Әл-Фараби атындағы Қазақ Ұлттық университеті

ӘОЖ 621.383 Қолжазба құқығында

СҚАБЫЛОВ ӘЛІШЕР ӘЛИҰЛЫ

ПЛИС негізінде шифрлеуге арналған аппараттық бағдарламалық қамтамасыздандыру

8D06201 – Радиотехника, электроника және телекоммуникациялар

Философия докторы (PhD)
дәрежесін алу үшін дайындалған диссертация

Ғылыми кеңесшілер:
Әл-Фараби атындағы ҚазҰУ
PhD, Налибаев Еркебулан
Дюсенбекұлы

Кенпук Ұлттық университеті,
профессор
Джонгтае Парк

Қазақстан Республикасы
Алматы, 2024

МАЗМҰНЫ

АНЫҚТАМАЛАР, БЕЛГІЛЕУЛЕР ЖӘНЕ ҚЫСҚАРТУЛАР	3
КІРІСПЕ	4
1 ПЛИС ЖӘНЕ АҚПАРАТТЫҚ-КОММУНИКАЦИЯЛЫҚ ТЕХНОЛОГИЯЛАРДАҒЫ ШИФРЛЕУ АЛГОРИТМДЕРІ ЖӘНЕ ОЛАРДЫ ЗЕРТТЕУ ӘДІСТЕРІНІҢ РӨЛІ	10
1.1 Заманауи шифрлеу алгоритмдеріне кіріспе және олардың деректер қауіпсіздігін қамтамасыз етудегі рөлі	10
1.2 Шифрлеу алгоритмдерінің сенімділігін бағалау критерийлері және олардың қауіпсіздігін талдау әдістеріне шолу	19
1.3 Шифрлеу алгоритмдерін аппараттық құралдармен жүзеге асыру құралы ретінде ПЛИС қолдану	21
Бөлім бойынша қорытынды	26
2 КӨПМҮШЕЛІК КОЭФФИЦИЕНТТЕРГЕ НЕГІЗДЕЛГЕН СИММЕТРИЯЛЫ БЛОКТЫҚ ШИФРЛЕУ АЛГОРИТМІ	27
2.1 Ауыстыру–алмастыру жүйесі негізінде симметриялы блоктық шифрлеу алгоритмі	27
2.2 Келтірілмейтін көпмүшелер модулі бойынша көпмүшелік көбейткіштерді құру	31
2.3 Құрылған симметриялық блоктық шифрлеу алгоритмінің сенімділігін зерттеу	36
2.4 Кілттерді қорғауды жақсарту үшін динамикалық хаосты қолдану	40
Бөлім бойынша қорытынды	43
3 МӘЛІМЕТТЕРДІ ШИФРЛЕУ ҮШІН БАҒДАРЛАМАЛЫҚ ЖӘНЕ АППАРАТТЫҚ ҚҰРАЛДАРДЫ ӘЗІРЛЕУ	44
3.1 Цифрлық технологиялардағы тиімділік пен қауіпсіздікті арттыру	44
3.2 ПЛИС негізіндегі хаостық генераторларды пайдалана отырып, псевдокездейсоқ сандарды алу	51
3.3 ПЛИС негізіндегі ашық кілтпен шифрлеу жүйелерін жеделдету бойынша жұмыс	59
3.4 ПЛИС негізінде деректерді шифрлеу үшін бағдарламалық және аппараттық қамтамасыз ету	67
Бөлім бойынша қорытынды	77
ҚОРЫТЫНДЫ	79
ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ	80
ҚОСЫМША А	89
ҚОСЫМША Ә	91
ҚОСЫМША Б	93
ҚОСЫМША В	95
ҚОСЫМША Г	103
ҚОСЫМША Ғ	105
ҚОСЫМША Д	107
ҚОСЫМША Е	112

АНЫҚТАМАЛАР, БЕЛГІЛЕУЛЕР ЖӘНЕ ҚЫСҚАРТУЛАР

Берілген диссертациялық жұмыста келесі терминдер мен анықтамалар қолданылды:

АКТ – ақпараттық-коммуникациялық технологиялар
АҚЖ – ақпараттық қауіпсіздік жүйелері
АККҚ – ақпаратты криптографиялық қорғау құралдары
ЕҮОБ – ең үлкен ортақ бөлгіш
ЖҚҚ – жартылай қалдықтарды қалыптастырушы
ПЕКСЖ – позициялық емес көпмүшелік санау жүйесі
ПКТ – псевдокездейсоқ тізбек
ПЛИС – (ағылш. FPGA, field-programmable gate array) программаланатын логикалық интегралды схема
ЭҚК – эллиптикалық қисықтардағы криптография
AES – (ағылш. Advanced Encryption Standard), АҚШ-тың деректерді шифрлеу стандарты;
ASIC – (ағылш. Application-specific integrted circuit), нақты қолдануға арналған интегралды схема
DSA – (ағылш. Digital Signal Algoritm), сандық қолтаңба алгоритмі
DES – (ағылш. Data Encryption Standard), АҚШ-тың деректерді шифрлеу стандарты
DSP – (ағылш. Digital Signal Processor), сандық сигналдарды өңдеу процессоры блоктары
SoC – (ағылш. System on Chip), чиптегі жүйелер
LUT – (ағылш. Lookup Table), ақиқат кестесі
NIST – (ағылш. National Institute of Standards and Technology), АҚШ-тың Ұлттық стандарттар және технологиялар институты
RSA – (ағылш. Rivest, Shamir және Adleman), Ашық кілтті криптографиялық алгоритм
FIPS – (ағылш. Federal Information Processing Standards) – ақпаратты өңдеудің федералдық стандарты
IDEA – (ағылш. International data Encryption Algorithm, деректерді шифрлеудың халықаралық алгоритмі) — деректерді шифрлеудың симметриялық блоктық алгоритмі

КІРІСПЕ

Жұмыстың жалпы сипаттамасы

Осы зерттеуде талқыланған докторлық диссертация телекоммуникациялық желілерде, соның ішінде заттар интернеті мен қауіпсіз байланыс жүйелерінде деректерді шифрлеуға арналған аппараттық және бағдарламалық жасақтаманы әзірлеуге бағытталған. Зерттеу параллельді және конвейерлі деректерді өңдеу арқылы жоғары жылдамдықты және сенімді шифрлеуға қол жеткізу үшін программаланатын логикалық интегралды схемаларды (ПЛИС) пайдалануға негізделген тәсілді ұсынады. Жұмыстың негізгі аспектісі шифрланған ақпараттың қауіпсіздігін едәуір жақсартатын кілт ретінде пайдаланылатын псевдокездейсоқ сандарды құру үшін хаос генераторларын қолдануды қамтиды. Диссертацияда ұсынылған шифрлеу алгоритмі позициялық емес көпмүшелік санау жүйесіне (ПЕКСЖ) негізделген және дәстүрлі шифрлеу әдістерінен ерекшеленетін деректерді қорғаудың жаңа шешімдерін ұсынады. Бұл алгоритмді ПЛИС-ке енгізу арқылы зерттеу технологиялық тәуелсіздік пен қауіпсіздікті қамтамасыз ететін ақпараттық қауіпсіздіктің заманауи стандарттарына сәйкес келетін отандық шифрлеу құрылғысын құруға бағытталған. Тұтастай алғанда, бұл зерттеу ақпараттық-коммуникация технологияларға (АКТ) негізделген қауіпсіздікті жақсарту үшін деректерді шифрлеу технологиялары мен сенімділік шешімдерін дамытуға айтарлықтай үлес қосады. Бұл үлес қазіргі байланыс жүйелеріндегі деректерді қорғаудың негізгі аспектілеріне әсер ететін ақпараттық қауіпсіздік пен технологиялық құпиялылық саласында өте маңызды.

