TARTU ÜLIKOOL

Loodus- ja tehnoloogiateaduskond

Tehnoloogiainstituut

Hendrik Türk

AES ALGORITMI REALISEERIMINE VHDL-IS NING TESTIMINE FPGA RIISTVARAL

Bakalaureusetöö (12 EAP)

Juhendaja:

MSc Margus Rosin

Tartu 2015

Resümee

Andmeturbeseadmeid tootvate ettevõtete pakutavad krüptokiirendid on suhteliselt kallid ning neid on keeruline konfigureerida. Bakalaureusetöö eesmärgiks on teha ettevalmistusi eraldiseisva krüptokiirendi loomiseks, mis suudaks võistelda olemasolevate krüptokiirendite hinna, jõudluse ja turvalisusega. Probleemi paremaks mõistmiseks uuriti krüpteerimise olemust, kasutatavaid algoritme ning krüptokiirendeid. Töö praktilise osana valmis AES algoritmi simulatsioon ning testiti krüpteerimisoperatsioone Basys 3 arendusplaadil.

Tulemustest selgus, et riistvaratestides kasutatud programmi loomisel on AES-i krüpteerimisalgoritm rakendatud õigesti ning selle edasine arendamine on põhjendatud. Edasisteks arendusteks nähakse AES algoritmi täiemahulist realiseerimist FPGA riistvaral, täielikku AES algoritmi kasutava prototüübiga reaalsete kiirus- ning turbetestide tegemist võrguliikluse krüpteerimisel ja dekrüpteerimisel ning saadud tulemuste võrdlemine integreeritud krüpteerimismooduleid või eraldiseisvaid krüptokiirendeid kasutavate võrguseadmete tulemustega.

Sisukord

Jooniste loetelu

Tabelite loetelu

Lühendid ja mõisted

AES (*Advanced Encryption Standard*) – NIST-i salajase võtmega krüpteerimismeetodi standard, mis kasutab 128, 192 ja 256-bitiseid võtmeid ning Rijndaeli algoritmi. Asendas 2001. aastani kasutusel olnud 3DES meetodi. AES võimaldab teostada krüpteerimist erinevalt 3DESi kolme tsükli asemel ühe tsükliga ning selle võti on pikem kui 3DES-i 168-bitine võti. [1]

ASCII (*American Standard Code for Information Interchange*) – standardne 7-bitine kooditabel inglise tähestiku ja teiste klaviatuuril esinevate sümbolite esitamiseks digitaalsel kujul. [2]

ASIC (*Application Specific Integrated Circuit*) – mingiks kindlaks otstarbeks, näiteks andmesideprotokolli, digikaamera või elektronmärkmiku jaoks projekteeritud kiip (pooljuht-mikroskeem). [3]

Asümmeetriline krüptograafia – sellise krüptoalgoritmi kasutamine, kus šifreerimiseks ja dešifreerimiseks kasutatakse erinevaid võtmeid. Avalik ja salajane võti täiendavad üksteist ning ühest ei saa tuletada teist. [4]

CAVP (*Cryptographic Algorithm Validation Program*) – krüptograafiliste algoritmide valideerimise programm.

DES (*Data Encryption Standard*) – populaarne sümmeetriliste võtmetega krüpteerimismeetod, mis töötati välja 1975. aastal ja millest 1981. aastal sai ANSI standard ANSI X3.92. DES kasutab 56-bitist võtit. 3DES meetodi puhul kasutatakse mitmekordset šifreerimist: andmed šifreeritakse ühe võtmega, dešifreeritakse teisega ja šifreeritakse uuesti kolmanda võtmega. [5]

FIPS (*Federal Information Processing Standards*) – krüptograafilisi algoritme ja teisi infotehnoloogilisi standardeid hõlmav kogu.

FPGA (*Field Programmable Gate Array*) – integraalskeem, mida saab programmeerida ka pärast tehasest väljastamist. Programmeeritavad ventiilmaatriksid sarnanevad programmeeritavatele püsimälukiipidele (PROM), kuid neil on palju laiemad kasutusvõimalused. Insenerid saavad programmeeritavaid ventiilmaatrikseid kasutada spetsialiseeritud integraalskeemide projekteerimiseks, mida hiljem võib tellida suurtes kogustes juba valmisühendustega. Ideaalis võiksid arvutikasutajad ise teha oma vajadustele vastavaid mikroprotsessoreid. [6]

IBM (*International Business Machines*) – rahvusvaheline tehnoloogia arendamisega tegelev korporatsioon.

Integraalskeem (*integrated circuit*) – mikrokiip.

LUT (*Look Up Table*) – otsingutabel, mis koosneb eeldefineeritud funktsiooni väärtustest.

SSL (*Secure Sockets Layer*) – infoturbe protokoll üle Interneti edastatavate andmete turvalisuse tagamiseks. Sõna "sokkel" viitab sellele, et andmete edasi-tagasi saatmine klient- ja serverprogrammi vahel toimub soklikihi programmi kaudu ja meenutab elektripirni pesasse sisse- ja väljakeeramist. SSL kasutab RSA kahe võtmega (avalik ja privaatvõti) krüpteerimissüsteemi. RSA süsteemi juurde kuulub ka digitaalne sertifikaat ehk isikutunnistus. SSL protokolli töötas välja Netscape ja seda kasutatakse laialdaselt näiteks krediitkaardiinfo edastamiseks elektrooniliste äritehingute puhul. [7]

Sümmeetriline krüptograafia – sümmeetriline šiffer on selline šiffer, mille puhul nii krüpteerimiseks kui dekrüpteerimiseks kasutatakse üht ja sama võtit. Seepärast on vaja leida turvaline meetod, mis võimaldaks enne krüpteerimist saatjal ja vastuvõtjal võtme osas omavahel kokku leppida. [8]

Šiffer – šifriks nimetatakse igasugust teksti krüpteerimise meetodit teksti loetavuse ja tähenduse varjamiseks. Terminit kasutatakse ka krüpteeritud tekstisõnumi enda kohta, kuigi sel juhul tuleks kasutada termineid "krüptogramm" või "šiffertekst". Sõna ise tuleb araabia keelest, kus see tähendab tühjust või nulli. [9]

NIST (*National Institute of Standards and Technology*) – Rahvuslik Standardite ja Tehnoloogia Instituut.

USB (*Universal Serial Bus*) – universaalne järjestiksiin. [10]

VHDL (*Very High Speed Integrated Circuit Hardware Description Language*) – IEEE (*Institute of Electrical and Electronic Engineers*) poolt standardiseeritud riistvarakirjelduskeel, mis sai 1980. aastatel alguse Ameerika Ühendriikide väga kiirete integraalskeemide loomise programmist.

1 Sissejuhatus

Üheks tähtsamaks krüpteerimisalgoritmide eesmärgiks on informatsiooni kaitsmine. [11] Teoreetiliselt on piisava ajalise ning arvutusliku ressursiga võimalik jõumeetodil dekrüpteerida kõik salakirjad. [12] Turvalisuse tagamiseks peavad krüptograafilised algoritmid olema piisavalt keerulised, et nendega krüpteeritud informatsiooni ei oleks võimalik mõistliku aja jooksul lahti murda. Keerulisus teeb algoritmid küll turvaliseks, kuid tihti on nende rakendamine olemasoleva riistvara jaoks koormav. [13]

Kuni II maailmasõjani kasutati krüptograafiat põhiliselt sõjalistel eesmärkidel. 20. sajandi keskel hakkasid andmete salastamisest rohkem huvituma ka erinevad tsiviilettevõtted. [14] Pankades, riigiasutustes ja firmades toimub pidev informatsiooni liikumine, mis peab olema turvaline, suure läbilaskvusega ning kuluefektiivne. Parema kiiruse tagamiseks kasutatakse serveritel eraldiseisvaid krüptokiirendeid või on krüptokiirendite funktsionaalsus ehitatud keskprotsessori arhitektuuri. [15]

1.1 Probleemi tutvustus

Krüptokiirendi on eraldiseisva protsessoriga põhisüsteemile lisatav laienduskaart, mis vabastab põhisüsteemi krüpteerimis- ja dekrüpteerimisülesannetest. Vabanenud arvutusjõudlus lubab põhisüsteemil keskenduda põhitööle ning suurendab andmete läbilaskvust. Seega lubavad krüptokiirendid konfidentsiaalset infot käitlevas ettevõttes luua kiiremad andmeside kanalid.

Krüptokiirendid on suhteliselt kallid ning nende seadistamine keeruline. Lisaks kasutatakse suuremates asutustes rohkem kui ühte krüptokiirendit vajavat seadet, mis muudab nende hinna ja konfigureerimise lihtsuse veelgi tähtsamaks. Kõik see teeb krüptokiirendite kasutuselevõtu kulukaks ning vähendab asutuste valmisolekut nendesse investeerida.

1.2 Töö eesmärk ja ülevaade

Bakalaureusetöö eesmärgiks on teha ettevalmistusi eraldiseisva krüptokiirendi loomiseks, mis suudaks võistelda olemasolevate krüptokiirendite hinna, jõudluse ja turvalisusega. Töö tulemuste põhjal järeldatakse, kas bakalaureusetöö käigus valitud meetodil on mõistlik jätkata krüptokiirendi arendamist ning pakutakse edasised tegevused selle loomiseks. Järelduste tegemiseks tuleb luua turvalist krüpteerimis- ja dekrüpteerimisalgoritmi kasutav programm, simuleerida programmi suhtlust väliskeskkonnaga ning testida loodud programmi sobival riistvaraplatvormil.

Bakalaureusetöö sisuline osa jaotub neljaks struktureeritud peatükiks. Esimeses räägitakse põgusalt krüptograafia ajaloost, uuritakse enim kasutatavaid krüpteerimisalgoritme ning tutvustatakse riistvaraliste krüptokiirendite olemust. Teises kirjeldatakse kasutatavat riistvara, tarkvara ning töö käiku. Sisulise osa kolmandas ja neljandas peatüki moodustavad saadud tulemused ning nende põhjal tehtud järeldused.

2 Ülevaade probleemist

Probleemi paremaks mõistmiseks tuleb omada ülevaadet krüpteerimisalgoritmidest ja riistvaralistest krüptokiirenditest. Ülevaate saamiseks ei ole mõistlik kirjeldada kõiki olemasolevaid riistvaralahendusi ja šifreid. Lisaks ei ole vajadust võrrelda erinevate tootjate poolt pakutavaid krüptokiirendid, kuna nende jõudlus ja maksumus sõltub neid kasutavatest süsteemidest. Siiski tuleks välja tuua mõned riistvaraliste krüptokiirendite tootjad ning uurida nende poolt pakutavaid riistvaralahendusi ja levinumaid krüpteerimisalgoritme. Kirjeldatavate riistvaralahenduste ning krüptograafiliste algoritmide valimisel on lähtutud ka nende kirjelduste kättesaadavusest ja mõjust tehtud tööle.

2.1 Krüptograafia areng

Krüptograafia võimalusi on turvaliseks andmeedastuseks kasutatud tuhandeid aastaid. Enne digitaalset ajastut oli krüptoloogia arengu suurimaks mõjutajaks sõjapidamisega kaasnev militaarne kommunikatsioon. Sellega seonduvalt kasutati krüptograafia võimalusi põhiliselt sõjanduses ja diplomaatias. 1960. aastatel loodud ning sajandi lõpus populaarsust kogunud Internet muutis turvalise andmevahetuse paljudele igapäevaseks probleemiks. Kindlustunde tagamiseks on loodud mitmeid informatsiooni krüpteerimise standardeid, mille algoritme pidevalt uuritakse ning testitakse. [12]

Tänapäeval on krüptograafia põhilisteks eesmärkideks kaitsta edastatava sõnumi salastatust, terviklikkust, õigsust ning panna sõnumi saatja sõnumi sisu eest vastutama. Krüpteerimisalgoritmid jagunevad sünkroonseteks ning asünkroonseteks. Sünkroonsed krüpteerimisalgoritmid kasutavad ühte salajast võtit nii andmete šifreerimiseks kui dešifreerimiseks. Asünkroonsete algoritmide puhul kasutatakse andmete krüpteerimiseks avalikku võtit ning dekrüpteerimiseks salajast võtit. Vajadus asünkroonsete šifrite järele tekkis digitaalsel ajastul, kus ei ole alati võimalik sümmeetrilist võtit turvaliselt jagada. [16]

2.2 Krüpteerimine

Krüpteerimine ehk šifreerimine on informatsiooni muutmine loetamatuks, kasutades selleks võtit ning krüpteerimisloogikat. Kasutatava võtme suurus, keerukus ja rakendamise meetod sõltub krüpteerimisalgoritmi parameetritest. Krüptograafilised algoritmid jaotuvad privaatse võtmega sümmeetrilisteks ning privaatse ja avaliku võtmega asümmeetrilisteks šifriteks. Šifreerimise pöördtegevuse dešifreerimise käigus taastatakse algsed andmed. [12]

Šifreerimise algvõtteteks on substitutsioon ehk asendusmeetod ning transpositsioon ehk sümbolite järjekorra muutmise meetod. Pelgalt asendus- või järjekorrameetodil põhinevaid

algoritme on lihtne kasutada, kuid nende nõrkusteks on krüpteerimise käigus tekkivad mustrid. Seaduspärasusi põhjustab erinevate tähemärkide ja -ühendite esinemise sagedus esialgses tekstis. [17] Nii on võimalik loogiliste järelduste põhjal ning krüpteerimisvõtit kasutamata salastatud tekst dešifreerida. Šifrite lahti murdmise ja nende analüüsimise protsessi kutsutakse krüptoanalüüsiks. [18]

Turvalised krüpteerimisalgoritmid peavad arvestama lisaks krüpteerimise käigus tekkivatele mustritele ka saadaoleva arvutusliku võimsusega, mida pakuvad meile erinevad integraalskeemid. Seega on asendus- või rotatsioonimeetodit mõistlik kasutada mitte eraldi krüpteerimisalgoritmina, vaid keerukama algoritmi ühe osana. Teoorias on võimalik kõik krüpteeritud tekstid jõumeetodil lahti murda. Sisuliselt tähendab see võtme leidmist võtmeruumi täielikul läbivaatusel. Usutakse, et hetkel turvalisi krüpteerimisalgoritme nagu AES ei ole võimalik tavalise transistorarvuti abil mõistliku aja jooksul lahti murda. Küll peaks see olema võimalik tulevikus kvantarvutitega. [12]

2.3 Krüpteerimisstandardid ja -algoritmid

Sümmeetrilised krüpteerimisalgoritmid jagunevad plokk- ja jadašifriteks. Sümmeetriliste plokkšifrite puhul jaotatakse krüpteeritavad andmed plokkidesse. Igale plokile rakendatakse võtit ning krüpteerimisloogikat eraldi ning selle tulem sõltub maksimaalselt ainult eelneva ploki krüpteerimisest. Jadašifrite puhul jaotataks krüpteeritavad andmed väiksematesse plokkidesse ning šifreerimisel sõltub iga järgnev plokk eelnevatest plokkidest. Jadašifrid on üldiselt kiiremad, kuid neid peetakse plokkšifritest vähem turvalisteks. [19] Plokkšifritel põhinevad algoritmid on seetõttu moodsas krüptograafias laialdasemalt levinud. Sümmeetriliste šifrite turvalisus sõltub lisaks krüpteerimisalgoritmile ka kasutatava võtme efektiivsest pikkusest ning selle juhuslikkusestJoonis 3. [20]

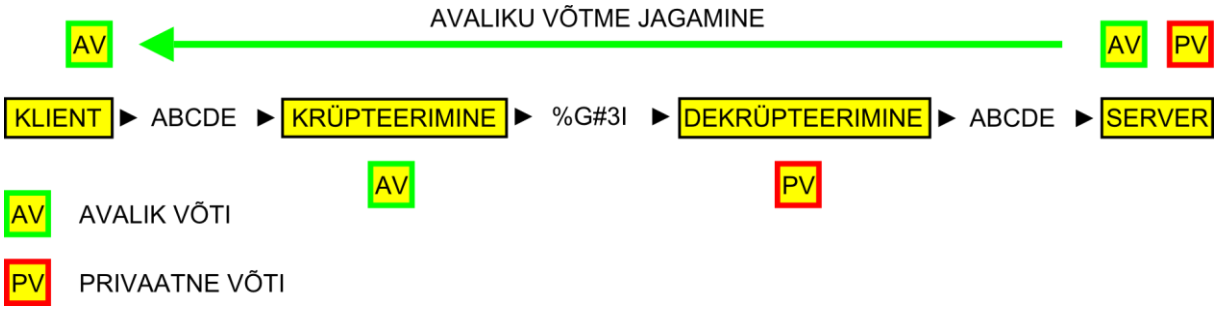
Asümmeetriliste krüpteerimisalgoritmide loogika põhineb erinevate algebraliste ja arvuteoreetiliste ülesannete lahendamisel. Seetõttu on need sümmeetrilistest krüpteerimisalgoritmidest aeglasemad. Tihti kasutatakse asümmeetrilisi algoritme turvaliseks sümmeetrilise algoritmi võtme edastamiseks. [21]

2.4 Krüpteerimisalgoritmid serverisüsteemides

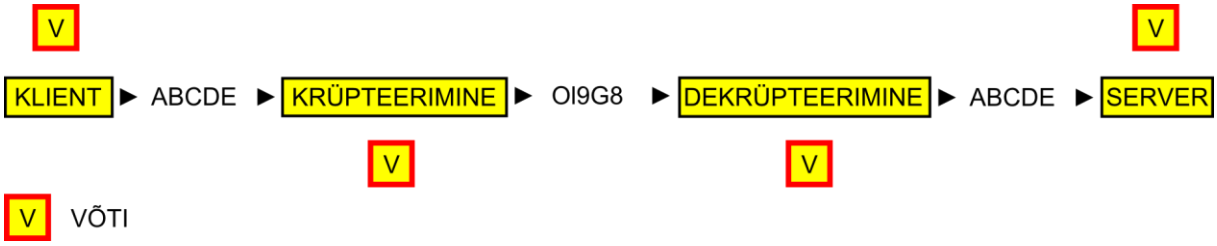
SSL on kliendi ja serveri vahelise krüpteeritud ühenduse loomise tehnoloogia. Turvaline ühendus võimaldab jagada paroole, teksti- ja meediafaile ning muud informatsiooni, mida soovitakse salajas hoida. [22]

Krüpteeritud SSL ühenduse loomiseks saadab server kliendile asümmeetrilise avaliku võtme. Klient kasutab saadud avaliku võtit sümmeetrilise sessioonivõtme krüpteerimiseks, mis saadetakse uuesti serverisse. Server dekrüpteerib selle asümmeetrilise privaatvõtmega. Edasi saavad klient ja server suhelda sümmeetrilist krüpteerimisalgoritmi kasutades. [22]

