kliūtimi yra atminties prieiga. Todėl, luma, ir chroma komponentų skaičiavimas tuo pačiu laiku neturi daug naudos nebent taip pat lygiagretinsime luma ir chroma nuorodos pikselio pernešimą. Tik tada, kai pralaidumas yra didesnis už numatytą projekte, lygiagreti apskaičiavimo architektūrą gali būti pritaikyta.

Šiame darbe, visi 16 luma 4×4 blokai yra apdirbti prieš pradedant chroma apskaičiavimą. Luma nurodomieji duomenys yra pernešami iš išorinės atminties ir skiriami 4×4 ar 8×8 blokų apskaičiavimui, tuo metu, kai įterpimas pritaikomas 4×4 bloko bazei. Chroma nukreipiamieji duomenys yra priskirti 4×4 ar 2×2 po vieną apskaičiavimą vienu metu, kai interpoliacija toliau dirba 2×2 bloko bazėje. Visi šitie duomenų pernešimo ir interpoliavimo ciklai yra prisitaikantys išskyrus chromą interpoliavimą, kuris yra fiksuotas vieno ciklo kiekvienam 2×2 blokui.

3.6.7. CAVLC dekoderio projektavimas

CAVLC yra taikomas užkoduoti liekanoms, zigzagine tvarka pateiktiems 4x4 ir 2x2, blokų transformacijos koeficientams. Taip įvedama konteksto modelio sąvoką, kad padidintų suspaudimo efektyvumą. Jis suprojektuotas taip kad išnaudotu kelias kvantuojamų 4x4 blokų savybes.[21]:

1) Po apskaičiavimo, transformacijos, ir kvantizacijos blokai paprastai yra išskaidyti ir savyje daugiausia turi nulius. CAVLC pasitelkia run-level kodavimą, kad pateiktu nulių eiles kompaktiškai.

2) Aukščiausio dažnio ne nenuliniai koeficientai po zigzaginio peržiūros dažniausiai yra ±1 sekos ir CAVLC signalizuoja kelis aukštojo dažnio ±1 koeficientus patogiu būdu.

3) Keli nenuliniai koeficientai kaimyniniuose blokuose dažniausiai yra koreliuojami.Koeficientai yra užkoduojami pasitelkiant LUT ir LUT pasirinkimas priklauso nuo nenulinių koeficientų skaičiaus kaimyniniuose blokuose.

4) Nenulinių koeficientų dydis yra linkęs būti didesnis pradžioje ir mažesnis esant aukštiems dažniams. CAVLC tuo pasinaudoja ir pritaiko pasirinkimą VLC LUT lygio parametrams priklausomai nuo pastarojo kodavimo lygio dydžio.

Kadangi sistemos silpnoji vieta paprastai būna apskaičiavimo modulyje, CAVLC dekoderis kartu su nuosekliai sujungtu IQIT dekoderiu veikia kaip lygiagretus konvejeris kartu su apskaičiavimo moduliu.

3.6.8. Lusto atmintis

Yra du lusto atminčių tipai, registrų failai (RF) ir SRAM. RTL kodavime, visi RF yra pateikiami iš elgesio modulio ram_async_1r_sync_1w, tuo metu, kai visi SRAMs yra pateikiami iš elgesio modulio ram_sync_1r_sync_1w.

7 lentelė. RF panaudojimas

Vardas	**Aukštis**	**Plotis**	**priėjimas**
Intra4x4_PredMode_RF	11	16	Async nuskaitymas, sync įrašymas
LumaLevel_mbAddrB_ RF	11	20	Async nuskaitymas, sync įrašymas
ChromaLevel_Cb_mbAddrB_ RF	11	10	Async nuskaitymas, sync įrašymas
ChromaLevel_Cr_mbAddrB _RF	11	10	Async nuskaitymas, sync įrašymas
mvx_mbAddrB_RF	11	32	Async nuskaitymas, sync įrašymas
mvy_mbAddrB_RF	11	32	Async nuskaitymas, sync įrašymas
mvx_mbAddrC_RF	10	8	Async nuskaitymas, sync įrašymas
mvy_mbAddrC_RF	10	8	Async nuskaitymas, sync įrašymas