Зерттеудің өзектілігі

Бұл зерттеудің өзектілігі электрондық ақпараттық ресурстарды, ақпараттық жүйелерді және байланыс инфрақұрылымын қорғауды қамтамасыз етуге қойылатын талаптардың артуына және құпия байланыс пен бейне байланыс сияқты мамандандырылған жүйелерге байланысты. Цифрлық ақпараттың көлемі үнемі өсіп келе жатқан сайын және оның стратегиялық маңыздылығы жағдайында, деректерді тиімді шифрлеу мәселелері бірінші кезектегі маңызға ие болуда.

ПЛИС икемділігі, ауқымдылығы және өнімділігі жоғары параллельді деректерді өңдеу қабілетінің арқасында осы мәселелерді шешудің бірегей мүмкіндіктерін ұсынады. Ақпаратты қорғау жүйелеріне ПЛИС деректерді шифрлеу жылдамдығын айтарлықтай жақсартуға мүмкіндік береді, сонымен бірге олардың құпиялылығы мен тұтастығының жоғары дәрежесін қамтамасыз етеді. Бұл ПЛИС-ті әсіресе қауіпсіздік пен өнімділікке қойылатын талаптар жоғары болатын арнайы байланыс және бейне байланыс жүйелері үшін де таңдау шешімі етеді.

Бұл зерттеудің негізгі аспектілерінің бірі-шифрлеу кілттерін жасау үшін псевдокездейсоқ сандардың көзі ретінде хаос генераторларын пайдалану. Бұл

тәсіл деректердің қауіпсіздігін едәуір арттырады, өйткені хаостық тізбектерді болжау және талдау қиын, бұл тиісті кілтсіз шифрды ашу процесін өте күрделі етеді.

Осы зерттеу аясында жасалған және ПЕКСЖ негізделген шифрлеу алгоритмі дәстүрлі шифрлеу әдістерінен айтарлықтай ерекшеленетін шешім болып табылады. Бұл шифрлеудың қауіпсіздік деңгейін арттырып қана қоймайды, сонымен қатар деректерді өңдеу процестерін оңтайландыру арқылы оның тиімділігін арттырады.

Шифрлеудың бұл тәсілі ақпаратты қорғаудың заманауи талаптарына жауап беретін отандық шифрлеу құрылғысын жасауға мүмкіндік береді. Қазіргі әлемде технологиялық егемендік ұлттық қауіпсіздікті қамтамасыз ету үшін барған сайын өзекті бола бастаған кезде, әр ел үшін өзінің шифрлеу алгоритмдері мен технологияларын құру мүмкіндігі стратегиялық артықшылықты білдіреді. Деректерді шифрлеу саласында өз шешімдерін әзірлеу және енгізу ақпараттың сенімді қорғалуын қамтамасыз етіп қана қоймай, сонымен қатар елдің технологиялық тәуелсіздігі мен қауіпсіздігін нығайта отырып, ұлттық технологиялық және ғылыми құзыреттердің дамуына ықпал етеді.

Осылайша, бұл зерттеудің өзектілігі деректерді шифрлеудың тиімді және сенімді әдістерін әзірлеудің техникалық аспектілеріне ғана емес, сонымен қатар ақпараттық қауіпсіздік саласындағы технологиялық егемендікті қамтамасыз етудің стратегиялық қажеттілігіне байланысты.

Диссертациялық жұмыстың мақсаты ПЛИС негізінде деректерді шифрлеудың кешенді ақпараттық-бағдарламалық жасақтаманы әзірлеу және енгізу болып табылады.

Зерттеу объектісі

1. Деректерді шифрлеу жүйелері:

- Телекоммуникациялық жүйелерде қолданылады;
- Заттар интернеті саласында;
- Арнайы және бейне байланыс жүйелерінде.

2. Программаланатын логикалық интегралды схемалар (ПЛИС):

- Шифрлеу алгоритмдерін іске асырудың негізі ретінде.

3. Хаос генераторлары:

- Шифрлеу кілттері ретінде қызмет ететін псевдокездейсоқ сандарды жасау үшін пайдалану.

4. Шифрлеу алгоритмдері:

- Позициялық емес көпмүшелік санау жүйесіне негізделген.

Зерттеу пәні телекоммуникациялық жүйелер және арнайы байланыс жүйелері:

- программаланатын логикалық интегралдық схемалар (ПЛИС);

- позициялық емес көпмүшелік санау жүйелері;
- псевдокездейсоқ сандар генераторы.

Зерттеу міндеттері

1. Деректерді шифрлеу алгоритмдерін зерттеу: деректерді шифрлеу үшін қолданылатын дәстүрлі және заманауи шифрлеу әдістерін зерттеу және салыстыру;
2. Бағдарламалық және аппараттық құралдарды әзірлеу: деректерді шифрлеу жүйелерінің негізі ретінде ПЛИС пайдалану потенциалын бағалау. ПЛИС негізінде аппараттық архитектураны құру, шифрлеу алгоритмдерін оңтайландыру және бағдарламалық және аппараттық құрамдас бөліктерді біріктіру;
3. Шифрлеу алгоритмін әзірлеу: позициялық емес көпмүшелік санау жүйесіне негізделген және ПЛИС-те іске асыруға бейімделген деректерді шифрлеудың тиімді алгоритмін құру;
4. Хаос генераторларын қолдану: сенімді шифрлеу кілттері ретінде қызмет ететін псевдокездейсоқ сандарды құру үшін хаос генераторларын қолдану әдістемесін әзірлеу;
5. Қауіпсіздік пен тиімділікті бағалау: ақпараттық-коммуникациялық технологиялардағы деректерді шифрлеудың басқа әдістерімен салыстырғанда әзірленген құралдардың өнімділігін, күші мен тиімділігін талдау;
6. ПЛИС-ке алгоритмді енгізу және тестілеу: оның практикалық қолданылуы мен өнімділігін бағалау үшін ПЛИС-та әзірленген шифрлеу алгоритмін енгізу және тестілеу.

Диссертациялық жұмыстың **жаңалығы** мен ерекшелігі, **алғаш рет:**

1. ПЕКСЖ негізінде ПЛИС арқылы шифрлеу алгоритмі енгізілді және тексерілді;
2. Шифрлеу кілттері ретінде қызмет ететін жоғары энтропиялық псевдокездейсоқ сандарды жасау үшін хаос генераторларын тікелей ПЛИС-та қолдану алғаш рет жүзеге асырылды, бұл жүйенің қауіпсіздік деңгейін едәуір арттырады;
3. Алғаш рет қарапайым телекоммуникациялық жүйелерде деректерді жоғары өнімді және қауіпсіз шифрлеуды қамтамасыз етуге қабілетті ПЛИС аппараттық құралдары мен мамандандырылған бағдарламалық жасақтаманы біріктіретін кешенді шешім ұсынылды;
4. Шифрлеу алгоритмдерінде қолдануға арналған келтірілмейтін көпмүшелік мәліметтер базасы ПЛИС-те алғаш рет құрылды және жүйеленді, бұл шифрлеу кілттерін құру үшін көпмүшелерді таңдаудың негізгі процесі. Көпмүшелердің келтірілмейтіндігін аппараттық тексеру алғаш рет ПЛИС-ке тікелей енгізілді, бұл көпмүшелерді тексеру процесін автоматтандыруға және жеделдетуге мүмкіндік береді, соның ішінде деректерді шифрлеу үшін кілттерді таңдаудың жоғары сенімділігі мен жеделдігін қамтамасыз етеді.