Seega kasutatakse asümmeetrilist krüpteerimisalgoritmi tavaliselt sümmeetrilise võtme turvaliseks jagamiseks. Põhilise andmevahetuse krüpteerimiseks kasutatakse sümmeetrilisi algoritme, kuna need on reeglina asümmeetrilistest kiiremad. Populaarseteks sümmeetrilisteks krüpteerimisalgoritmideks on AES ning vananenud 3DES. Asümmeetriliseks krüpteerimisalgoritmiks kasutatakse tihti RSA-d. Asümmeetrilise võtmega krüpteerimisalgoritmi kasutamist kujutatakse Joonis 1 ning sümmeetrilise võtmega krüpteerimisalgoritmi selgitava skeemi leiab Joonis 2. [22]



Joonis 1. Asümmeetrilise võtmega šifri selgitav skeem



Joonis 2. Sümmeetrilise võtmega šifri selgitav skeem

DES on 1970. aastatel IBM-i poolt arendatud ning Ameerika Ühendriikide Rahvusliku Standardite ja Tehnoloogia Instituudi defineeritud sümmeetriline plokkšiffer. Avaldamise ajal oli see esimene avalikkusele kättesaadav kõrge turvalisusega standardiseeritud krüpteerimisloogika. DES-i standardit uuendati kolmel korral kuni 1999. aastani. Hiljem leiti, et see, 56-bitise efektiivse võtmepikkusega algoritm ei ole enam piisavalt turvaline. [23] Alates 1990. aastate lõpust on teostatud mitmeid edukaid DES algoritmiga krüpteeritud tekstide lahti murdmise operatsioone. Tavaliselt kasutatakse võtmete leidmiseks ASIC või FPGA riistvaral põhinevaid süsteeme. [24]

USA valitsusel on kuni 2030. aastani lubatud kasutada 3DESi, mis on sisu poolest kolmekordne DES-i järjestikune rakendamine. [23] Hetkel peetakse 3DESi turvaliseks, kuid aeglaseks šifreerimisloogikaks. Vana DES standardi väljavahetamiseks kuulutati 1997. aastal välja AES konkurss täiustatud krüpteerimisalgoritmi leidmiseks. [25]

AES on 2001. aastal välja kuulutatud Rijndaeli loogikat kasutav DES-i järeltulija. Sarnaselt DES-ile on uus standard avalikkusele kättesaadav sümmeetriline plokkšiffer. AES algoritmis töödeldakse 128-bitiseid andmeplokke 128-, 192- või 256-bitise võtmega. Pärast standardistaatuse saavutamist on täiustatud krüpteerimisstandard muutunud kõige laialdasemalt kasutatavaks krüpteerimisalgoritmiks. [25]

Suur populaarsus teeb AES-i ahvatlevaks krüptoanalüüsi sihtmärgiks. Siiani ei ole veel teadaolevalt ühegi võtmepikkusega AES algoritmi lahti murtud. 128-bitise võtmepikkuse puhul on kõikide võtmekombinatsioonide koguarv 2^{128}. Selline kombinatsioonide arv tähendab, et kasutades kahendarvudel töötavat superarvutit, on 128-bitise võtme leidmiseks kuluv aeg võrreldav kogu teadaoleva universumi eksisteerimise ajaga. [26]

RSA on 1977. aastal publitseeritud asümmeetriline krüpteerimisloogika. See šiffer on mõeldud andmete salajasuse tagamiseks ning ehtsuse tõestamiseks. Sarnaselt AES-i ja DES-iga on ka RSA-d põhjalikult analüüsitud. Ainult korrektselt kasutatud RSA algoritmi võib pidada turvaliseks. Igapäevaselt kasutatakse RSA-d serverites ja brauserites turvalise internetiliikluse tagamiseks. [27]

2.5 Ülevaade riistvaralistest krüptokiirenditest

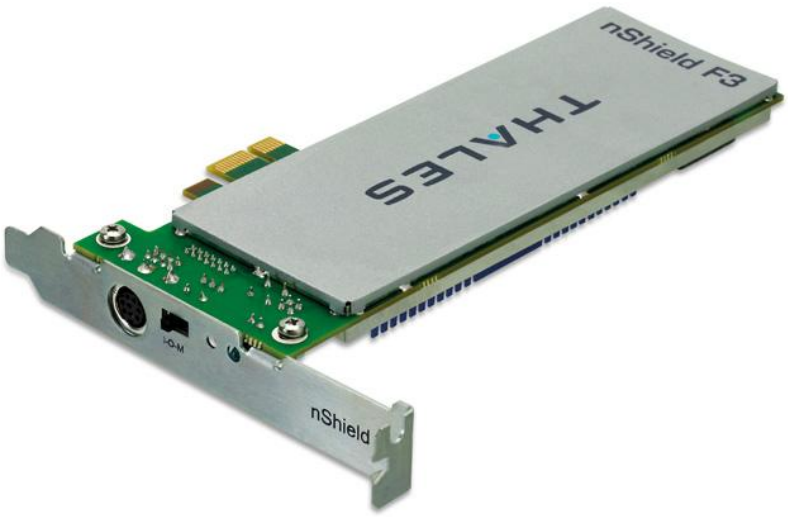
Riistvaralised krüptokiirendid on krüpteerimisjõudluse suurendamise eesmärgil põhiliselt serverisüsteemidesse lisatavad riistvaramoodulid. [28] Taolisi seadmeid kasutatakse SSL ühenduste kiirendamiseks, digitaalallkirjade autentimiseks, andmebaasides olevate andmete turvaliseks krüpteerimiseks ja teisteks sarnasteks protseduurideks. Lai kasutusvaldkond muudab riistvaraliste krüptokiirendite konfigureerimise aeganõudvaks ning eeldab eelteadmisi nii kasutatava riistvarakiirendi kui põhisüsteemi kohta. Krüptokiirendite valimisel tuleks jälgida, et seadmetes kasutataks laialdaselt levinud krüpteerimisalgoritme, kuna nii on täpselt teada kasutatavate algoritmide tugevused ja nõrkused. [29]

Riistvaraliste krüptokiirendite suurimaks nõrkuseks on nende hind, mis võib ulatuda tuhandete eurodeni. [30] Tihti ei avalikusta seadmete tootjad kogu informatsiooni nende turvaseadmetes kasutatud tehnoloogiate kohta. Mõned turvavead võivad välja tulla alles pärast seadme

soetamist. Kui turvaviga leitakse kasutatavates krüpteerimismoodulites, on vigade eemaldamine või seadmete uuendamine kulukas. [29]

Krüpteerimismoodulites on loogikatehete tegemiseks tavaliselt lühikese latentsusajaga FPGA mikrokiibid. Andmete krüpteerimisel ja dekrüpteerimisel võib korrektselt programmeeritud FPGA seade olla tavalisest mikroprotsessoriga seadmest tunduvalt kiirem. Kui mikroprotsessorite viiteaegu mõõdetakse millisekundites, siis FPGA integraalskeemides kasutatakse latentsuse mõõtmiseks mikrosekundeid. [31]

Ühed suuremad riistvaralisi andmeturbelahendusi pakkuvad ettevõtted on Thales ja SafeNet. Nende tootevalikus on lisaks krüptokiirenditele ka teisi serverisüsteemidele mõeldud seadmeid. [13] Mõlema ettevõtte erinevad tooteseeriad kasutavad laia valikut krüpteerimisalgoritme, mille hulka kuuluvad ka 3DES ja AES sümmeetrilised krüpteerimisstandardid ning osade toodete puhul erinevad asümmeetrilised algoritmid nagu RSA. [32]



Joonis 3. Illustreeriv pilt Thales'i nShield Solo seeria krüptokiirendist

Thales'i tootevalikust on käesoleva töö eesmärkidele kõige ligilähedasema funktsionaalsusega nShield Solo seeria krüpteerimismoodulid. Need FIPS 140-2 sertifikaadiga serverisüsteemidele mõeldud PCI-E laienduskaardid on võimelised kasutama AES, Aria, Camelia ja 3-DES sümmeetrilisi ning RSA, Diffle-Hellman, DSA ja ECC Suite B asümmeetrilisi algoritme. Kirjeldusi seadmete riistvaralistest lahendustest tootja poolt avalikustatud ei ole. [32]

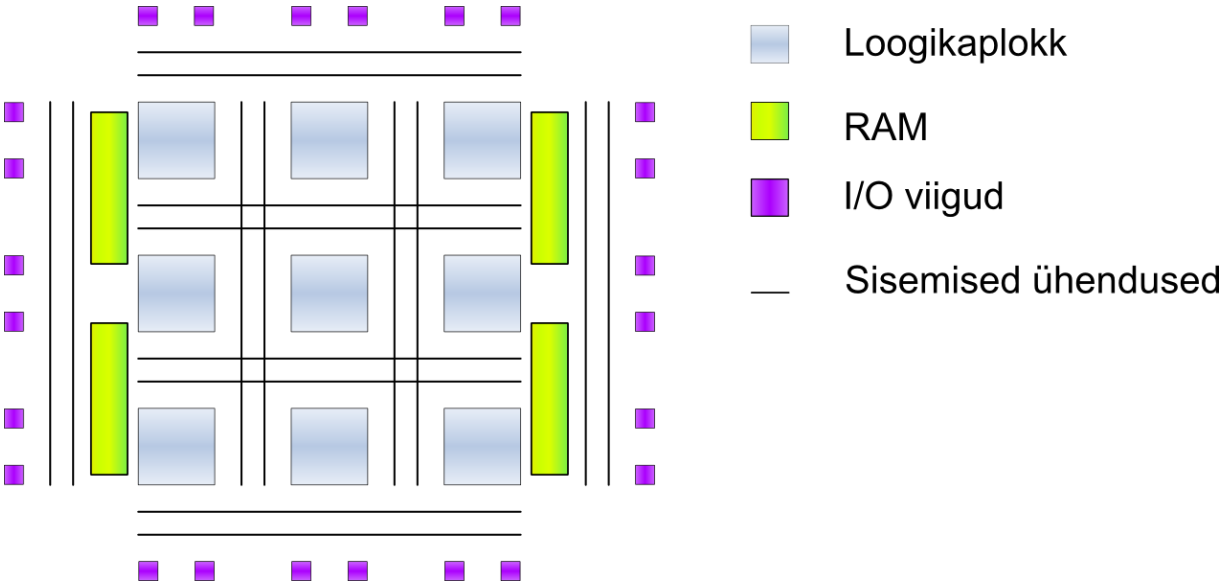
SafeNet'i krüpteerimismoodulite hulgast leiame Thales'i nShield solo seeriaga sarnased FIPS 140-2 sertifikaadiga Luna PCI-E laienduskaardid. Luna PCI-E moodulid on võimelised kasutama AES, RC2, RC4, RC5, CAST, DES, 3DES, ARIA ja SEED sümmeetrilisi ning RSA, DSA, Diffie-Hellman ja KCDSA asümmeetrilisi algoritme. Vaadeldavate Luna seadmete riistvaraliste lahenduste kirjeldusi tootja poolt avalikustatud ei ole. [33]

Lisaks eelnevalt nimetatud ettevõtetele leiame erinevaid krüpteerimislahendusi ja krüptokiirendeid ka Inteli tootevalikust. Uued Inteli serveriklassi Xeon protsessorid toetavad AES-NI tehnoloogiat, mis võimaldab varasemates serverites kasutatavatest protsessoritest kordades kiiremat AES algoritmi rakendamist. [34] Inteli poolt pakutavate krüptokiirendite hulgast leiame Inteli QuickAssist tehnoloogial põhinevad seadmed, mis pakuvad Thales'i ja SafeNet'i krüptokiirenditega sarnast funktsionaalsust. [35]

2.6 FPGA

FPGA on seadistatavatest loogikaplokkidest (*Configurable Logic Block*), programmeeritavatest vaheühendustest, sisend-väljundviikudest ning mälust koosnev integraalskeem. [36] Loogikaplokist leiab lülitite maatriksi ning käesoleva töö raames kasutatavast FPGAst neli tükki (*slice*). Iga tükk jaotub enamlevinult nelja sisendi ja ühe väljundiga LUT-ideks (*Look Up Table*) ja mäluelementideks. Iga LUT suudab realiseerida mistahes kahendkoodis funktsiooni, millel on neli sisendit ja üks väljund. Programmeeritavad vaheühendused juhivad loogikaplokkide ning sisend-väljundviikude vahelisi signaale. Tavaliselt võimaldavad FPGA-de programmeerimiskeskkonnad automatiseeritud vaheühenduste seadistamist. [36]

FPGA kiipides olev mälu jaotub plokk (*block*) RAM-iks ja jagatud (*distributed*) RAM-iks. Jagatud RAM kasutab mälu realiseerimiseks loogikaplokke, mis võimaldab paindlikku mälu jaotamist, kuid suurte andmehulkade puhul muutub see ebaefektiivseks. Plokk RAM on FPGA-s eraldiseisev element, mida saab kasutada ainult mäluna. Tavaliselt paigutatakse plokk RAM-idesse andmed, mida ei ole mõttekas nende suuruse või tüübi tõttu jagatud RAM-is hoida. [37]



Joonis 4. FPGA sisemist ehitust illustreeriv skeem

Šifrite murdmisel ja nende rakendamisel on arvutuslikult kõige nõudlikumad pidevalt korduvad lihtsad operatsioonid. [38] Tavalistes mikrokontrollerites toimub ülesannete lahendamine lineaarselt. Ühes FPGA integraalskeemis on korraga võimalik konfigureerida mitu paralleelselt töötavat arvutusmasinat. Nii saab sama ajaühiku jooksul, mis kulub ühe lihtsa operatsiooni tegemiseks tavalisel mikrokontrolleril, lahendada mitu samasuguse raskusastmega operatsiooni FPGA-l. [39]

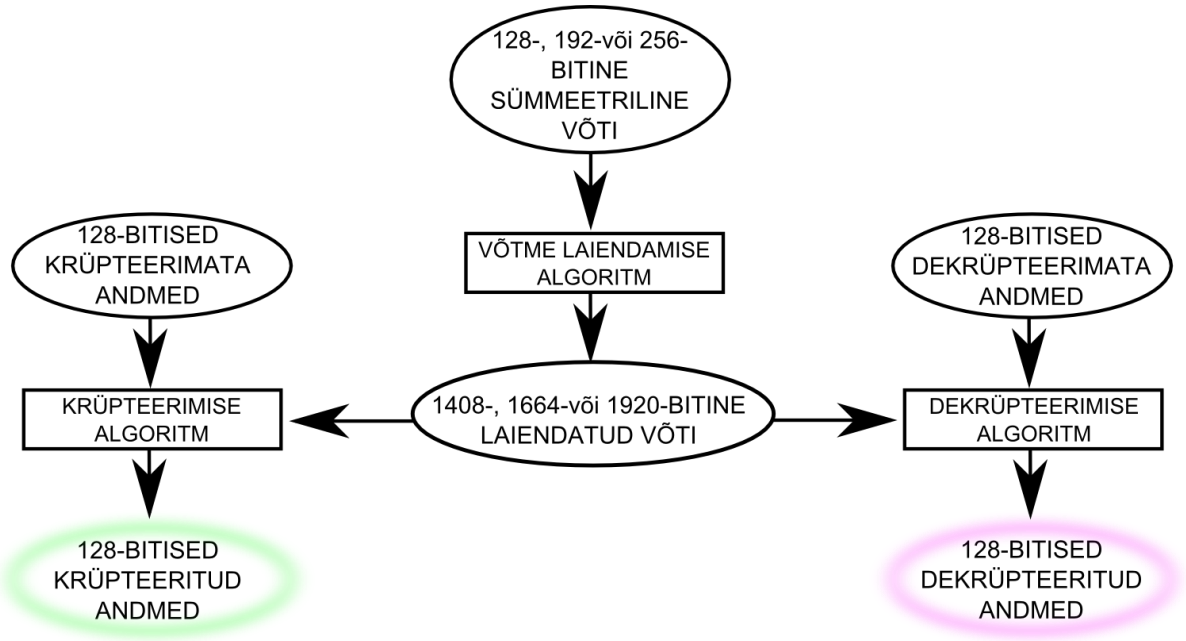
Lisaks ühe ülesande paralleelsele lahendamisele võimaldab FPGA arhitektuur sisemise loogika optimeerimist ning muutmist vastavalt vajadusele. Selline paindlikkus lubab ühte seadet kasutada mitme erineva ülesande lahendamiseks. Teiste sarnaste süsteemide (nagu ASIC) optimeerimine või muutmine tähendaks tihtipeale riistvarakomponentide väljavahetamist. Uute komponentide ostmine nõuab rohkem aega ning raha kui FPGA lõppkasutaja poolt ümberprogrammeerimine. Lihtne seadistamine ning paindlikkus teevad FPGA seadmed heaks prototüüpimise ja arendustöö keskkonnaks. [40]

Riistvaralisi andmeturbelahendusi pakuvad ettevõtted, nagu Thales, kasutavad FPGA integraalskeeme nende poolt pakutavates krüptokiirendites krüpteerimisoperatsioonide läbiviijatena. Kõike eelnevat arvesse võttes tundub otstarbekas ka õppe eesmärgil seadme loomisel kasutada FPGA-d. [41]

3 Metoodika

Eelneva põhjal võib väita, et olemasolevatest sümmeetrilistest krüptograafilistest algoritmidest on AES üks turvalisemaid ning krüptokiirendite seas laialdaselt levinud. Töö käigus uuritakse AES krüpteerimisstandardit, luuakse sellele simulatsioon Vivado programmeerimis- ja simuleerimiskeskkonnas ning testitakse tehtud AES programmi Artix-7 FPGA-d kasutaval Basys 3 arendusplaadil.