8 lentelė. SRAM panaudojimas

Vardas	**Aukštis**	**Plotis**	**priėjimas**
Intra_mbAddrB_RAM	88	32	Sync nuskaitymas, sync įrašymas
DF_mbAddrA_RAM	32	32	Sync nuskaitymas, sync įrašymas

Vardas	Aukštis	Plotis	priėjimas
DF_mbAddrB_RAM	352	32	Sync nuskaitymas, sync įrašymas
rec_DF_RAM0	96	32	Sync nuskaitymas, sync įrašymas
rec_DF_RAM1	96	32	Sync nuskaitymas, sync įrašymas

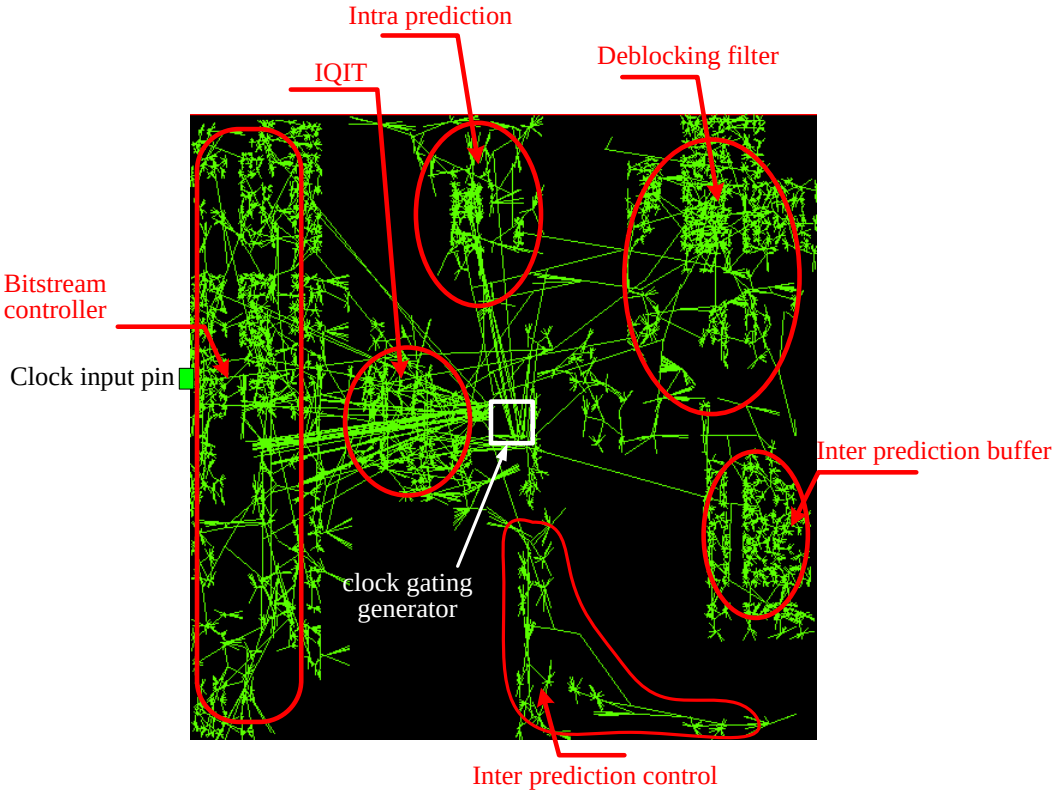
3.6.9. Periodas

Nova yra pilnai sinchroninis projektas su viena periodo įvestimi, kuri valdo visus DFF ir lusto atmintys SRAM.