Қорғауға шығарылатын негізгі тұжырымдар

1. Аппараттық қорғаныста қолдануға арналған келтірілмейтін көпмүшелер генераторы ПЛИС Artix-7 құрылғысы арқылы 16 биттік кілтті жасауды қамтамасыз етеді;

2. ПЛИС негізінде жүзеге асырылатын хаостық сигналдар мен псевдокездейсоқ сандар генераторлары жіберуші және қабылдаушы құрылғылардағы ақпараттық блоктардың ұзындығын кездейсоқ бөлу арқылы шифрланған ақпараттың крипто-төзімділігін арттырады;

3. Паралельді және конвейерлі есептеу әдістерінің арқасында ПЛИС негізіндегі ашық кілтті шифрлау алгоритмін аппараттық іске асырудың жылдамдығы бір тактыні 6,239 нс. дейін жеткізуге болады.

Жұмыстың теориялық және практикалық маңыздылығы

Бұл зерттеудің теориялық маңыздылығы ақпараттық-коммуникациялық технологиялар (АКТ) және олардың мәліметтерді шифрлеу мен шифрын шешуде, сондай-ақ ақпараттық қауіпсіздікте қолдану саласындағы теориялық негіздері мен түсінігін дамытуда жатыр. Зерттеу АКТ жұмысының принциптерін, деректерді шифрлеу және шифрын ашу ерекшеліктерін тереңірек зерттеуге, сондай-ақ ақпараттық қауіпсіздікті қамтамасыз етудің жаңа әдістері мен алгоритмдерін әзірлеуге мүмкіндік береді.

Зерттеудің тәжірибелік маңыздылығы тиімді және сенімді деректерді шифрлеу және дешифрлеу жүйелерін енгізу үшін ПЛИС негізінде аппараттық және бағдарламалық қамтамасыз етуді құруда жатыр. Бұл ақпаратты рұқсат етілмеген қол жеткізуден қорғауды қамтамасыз етеді және берілетін және сақталған деректердің құпиялылығын қамтамасыз етеді.

Бұл ғылыми зерттеудің нәтижелері әртүрлі салаларда, соның ішінде телекоммуникация, электроника, ақпараттық жүйелер және т.б. ПЛИС негізіндегі аппараттық және бағдарламалық шешімдер, деректер қауіпсіздігі мен ақпаратты қорғауды қамтамасыз ету үшін әртүрлі АКТ құрылғылары мен жүйелеріне біріктірілуі мүмкін. Бұл олардың сенімділігін, құпиялылығын және тұтастығын қамтамасыз ететін заманауи байланыс және деректерді беру жүйелерін дамытуға жаңа мүмкіндіктер ашады.

Алынған нәтижелер және ПЛИС негізіндегі деректерді шифрлеу және дешифрлеудің әзірленген әдістері мен алгоритмдері ақпараттық-коммуникациялық технологияларда, бұлтты сақтауда, электрондық коммерция жүйелерінде және т.б. ақпаратты қорғауды қажет ететін тәжірибелік қолданбаларда қолданылуы мүмкін. Бұл ақпараттық қауіпсіздікті дамытуға және деректерді қорғаудың неғұрлым сенімді және тиімді жүйелерін құруға ықпал етеді.

Алынған нәтижелердің практикалық маңыздылығы халықаралық рецензияланатын басылымдардағы жарияланымдармен және тұрақты халықаралық конференцияларға қатысумен, сондай-ақ ҚР № 22917 авторлық куәлігімен расталады.

Автордың жеке үлесі Сқабылов Ә.Ә. зерттеу жұмысының барлық көлемін, зерттеу әдісін таңдауды, эксперимент жүргізуді, алынған нәтижелерді өңдеуді, алынған мәліметтерді талдауды өз бетінше жүзеге асырды.

Докторант барлық зерттеулерге және нәтижелерді интерпретациялауға қатысты және кіріспе, әдістеме, нәтижелер, қорытындылар мен графиканы сипаттайтын мақалалардың алғашқы нұсқаларын дайындады. Сонымен қатар, журнал талаптарына сәйкес мақалаларды ресімдеуге және рецензияның әрбір кезеңінен кейін мақаланы жетілдіруге қатысты. Тапсырма қою нәтижелерді талқылау ғылыми жетекшілермен бірлесіп жүргізілді.

Алынған нәтижелердің дұрыстығы мен негізділігі

Диссертациялық жұмыста алынған нәтижелердің дұрыстығы ҚР БжҒМ Білім және ғылым саласында сапаны қамтамасыз ету комитеті (БҒССҚЕК) ұсынған басылымдарда, импакт-факторы нөлден жоғары алыс шетелдердің журналдарында және халықаралық конференциялардың еңбектерінде жарияланымдардың болуымен расталады.

Диссертациялық жұмыстың аппробациясы

Диссертациялық жұмыс материалдары негізінде 10 баспа жұмыстары жарық көрді:

Thomson Reuters деректер базасы бойынша немесе Scopus халықаралық ғылыми деректер базасына енетін басылымдарда жоғары импакт-факторлы мақалалар:

1. Ibraimov M.K., Tynymbayev S.T., **Skabylov A.A.**, Kozhagulov Y., Zhexebay D.M. Development and design of an FPGA-based encoder for NPN //Cogent Engineering. – 2022. – Vol. 9, № 1. – 2008847 **[Q2, 67%]**

2. Aitkhozhayeva Y.Zh., Tynymbayev S., Adilbekkyzy S., Skabylov A., Ibraimov, M. Design and research of the behavioral model for the modular reduction device // Eurasian Physical Technical Journal 7. – 2020. – Vol. 17, № 1. – С. 151–156. **[Q3, 25%]**

ҚР ҒЖМБ ҒЖБССҚК ұсынған басылымдарда жарияланған мақалалар:

1. Kalimoldayev M., Tynymbayev S., Almen D., Skabylov A. Implementation of the nonconductivity check algorithm for a given binary polynomial on the FPGA // Вестник КазНИТУ. – 2020. – Т. 141, № 5. – С. 629-635.

2. Скабылов А.А., Ибраимов М.К., Жексебай Д.М., Кожагулов Е.Т. Программно-аппаратная реализация qpsk на базе zynq и matlab // Вестник КазНПУ им. Абая, серия «Физико-математические науки», №1(77), 2022 г.

3. Скабылов А.А., Жексебай Д.М., Ибраимов М.К., Налибаев Е.Д. Применение хаотических генераторов для секретного ключа в приемо-передающих электронных устройствах // Вестник КазНПУ. – 2023. – Т. 82, № 2. – С. 194-201

Халықаралық конференциялар тезистері жинақтарындағы жарияланымдар:

1. Әлмен Д.Б., Сқабылов Ә.Ә. ПЛИС негізінде GF(2) ақырғы өрістегі келтірілмейтін көпмүшелер базасын құру //Международная научная конференция студентов и молодых ученых, «ФАРАБИ ӘЛЕМІ», 2020. – С. 269.

2. Сқабылов Ә.Ә., Әлмен Д.Б., Жексебай Д.М., Азамат Р.М. Модуль бойынша бөлу құрылғысының әрекеттік моделін жобалау және зерттеу

//Международная научная конференция студентов и молодых ученых, «ФАРАБИ ӘЛЕМІ», 2021. – С. 217.

3. Сқабылов Ә.Ә., Әлмен Д.Б., Жексебай Д.М., Азамат Р.М. ПЛИС негізінде шифрлеу және дешифрлеу алгоритмін аппараттық жүзеге асыру //Международная научная конференция студентов и молодых ученых, «ФАРАБИ ӘЛЕМІ», 2021. – С. 218.

4. Сқабылов Ә.Ә., Жексебай Д.М., Әлмен Д.Б., Ихсан Г.Б. FPGA негізінде qpsk бағдарламалық-ақпараттық комплекс //Международная научная конференция студентов и молодых ученых, «ФАРАБИ ӘЛЕМІ», 2022. – С. 215.