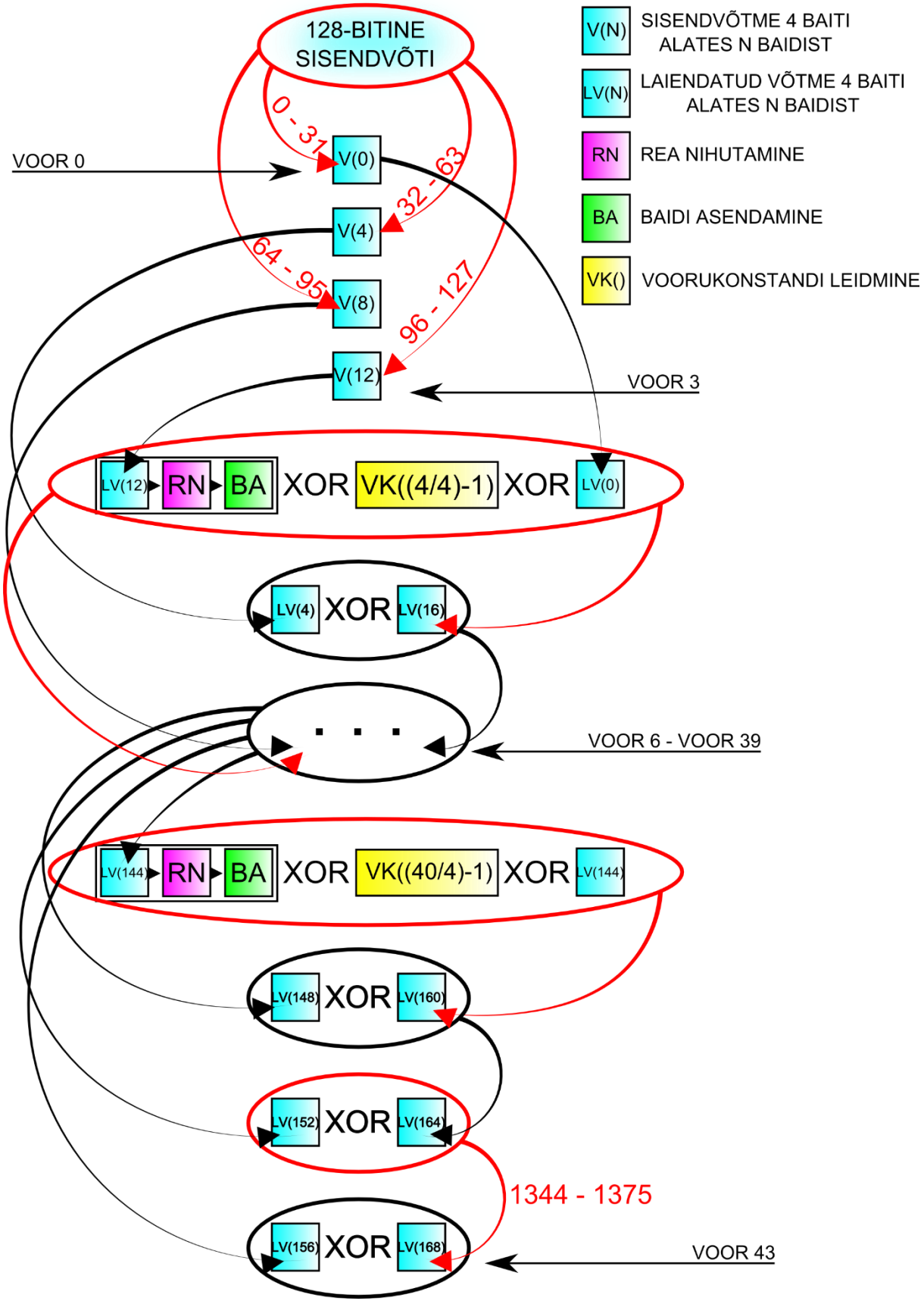
AES algoritm koosneb võtme laiendamise, krüpteerimise ning dekrüpteerimise komponentidest. Võtme laiendamine on eeltegevuseks nii krüpteerimis- kui dekrüpteerimisoperatsioonidele. Krüpteerimine ja dekrüpteerimine jaotuvad olenevalt võtme pikkusest kümneks, kaheteistkümneks või neljateistkümneks vooruks, kus igas voorus kasutatakse ühte 128-bitist laiendatud võtme vektorit. AES algoritmi ülevaatliku skeemi leiab Joonis 5.



Joonis 5. AES algoritmi ülevaatlik skeem

Algoritmi arvutusliku osa lihtsustamiseks kasutatakse mitmeid eelnevalt välja arvutatud asendustabeleid. Asendustabelitest info leidmine on sarnane põhikoolis korrutustabeli kasutamisega. Kasutatud asendustabelid leiab lisadest 2, 3, 4 ja 5.

3.1 AES-i võtme laiendamise algoritm

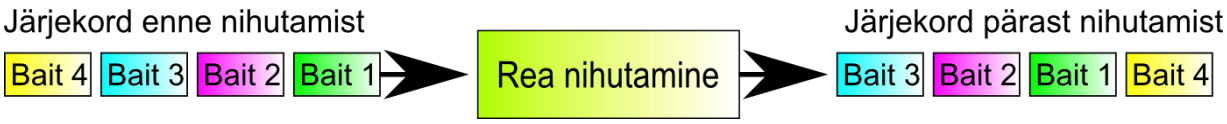


Joonis 6. 128-bitise võtme laiendamist selgitav skeem

Võtme laiendamise algoritmiga saadakse esialgsest võtmest kordades pikem laiendatud võti. Uus pikk võti eraldab iga AES-i krüpteerimise või dekrüpteerimise vooru jaoks erineva lühema 128-bitise võtme. Taoline võtme kasutamine peaks tõstma krüpteerimisalgoritmi turvalisust, kuna vähendab mustrite tekkimise võimalust. 128-, 192- või 256-bitine sisendvõti laiendatakse vastavalt 1408-, 1664- või 1920-bitini. 128-, 192- ja 256-bitiste sisendvõtmete korral jaotub võtme laiendamine kas neljakümne neljaks, viiekümne kaheks või kuuekümneks vooruks. Olenevalt sisendvõtme pikkusest jagatakse esimestes voorudes see kas neljaks, kuueks või kaheksaks võrdseks 32-bitiseks osaks. Nendest saab laiendatud võtme esimesed 128, 192 või 256 bitti. Igas järgnevas võtme laiendamise voorus leitakse laiendatava võtme järgmised 4 baiti ehk 32 bitti.

Tulenevalt sisendvõtme pikkusest kasutatakse võtme laiendamisel kas iga nelja, kuue või kaheksa vooru järel rea nihutamise, baidi asendamise ja voorukonstandi leidmise funktsioone. Vahepealsetes voorudes, välja arvatud neljas, kuues või kaheksas esimeses (olenevalt sisendvõtme pikkusest), teostatakse eelmise ning kas neli, kuus või kaheksa vooru tagasi leitud vooru tulemuste vahel XOR tehe. Võtme laiendamise ülevaatliku skeemi 128-bitise sisendvõtme korral leiab Joonis 6, kus voorude nummerdamist on alustatud nullist.

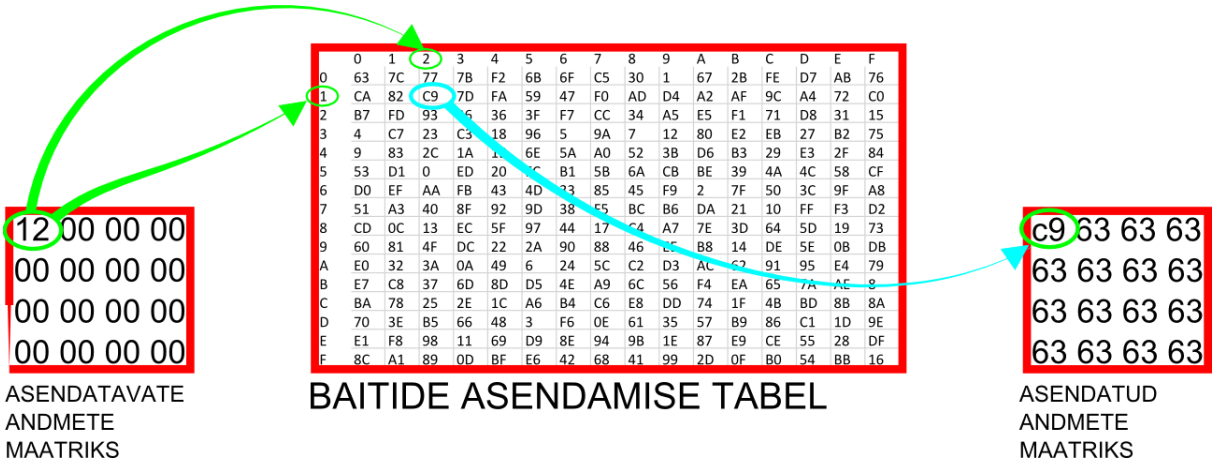
AES-i võtme laiendamise algoritmi paremaks mõistmiseks tuleb defineerida kaks lisafunktsiooni, mida saab kasutada võtme eri piirkondadest neljabaidiliste ehk 32-bitiste sõnade leidmiseks. Need funktsioonid on Joonis 6 märgitud kui V ja LV. Võtme laiendamisel kasutatavate funktsioonide täpne kasutamise skeem 128-bitise sisendvõtme korral on leitav lisast 1. [42, 43]



Joonis 7. Võtme laiendamisel kasutatava rea nihutamise operatsiooni illustreeriv skeem

Rea nihutamise funktsiooni sisendiks on algse või juba laiendatud sisendvõtme neljast baidist koosnev rida. Võtme laiendamisel tehakse rea nihutamise operatsioone ainult nende nelja baidiga. Baitide sees olevaid bitte ühe rea nihutamise operatsiooniga ümber ei tõsteta, muudetakse vaid baitide järjekorda. Õige tulemuse saamiseks nihutatakse sissevõetud baitide järjestust vastavalt Joonis 7. Lisas 1 ja Joonis 6 on rea nihutamise funktsioon märgitud lühendiga RN. [42, 43]

Baitide asendamise funktsiooni sisendiks on algse või juba laiendatud sisendvõtme neljast baidist koosnev rida. Joonis 8 kujutab näidet asendusmaatriksi kasutamisest. Sama asendusmaatriksi suurendatud kujul leiab lisast 2. Lisas 1 ja Joonis 6 on baidi asendamise funktsioon märgitud lühendiga BA. [42, 43]



Joonis 8. Ülevaatlik skeem baitide asendamise tabeli kasutamisest

Voorukonstandi funktsiooni sisendi leidmise valem:

$$Sisend = \left(\frac{Voor}{\frac{VõtmePikkus}{4}} \right) - 1 \quad (1)$$

Voorukonstandi leidmise funktsioon väljastab valemist 1 saadud sisendi põhjal eeldefineeritud konstandi. Eeldefineeritud konstandid ja nendele vastavad sisendväärused on kujutatud tabelis 1, kus sisenditeks on kümnendarvud nullist neljateistkümneni ning konstantideks neljabaidised kuueteistkümnendarvud. Voorukonstanti kasutatakse perioodiliselt võtme laiendamise algoritmis vooru tulemusega XOR tehte tegemisel. Funktsiooni väljakutsumise tihedus sõltub sisendvõtme pikkusest. Lisas 1 ja Joonis 6 on voorukonstandi leidmise funktsioon märgitud lühendiga VK. [42, 43]

Sisend	Vastav konstant
0	01000000
1	02000000
2	04000000
3	08000000
4	10000000

Sisend	Vastav konstant
5	20000000
6	40000000
7	80000000
8	1B000000
9	36000000

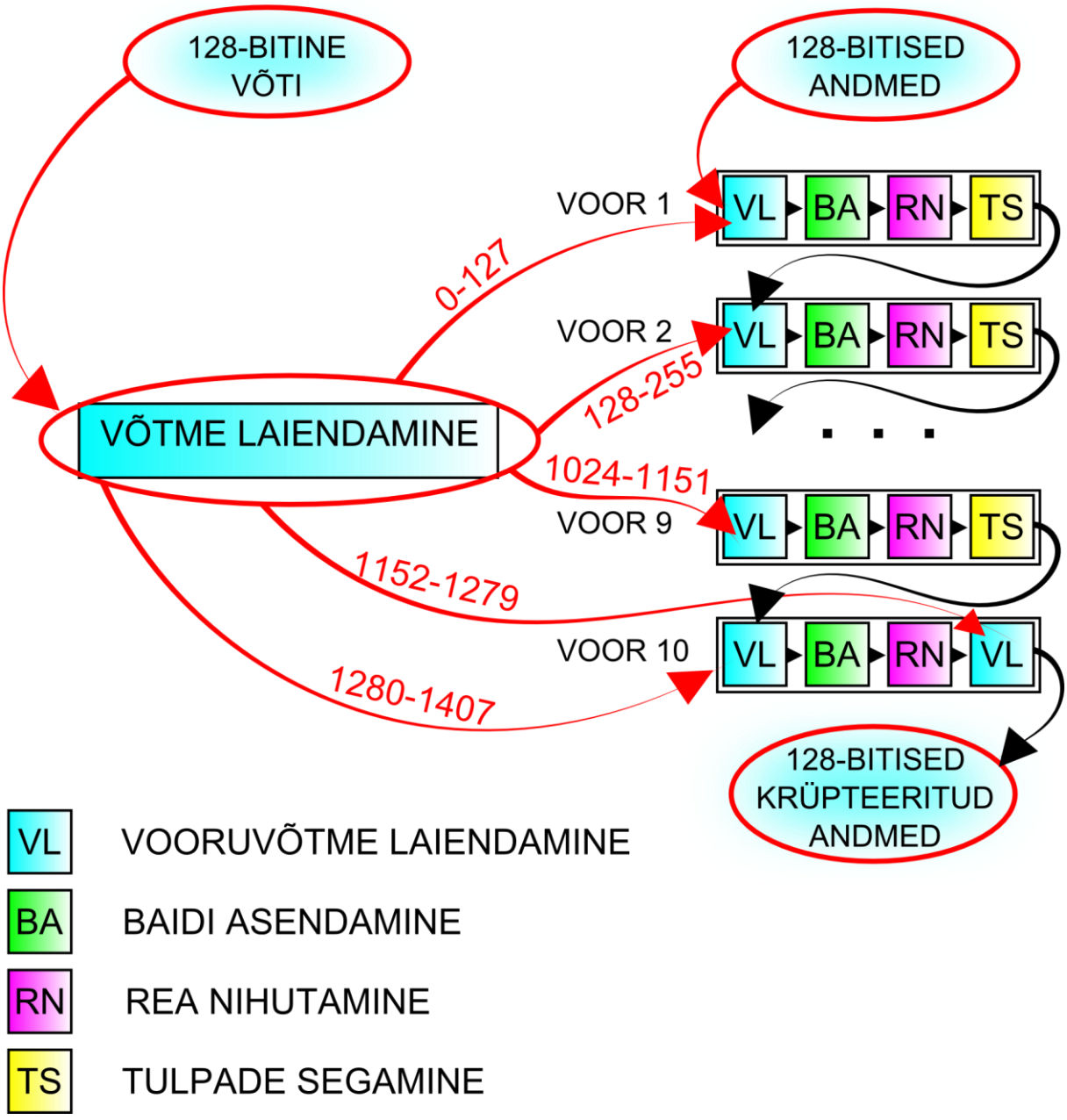
Sisend	Vastav konstant
10	6C000000
11	D8000000
12	AB000000
13	4D000000
14	9A000000

Tabel 1. Voorukonstandi funktsiooni sisendid ja nendele vastavad konstandid

Lisas 1 ja Joonis 6 kasutatavad funktsioonid V ja LV on mõeldud algsest võtmest ning juba laiendatud võtme osadest neljabaidiste ridade leidmiseks. Neid kasutatakse kõigis võtme

laiendamise voorudes. Funktsioonidega V ja VL leitavateks ridadeks on varasemate laiendusvoorude 32-bitised tulemused. [42, 43]

3.2 AES-i krüpteerimise algoritm



Joonis 9. AES algoritmi krüpteerimisprotsessi ülevaatlik skeem

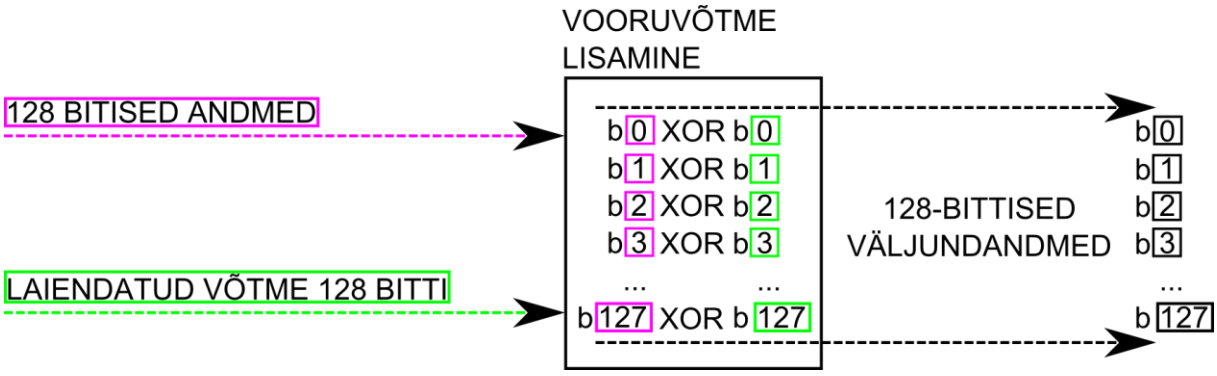
AES-i krüpteerimisalgoritm kasutab vooruvõtme lisamise, baidi asendamise, rea nihutamise ja tulpade segamise funktsioone, mis kokkupanduna moodustavad ühe krüpteerimisvooru. Krüpteerimisvoorude arv oleneb kasutatava võtme pikkusest. Krüpteerimisvoorude arvuline sõltuvus sisendvõtme pikkusest on leitav tabelist 2. AES krüpteerimisalgoritm algab alati vooruvõtme lisamisega, millele järgnevad baidi asendamine, rea nihutamine ja tulpade

segamine. Viimases krüpteerimisvoorus jäetakse vahelt ära tulpade segamine ning selle asemel tehakse viimane vooruvõtme lisamise operatsioon. Krüpteerimisfunktsioonide kasutamist 128-bitise sisendvõtme korral kujutatakse ülevaatlikult Joonis 9 ning täpsemalt lisas 6. Kirjeldatav AES-i krüpteerimisalgoritm muudab krüpteerimata sisendandmed krüpteeritud väljundandmeteks. [42, 43]

Võtme pikkus		Krüpteeritavate andmete pikkus		Krüpteerimisvoorude arv
baitides	bittides	baitides	bittides	
16	128	16	128	10
24	192	16	128	12
32	256	16	128	14

Tabel 2. Krüpteerimisvoorude arvuline sõltuvus võtme pikkusest

Vooruvõtme lisamise käigus teostatakse krüpteerimisvooru 16-baidise ehk 128-bitise andmevektori ning sellele vastava laiendatud võtme 16-baidise ehk 128-bitise vektori vahel XOR tehe. Igas uues krüpteerimisvoorus võetakse XOR tehte jaoks laiendatud võtme järgmised 128 bitti. Lisavooruna võib kujutada laiendatud võtme viimase 128 biti lisamist kümnendas krüpteerimisvoorus, kus see järgneb tulpade segamise asemel rea nihutamise operatsioonile. XOR tehte võimalikud tulemused on kujutatud tabelis 3 ning Joonis 10 leiab vooruvõtme lisamist selgitava skeemi. [42, 43]