9 lentelė. Periodas

Vardas	Šaltinis	Dažnis (MHz)			Pastabos	Aprašymas
		Maks	Min	Resolution		
clk	Įvestis	10	1	NA	Veikimo ciklas 50/50.	Vienas periodas visam projektui

Galutinė medžio schema yra pateikta apačioje:



13 Pav. Medžio schema

3.6.10. Įėjimo/išėjimo jungtys

Šis skyrius apibrėžia pagrindines įėjimo/išėjimo jungtis.

10 lentelė. Įėjimo/išėjimo jungtys

Jungtis	Plotis	Kriptis	Aprašymas
clk	1	įėjimas	Sistemos periodas
reset_n	1	įėjimas	Sistemos perkrovimas, žemas aktyvumas
BitStream_buffer_input	16	įėjimas	BitStream_buffer duomenų įvedimas
BitStream_ram_ren	1	išėjimas	BitStream_buffer nuskaitymo įgalinimas, žemas aktyvumas
BitStream_ram_addr	17	išėjimas	BitStream_buffer adresas
pin_disable_DF	1	įėjimas	Išoriškai įjungia/išjungia deblocking filtrą. =1 deblocking filtras yra išjungtas

Jungtis	Plotis	Kriptis	Aprašymas
			=0 iššifruotas bitstream nuspręs išjungi ar įjungti deblocking filtrą
freq_ctrl0	1	įėjimas	Dažnio valdymo įvestis
freq_ctrl1	1	įėjimas	Dažnio valdymo įvestis
pic_num	6	išėjimas	Žemas 6 bitų einamojo iššifravimo paveikslo skaičius. Derinimo tikslam
ext_frame_RAM0_data	32	įėjimas	Išoriniai RAM0 duomenys
ext_frame_RAM0_cs_n	1	išėjimas	Išorinio RAM0 pasirinkto lusto, žemas aktyvumas
ext_frame_RAM0_wr	1	išėjimas	Išorinio RAM0 įrašymo valdymas, aukštas aktyvumas
ext_frame_RAM0_addr	14	išėjimas	Išorinis RAM0 adresas
ext_frame_RAM1_data	32	įėjimas	Išoriniai RAM1 duomenys
ext_frame_RAM1_cs_n	1	išėjimas	Išoriniai RAM1 pasirinkto lusto, žemas aktyvumas
ext_frame_RAM1_wr	1	išėjimas	Išorinis RAM1 įrašymo valdymas, aukštas aktyvumas
ext_frame_RAM1_addr	14	išėjimas	Išorinis RAM1 adresas

4. REZULTATAI

4.1. Nova atlikimas

Nova buvo patikrintas FPGA aplinkoje ir ant tikro silicio:

- ✓ FPGA įgyvendinimo rezultatai yra pateikti apačioje:

11 lentelė. FPGA atlikimas

Prietaisas	Virtex-4 xc4vlx200-10ff1513
Skilčių skaičius	18,377 (panaudojimas 20%)
4-įėjimų LUTs skaičius	32,711 (panaudojimas 18%)
Loginių elementų skaičius	10,405,748

✓ ASIC įgyvendinimo rezultatai:

13 lentelė. ASIC igyvendinimas

Technologija	0.18µm CMOS 1P6M	
Tiekiama įtampa	1.8V branduolio, 3.3V Įėjimui/išėjimui	
Dydis	4.4×4.4mm^2	
Paketas	CQFP 208	
Projektavimo išlaidos	Ligoniai elementai	169K
	Atmintis	2.5K baitų SRAM
Veikimo dažnis	1.5MHz QCIF @ 30fps	
Išmatuota galia	293µW @ 1.0V, 973µW @ 1.8V	

4.2. Programos modeliavimas

1. Dešimt 300 kadrų trukmės QCIF vaizdo sekų yra panaudotos bandymams. Jos yra užkoduotos JM94 programine įranga (http://iphome.hhi.de/suehring/tml/).