5. Сқабылов Ә.Ә., Налибаев Е.Д., Ибраимов М.К., Данабеков Б., Азамат Р.М //«Бейсызық жүйелердегі хаос және құрылымдар. Теория және тәжірибе» XII Халықаралық ғылыми конференциясы, 2022. – С. 155.

Авторлық куәлік:

1. Сқабылов Ә.Ә., Ибраимов М.К., Сахыбай Т., Жексебай Д.М., Кожагулов Е.Т. Аппаратный комплекс шифрования данных на основе ПЛИС //Авторлық куәлік, 2022. № 22917.

Аяқталған ғылыми-зерттеу жұмысын оқу процесіне енгізу туралы Акт:

1. 2023 жылғы 31 тамыздағы №1 хаттама

Диссертациялық жұмыстың құрылымы мен көлемі

Диссертациялық жұмыс кіріспеден, үш тараудан, қорытындыдан және пайдаланылған әдебиеттер тізімінен тұрады. Жұмыс 54 сурет, 20 формула, 6 кесте, 117 әдебиеттер тізімі мен 114 бетті қамтиды.

1 ПЛИС ЖӘНЕ АҚПАРАТТЫҚ-КОММУНИКАЦИЯЛЫҚ ТЕХНОЛОГИЯЛАРДАҒЫ ШИФРЛЕУ АЛГОРИТМДЕРІ ЖӘНЕ ОЛАРДЫ ЗЕРТТЕУ ӘДІСТЕРІНІҢ РӨЛІ

Деректерді өңдеу және беру шешуші рөл атқаратын қазіргі ақпараттық қоғамда ақпараттың қауіпсіздігі мен құпиялылығын қамтамасыз ету ақпараттық-коммуникациялық технологиялардың ажырамас бөлігіне айналады. Осы тұрғыда, ПЛИС [1] және шифрлеу [2] алгоритмдерін қолдану ақпараттық қауіпсіздікті қамтамасыз етудің маңызды элементіне айналады. ПЛИС мамандандырылған аппараттық шешімдерді құрудың қуатты құралдарын ұсынады, ал шифрлеу алгоритмдері деректерді рұқсатсыз кіруден сенімді қорғауды қамтамасыз етеді.

Бұл жұмыстың маңызды бөлігі – ПЛИС аппараттық шешімдерінің өнімділігі мен қауіпсіздігін бағалау үшін, сондай-ақ, олардың әртүрлі қолданбаларға жарамдылығын анықтау үшін қолданылатын бар зерттеу әдістерін талдау [3]. Бұл талдау ПЛИС шифрлеу жүйелерін жобалау және талдау әдістемесін түсінуді жақсартады.

Біздің зерттеулер ақпараттық қауіпсіздікті қамтамасыз ету және ақпараттық-коммуникациялық технологияларды дамыту контекстінде үлкен маңызға ие. ПЛИС және шифрлеу алгоритмдеріне қатысты мәселелерді зерттеу арқылы біз академиялық және өндірістік контексттерде пайдалы болуы мүмкін жаңа білім мен практикалық ұсыныстар беруді мақсат етеміз.

Диссертацияның келесі бөлімдерінде ақпараттық-коммуникациялық технологиялардағы ПЛИС және шифрлеу алгоритмдерінің рөлі толығырақ қарастырылған, сонымен қатар жұмыста қолданылған зерттеу әдістемесін және болашақ зерттеулердің перспективаларын талқыланған.

1.1 Заманауи шифрлеу алгоритмдеріне кіріспе және олардың деректер қауіпсіздігін қамтамасыз ету

Қазіргі ақпараттық қоғам деректердің қарқынды алмасуымен, ақпараттың құпиялылығы мен тұтастығының маңыздылығымен, сондай-ақ кибершабуылдар қаупімен сипатталады. Бұл тұрғыда шифрлеу алгоритмдері деректердің қауіпсіздігін қамтамасыз етудің негізгі элементіне айналады. Бұл тарауда біз шифрлеу алгоритмдерінің негіздері енгізілген және олардың заманауи ақпараттық-коммуникациялық технологиялардағы рөлі қарастырылған.

Шифрларды жасау және ашу әдістері туралы білім саласы криптография деп аталады [4].

Қазіргі заманғы шифрлеу-дешифрлеу алгоритмдері өте күрделі және оларды қолмен орындау мүмкін емес. Алдымен, ақпаратты қорғау үшін криптографиялық алгоритмдердің қандай түрлері қолданылатыны талқыланды (1.1-сурет).



Сурет 1.1 – Криптографиялық алгоритмдердің классификациясы [5]

Криптографиялық алгоритмдер негізінен үш санатқа бөлінеді:

- криптографиялық түрлендіру процесінде ешбір кілтті пайдаланбайтын кілтсіз алгоритмдер;
- есептеулерінде белгілі бір құпия кілтті пайдаланатын бір кілтті алгоритмдер;
- екі кілтті алгоритмдер, оларда есептердің әртүрлі кезеңдерінде кілттердің екі түрі қолданылады: құпия және ашық.

Хэш функциялары. Олар айнымалы ұзындықтағы деректердің бекітілген өлшем тізбегіне «жиынтығын» орындайды – шын мәнінде, бұл белгілі бір кілттің қатысуымен де, онсыз да орындалуы мүмкін деректердің бақылау сомасы. Мұндай функциялар компьютерлік ақпаратты қорғау саласында кеңінен қолданылады, мысалы: электрондық қолтаңбаны пайдалану мүмкін болмаған (мысалы, ресурстардың жоғары қарқындылығына байланысты) немесе артық болған жағдайларда кез келген деректердің тұтастығын растау.

Кейбір шифрлеу алгоритмдері шифрлеу алгоритмінің өзі құпия болып табылатындығына негізделген. Қазіргі уақытта мұндай әдістер тек тарихи қызығушылық тудырады және тәжірибелік құндылығы жоқ. Барлық заманауи алгоритмдер шифрлеуды және дешифрлеуды басқару үшін кілтті пайдаланады; кілт белгілі болған жағдайда ғана хабарламаның шифрын сәтті шешуге болады. Дешифрлеу үшін қолданылатын кілт шифрлеу үшін қолданылатын кілтпен бірдей болмауы мүмкін, бірақ алгоритмдердің көпшілігінде кілттер бірдей.

Кілт – қолданылатын шифрлеу әдісі үшін мүмкін болатындар жиынтығынан бір түрлендіруді таңдауды қамтамасыз ететін белгілі бір параметрдің (параметрлердің) нақты құпия күйі.

Шифр – хабарды рұқсатсыз оқудан қорғау мақсатында оның пішінін қайтымды түрлендірулер жиынтығы. Шифрда әрқашан екі элемент болады: алгоритм және кілт [6,7].

Шифрлеуды келесі формулалар түрінде көрсетуге болады:

$$C = E_{k1}(M) \tag{1.1}$$

$$M' = D_{k2}(C) \tag{1.2}$$

мұнда: M (message) — ашық ақпарат, ақпараттық қауіпсіздік бойынша әдебиеттерде анық мәтін тіркесі жиі белгіленеді; C (cipher text)– шифрлеу нәтижесінде алынған шифрлық мәтін (немесе криптограмма); E (encryption) – бастапқы мәтінде криптографиялық түрлендірулерді орындайтын шифрлеу функциясы; k_1 (key) – шифрлеу кілті деп аталатын E функциясының параметрі; M' — шифрды шешу нәтижесінде алынған ақпарат; k_1 (key) – ақпараттың шифрын ашу үшін қолданылатын кілт; D (дешифрлеу) шифрлық мәтін бойынша шифрлеуға кері криптографиялық түрлендірулерді орындайтын шифрды шешу функциясы болып табылады [8-10].