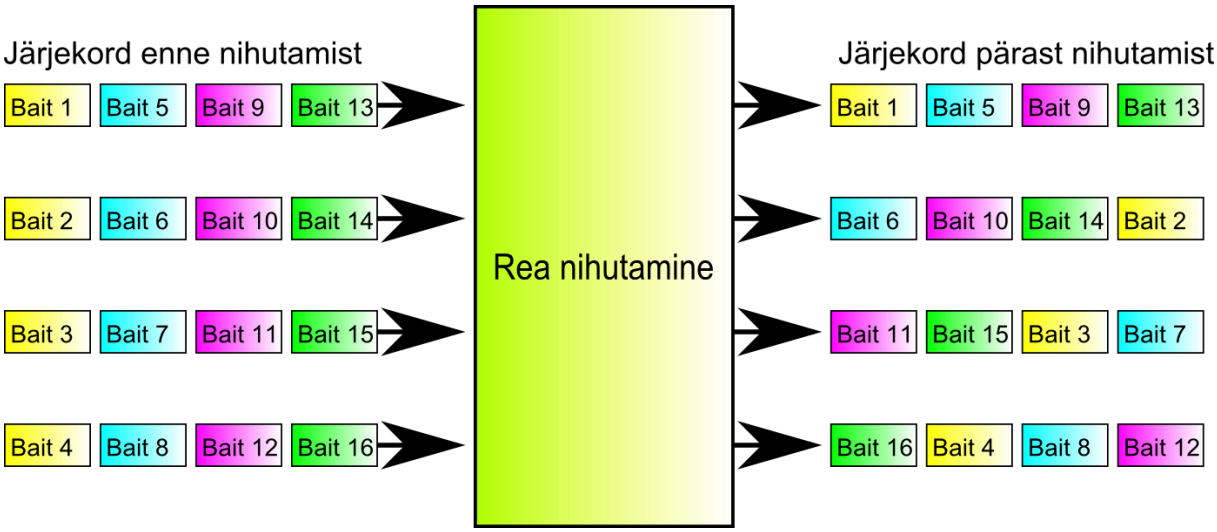


Joonis 10. Vooruvõtme lisamist selgitav skeem

Vooruvõtme lisamise järel asendatakse saadud andmevektori baidid asendustabeli vastavate baitidega nii, nagu seda tehti ka laiendatud võtme algoritmi baitide asendamise operatsioonis. Vajamineva asendustabeli leiab lisast 2. [42, 43]

Rea nihutamise etapis nihutatakse kuueteistkümnest baidist koosneva neljarealise maatriksi iga rida vastavalt 0, 1, 2 või 3 kohta. Krüpteerimisel kasutatav rea nihutamise operatsioon võtab sisse kõik ühe vooru baidi asendamise operatsioonist saadud 16 baiti ehk 128 bitti.

Nihutamisoperatsioone tehakse ainult baitidega, baitide sees olevaid bitte ümber ei tõsteta. Rea nihutamist krüpteerimisel demonstreerib Joonis 11. [42, 43]

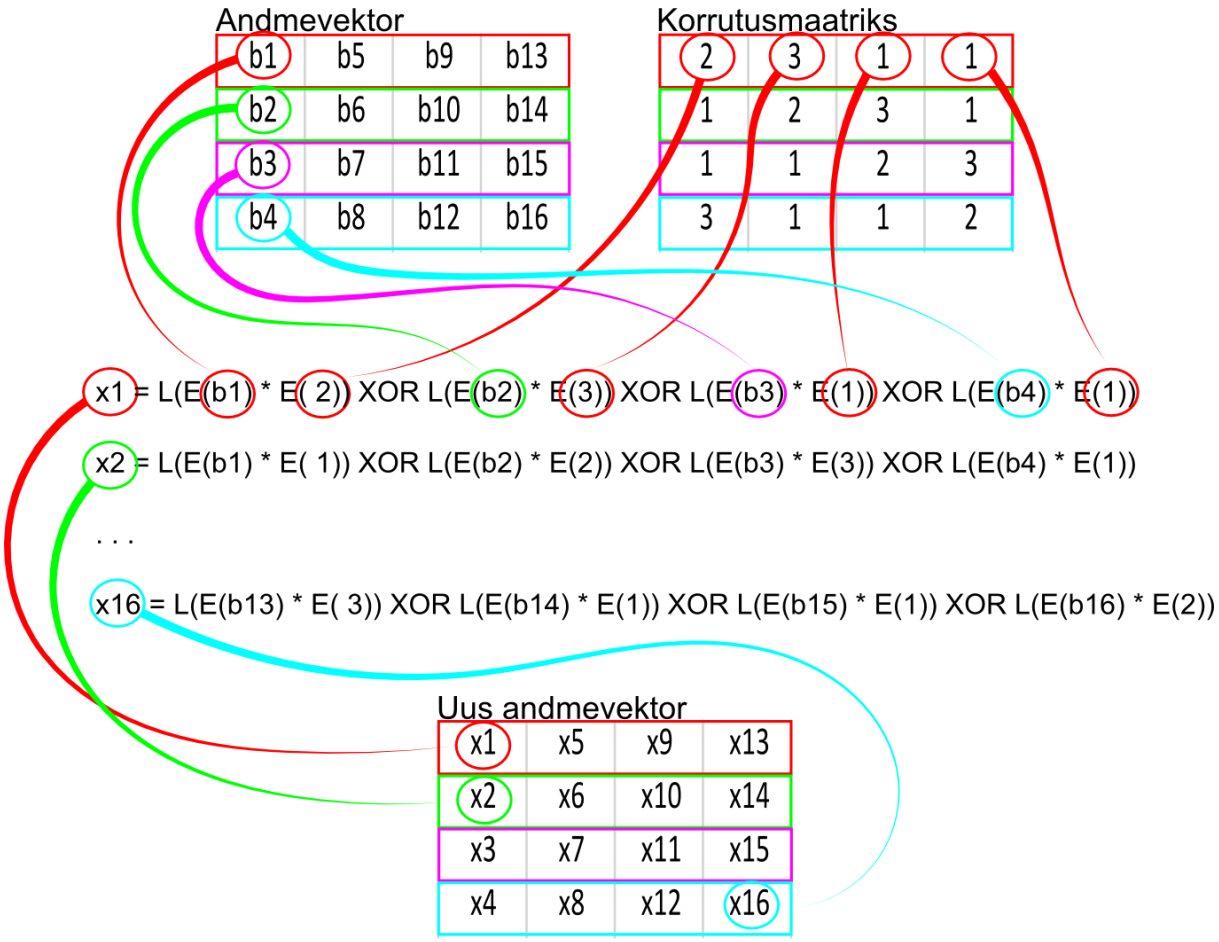


Joonis 11. Krüpteerimisel kasutatava rea nihutamise operatsiooni illustreeriv skeem

Tulpade segamiseks korrutatakse maatriksisse paigutatud andmevektorit ettemääratud korrutusmaatriksiga. Uue 16-baidise andmevektori iga baidi väärtus sõltub sisendvektori ühe tulba baitide väärtusest. Andmete korrutusmaatriksiga läbikorrutamise näide on kujutatud Joonis 12, kus on näidatud uue andmevektori esimese, teise ja kuueteistkümnenda baidi väärtuse leidmine. Korrutustulemuste leidmiseks pöördutakse vaadeldavas näites L ja E funktsioonidega lisades 4 ja 5 asuvate asendustabelite poole, millest andmed saadakse sarnaselt Joonis 8 kujutatuga. [42, 43]

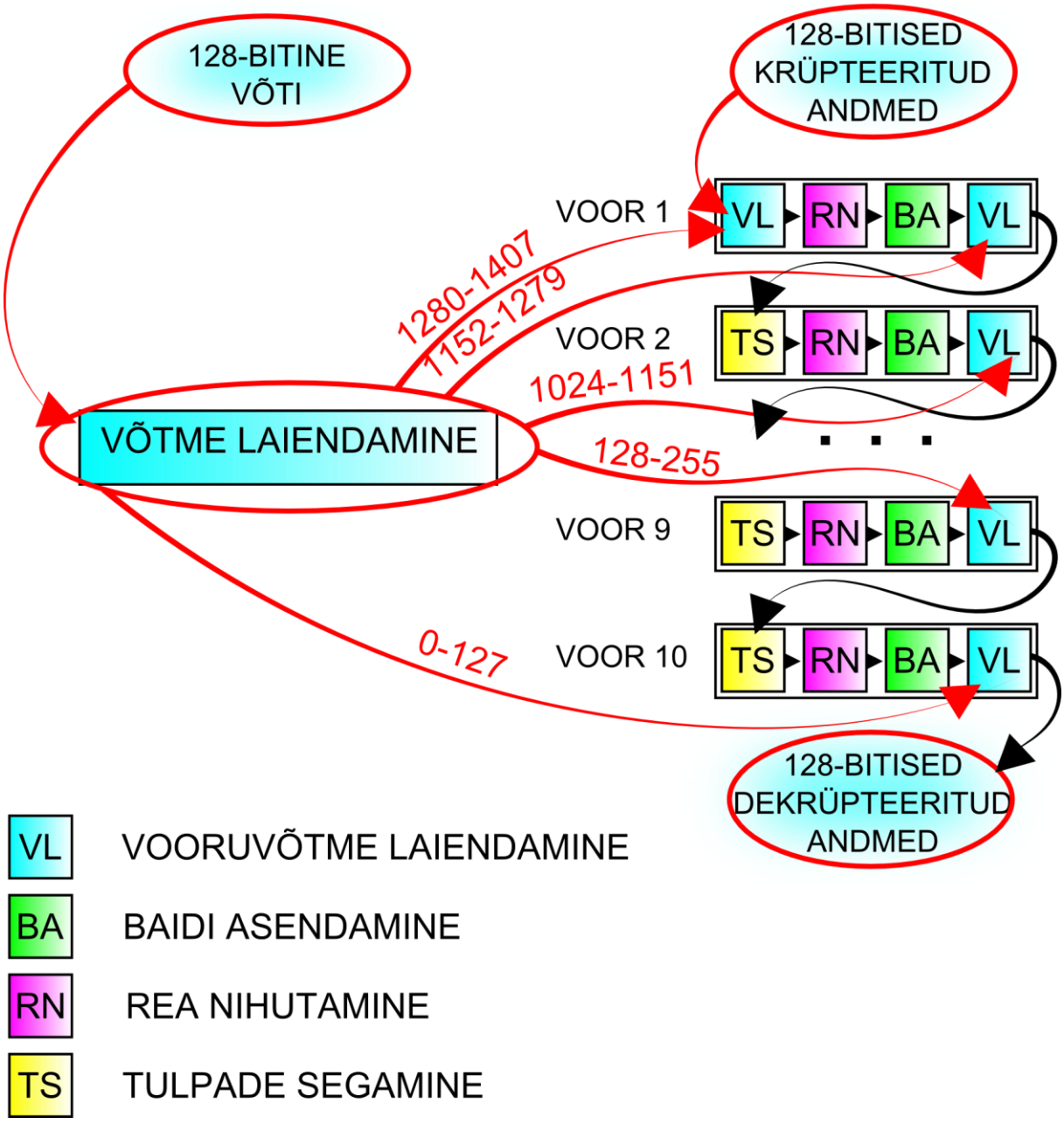
A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Tabel 3. XOR tehte olekutabel



Joonis 12. AES-i küpteerimisalgoritmi tulpade segamist illustreeriv skeem

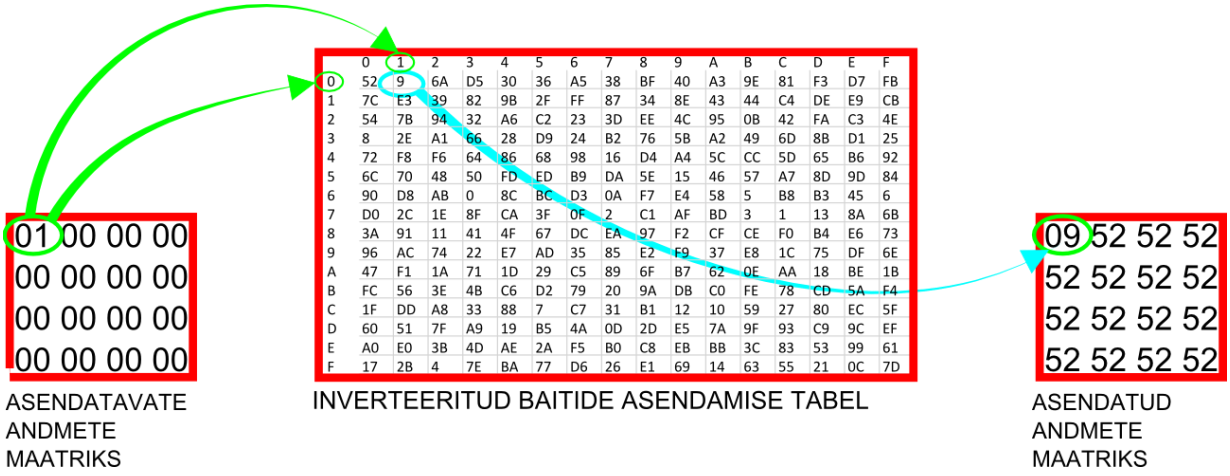
3.3 AES-i dekrüpteerimise algoritm



Joonis 13. AES algoritmi dekrüpteerimisprotsessi ülevaatlik skeem

AES-i dekrüpteerimise algoritm koosneb, nagu ka krüpteerimise algoritm, vooruvõtme lisamise, baidi asendamise, rea nihutamise ja tulpade segamise funktsioonidest. Dekrüpteerimisfunktsioonide kasutamist 128-bitise sisendvõtme korral kujutatakse ülevaatlikult Joonis 13 ning täpsemalt lisas 7. Kirjeldatav algoritm muudab krüpteeritud sisendandmed dekrüpteeritud väljundandmeteks. [42, 43]

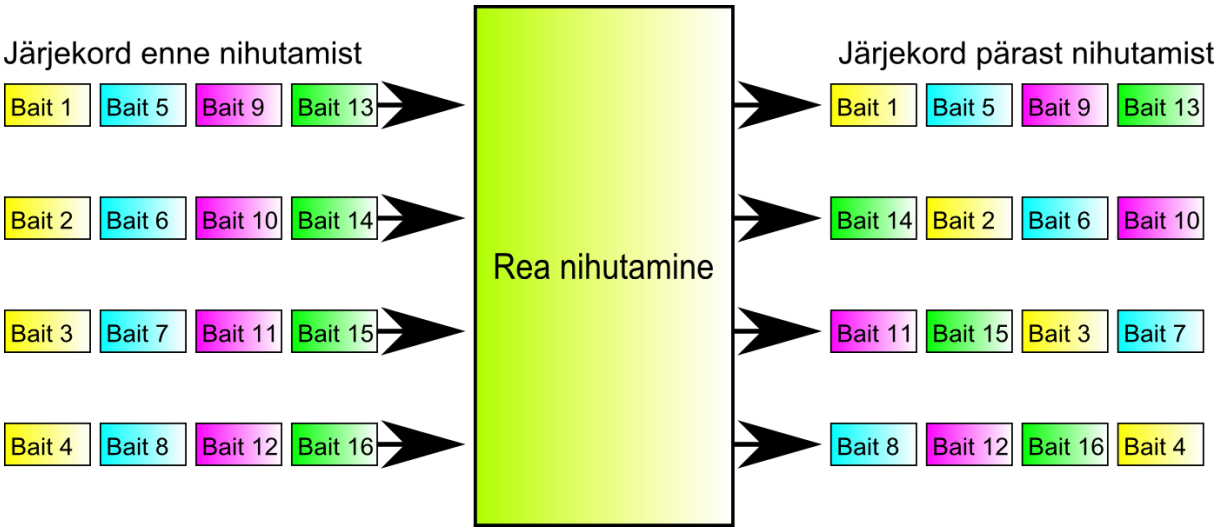
Vooruvõtme lisamise operatsioon dekrüpteerimisel on identne vooruvõtme lisamisega krüpteerimisel. Kuna XOR tehte puhul muudetakse bittide seisu ainult olukorras, kus võtme ja andmevektori bitt paikneb erinevas olekus, on lihtne taastada algset bittide seisu. [42, 43]



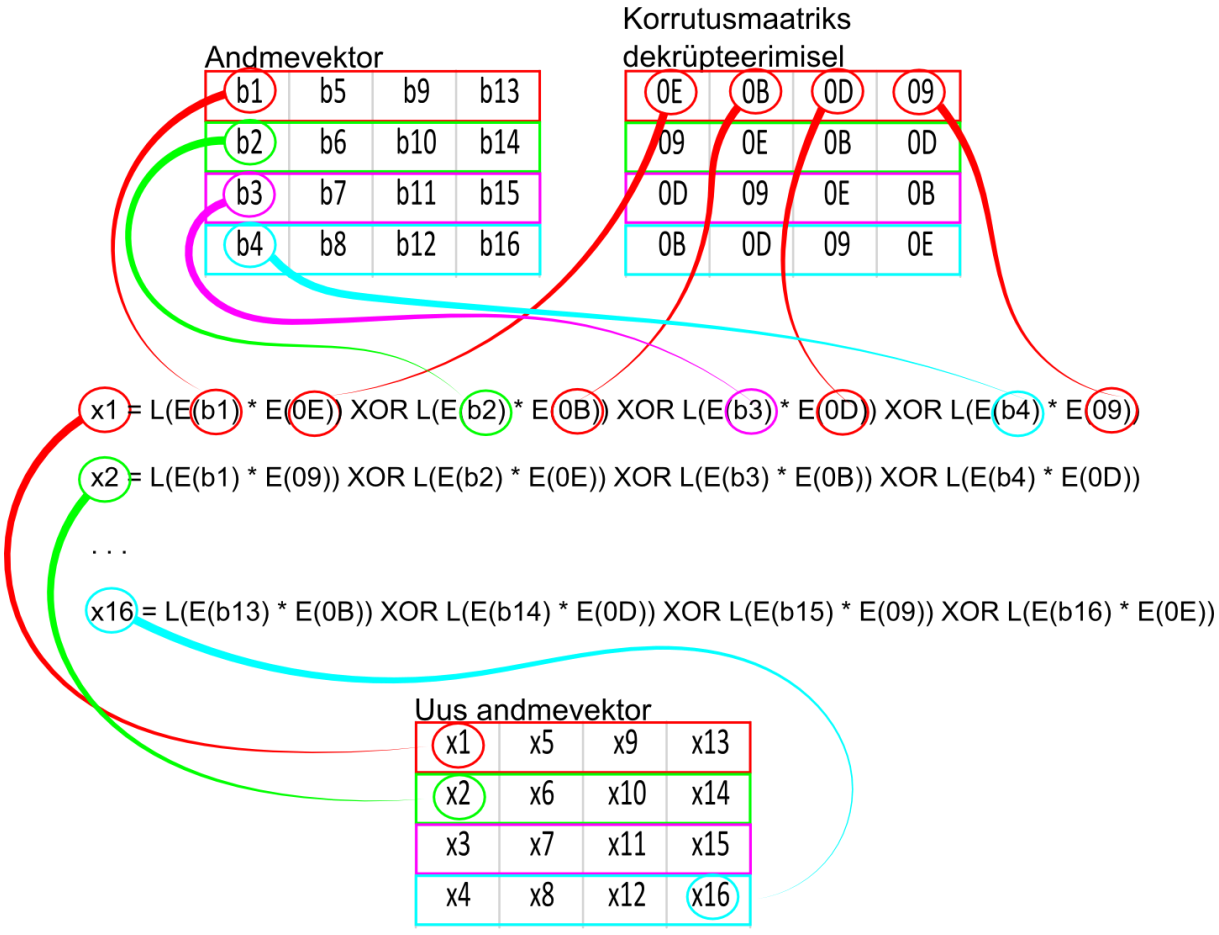
Joonis 14. Ülevaatlik skeem inverteeritud baitide asendamise tabeli kasutamisest