14 lentelė. QCIF bandomų sekos

	KP (Kvantavimo parametras)	Kokybė (kb/s)	Bitai/kadras (Intra)	Bitai/kadras (Inter)	SNR Y (dB)	SNR U (dB)	SNR V(dB)
Mother & daughter	24	78.92	24,215	2,512	40.03	43.37	43.94
News	26	94.8	32,161	3,018	38.05	40.97	41.56
Akiyo	28	24.83	19,005	721	38.03	40.82	41.68
Claire	28	30.68	12,939	937	39.6	39.4	42
Foreman	28	130.24	21,911	4,177	36.16	40.54	41.64
Silent	30	65.06	21,350	2,058	34.19	37.7	38.99
Container	30	28.65	20,768	843	34.36	39.93	39.64
Hall	32	30.75	15,644	931	34.44	37.97	40.17
Coastguard	34	65.39	13,107	2,097	29.7	39.8	42.14
Carphone	36	45.19	10,418	1,431	30.69	36.635	37.135

2. JM94 programa pateikia .264 formato failus, kurie negali būti nuskaityti tiesiogiai į verilog. Failo formato pakeitimas, bin2hex.pl, yra pateiktas bandomajame kataloge.. Pavadinimas konvertuoto teksto failo turi būti nurodytas aštuntoje bin2hex.pl. eilutėje.

3. Teksto failas būtų skaitomas per Beha_BitStream_ram: $readmemh ("nova/test/bitstream/Testas.txt", BitStream_ram); Keisti failo vietą, jei nukreipta į kitus kelius.

4. Iššifruoti išėjimai yra sukuriami iš ext_frame_RAM0_wrapper ir ext_frame_RAM1_wrapper. “nova_display.log” yra naudojamas atvaizdavimui, tuo metu, kai “nova_MB_output.log” naudojamas derinimui.

5. C programa yra pateikta bandymų kataloge, kad paverstų tekstą atgal į dvejetainį.

6. Taip gauname “nova300.yuv” kuri galima paleisti per YUVViewer.

5. IŠVADOS

Šio darbo tikslas buvo išanalizuoti jau esamus metodus, išsiaiškinti jų pagrindinius bruožus bei galimybes. Darbe labiau susitelkta ties vaizdo dekoderio veikimo tyrimu. Pasirinkto H.264 vaizdo dekoderio gilesnė analizė ir tyrimas. Buvo įsiaiškintas jo veikimo principas. Ištirti pagrindiniai mechanizmai leidžiantys iššifruoti vaizdą. Lygiagretumas, konvejeris ir hierarchinis atminties valdymas yra trys plačiausiai taikomi metodai.

Išanalizuoti duomenų suspaudimo principai. Du fundamentalūs suspaudimo principai yra pertekliškumo ir nereikšmingumo mažinimas.

Atliktas H.264 standarto palyginimas su prieš tai buvusiais. Jis nepateikia nei vieno kodavimo elemento, kuris ženkliai būtu patobulintas palyginus su ankstesniais vaizdo kodavimo standartais. Tai daugiau yra maži patobulinimai, kurie suteikia esminį pagerėjimą. H.264 standartas dirba žymiai geriau už visus ankstesnius standartus, dėl padidėjusio kodavimo lankstumo ir sudėtingumo.

Pasinaudojant vienų iš pasirinktu H.264 vaizdo iššifravimo projektu, sėkmingai atliktas vaizdo iššifravimas.

Vaizdo iššifravimo pagrindinis panaudojimo tikslas yra šiandieniniai portatyvus prietaisai.

6. ŠALTINIAI IR LITERATŪRA

1. Saha, S. Image Compression – from DCT to Wavelets: A Review // ACM Crossroads Student Magazine, Nr. 4, 2000.

2. Mullen, K.T. The contrast sensitivity of human color vision to red-green and blue-yellow chromatic gratings. // J. Physiol, 1985, Nr. 359, 381-400 p.