Шифрдың ашуға қарсы тұру қасиеті (оның шифрды шешуге төзімділігі) криптографиялық беріктік (немесе сенімділік) деп аталады және әдетте шифрды шешу алгоритмінің күрделілігімен өлшенеді.

Кез келген криптографиялық шифрлеу жүйесі келесі талаптарға толық жауап беретіндей ғана қауіпсіз болады [11]:

- тіпті белгілі мәтінмен оны ашудың мүмкін еместігі, ал хабарлама ашылған жағдайда, бұрын жіберілген және болашақта берілетін хабарламалардың қауіпсіздігінің кепілі;
- қазіргі есептеуіш құралдарды қолданудың өзінде ақпараттың шынайы мазмұнын ашуға мүмкіндік бермейтін шифрлеу опцияларының жеткілікті үлкен саны;
- әртүрлі математикалық әдістерді қолдану арқылы ашуға мүмкіндік бермейтін шифрдің жоғары күрделілігі;
- кілтті және шифрлеу алгоритмін, сондай-ақ шифрлеу құрылғыларының өздерін сақтаудың кепілдік берілген сенімділігі [12].

Мәтінді қорғау әдістерін бағдарламалық және аппараттық түрде іске асыру мүмкін.

Бағдарламалық қамтамасыз етуді жүзеге асыру шифрлеу сатысында да, шифрды шешу сатысында да есептеу процестерінің негізінде жүзеге асырылады. Аппараттық құралдарды іске асыру арнайы аппараттық құралдарды пайдалануға негізделген.

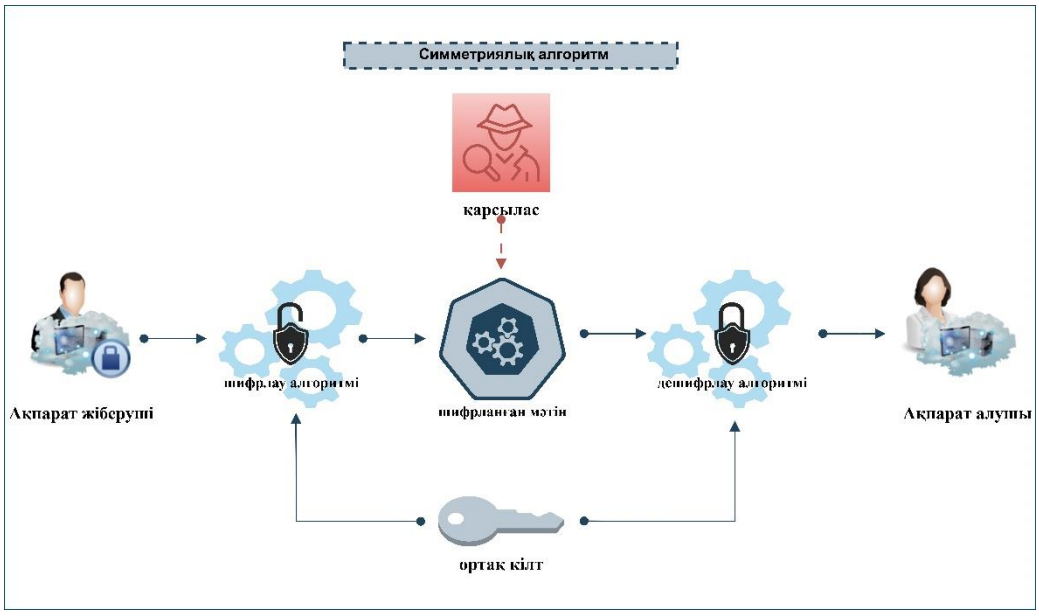
Криптографиялық қорғаудың негізгі міндеті потенциалды қарсыластың криптограммаларды ұстап алуға және жазуға қабілетті кез келген техникалық құрал-жабдықтары болған жағдайда және ол криптограммалардың кейбір үзінділерін және бастапқы хабарламаның сәйкес бөліктерін білетін жағдайда ақпаратқа қол жеткізудің мүмкін еместігін қамтамасыз ету болып табылады [13].

Шифрлеу алгоритмдерінің екі негізгі категориясы бар: симметриялық кілт және ашық кілт (асимметриялық кілт) [14-15].

Симметриялық кілт алгоритмдері – шифрлеу үшін де, шифрды ашу үшін де бір кілтті қолданатын шифрлеу алгоритмінің бір түрі [16]. Бұл кілт құпия сақталады және қарым-қатынасқа қатысушы тараптар арасында берілуі керек. Симметриялық кілт алгоритмдері асимметриялық аналогтарына қарағанда жылдамырақ болады, бұл оларды үлкен көлемдегі деректерді шифрлеуға өте ыңғайлы етеді.

Симметриялық кілт алгоритмдерінің жылдамдығынан басқа да артықшылықтары бар. Олар асимметриялық алгоритмдерге қарағанда өңдеу қуатын аз талап етеді және белгілі бір шабуыл түрлеріне аз сезімтал. Дегенмен, симметриялық кілт алгоритмдерінің кейбір шектеулері де бар. Ең маңызды шектеу – құпия кілтті байланысқа қатысатын тараптар арасында бөлісу қажеттілігі. Мұны қауіпсіз орындау қиын болуы мүмкін, әсіресе тараптар бір-біріне сенбейтін жағдайларда.

Бұл шектеуді жеңу үшін кілттер алмасу алгоритмдері әзірленді. Бұл алгоритмдер екі тарапқа симметриялық кілтті қауіпсіз емес арна арқылы қауіпсіз алмасуға мүмкіндік береді. Симметриялық шифрлеу кезінде құпия кілт деректерді шифрлеу және шифрын ашу үшін қолданылады (1.2 сурет).



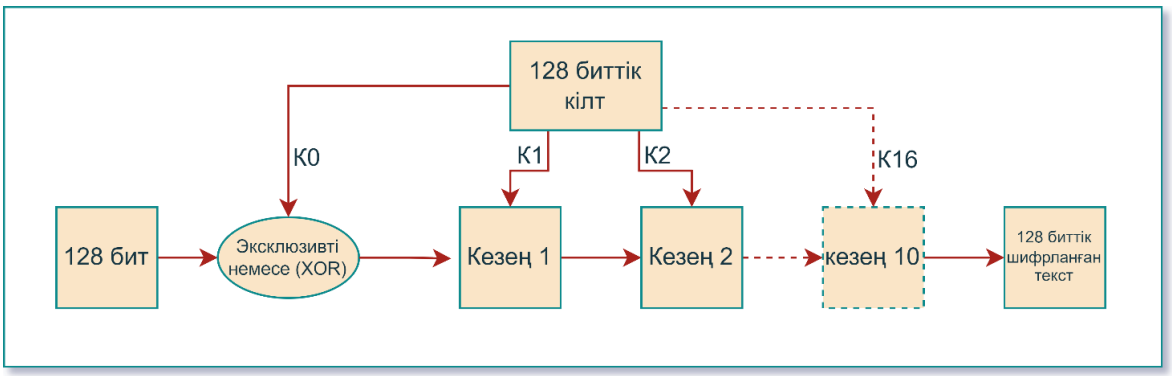
Сурет 1.2 – Симметриялық деректерді шифрлеу алгоритмінің схемасы

Осылайша, симметриялық кілт алгоритмдері шифрлеу және шифрды шешу үшін бірдей кілтті пайдаланатын шифрлеу алгоритмінің түрі болып табылады.