Vooruvõtme lisamise järel asendatakse saadud andmevektori baidid inverteeritud asendustabeli vastavate baitidega. Inverteeritud asendustabeli kasutamise ülevaatliku skeemi leiab Joonis 14. Nagu näha toimub inverteeritud baitide asendustabeli kasutamine sarnaselt tavalise baitide asendustabeli kasutamisele, kuid inverteeritud baitide asendustabel on tavalise baitide asendamise tabeli pöörd kuju. See tähendab, et kui inverteeritud tabelis vastab arvule n arv m, on tavalises asendustabelis arvuga m leitav arv n. Vajamineva inverteeritud asendustabeli leiab täismahus lisast 3. [42, 43]

Rea nihutamise etapis nihutatakse kuueteistkümnest baidist koosneva neljarealise maatriksi iga rida vastavalt 0, 1, 2 või 3 kohta. Võrrelduna krüpteerimisel kasutatava rea nihutamise operatsiooniga teostatakse dekrüpteerimisel rea nihutamised vastupidises suunas. Kui ühel juhul nihutatakse baite 4, 8, 12, 16 kolm kohta vasakule, siis dekrüpteerimisel kolm kohta paremale. Rea nihutamise operatsioone dekrüpteerimisel demonstreerib Joonis 15. [42, 43]



Joonis 15. Dekrüpteerimisel kasutatava rea nihutamise operatsiooni illustreeriv skeem



Joonis 16. AES-i deküpteerimisalgoritmi tulpade segamist illustreeriv skeem

Dekrüpteerimise tulpade segamise operatsioon erineb krüpteerimise tulpade segamise operatsioonist ainult kasutatava korrutusmaatriksi poolest. Dekrüpteeritavate andmete korrutusmaatriksiga läbikorrutamise näide on kujutatud Joonis 16, kus on näidatud uue

andmevektori esimese, teise ja kuueteistkümnenda baidi väärtuse leidmine. Ülejäänud sammud jäävad krüpteerimise tulpade segamise operatsiooniga samaks. [42, 43]

3.4 Kasutatav tarkvara

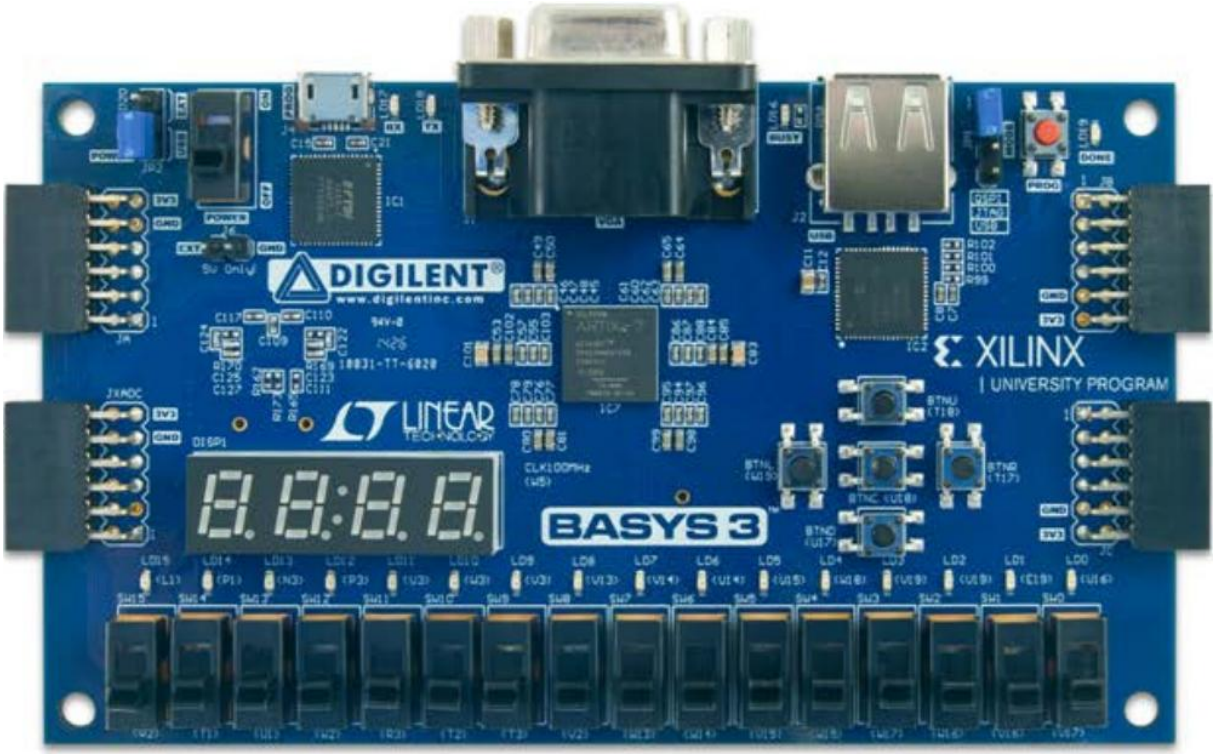
Programmeerimise ja simulatsiooni keskkonnana kasutatakse Ultrascale, Virtex-7, Kintex-7, Artix-7 ja Zynq-7000 seadmete konfigureerimiseks mõeldud Vivado WebPACK versiooni. [44]. Kasutatav WebPACK versioon võimaldab programmide kirjutamist, nende simuleerimist, sünteesimist ja paigaldamist sobivatele FPGA seadmetele ning toetab VHDL ja Verilog programmeerimiskeeli.

Lisaks kasutatakse kahe abiskripti tegemiseks Python'i 3.4.3 versiooni integreeritud programmeerimiskeskkonda IDE.

3.5 Kasutatav riistvara

Riistvaratestide tegemiseks kasutatakse Artix-7 FPGA kiibiga Basys 3 arendusplaati. [45] Arendusplaat on spetsiaalselt mõeldud kasutamiseks koos Vivado programmeerimiskeskkonnaga, kus saab sünteesitud programmi otse seadmele paigaldada. Ülevaade Basys 3 arendusplaadi omadustest:

- 33 280 loogikarakku 5200 tükis ja 1 800 kilobitti RAM mälu;
- 16 liuglülitit, 16 LED-i, 5 nuppu, 4-kohaline 7-segmendiline ekraan, 12-bitine VGA väljund, 4 Digilenti välise mooduli konnektorit ja USB port hiire, klaviatuuri või mälupulgaga ühendamiseks;
- USB-JTAG port FPGA konfigureerimiseks ja andmevahetuseks.



Joonis 17. Basys 3 arendusplaati illustreeriv pilt

3.6 AES algoritmi simulatsioon

Simulatsiooni loomise eesmärgiks on realiseerida AES algoritmi 128-bitise võtmega krüpteerimise ja dekrüpteerimise funktsionaalsus VHDL programmeerimiskeeles nii, et tehtud sünteesitavaid programmilisi lahendusi saaks kasutada ka riistvaratestide tegemisel. VHDL programmeerimiskeele süntaksist kasutatakse ainult selliseid koodielemente, mis lisaks simuleerimisele on ka sünteesitavad. Algoritmi korrektse töö kontrollimiseks kasutatakse NIST-i poolt väljastatud testvektoreid. [46] Simulatsiooniprotsessi paremaks mõistmiseks tehakse simulatsioon ka töö autori poolt valitud ASCII kodeeringus tekstivektoritega.

Simulatsiooni läbiviimiseks kasutatakse tekstifailidesse salvestatud testvektoreid, simulatsiooniprogrammi ja kahte Pythoni skripti. Kasutatavad testvektorid, simulatsiooniprogrammi ja Pythoni skriptid leiab lisast 8.

Simulatsiooniprogrammi testitakse lisaks NIST-i poolt väljastatud 128-bitistele testvektoritele ka töö autori poolt valitud väljalõigetega inglisekeelsest Vikipeedia artikklist „AES“. [47] NIST-i CAVP (*Cryptographic Algorithm Validation Program*) tegeleb AES algoritmi testimise ning valideerimisega. Seega on mõistlik kasutada just nende poolt väljastatud testvektoreid. [46] Töö autori poolt valitud testvektoritega soovitakse näidata AES algoritmi kasutamist selle kasutajale arusaadavamal viisil. Lisaks võib vabalt valitud artiklist (antud töö raames

Vikipeedia artiklist) võetavaid andmeid pidada juhuslikeks testvektoriteks, kuna täpne bittide jada selgub alles pärast ASCII koodi teisendamist binaarkujule.

Pythoni ASCII_to_BIN skripti kasutatakse tekstikujul olevate testandmete konverteerimiseks simulatsiooniprogrammile sobivate pikkustega binaararvulisteks andmevektoriteks. BIN_to_ASCII skript muudab dekrüpteeritud binaararvulised väljundandmed kasutajale arusaadavaks ASCII tekstiks.

Simulatsiooniprogramm koosneb üheksast vhd laiendiga failist (ST, F1AES, F2AES, F3AES, KeyExp, DeCryptAES, CryptAES, TopAES ja TopAESTB) ning neljast txt laiendiga failist (key, plain, krypt, dekrypt), mida kasutatakse VHDL-i TEXTIO teegi funktsioonide kaudu väliskeskkonnaga suhtlemiseks

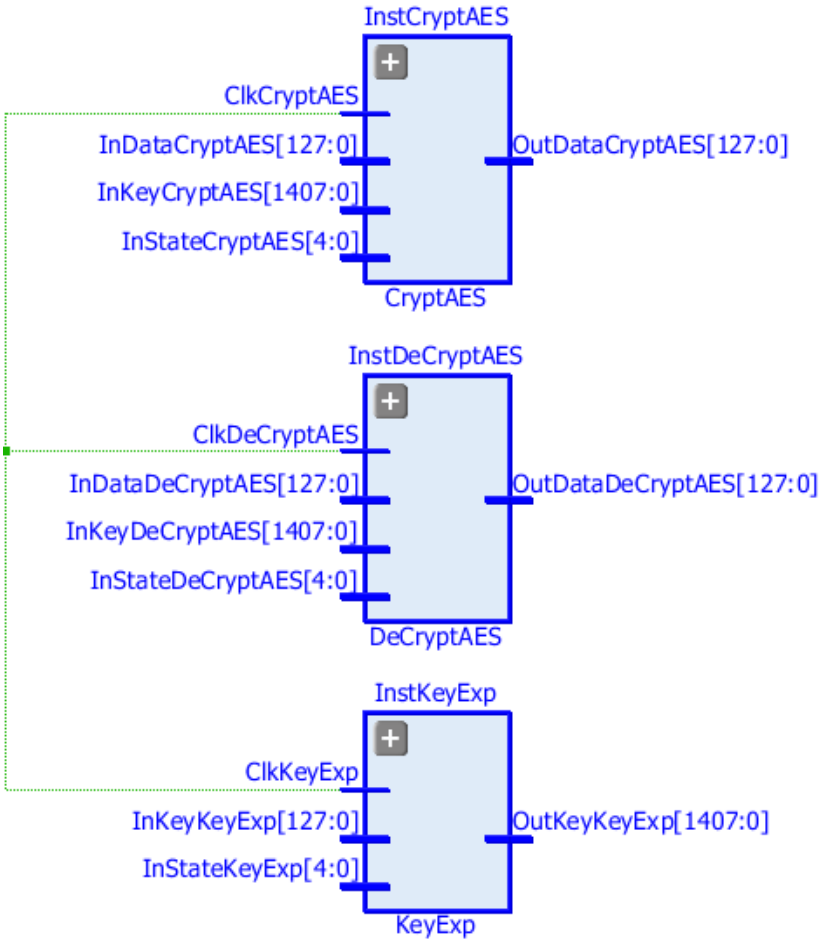
ST, F1AES, F2AE ja F3AES failid on kasutaja poolt kirjeldatud pakid (*package*), milles defineeritakse korduvalt kasutatavad funktsioonid ning muutujad. Teistele vhd laiendiga failidele lisatakse pakke sarnaselt VHDL teekidega. ST-s hoitakse lisades 2, 3, 4 ja 5 kirjeldatud asendustabeleid ning tabelis 1 kujutatud voorukonstante. F1AES, F2AE ja F3AES pakkides kirjeldatakse krüpteerimiseks ja dekrüpteerimiseks vajaminevaid funktsioone. F1AES-is on defineeritud simulatsiooni teostamiseks vajaminevad tekstifailist lugemise ja tekstifaili kirjutamise funktsioonid.

TopAES kontrollib 5-bitiste juhtvektoritega KeyExp, DeCryptAES ja CryptAES failide tööd. Kuna esialgu on tegu ainult simuleeritava AES algoritmiga, ei ole riistvaraliste piirangutega arvestatud. Nii on TopAES failil kasutuses üle viiesaja sisend-väljundviigu, mille kaudu saadakse sisenditeks 128-bitised krüpteerimata ja krüpteeritud andmed, kasutatav võti ning faili juhtiv 4-bitine vektor. TopAES-i väljunditeks on krüpteeritud ja dekrüpteeritud andmete 128-bitised vektorid. Joonis 18 kujutatakse simulatsiooniprogrammi KeyExp, DeCryptAES ja CryptAES failide võrdlust. Joonisel on näha kõigi kolme alamfaili sisend- ja väljundsignaale.

KeyExp failis määratakse 128-bitise sisendvõtme laiendamiseks mõeldud funktsioonide kasutamise järjekord ning omavaheline sõltuvus. Faili sisenditeks on TopAES failist saadav taktsignaal, 128-bitine sisendvõti ning protsessis paikneva olekumasina 5-bitine juhtvektor. Ainukeseks väljundiks on TopAES faili saadetav 1408-bitine laiendatud võti.

CryptAES faili sisenditeks on TopAES failist saadav taktsignaal, laiendatud võti, krüpteerimata andmed ja protsessis paikneva olekumasina 5-bitine juhtvektor. Kümne krüpteerimisfunktsiooni rakendamise järel väljastatakse 128-bitine krüpteeritud andmevektor.

DeCryptAES faili sisenditeks on TopAES failist saadav taktsignaal, laiendatud võti, krüpteeritud andmed ja protsessis paikneva olekumasina 5-bitine juhtvektor. Kümne dekrüpteerimisfunktsiooni rakendamise järel väljastatakse 128-bitine dekrüpteeritud andmevektor.



Joonis 18. Visualiseering simulatsiooniprogrammi alamfailidest

TopAESTB failis kirjeldatakse simulatsioonifail, mille käigus võetakse txt laiendiga tekstfailist binaarkujul olevad krüpteeritavad andmevektorid ja väljastatakse binaarkujul olevad krüpteeritud tulemused uude txt laiendiga faili. Edasi kasutatakse saadud krüpteeritud andmeid dekrüpteerimisalgoritmi sisendina. Simulatsiooni tulemuseks on tekstifailid krüpteeritud ja dekrüpteeritud andmetega.

Simulatsiooniprogrammi nõuded:

- simulatsiooniprogrammi võtmevektoreid või vektorit sisaldav fail peab olema nimega „key.txt“;
- simulatsiooniprogrammi andmevektoreid või vektorit sisaldav fail peab olema nimega „plain.txt“;
- „key.txt“ ja „plain.txt“ failides peab informatsioon olema binaarkujul;
- „key.txt“ ja „plain.txt“ failidesse võib andmeid paigutada 128-bitiste plokkidena;

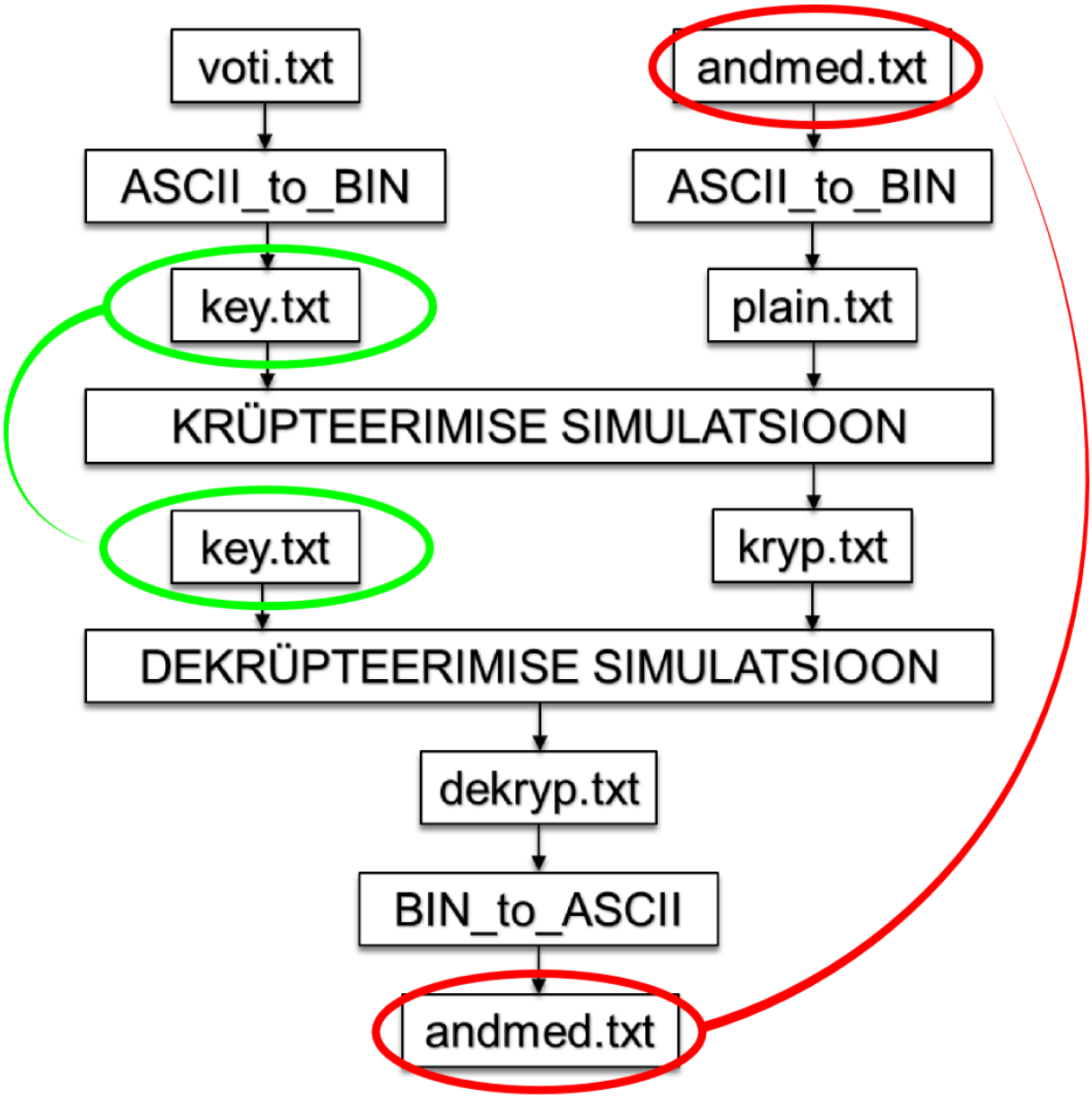
- „key.txt“ ja „plain.txt“ failides võib igale reale paigutada ainult ühe 128-bitise andmeploki.