3. Oppenheim, A.V. Discrete-Time Signal Processing 2nd ed. / A.V. 3, R.W. Shafer. Prentice Hall, 1999. 870 p.

4. Ifeachor, E.C. Digital Signal Processing. A Practical Approach / E.C. 4, B.W. Jervis. Addison-Wesley, 1993. 760 p.

5. Vatterli, M. Wavelets and Subband coding / M. 5, J. Kovacevic. Prentice Hall, 1995. 488 p. ISBN 0130970808

6. Nelson, M. Data Compression Book / M. 6, J.L. Gailly. Hungry Minds, 1995. 557 p.

7. T. Wiegand, G. J. Sullivan, G. Bjontegaard, A. Luthra, "Overview of the H.264/AVC video c oding standard", IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7, July 2003, pp.560-576.

8. T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, G. J. Sullivan, "Rate-constrained coder control and comparison of video coding standards", IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7, July 2003, pp.688-703.

9. S. H. Wang, et al., "A platfo rm-based MPEG-4 Advanced Video Coding (AVC) decoder with block level pipelining", Proceedings on IEEE ICICS-PCM , Vol. 1, 2003, pp. 51 –55.

10. M. Horowitz, A. Joch, F. Kosse ntini, A. Hallapuro, "H.264/AVC baseline profile decoder complexity analysis", IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7, July 2003, pp.704-716

11. V. Lappalainen, A. Hallapuro, T. D. Hamalainen, "Complexity of optimized H.26L video decoder implementation", IEEE Transactions of Circuits and Systems on Video Technology , vol. 13, pp. 717 - 725, July 2003.

12. Y. W. Huang, C. Y. Chen, C. H. Tsai, C. F. Shen, L. G. Chen, "Survey on block matching motion estimation algorithms and architectures with new results", Journal of VLSI Signal Pro cessing Systems, vol. 42, no. 3, pp. 297 – 320, March 2006.

13. C. J. Lian, P. C. Tseng, L. G. Chen, "Low-power and power-aware video codec design: an overview", China Communications, Oct. 2006, pp. 45 - 51.

14. H. C. Chang, et al., “A 7mW- to-183mW dynamic quality-scalable H.264 video encoder chip”, IEEE International Solid-State Circuits Conference, 2007, pp. 280 – 281.
15. M. Pedram, J. Rabaey, “Power aware design methodologies”, Kluwer Academic , 2002.
16. T. Sakurai, “Perspective on power-aware electronics”, IEEE International Solid-State Circuits Conference , 2003, vol. 1, pp. 26 - 29.
17. David A. Patterson, John L. Hennessy, “Computer Architecture: A Quantitative Approach, Second edition”, China Machine Press, 1999.
18. T. C. Chen, Y. W. Huang, L. G. Chen, “Analysis and design of macroblock pipelining for H.264/AVC VLSI architecture”, IEEE International Symposium on Circuit and Systems, May 2004, vol. 2, pp. 273 - 276.
19. H. Y. Kang, K. A. Jeong, J. Y. Bae, Y. S. Lee, S. H. Lee, “MPEG4 AVC/H.264 decoder with scalable bus architecture and dual memory controller”, IEEE International Symposium on Circuits and Systems, May 2004, pp. 145 – 148.
20. T. A. Lin, S. Z. Wang, T. M. Liu and C. Y. Lee, “An H.264/AVC decoder with 4x4 block level pipeline”, IEEE International Symposium on Circuits and Systems, May 2005, pp. 1810 – 1813.
21. I. E. G. Richardson, “H.264 and MPEG-4 video compression”, John Wiley & Sons Ltd., 2003.
22. OpenCores [žiūrėta 2012-04-08]. Prieiga per internetą: < http://opencores.org/project,nova>.