Олар жылдам, тиімді және үлкен көлемдегі деректерді шифрлеуға жарамды. Ең көп қолданылатын симметриялық блокты шифрлеу алгоритмі – Advanced Encryption Standard (AES) [17], Оны 2001 жылы АҚШ ұлттық стандарттар және технологиялар институты (NIST) деректерді шифрлеу стандартының Data Encryption Standard (DES) [18] орнына таңдады. AES үш кілт өлшеміне ие: 128, 192 және 256 бит. Кілттің өлшемі неғұрлым үлкен болса, шифрлеу соғұрлым сенімді болады. 1.3-суретте AES алгоритмінің құрылымдық схемасы көрсетілген. Бұл әдіс әртүрлі блоктардан тұрады, мысалы:

– Кілтті генерациялау;
– 128-биттік кіріс деректері;
– Бірнеше айналым;
– Шифрланған деректер.

AES құрылымы ауыстыру желісіне негізделген. Кіріс деректері кілт өлшеміне байланысты шифрлеу процесінің бірнеше айналымынан өтеді. Әрбір айналым байтты ауыстыру, жолды ауыстыру, бағанды араластыру және пернені қосу операцияларынан тұрады.



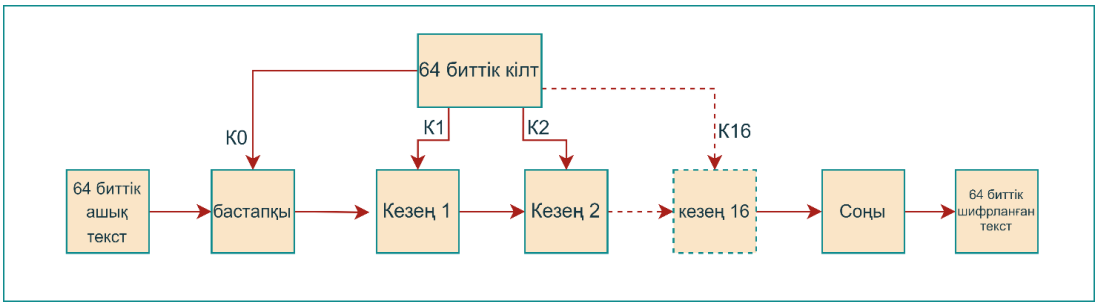
Сурет 1.3 – AES алгоритмінің құрылымдық схемасы

Одан бөлек басқа танымал алгоритмдерге DES жатады.
Бұл әдіс әртүрлі блоктардан тұрады, мысалы:

– Кілт генераторы;
– 64-биттік кіріс деректері;
– Бастапқы ауыстыру;
– Он алты айналым;
– Соңғы қайта реттеу;
– Шифрланған деректер.

Бастапқы ауыстыру ашық мәтінде орындалады және бастапқы ауыстыру кестесін пайдаланып мәтіндік блоктың екі жартысын жасайды. Алынған бөліктер шифрлеу процесінің 16 айналымынан өтеді. Әрбір айналым кеңейту ауыстыруынан, S-қорапты ауыстырудан, P-боксты ауыстырудан, xoring және ауыстыру операцияларынан тұрады. 1.4-суретте DES алгоритмінің құрылымдық схемасы көрсетілген. Ол 64 бит ақпарат алады (кез келген сәтте 8 пиксель) және оған бастапқы ауыстыруды (IP) қолданады. Орналастырылған ақпарат әр түрлі

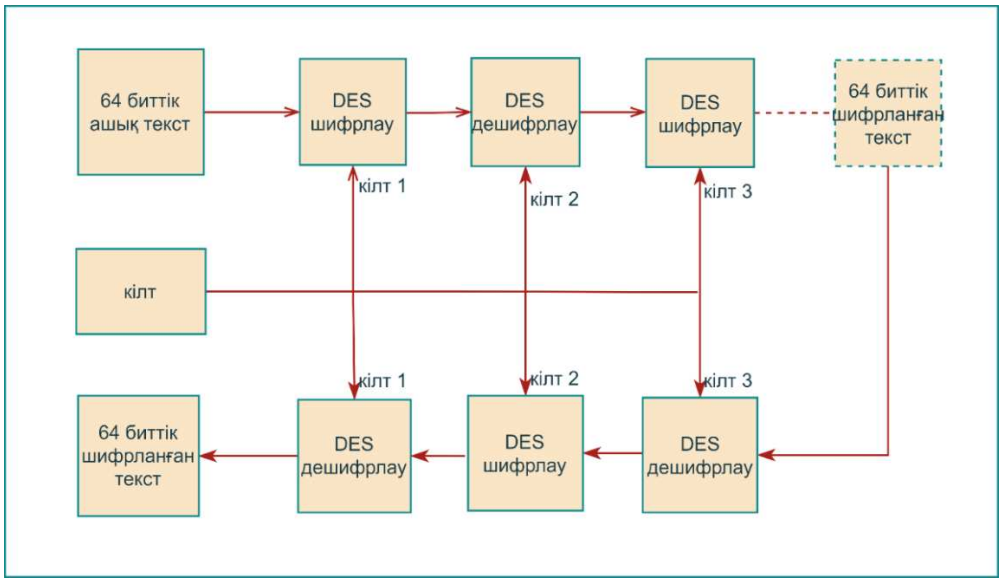
кілттерді (48 бит) пайдаланып, соңғы шифр мәтінін жасау үшін ауыстырылатын он алты циклдік операциядан өтеді. Бастапқы кілт ұзындығы 64 бит, оның 8 бит тақтылықты тексеру үшін сақталады. Шифрды шешу шифрлеуға ұқсас айналым механизмі пайдаланылады, бірақ қосымша кілттердің реті өзгертіледі.Тікелей және кері процедуралар көп айналымда орындалғанымен, DES қауіпсіздік механизмін көптеген жолдармен бұзуға болады. Дөрекі күш шабуылдары және ашық мәтіндік шабуылдар ең кең таралған әдістемелер болып табылады.



Сурет 1.4 – DES алгоритмінің құрылымдық схемасы

3DES (Үштік DES (TDES)) алгоритмі.

Ол үштік деректерді шифрлеу алгоритмі (TDEA) ретінде де белгілі. TDES – симметриялы кілттер блогының шифры. 1.5-суретте TDES үштік алгоритмінің блок-схемасы көрсетілген. Аты айтып тұрғандай, алгоритм DES алгоритмін шифрлеу, дешифрлеу және кілттерді генерациялау процестерінде үш рет пайдаланады.



Сурет 1.5 – TDES үштік алгоритмінің блок-схемасы

TDES шифрлеу және шифрды шешу үшін үш 56 биттік кілтті пайдаланады. TDES процедурасы төмендегідей пернелерді пайдаланудың үш нұсқасын береді:

1. 1-кілтті таңдау: Барлық үш перне бір-бірінен тәуелсіз. Бұл кілтке ең жақсы балама және кез келген белгілі практикалық шабуылдарға осал емес.

2. 2-кілт таңдау: K3 K1 сияқты, K1 және K2 автономды болып табылады. Ол ортадағы кездесу шабуылдарынан қорғалған, бірақ таңдалған ашық мәтіндік шабуылдарға қарсы дәрменсіз. Ол 2DES ретінде де белгілі.