Simulatsiooniprogrammi omadused:

- simulatsiooni tulemuseks on tekstifailid „krypt.txt“ ja „dekrypt.txt“;
- „krypt.txt“ ja „dekrypt.txt“ failidesse salvestatakse info binaarkujul;
- „krypt.txt“ ja „dekrypt.txt“ failidesse salvestatakse info 128-bitiste plokkidena;
- „krypt.txt“ ja „dekrypt.txt“ failidesse salvestatakse igale reale maksimaalselt üks 128-bitine andmeplokk;
- „krypt.txt“ faili salvestatakse krüpteerimise tulemus;
- „dekrypt.txt“ faili salvestatakse dekrüpteerimise tulemus.

Teostatav simulatsioon jaotub AES-i küpteerimis- ja dekrüpteerimisalgoritmi simulatsiooniks. Krüpteerimise korral võetakse tekstifailidest 128-bitine võti ja 128-bitised sisendandmed ning suunatakse need krüpteerimissimulatsiooni, kus teostatakse võtme laiendamine ja andmete krüpteerimine. Kui 128-bitiseid sisendandmete plokke on rohkem kui üks, tehakse nende krüpteerimine järjestikku 128-bitiste plokkide kaupa. Võtme laiendamise ja andmete krüpteerimise järel kirjutatakse krüpteeritud bitivektorid tühja tekstifaili. Pärast krüpteerimisalgoritmi töö lõppu alustatakse dekrüpteerimisega.

Dekrüpteerimise korral võetakse tekstifailidest 128-bitine võti ja krüpteeritud andmed ning tehakse sisendandmete dekrüpteerimine. Dekrüpteeritud tulemus kirjutatakse uude tekstifaili. Simulatsioonide järel analüüsitakse saadud tulemusi ning jätkatakse riistvaraliste testidega. Simulatsiooni läbiviimise skeemi leiab Joonis 19. Skeemis on kujutatud ka sisendandmete binaarkujule ja väljundandmete tekstikujule teisendamist.



Joonis 19. Simulatsiooni skeem

3.7 Testid Basys 3 arendusplaadil

Riistvaratestide eesmärgiks on seatud AES algoritmi operatsioonide korrektne jooksutamine FPGA-l, mis loob eeldused programmi hilisemaks optimeerimiseks. Selleks tuleb tehtud simulatsiooniprogrammi krüpteerimisalgoritmi funktsioonid kohandada Basys 3-le paigaldamiseks. FPGA loogikaelementide kasutamise vähendamiseks plaanitakse kasutatavad asendustabelid paigutada plokk RAM-i. Kuna krüpteerimine ja dekrüpteerimine on ehituselt sarnased, tehakse töö käigus riistvarasimulatsioone ainult krüpteerimisalgoritmi loogikale.

Basys 3 arendusplaadil testitakse eraldiseisvalt võtme laiendamise, vooruvõtme lisamise, baidi asendamise ja rea nihutamise operatsioone ning tehakse nendele ka koostöötest. Riistvaratestide

kontrollimiseks võetakse üks NIST-i poolt väljastatud testvektor, mida kasutati simulatsiooniprogrammi testimisel ning autori poolt defineeritud testvektorid piirjuhtumite jaoks. Piirjuhtudeks võetakse võimalikest sisenditest kõige väiksema, keskmise ja kõige suurema väärtusega arv. Erinevate alamoperatsioonide testimiseks kasutatakse testvektorite esimese vooru tulemusi, mis selgitatakse välja töö käigus loodud simulatsiooniprogrammi abil. Riistvaratestide testvektorid leiab lisast 8.

Võtme laiendamise riistvaratestiks luuakse programm, mis tegeleb AES algoritmi võtme laiendamise operatsiooni läbiviimisega Basys 3 arendusplaadil. Loodava programmi sisenditeks on 128-bitine võtmevektor, arendusplaadi kellasignaal ja 16 lülitit, millega peab saama muuta 128-bitise sisendvõtme viimast 16 bitti. Väljundina kuvatakse laiendatud võtme viimased 16 bitti ehk 4 kuueteistkümnendarvu 7-segmendilisel ekraanil. AES-i seisukohalt järgneb võtme laiendamisele krüpteerimisalgoritm. Võtmevektoriks kasutatakse NIST-i poolt väljastatud testvektori võtit ja töö autori poolt defineeritud sisendvõtme piirjuhtusid. Tulemusi võrreldakse simulatsiooniprogrammi abil leitud tulemustega.

Vooruvõtme lisamise riistvaratestiks luuakse programm, mis tegeleb AES algoritmi vooruvõtme lisamise operatsiooni läbiviimisega Basys 3 arendusplaadil. Loodava programmi sisenditeks on 128-bitine võtme- ning andmevektor, arendusplaadi taktsignaal ja 16 lülitit, millega peab saama muuta 128-bitise sisendvõtme viimast 16 bitti. Väljundina kuvatakse laiendatud võtme viimased 16 bitti ehk 4 kuueteistkümnendarvu 7-segmendilisel ekraanil. Võtmevektoriks on NIST-i poolt väljastatud testvektori laiendatud võtme esimesed 128 bitti. AES-i seisukohalt eelneb vooruvõtme lisamise operatsioonile võtme laiendamine ning sellele järgneb baitide asendamine. Andmevektoriks võetakse NIST-i poolt väljastatud testvektori ning kasutaja poolt defineeritud piirjuhtumite andmevektorid. Saadud testitulemusi võrreldakse simulatsiooniprogrammi abil leitud tulemustega.

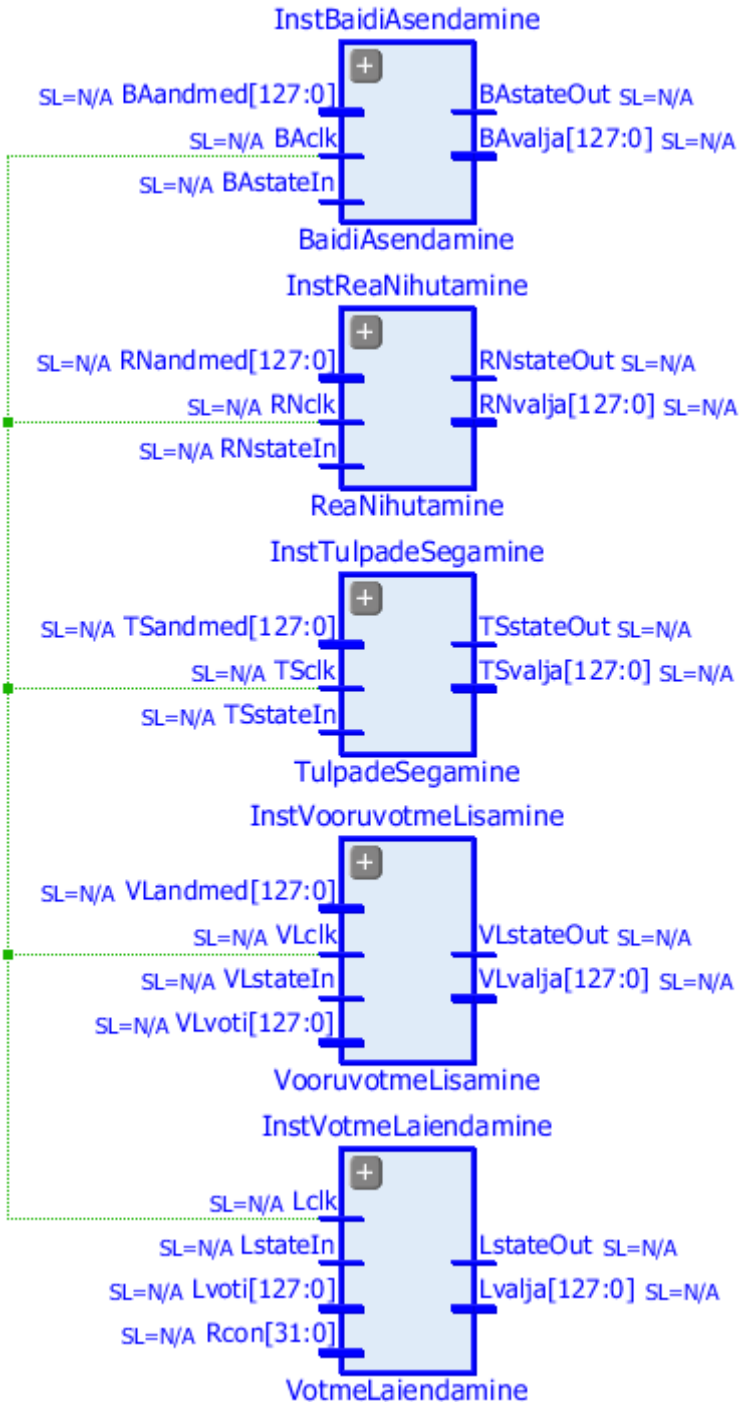
Baidi asendamise riistvaratestiks luuakse programm, mis teostab AES algoritmi baidi asendamise operatsiooni Basys 3 arendusplaadil. Loodava programmi sisenditeks on arendusplaadi taktsignaal ja 16 lülitit, millega peab saama muuta sisendandmete 16-bitti. Väljundina kuvatakse saadud tulemus 7-segmendilisel ekraanil. Lülititega 15 – 8 sisestatud oleku asendusbait kuvatakse 7-segmendilise ekraani kahe vasakpoolse kuueteistkümnendarvuga ning lülititega 7 – 0 sisestatud oleku asendusbait kuvatakse 7-segmendilise ekraani kahe parempoolse kuueteistkümnendarvuga. Nii on võimalik läbi proovida kõik baidi asendamise testvektori elemendid. AES-i seisukohalt eelneb baidi

asendamisele vooruvõtme lisamine ning sellele järgneb ridade nihutamise operatsioon. Testid tehakse NIST-i poolt väljastatud testvektori ja töö autori poolt defineeritud sisendvõtmete viimase 16 bitiga ning neid võrreldakse simulatsiooniprogrammi abil saadud tulemustega.

Rea nihutamise riistvaratestiks luuakse programm, mis tegeleb AES algoritmi ridade nihutamise operatsiooni läbiviimisega Basys 3 arendusplaadil. Loodava programmi sisenditeks on 128-bitine andmevektor ja arendusplaadi taktsignaal. Väljundina kuvatakse saadud tulemuse viimased 16 bitti ehk 4 kuueteistkümnendarvu 7-segmendilisel ekraanil. AES-i seisukohalt eelneb rea nihutamise operatsioonile baitide asendamine ning sellele järgneb tulpade segamine. Testitavateks andmevektoriteks on NIST-i poolt väljastatud ja töö autori poolt defineeritud testvektorite esimeste voorude võtme lisamise operatsioonide tulemused. Rea nihutamise operatsiooni testide tulemusi võrreldakse simulatsiooniprogrammi abil saadud tulemustega.

Tulpade segamise riistvaratestiks luuakse programm, mis tegeleb AES algoritmi tulpade segamise operatsiooni läbiviimisega Basys 3 arendusplaadil. Loodava programmi sisenditeks on 128-bitine andmevektor ja arendusplaadi taktsignaal. Väljundina kuvatakse saadud tulemuse viimased 16 bitti ehk 4 kuueteistkümnendarvu arendusplaadi 7-segmendilisel ekraanil. AES-i seisukohalt eelneb tulpade segamisele ridade nihutamine ning sellele järgneb järgmise vooru vooruvõtme lisamise operatsioon. Sisse võetavaks andmevektoriks on NIST-i poolt väljastatud ja töö autori poolt defineeritud testvektorite esimeste voorude ridade nihutamiste operatsioonide tulemused. Tulpade segamise operatsiooni testide tulemusi võrreldakse simulatsiooniprogrammi abil saadud tulemustega.

Funktsioonide koostöö testis ühendatakse võtme laiendamise, vooruvõtme lisamise, baidi asendamise, rea nihutamise ja tulpade segamise operatsioonid tervikliku AES-i krüpteerimisalgoritmi testimiseks Basys 3 arendusplaadil. Loodav programm kasutab eelnevalt testitud võtme laiendamise, vooruvõtme lisamise, baidi asendamise, rea nihutamise, tulpade segamise operatsioone ning võtab sisenditeks 128-bitised andme- ja võtmevektorid ning 16 lülitit. Lülititega saab soovi korral muuta 128-bitise võtme viimast 16 bitti, mille tagajärjel muutub ka saadav tulemus. Koostöö testis kasutatav programm teostab AES krüpteerimisalgoritmi põhjal sisendandmete täieliku krüpteerimise. Testvektoritena kasutatakse NIST-i poolt väljastatud testvektorit ning kasutaja poolt defineeritud piirjuhtumite andmevektorid. Saadud krüpteeritud tulemusi võrreldakse simulatsiooniprogrammi abil saadud tulemustega.



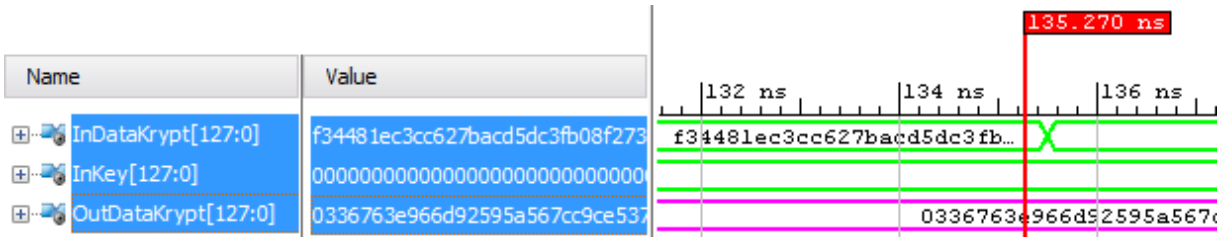
Joonis 20. Vivado visualiseering koostöö testi alamfailidest

4 Tulemused

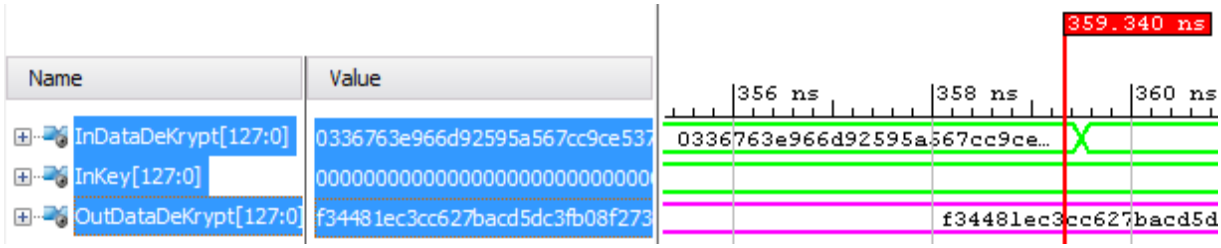
4.1 Simulatsioonide tulemused

Simulatsioonis kasutati sisenditena txt faililaiendiga *plain* ja *key* nimelisi tekstifaile. Tulemuseks saadi txt faililaiendiga *krypt* ja *dekrypt* nimelised tekstifailid. Nii sisend- kui väljundfailides on andmed kujutatud 128-bitiste plokkidena. Sisend- ja väljundfailide igal real paikneb maksimaalselt üks 128-bitine andmevektor, mis on tingitud simulatsiooniprogrammi nõuetest ning omadustest. Simulatsiooni sisendfailid, saadud tulemusfailid ja kasutatud testvektorid leiab lisast 8.

Saadud simulatsiooni tulemused ühtisid NIST-i poolt väljastatud testvektoritega. Joonis 21 ja Joonis 22 kujutatakse väljalõikeid simulatsiooni krüpteerimis- ja dekrüpteerimistulemustest Vivado simulatsioonikeskkonnas.

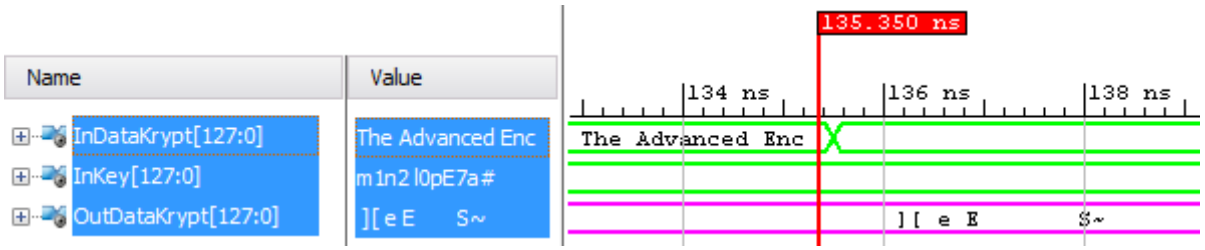


Joonis 21. Väljalõige NIST testvektorite krüpteerimisest

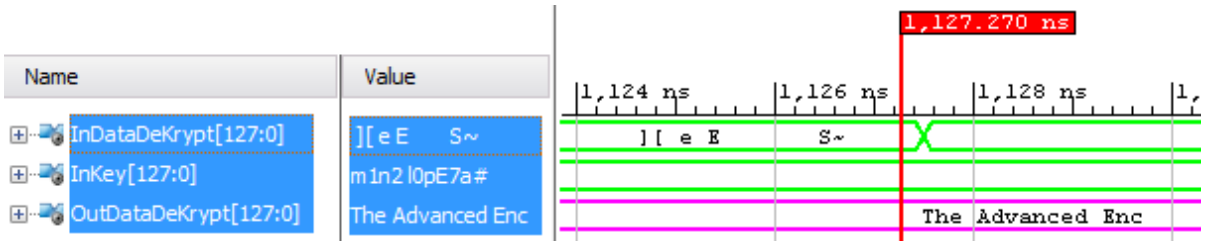


Joonis 22. Väljalõige NIST testvektorite dekrüpteerimisest

Inglisekeelse Vikipeedia artiklist „AES“ võetud testvektorite krüpteerimissimulatsiooni krüpteeritud andmevektorite põhjal taastati dekrüpteerimise tulemusena algsed andmevektorid.[47] Joonis 23 ja Joonis 24 kujutatakse väljalõikeid simulatsiooni krüpteerimis- ja dekrüpteerimistulemustest Vivado simulatsioonikeskkonnas.



Joonis 23. Väljalõige tekstvektorite krüpteerimisest



Joonis 24. Väljalõige tekstvektorite dekrüpteerimisest

4.2 Riistvaratestide tulemused

Võtme laiendamise, vooruvõtme lisamise, baidi asendamise, rea nihutamise ja tulpade segamise operatsioonide riistvaratestide tulemused ühtisid oodatud testvektorite tulemustega. Tulemuste jäädvustamiseks tehti koostöötestist video.. Lisaks koostas Vivado keskkond iga tehtud testiprogrammi kohta sünteesiaruanded, mille hulgast leiab ka riistvara kasutuse aruande. Koostöötesti video ja Vivado poolt koostatud riistvara kasutamise aruanded leiab lisast 8.

Vivado riistvara kasutamise aruanne näitab, milliseid ja kui palju FPGA ressursse sünteesitud programm kasutab. Vastupidiselt ootustele Vivado sünteesimise käigus programmis kirjeldatud asendustabeleid ühegi riistvaratesti puhul plokk RAM-idesse ei kirjutanud. Süntaksi poolest õigesti konfigureeritud asendustabelid paigutati jagatud mällu. Teiste ressursside kasutamine oli ootuspärane.

5 Tulemuste analüüs ja järeldused

Simulatsioonide tulemused tõestavad AES algoritmi rakenduslikult õiget kasutamist nii levinud testvektorite kui juhuslikult valitud andmete töötlemisel. Simulatsioonid tekstifailidega näitasid töö käigus valminud programmi suutlikkust suhelda välise keskkonnaga. Teostati väliskeskkonnast andmete sisse lugemine, nende töötlemine AES algoritmiga ja saadud tulemuste väliskeskkonda laadimine. Kuna algoritmi rakendamisel VHDL-is lähtuti ametlikest allikatest ja avalikest kirjeldustest ning tulemused testvektoritega olid korrektsed, võib järeldada, et tehtud programm oskab AES algoritmi õigesti implementeerida. Seetõttu on mõistlik simulatsiooniprogrammi kasutada ka riistvaratestide tulemuste kontrollimiseks.

Riistvaratestide tulemuste põhjal töötasid AES-i operatsioonid korrektselt. Riistvara kasutamise aruannetest selgub, et eraldiseisva operatsioonina oli kõige vähem nõudlik rea nihutamine, mis kasutas 0,05% Artix-7 FPGA loogikaplokkidest. Selle põhjusteks võib pidada rea nihutamise ülesande lihtsust, staatilisi sisendandmeid ja lühikest väljundvektorit. Lühike 16-bitine väljundvektor võis tingida ka osade rea nihutamise operatsiooni tehete välja optimeerimise, kuid seda testide tegemisel ei kontrollitud. Siiski oli saadud tulemus õige ning rea nihutamise operatsioon ei tinginud vigu koostöötestis.

Kõige riistvaranõudlikumaks eraldiseisvaks operatsiooniks oli võtme laiendamine, mis kasutas 3,56% Artix-7 FPGA loogikaplokkidest. Testiti kogu 1408-bitise võtme laiendamist ning salvestati saadud tulemus ühte 1408-bitisesse muutujasse. Teiste eraldiseisvate operatsioonide testid töötlesid andmeid ühe krüpteerimisvooru ulatuses, mida võib pidada nende väiksema riistvarakasutuse peamiseks põhjuseks. Koostöötestis jaotati võtme laiendamine krüpteerimisvoorude eri osade vahel nii, et igas krüpteerimisvoorus leiti laiendatud võtme järgmised 128 bitti. Seega, terviklahenduses tegeles võtme laiendamise operatsioon korraga ainult ühele krüpteerimisvoorule võtme leidmisega.

Teistest operatsioonidest oli kõige suurema riistvara kasutusega baidi asendamine, mis kasutas 0,44% Artix-7 FPGA loogikaplokkidest. Selle peamiseks põhjuseks võib pidada kasutatava asendustabeli hoidmist jagatud mälus. Baidi asendamist kirjeldavas VHDL koodis defineeriti ka asendustabelite salvestamine plokk RAM-i, kuid Vivado keskkond etteantud riistvarale seda ei sünteesinud. Selle põhjuseks võivad olla programmi sünteesimisel tekkivad vastuolud, mille analüüsimine jääb antud töö mahust välja.

AES-i riistvaral rakendamise koostöö testis kasutati kokku 21,22% Artix-7 FPGA loogikaplokkidest, mis on alamoperatsioonide loogika kasutusest tunduvalt suurem. Siiski on

see tulemus ootuspärane, kuna täieliku AES krüpteerimisalgoritmi rakendamiseks tuleb korduvalt kasutada kõiki väiksema riistvaralise nõudlusega alamoperatsioone, nagu baitide asendamine või vooruvõtme lisamine. Sellest tulenevalt on nii ka salvestamist vajavaid andmeid rohkem. Lisaks nõuab erinevate vahetulemuste väljastamise funktsioon koostöötestilt eraldiseisvatest testidest rohkem dünaamikat.

Lähtudes AES-i krüpteerimisalgoritmi riistvarakasutuse protsendist, mis on väiksem kui üks neljandik kogu Artix-7 FPGA riistvara ressursist, võib eeldada, et ka ehituselt sarnase AES-i dekrüpteerimise algoritmi saaks lisada testide tegemisel kasutatud seadmele. Arendades taolisele seadmele lisaks väliskeskkonnaga suhtlemise mooduli, mis võimaldaks andmepakettide vahetamist, olekski lihtne AES algoritmi kasutav krüpteerimist ja dekrüpteerimist võimaldav seade olemas. Väliskeskkonnaga suhtlemise mooduli üheks osaks võiks olla Digilenti võrguga ühendamise kontroller PmodNIC. [47]

5.1 Edasised arendused

Edasisteks arendusteks võiksid olla:

- AES algoritmi täiemahuline realiseerimine FPGA riistvaral;
- reaalsete kiirus- ning turbetestide tegemine võrguliikluse krüpteerimisel või dekrüpteeerimisel, kasutades selleks PmodNIC kontrollerit;
- saadud tulemuste võrdlemine integreeritud krüpteerimismooduleid või eraldiseisvaid krüptokiirendeid kasutavate netiseadmete tulemustega.

Kokkuvõte

Lõputöö eesmärgiks oli teha ettevalmistusi krüptokiirendi loomiseks, mis oleks realiseeritud FPGA riistvaral. Selleks tuli kõigepealt mõista krüptoloogia olemust ning teostada ülevaade levinumatest krüpteerimisalgoritmidest ja neid kasutatavatest krüptokiirenditest. Kogutud informatsiooni põhjal valiti edasiseks uurimiseks ülemaailmselt populaarne AES algoritm. Algoritmi tööpõhimõtte väljaselgitamise järel alustati bakalaureusetöö praktilise osaga, mille eesmärkideks seati AES-i rakendamine VHDL-is ning simulatsiooni- ja riistvaratestide tegemine. Töö tulemuste põhjal järeldati, kas bakalaureusetöö käigus valitud meetodil on mõistlik jätkata krüptokiirendi arendamist, ning millised oleksid edasised tegevused selle loomiseks.

Praktiline osa jagunes töö käigus loodud simulatsiooniprogrammi testimiseks ning AES-i krüpteerimisalgoritmi rakendamiseks Basys 3 arendusplaadil. Dekrüpteerimise algoritmiga riistvarateste ei tehtud, kuna krüpteerimisel ja dekrüpteerimisel kasutatavad funktsioonid on võrdse keerukusega ning dekrüpteerimise realiseerimine riistvaral oleks järgmine samm koos väliskeskkonnaga suhtluse realiseerimisega. Simulatsiooni- ja riistvaraprogrammid valmisid Vivado programmeerimis- ja simuleerimiskeskkonnas. Simulatsiooniprogrammi testiti erinevate NIST-i poolt väljastatud ning töö autori poolt defineeritud juhuslike testvektoritega. Riistvaratestide tegemiseks valiti juhuslik NIST-i poolt väljastatud testvektor ja töö autori poolt defineeritud testvektorid piirjuhtumite jaoks.

Riistvara- ja simulatsioonitestide analüüsist järeldus, et töö käigus valitud meetodil on võimalik luua välise keskkonnaga suhtlev seade, mis võimaldab andmeid krüpteerida ning dekrüpteerida. Kuna see on ka olemasolevate krüptokiirendite põhiliseks funktsiooniks, on mõistlik defineerida tegevused töö käigus valminud ja esialgsed testid läbinud AES-i krüpteerimisalgoritmi rakendamise programmi edasiseks arendamiseks.

Basys 3 arendusplaadil testitud AES krüpteerimisalgoritmile tuleks edasise arendamise eesmärgil kõigepealt lisada dekrüpteerimine. Järgmisena oleks vajalik kiirus- ning andmeturbetestide tegemine võrguliikluse krüpteerimisel ja dekrüpteerimisel, kasutades võrguga ühendamiseks näiteks Digilenti PmodNIC moodulit. Saadud tulemusi peaks edasiste järelduste tegemiseks võrdlema integreeritud krüpteerimismooduleid või eraldiseisvaid krüptokiirendeid kasutavate netiseadmete tulemustega.

Summary

The realization of AES algorithm in VHDL and testing it on FPGA hardware

Hendrik Türk

The thesis objective was to make preparations for the creation of a cryptographic accelerator device that would have run on FPGA hardware. In order to achieve this, it was necessary to understand the nature of cryptology and to have an overview of the most common cryptographic algorithms and the cryptographic accelerator hardware that uses them. Having gathered enough information about these topics, AES algorithm was chosen for further inspection. After elaborating how AES should be implemented, the practical part of the bachelor thesis started. Its objectives were to implement AES in VHDL and to perform simulation and hardware tests on the implemented AES program. The results about the success of the selected implementing methods and testing - if they were enough to support further development - and further steps in achieving the main objective are stated in the conclusion.

The practical part was divided between simulation tests and AES algorithm implementation on Basys 3 development board. Hardware tests were not done for the decryption part of AES mainly because the encryption and decryption algorithms are very similar in complexity. Simulation and hardware programs were made in Vivado programming and simulating environment. The AES simulation program was tested with different test vectors issued by NIST and defined by the author of this paper. For the hardware tests one test vector from NIST was chosen in conjunction with test vectors for border cases.

It was concluded that the method chosen for the implementation of AES can be used in order to make an encryption and decryption device that could encrypt and decrypt incoming or outgoing data. As this is the main functionality of existing cryptographic accelerators, it would be wise to define activities and recommendations for further development of this device.

Firstly, the AES should be fully implemented on Basys 3 by adding the decryption algorithm in addition to the already fully working encryption algorithm. Secondly, it would be necessary to have real encryption and decryption over a network in order to test the encryption and decryption speeds. For example, Digilents' PmodNIC could be used as the controller for connecting to the network. Then the results should be compared with other networking devices that use cryptographic accelerators and devices that would rely on their integrated encryption methods.

Viited

[1] Vallaste, „AES (Advanced Encryption Standard),“ Vallaste, [Võrgumaterjal]. Saadaval: http://www.vallaste.ee/index.htm?Type=UserId&otsing=4393. [Kasutatud 20. 05. 2015].

[2] Vallaste, „ASCII (American Standard Code for Information Interchange),“ Vallaste, [Võrgumaterjal]. Saadaval: http://www.vallaste.ee/index.htm?Type=UserId&otsing=42. [Kasutatud 20 05 2015].

[3] Vallaste, „ASIC (Application Specific Integrated Circuit),“ Vallaste, [Võrgumaterjal]. Saadaval: http://www.vallaste.ee/index.htm?Type=UserId&otsing=453. [Kasutatud 20. 05. 2015].

[4] Vallaste, „asymmetric cryptography,“ Vallaste, [Võrgumaterjal]. Saadaval: http://www.vallaste.ee/index.htm?Type=UserId&otsing=3149. [Kasutatud 20. 05. 2015].

[5] Vallaste, „DES (Data Encryption Standard),“ Vallaste, [Võrgumaterjal]. Saadaval: http://www.vallaste.ee/index.htm?Type=UserId&otsing=113. [Kasutatud 20. 05. 2015].

[6] Vallaste, „FPGA (Field-Programmable Gate Array),“ Vallaste, [Võrgumaterjal]. Saadaval: http://www.vallaste.ee/index.htm?Type=UserId&otsing=4183. [Kasutatud 20. 05. 2015].

[7] Vallaste, „SSL (Secure Sockets Layer),“ Vallaste, [Võrgumaterjal]. Saadaval: http://www.vallaste.ee/index.htm?Type=UserId&otsing=309. [Kasutatud 05. 20. 2015].

[8] Vallaste, „symmetric cryptography,“ Vallaste, [Võrgumaterjal]. Saadaval: http://www.vallaste.ee/index.htm?Type=UserId&otsing=1744. [Kasutatud 20. 05. 2015].

[9] Vallaste, „chiper,“ Vallaste, [Võrgumaterjal]. Saadaval: http://www.vallaste.ee/index.htm?Type=UserId&otsing=4397. [Kasutatud 20. 05. 2015].

[10] Vallaste, „USB (Universal Serial Bus),“ Vallaste, [Võrgumaterjal]. Saadaval: http://www.vallaste.ee/index.htm?Type=UserId&otsing=321. [Kasutatud 20. 05. 2015].

[11] G. C. Kessler, „An Overview of Cryptography,“ 21. 01. 2015. [Võrgumaterjal]. Saadaval: http://www.garykessler.net/library/crypto.html#purpose. [Kasutatud 21 05. 2015].

[12] N. G. McDonald, „PAST, PRESENT, AND FUTURE METHODS OF CRYPTOGRAPHY AND DATA ENCRYPTION,“ [Võrgumaterjal]. Saadaval: http://www.eng.utah.edu/~nmcdonal/Tutorials/EncryptionResearchReview.pdf. [Kasutatud 09. 05. 2015].

[13] Thales, „Thales e-Security Products and Services,“ Thales e-Security Products and Services, [Võrgumaterjal]. Saadaval: https://www.thales-esecurity.com/products-and-services. [Kasutatud 14. 05. 2015].

[14] H. Sidhpurwala, „A Brief History of Cryptography,“ 14. 08. 2013. [Võrgumaterjal]. Saadaval: https://securityblog.redhat.com/2013/08/14/a-brief-history-of-cryptography/. [Kasutatud 09. 05. 2015].

[15] K. K. Parhi, „Crypto Accelerators,“ [Võrgumaterjal]. Saadaval: http://www.ece.umn.edu/~parhi/research/crypto.html. [Kasutatud 09. 05. 2015].

[16] Cryptovision, „A short introduction to modern cryptography,“ [Võrgumaterjal]. Saadaval: https://www.cryptovision.com/fileadmin/media/documents/Whitepaper_Produkte/Modern_Cryptography.pdf. [Kasutatud 29. 10 2014].

[17] Thawte, „History of Cryptography,“ [Võrgumaterjal]. Saadaval: http://book.itep.ru/depository/crypto/Cryptography_history.pdf. [Kasutatud 09. 05. 2015].

[18] A. Shaun, „Vigenere Cryptanalysis,“ [Võrgumaterjal]. Saadaval: http://www.cs.virginia.edu/~cmt5n/Classwork/Crypt/Shaun/vigenerecrypt.html. [Kasutatud 09. 05. 2015].

[19] H. C. Hudde, „Building Stream Ciphers from Block,“ 18. 02. 2009. [Võrgumaterjal]. Saadaval: https://www.emsec.rub.de/media/crypto/attachments/files/2011/03/hudde.pdf. [Kasutatud 09. 05. 2015].

[20] J. Willemson, P. Laud, A. Jürgenson ja M. Laur, „Krüptograafiliste algoritmide kasutusvaldkondade ja elutsükli uuring,“ [Võrgumaterjal]. Saadaval: https://www.ria.ee/public/Programm/kryptoalgoritmide_elutsykli_uuring_15-07-2011.pdf. [Kasutatud 09. 05. 2015].

[21] D. Czagan, „Symmetric and Asymmetric Encryption,“ Infosec, [Võrgumaterjal]. Saadaval: http://resources.infosecinstitute.com/symmetric-asymmetric-encryption/. [Kasutatud 09. 05. 2015].

[22] Digicert, „Behind the Scenes of SSL Cryptography,“ Digicert, [Võrgumaterjal]. Saadaval: https://www.digicert.com/ssl-cryptography.htm. [Kasutatud 14. 05. 2015].

[23] U.S. DEPARTMENT OF COMMERCE/National Institute of Standards and Technology, „DATA ENCRYPTION STANDARD (DES),“ 25. 10. 1999. [Võrgumaterjal]. Saadaval: http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf. [Kasutatud 10. 05. 2015].

[24] R. Clayton ja M. Bond, „Experience Using a Low-Cost FPGA Design to Crack DES Keys,“ 2002. [Võrgumaterjal]. Saadaval: http://www.cl.cam.ac.uk/~rnc1/descrack/DEScracker.pdf. [Kasutatud 10. 05. 2015].

[25] NIST, „ADVANCED ENCRYPTION STANDARD (AES),“ 26. 11. 2001. [Võrgumaterjal]. Saadaval: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf. [Kasutatud 10. 05. 2015].