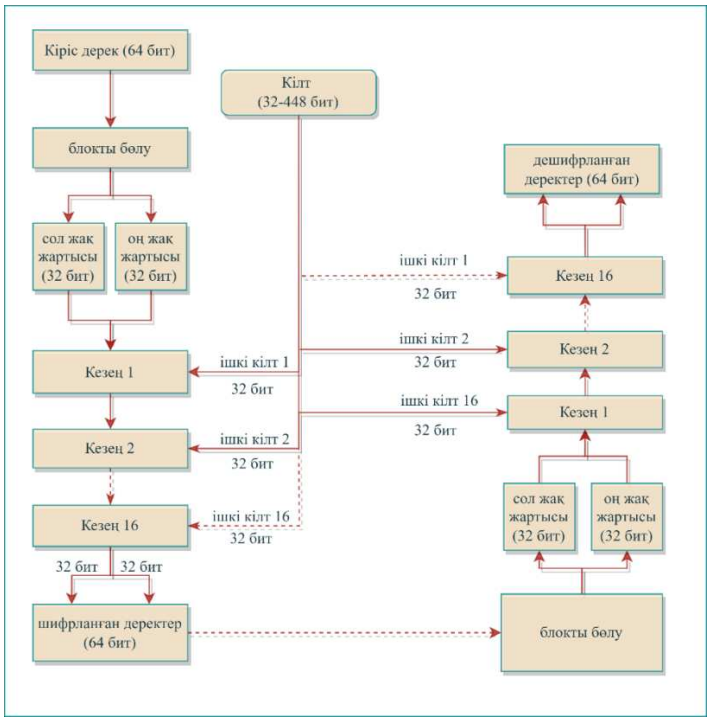
3. 3-кілтті таңдау: Барлық үш перне бір-бірінен ажыратылмайды. Бұл ең әлсіз енгізу опциясы.

Blowfish алгоритмі.

Бұл симметриялық кілт блокты шифрлеу әдісі. Оның негізгі құрамдас бөлігі 16 рет қайталанатын Feistel желісі болып табылады. Blowfish ешбір лицензияны қажет етпейтін алғашқы қауіпсіз блоктық шифрлардың бірі болды және осылайша кез келген адам пайдалана алады. Бұл артықшылық оның криптографиялық бағдарламалаудағы дәрежесін арттырды. Бұл әдіс әртүрлі блоктардан тұрады, мысалы:

– Кілтті генерациялау;
– Деректерді енгізу;
– Блокты бөлу;
– Он алты айналым;
– Шифрланған деректер.

Барлық блоктар мәндері бойынша ұқсас, бірақ, әрбір он алты айналым xoring және айырбастау сияқты әртүрлі операциялардан тұрады. 1.6-суретте Blowfish алгоритмінің блок-схемасы көрсетілген.



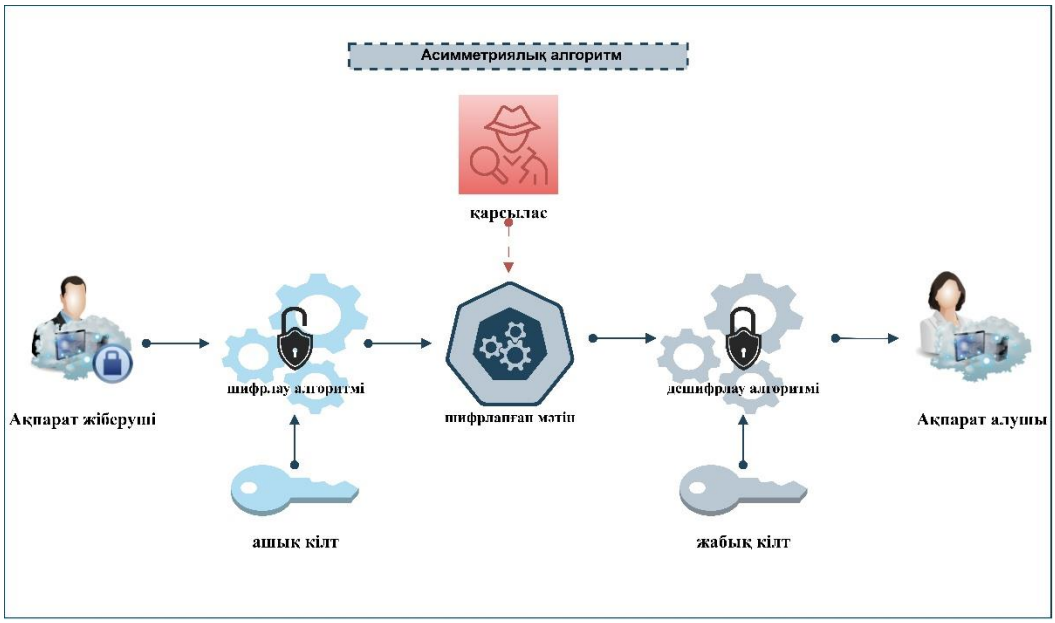
Сурет 1.6 – Blowfish алгоритмінің блок-схемасы

Қолданылатын блок өлшемі 64 бит. Ол 32-448 бит айнымалы кілт ұзындығын пайдаланады. Процесс 64 биттік енгізуді және әр айналымда басқа қосымша кілтті қабылдай отырып, 16 айналымға қайталанады. Кіріс деректері

екі 32 биттік бөлікке бөлінеді, олар сол жарты және оң жартысы ретінде ұсынылған. Бұл екі бөлік xoring және ауыстырудан тұратын 16 айналымнан өтеді. 16-шы айналымнан кейін шифрланған деректерді алу үшін сол және оң жартылар біріктіріледі. Шифры шешілген деректерді алу үшін кері операция орындалады. Бұл алгоритмдер олардың қауіпсіздігін қамтамасыз ету үшін мұқият зерттелген және талданған.

Асимметриялық кілт алгоритмдері деп те аталатын ашық кілт алгоритмдері қазіргі криптографияның маңызды құрамдас бөлігі болып табылады. Олар Интернет сияқты қорғалмаған байланыс арналары арқылы құпия ақпаратты қауіпсіз тасымалдау үшін пайдаланылады. Ашық кілт алгоритмдері екі кілтті қолдануға негізделген: ашық кілт және жеке кілт. Ашық кілт кез-келген адамға қол жетімді, ал жеке кілтті иесі құпия сақтайды.

Ашық кілтті криптография деп те аталатын асимметриялық шифрлеуда (1.7-сурет) көпмүшелер шифрлеу және шифрды шешу алгоритмдерін жасауда қолданылады.



Сурет 1.7 – Асимметриялық деректерді шифрлеу алгоритмінің схемасы

Асимметриялық шифрлеу схемалары деректерді шифрлеу және шифрын ашу үшін жұп кілттерді, ашық кілтті және жеке кілтті пайдаланады. Ашық кілт деректерді шифрлеу үшін, ал жеке кілт деректерді шифрлеу үшін қолданылады. Көпмүшелер шифрлеу қауіпсіздігін қамтамасыз ету үшін шифрлеу және шифрды шешу алгоритмдерін жасауда пайдалынады.

Ең танымал ашық кілт алгоритмі – Ривест-Шамир-Адлеман алгоритмі (RSA) [19], оны алғаш рет 1977 жылы Рон Ривест, Ади Шамир және Леонард Адлеман енгізген. RSA қауіпсіз байланыс, цифрлық қолтаңба және кілт алмасу үшін кеңінен қолданылады. Алгоритм үлкен жай сандарды көбейтудің күрделілігіне негізделген. RSA қауіпсіздігі екі үлкен жай санның көбейтіндісін қазіргі заманғы компьютерлердің көмегімен де көбейту қиын екендігіне

арналған. RSA-да қолданылатын қарапайым сандар неғұрлым көп болса, алгоритм соғұрлым қауіпсіз болады.

Тағы бір кеңінен қолданылатын ашық кілт алгоримі – 1976 жылы Уитфилд Диффи мен Мартин Хеллман енгізген Диффи-Хеллман кілттерімен алмасу. Алгоритм екі тарапқа қауіпті байланыс арнасы арқылы ортақ құпия кілтті орнатуға мүмкіндік береді. Кілт кілттермен алмасу процесінде айырбасталатын тараптардың ашық және жабық кілттерін қолдану арқылы жасалады. Диффи-Хеллман кілттерімен алмасу алгоритмі SSL/TLS, SSH және IPSec сияқты қорғалған байланыс протоколдарында кеңінен қолданылады.