[26] M. Arora, „How secure is AES against brute force attacks?,“ EETimes, 05. 07. 2012. [Võrgumaterjal]. Saadaval: http://www.eetimes.com/document.asp?doc_id=1279619. [Kasutatud 10. 05. 2015].

[27] D. Boneh, „Twenty Years of Attacks on the RSA Cryptosystem,“ [Võrgumaterjal]. Saadaval: http://crypto.stanford.edu/~dabo/pubs/papers/RSA-survey.pdf. [Kasutatud 10. 05. 2015].

[28] Signalogic, „Server Accelerator Cards,“ Signalogic, [Võrgumaterjal]. Saadaval: http://www.signalogic.com/index.pl?page=server_accelerator_cards. [Kasutatud 14. 05. 2015].

[29] SANS Institute, „An Overview of Hardware Security Modules,“ [Võrgumaterjal]. Saadaval: http://www.sans.org/reading-room/whitepapers/vpns/overview-hardware-security-modules-757. [Kasutatud 14. 05. 2015].

[30] J. Ivarsson ja A. Nilsson, „A Review of Hardware Security Modules,“ 31. 12. 2010. [Võrgumaterjal]. Saadaval: http://www.opendnssec.org/wp-content/uploads/2011/01/A-Review-of-Hardware-Security-Modules-Fall-2010.pdf. [Kasutatud 14. 05. 2015].

[31] J. Warren, „Network Encryption: Microprocessor (Software) vs. FPGA (Hardware),“ Thales, [Võrgumaterjal]. Saadaval: https://www.thales-esecurity.com/blogs/2014/april/network-encryption-microprocessor-vs-fpga. [Kasutatud 14. 05. 2015].

[32] Thales, „nShield Solo Specifications,“ Thales, [Võrgumaterjal]. Saadaval: https://www.thales-esecurity.com/products-and-services/products-and-services/hardware-security-modules/general-purpose-hsms/nshield-solo. [Kasutatud 14. 05. 2015].

[33] SafeNet, „Luna PCI-E – Cryptographic Acceleration from an Embedded HSM - See more at: http://www.safenet-inc.com/data-encryption/hardware-security-modules-hsms/luna-hsms-key-management/luna-pci-e/#content-left,“ SafeNet, [Võrgumaterjal]. Saadaval: http://www.safenet-inc.com/data-encryption/hardware-security-modules-hsms/luna-hsms-key-management/luna-pci-e/#content-left. [Kasutatud 15. 05. 2015].

[34] A. Shilov, „KitGuru,“ KitGuru, 09 09. 2014. [Võrgumaterjal]. Saadaval: http://www.kitguru.net/components/cpu/anton-shilov/intel-unveils-new-generation-xeon-e5-v3-processors-with-up-to-18-cores/. [Kasutatud 20 05. 2015].

[35] Intel, „Intel® QuickAssist Adapter Family for Servers,“ Intel, [Võrgumaterjal]. Saadaval: http://www.intel.com/content/www/us/en/network-adapters/quickassist-adapter-for-servers.html. [Kasutatud 20. 05. 2015].

[36] XILINX, „What is a FPGA?,“ XILINX, [Võrgumaterjal]. Saadaval: http://www.xilinx.com/fpga/. [Kasutatud 10. 05. 2015].

[37] fpga4fun, „Internal RAM,“ fpga4fun, [Võrgumaterjal]. Saadaval: http://www.fpga4fun.com/FPGAinfo3.html. [Kasutatud 20. 05. 2015].

[38] A. Coman ja R. Frățilă, „Cryptographic Applications using FPGA Technology,“ [Võrgumaterjal]. Saadaval: http://www.jmeds.eu/index.php/jmeds/article/viewFile/Cryptographic-Applications-using-FPGA-Technology/pdf. [Kasutatud 10. 05. 2015].

[39] D. Hulton ja D. Pellerin, „Accelerating cryptography with FPGA clusters,“ Military Embedded Systems, 14. 07. 2010. [Võrgumaterjal]. Saadaval: http://mil-embedded.com/articles/accelerating-cryptography-fpga-clusters/. [Kasutatud 10. 05. 2015].

[40] T. Wollinger, J. Guajardo ja C. Paar, „Cryptography on FPGAs: State of the Art Implementations and Attacks,“ [Võrgumaterjal]. Saadaval: https://www.ais.rub.de/media/crypto/veroeffentlichungen/2010/08/08/wollingeretal_acmtransembeddedsysfpgacryptooverview_final.pdf. [Kasutatud 10 05. 2015].

[41] Thales, „Network Encryption: Microprocessor (Software) vs. FPGA (Hardware),“ Thales, [Võrgumaterjal]. Saadaval: https://www.thales-esecurity.com/blogs/2014/april/network-encryption-microprocessor-vs-fpga. [Kasutatud 16. 05. 2015].

[42] A. Berent, „Advanced Encryption Standard by Example,“ [Võrgumaterjal]. Saadaval: http://www.adamberent.com/documents/aesbyexample.pdf. [Kasutatud 11. 05. 2015].

[43] Federal Information , „ADVANCED ENCRYPTION STANDARD (AES),“ 26. 11. 2001. [Võrgumaterjal]. Saadaval: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf. [Kasutatud 11. 05. 2015].

[44] XILINX, „Vivado Design Suite,“ XILINX, [Võrgumaterjal]. Saadaval: http://www.xilinx.com/products/design-tools/vivado.html. [Kasutatud 12. 05. 2015].

[45] DIGILENT, „Basys™3 Artix-7 FPGA Board,“ DIGILENT, [Võrgumaterjal]. Saadaval: http://www.digilentinc.com/Products/Detail.cfm?Prod=BASYS3. [Kasutatud 12. 05. 2015].

[46] NIST, „CRYPTOGRAPHIC ALGORITHM VALIDATION PROGRAM (CAVP),“ NIST, [Võrgumaterjal]. Saadaval: http://csrc.nist.gov/groups/STM/cavp/. [Kasutatud 13. 05. 2015].

[47] Vikipeedia, „Advanced Encryption Standard,“ Vikipeedia, [Võrgumaterjal]. Saadaval: http://en.wikipedia.org/wiki/Advanced_Encryption_Standard. [Kasutatud 15. 05. 2015].

[48] Digilent, „PmodNIC - Network Interface Controller,“ Digilent, [Võrgumaterjal]. Saadaval: http://www.digilentinc.com/Products/Detail.cfm?Prod=PMOD-NIC. [Kasutatud 20. 05. 2015].

[49] Thales Group, „Cryptographic Acceleration,“ Thales, [Võrgumaterjal]. Saadaval: https://www.thales-esecurity.com/solutions/by-technology-focus/cryptographic-acceleration. [Kasutatud 09. 05. 2015].

[50] J. Katz ja Y. Lindell, „Introduction to modern cryptography,“ Boca Raton, London, New York,, Chapman & Hall/CRC, Taylor & Francis Group,, 2008, pp. 162-170.

[51] W. C. Barker ja E. Barker, „Information Security,“ 01. 2012. [Võrgumaterjal]. Saadaval: http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.pdf. [Kasutatud 10. 05. 2015].

[52] SafeNet, „Hardware Security Modules (HSMs),“ SafeNet, [Võrgumaterjal]. Saadaval: http://www.safenet-inc.com/data-encryption/hardware-security-modules-hsms/. [Kasutatud 14. 05. 2015].

Lisad

Lisa 1. 128-bitise võtme laiendamine 1408-bitini

Voor	Laiendatud võtme baidid	Funktsioon
0	0 1 2 3	V(0)
1	4 5 6 7	V(4)
2	8 9 10 11	V(8)
3	12 13 14 15	V(12)
4	16 17 18 19	BA(RN(LV((4-1)*4))) XOR VK((4/4)-1) XOR LV((4-4)*4)
5	20 21 22 23	LV((5-1)*4) XOR LV((5-4)*4)
6	24 25 26 27	LV((6-1)*4) XOR LV((6-4)*4)
7	28 29 30 31	LV((7-1)*4) XOR LV((7-4)*4)
8	32 33 34 35	BA(RN(LV((8-4)*4))) XOR VK((8/4)-1) XOR LV((8-4)*4)
9	36 37 38 39	LV((8-1)*4) XOR LV((9-4)*4)
10	40 1 42 43	LV((10-1)*4) XOR LV((10-4)*4)
11	44 5 46 47	LV((11-1)*4) XOR LV((11-4)*4)
12	48 9 50 51	BA(RN(LV((12-4)*4))) XOR VK((12/4)-1) XOR LV((12-4)*4)
13	52 3 54 55	LV((13-1)*4) XOR LV((13-4)*4)
14	56 7 58 59	LV((14-1)*4) XOR LV((14-4)*4)
15	60 1 62 63	LV((15-1)*4) XOR LV((15-4)*4)
16	64 5 66 67	BA(RN(LV((16-4)*4))) XOR VK((16/4)-1) XOR LV((16-4)*4)
17	68 9 70 71	LV((17-1)*4) XOR LV((17-4)*4)
18	72 3 74 75	LV((18-1)*4) XOR LV((18-4)*4)
19	76 7 78 79	LV((19-1)*4) XOR LV((19-4)*4)
20	80 1 82 83	BA(RN(LV((20-4)*4))) XOR VK((20/4)-1) XOR LV((20-4)*4)
21	84 5 86 87	LV((21-1)*4) XOR LV((21-4)*4)
22	88 9 90 91	LV((22-1)*4) XOR LV((22-4)*4)
23	92 3 94 95	LV((23-1)*4) XOR LV((23-4)*4)
24	96 7 98 99	BA(RN(LV((24-4)*4))) XOR VK((24/4)-1) XOR LV((24-4)*4)
25	100 101 102 103	LV((25-1)*4) XOR LV((25-4)*4)
26	104 105 106 107	LV((26-1)*4) XOR LV((26-4)*4)
27	108 109 110 111	LV((27-1)*4) XOR LV((27-4)*4)
28	112 113 114 115	BA(RN(LV((28-4)*4))) XOR VK((28/4)-1) XOR LV((28-4)*4)
29	116 117 118 119	LV((29-1)*4) XOR LV((29-4)*4)
30	120 121 122 123	LV((30-1)*4) XOR LV((30-4)*4)
31	124 125 126 127	LV((31-1)*4) XOR LV((31-4)*4)
32	128 129 130 131	BA(RN(LV((32-4)*4))) XOR VK((32/4)-1) XOR LV((32-4)*4)
33	132 133 134 135	LV((33-1)*4) XOR LV((33-4)*4)
34	136 137 138 139	LV((34-1)*4) XOR LV((34-4)*4)
35	140 141 142 143	LV((35-1)*4) XOR LV((35-4)*4)
36	144 145 146 147	BA(RN(LV((36-4)*4))) XOR VK((36/4)-1) XOR LV((36-4)*4)
37	148 149 150 151	LV((37-1)*4) XOR LV((37-4)*4)
38	152 153 154 155	LV((38-1)*4) XOR LV((38-4)*4)
39	156 157 158 159	LV((39-1)*4) XOR LV((39-4)*4)
40	160 161 162 163	BA(RN(LV((40-4)*4))) XOR VK((40/4)-1) XOR LV((40-4)*4)
41	164 165 166 167	LV((41-1)*4) XOR LV((41-4)*4)
42	168 169 170 171	LV((42-1)*4) XOR LV((42-4)*4)
43	172 173 174 175	LV((43-1)*4) XOR LV((43-4)*4)

Lisa 2. Krüpteerimisel ja võtme laiendamisel kasutatav baitide asendustabel

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	1	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	4	C7	23	C3	18	96	5	9A	7	12	80	E2	EB	27	B2	75
4	9	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	0	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	2	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	6	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	8
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	3	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Lisa 3. Dekrüpteerimisel kasutatav inverteeritud baitide asendustabel

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	52	9	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	E9	CB
2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
3	8	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
6	90	D8	AB	0	8C	BC	D3	0A	F7	E4	58	5	B8	B3	45	6
7	D0	2C	1E	8F	CA	3F	0F	2	C1	AF	BD	3	1	13	8A	6B
8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
A	47	F1	1A	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1B
B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
C	1F	DD	A8	33	88	7	C7	31	B1	12	10	59	27	80	EC	5F
D	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
F	17	2B	4	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

Lisa 4. E-tabel

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	1	3	5	0F	11	33	55	FF	1A	2E	72	96	A1	F8	13	35
1	5F	E1	38	48	D8	73	95	A4	F7	2	6	0A	1E	22	66	AA
2	E5	34	5C	E4	37	59	EB	26	6A	BE	D9	70	90	AB	E6	31
3	53	F5	4	0C	14	3C	44	CC	4F	D1	68	B8	D3	6E	B2	CD
4	4C	D4	67	A9	E0	3B	4D	D7	62	A6	F1	8	18	28	78	88
5	83	9E	B9	D0	6B	BD	DC	7F	81	98	B3	CE	49	DB	76	9A
6	B5	C4	57	F9	10	30	50	F0	0B	1D	27	69	BB	D6	61	A3
7	FE	19	2B	7D	87	92	AD	EC	2F	71	93	AE	E9	20	60	A0
8	FB	16	3A	4E	D2	6D	B7	C2	5D	E7	32	56	FA	15	3F	41
9	C3	5E	E2	3D	47	C9	40	C0	5B	ED	2C	74	9C	BF	DA	75
A	9F	BA	D5	64	AC	EF	2A	7E	82	9D	BC	DF	7A	8E	89	80
B	9B	B6	C1	58	E8	23	65	AF	EA	25	6F	B1	C8	43	C5	54
C	FC	1F	21	63	A5	F4	7	9	1B	2D	77	99	B0	CB	46	CA
D	45	CF	4A	DE	79	8B	86	91	A8	E3	3E	42	C6	51	F3	0E
E	12	36	5A	EE	29	7B	8D	8C	8F	8A	85	94	A7	F2	0D	17
F	39	4B	DD	7C	84	97	A2	FD	1C	24	6C	B4	C7	52	F6	1

Lisa 5. L-tabel

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		0	19	1	32	2	1A	C6	4B	C7	1B	68	33	EE	DF	3
1	64	4	E0	0E	34	8D	81	EF	4C	71	8	C8	F8	69	1C	C1
2	7D	C2	1D	B5	F9	B9	27	6A	4D	E4	A6	72	9A	C9	9	78
3	65	2F	8A	5	21	0F	E1	24	12	F0	82	45	35	93	DA	8E
4	96	8F	DB	BD	36	D0	CE	94	13	5C	D2	F1	40	46	83	38
5	66	DD	FD	30	BF	6	8B	62	B3	25	E2	98	22	88	91	10
6	7E	6E	48	C3	A3	B6	1E	42	3A	6B	28	54	FA	85	3D	BA
7	2B	79	0A	15	9B	9F	5E	CA	4E	D4	AC	E5	F3	73	A7	57
8	AF	58	A8	50	F4	EA	D6	74	4F	AE	E9	D5	E7	E6	AD	E8
9	2C	D7	75	7A	EB	16	0B	F5	59	CB	5F	B0	9C	A9	51	A0
A	7F	0C	F6	6F	17	C4	49	EC	D8	43	1F	2D	A4	76	7B	B7
B	CC	BB	3E	5A	FB	60	B1	86	3B	52	A1	6C	AA	55	29	9D
C	97	B2	87	90	61	BE	DC	FC	BC	95	CF	CD	37	3F	5B	D1
D	53	39	84	3C	41	A2	6D	47	14	2A	9E	5D	56	F2	D3	AB
E	44	11	92	D9	23	20	2E	89	B4	7C	B8	26	77	99	E3	A5
F	67	4A	ED	DE	C5	31	FE	18	0D	63	8C	80	C0	F7	70	7

Lisa 6. Krüpteerimisvoorud 128-bitise sisendvõtme puhul

Voor	Funktsioon
1	TulpadeSegamine(ReaNihutamine(BaidiAsendamine(VõtmeLisamine(Andmed)))))
2	TulpadeSegamine(ReaNihutamine(BaidiAsendamine (VõtmeLisamine(Voor1))))
3	TulpadeSegamine(ReaNihutamine(BaidiAsendamine (VõtmeLisamine(Voor2))))
4	TulpadeSegamine(ReaNihutamine(BaidiAsendamine (VõtmeLisamine(Voor3))))
5	TulpadeSegamine(ReaNihutamine(BaidiAsendamine (VõtmeLisamine(Voor4))))
6	TulpadeSegamine(ReaNihutamine(BaidiAsendamine (VõtmeLisamine(Voor5))))
7	TulpadeSegamine(ReaNihutamine(BaidiAsendamine (VõtmeLisamine(Voor6))))
8	TulpadeSegamine(ReaNihutamine(BaidiAsendamine (VõtmeLisamine(Voor7))))
9	TulpadeSegamine(ReaNihutamine(BaidiAsendamine (VõtmeLisamine(Voor8))))
10	VõtmeLisamine(ReaNihutamine(BaidiAsendamine (VõtmeLisamine(Voor9)))

Lisa 7. Dekrüpteerimisvoorud 128-bitise sisendvõtme puhul

Voor	Funktsioon
1	VõtmeLisamine(BaidiAsendamine(ReaNihutamine(VõtmeLisamine(Andmed)))))
2	VõtmeLisamine(BaidiAsendamine(ReaNihutamine (TulpadeSegamine(Voor1))))
3	VõtmeLisamine(BaidiAsendamine(ReaNihutamine (TulpadeSegamine(Voor2))))
4	VõtmeLisamine(BaidiAsendamine(ReaNihutamine (TulpadeSegamine(Voor3))))
5	VõtmeLisamine(BaidiAsendamine(ReaNihutamine (TulpadeSegamine(Voor4))))
6	VõtmeLisamine(BaidiAsendamine(ReaNihutamine (TulpadeSegamine(Voor5))))
7	VõtmeLisamine(BaidiAsendamine(ReaNihutamine (TulpadeSegamine(Voor6))))
8	VõtmeLisamine(BaidiAsendamine(ReaNihutamine (TulpadeSegamine(Voor7))))
9	VõtmeLisamine(BaidiAsendamine(ReaNihutamine (TulpadeSegamine(Voor8))))
10	VõtmeLisamine(BaidiAsendamine(ReaNihutamine (TulpadeSegamine(Voor9)))

Lisa 8. CD

CD sisu:

1. Lõputöö pdf kujul
2. Riistvaratestide failid
3. Simulatsioonitestide failid

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üleüldsusele kättesaadavaks tegemiseks

Mina, Hendrik Türk

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

"AES ALGORITMI REALISEERIMINE VHDL-IS NING TESTIMINE FPGA RIISTVARAL"

mille juhendaja on Margus Rosin

(a) reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmargil kuni autoriõiguse kehtivuse tähtaja loppemiseni;
(b) üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace'i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile;
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, 16.05.2015