Эллиптикалық қисықтардағы криптография (ЭҚК) – басқа алгоритмдермен салыстырғанда кілт өлшемдерінің кішірек болуына және қауіпсіздіктің жоғары болуына байланысты танымал бола бастаған тағы бір ашық кілт алгоритмі. ЭҚК алғаш рет 1980 жылдардың ортасында Нил Коблиц пен Виктор Миллер енгізді. Алгоритм эллиптикалық қисық бойынша дискретті логарифм есебін шешудің қиындығына негізделген. ЭҚК жүйесінде пайдаланылатын кішірек кілт өлшемдері оны мобильді құрылғылар мен ендірілген жүйелер сияқты ресурстары шектеулі орталарда пайдалану үшін өте қолайлы етеді.

Сандық қолтаңба алгоритмі (ағылш. Digital Signal Algoritm, DSA) – сандық қолтаңбалар үшін пайдаланылатын ашық кілт алгоритмі. Алгоритмді 1991 жылы NIST енгізді және соңғы өрістегі дискретті логарифм есебіне негізделген. DSA SSL/TLS және SSH сияқты қауіпсіз байланыс протоколдарында кеңінен қолданылады[20].

Шифрлеу алгоритмдерінің сапасы қазіргі ақпараттық жүйелерде деректердің қауіпсіздігін қамтамасыз етудің маңызды аспектісі болып табылады. Шифрлеу алгоритмдерін бағалау олардың деректерді рұқсатсыз кіруден қорғаудағы тиімділігін қамтамасыз ету үшін қажет. Шифрлеу алгоритмдерінің сапасын бағалау үшін жылдамдықты, тұрақтылықты, кілт өлшемін, жад талаптарын және іске асырудың қарапайымдылығын қоса алғанда, бірнеше әдістер қолданылады.

Жылдамдық шифрлеу алгоритмдерінің сапасын бағалаудың маңызды факторы болып табылады. Бұл деректерді шифрлеуға және шифрын шешуге кететін уақытты білдіреді.

Шифрлеу алгоритмдерінің сапасын бағалаудың тағы бір маңызды факторы тұрақтылық болып табылады. Ол алгоритмнің өрескел күш шабуылдары немесе дифференциалды криптоталдау сияқты шабуылдарға қарсы тұру қабілетін білдіреді.

Кілт өлшемі де шифрлеу алгоритмдерінің сапасын бағалаудың маңызды факторы болып табылады. Шифрлеу кілтінің өлшемі алгоритмінің күшіне әсер етеді, ал ұзағырақ кілттер әдетте неғұрлым қауіпсіз болып саналады. Дегенмен, ұзағырақ кілттер деректерді шифрлеу және шифрын шешу үшін көбірек өңдеу қуатын қажет етеді, бұл алгоритм жылдамдығына әсер етуі мүмкін.

Жад талаптары да шифрлеу алгоритмдерінің сапасын бағалауда маңызды фактор болып табылады. Алгоритмге қажетті жад көлемі оның өнімділігіне әсер

етуі мүмкін. Төмен жад талаптары бар алгоритмдер әдетте артықшылық береді, себебі олар оңайырақ жүзеге асырылуы мүмкін және іске қосу үшін ресурстар аз қажет.

Шифрлеу алгоритмдерінің сапасын бағалау кезінде іске асырудың қарапайымдылығы да маңызды фактор болып табылады. Аппараттық немесе бағдарламалық жасақтамада оңай іске асырылатын алгоритмдер әдетте артықшылық береді, өйткені олар бар жүйелермен біріктіру оңайырақ. Бұл алгоритмді жүзеге асыруға кететін шығын мен уақытты азайтуға көмектеседі және оны пайдаланушылардың кең ауқымы үшін қолжетімді ете алады.

Шифрлеу алгоритмдері цифрлық ақпаратты қорғауда маңызды рөл атқарады. Бұл бөлімде бар шифрлеу алгоритмдері мен олардың сапасын бағалау әдістеріне шолу жасалды. Әрбір алгоритмнің күшті және әлсіз жақтарын бағалау және нақты пайдалану жағдайына негізделген ең қолайлысын таңдау маңызды [21].

1.2 Шифрлеу алгоритмдерінің сенімділігін бағалау критерийлері және олардың қауіпсіздігін талдау әдістеріне шолу

Деректер қауіпсіздігі бірінші орында тұрған қазіргі цифрлық әлемде шифрлеу алгоритмдері ақпараттың құпиялылығы мен тұтастығын қорғауда басты рөл атқарады. Бұл алгоритмдердің сенімділігі мен қауіпсіздігін қамтамасыз ету үшін оларды әзірлеу және қолдану ғана емес, сонымен қатар олардың тиімділігі мен әртүрлі шабуыл түрлеріне төзімділігін бағалау қажет.

Бұл шолу шифрлеу алгоритмдерінің сенімділігін бағалау критерийлерін, сондай-ақ, олардың қауіпсіздігін талдау әдістерін зерттейді. Біз криптографиялық күшті, ресурс қарқындылығы мен алгоритм тиімділігі мәселелерін зерттеуден бастап шифрлеу жүйелеріне әсер ететін ішкі және сыртқы қауіптерге назар аударуға дейінгі аспектілерді қарастырамыз.

Осы аспектілерді зерделеу цифрлық қауіпсіздік пен киберқорғаныс дәуірінде маңызды болып табылатын ақпараттық жүйелердің сенімділігі мен қауіпсіздігін қамтамасыз етудің принциптері мен әдістерін жақсы түсінуге мүмкіндік береді.

Ақпараттық қауіпсіздік жүйелерінің (АҚЖ) басқа жүйелерден айырмашылығы – ақпараттың сенімді қорғалуына кепілдік беретін қарапайым және бір мәнді тесттер жоқ. Сонымен қатар, АҚЖ-нің тиімділігі және олардың болуы тек негізгі жүйенің жұмысымен байланысты емес. Сондықтан, АҚЖ-нің тиімділігі мәселесін күнделікті тестілеу арқылы шешу мүмкін емес. Мысалы, байланыс жүйесін тестілеу оның жұмыс істейтіндігін растауы мүмкін болса да, оған енгізілген ақпаратты қорғаудың ішкі жүйесінің тиімділігіне толық сенімділік бермейді. Криптографиялық қорғау әдістерін қолданған кезде оның тиімділігін анықтау міндеті ақпаратты қорғау жүйесін құрудан гөрі көп күш пен арнайы білімді қажет етеді. Көбінесе жаңа шифрды талдау тек техникалық міндет емес, ғылыми мәселе болып табылады.

Шифрлеу алгоритмдерінің сенімділігін бағалау оларды таңдау мен пайдаланудың негізгі бөлігі болып табылады. Сенімділікті бағалау кезінде ескеру қажет маңызды критерийлерге мыналар жатады [22]:

1) Кілттерге төзімділік: бұл өте маңызды критерий. Сенімді алгоритм кілттерді бұзу шабуылдарына төзімді болуы керек, яғни шабуылдаушы кілтті оңай болжай немесе есептей алмайды;

2) Делдал шабуылдарына төзімділік: Алгоритм деректерді шабуылдаушы деректерді ұстап, талдай алатын делдал шабуылдарынан қорғауы керек;

3) Криптограмма мәтіндеріне қарсы шабуылға төзімділік: жақсы алгоритм шифр мәтінін талдауға негізделген шабуылдардан қорғауды қамтамасыз етуі керек;

4) Ортадағы адам шабуылына төзімділік: Алгоритм шабуылдаушының жіберуші мен деректерді алушы арасындағы араласуына байланысты шабуылдардың алдын алуы керек.

Қазіргі криптоталдау ықтималдықтар теориясы мен математикалық статистикаға негізделген, сонымен қатар алгебра, сандар теориясы, алгоритмдер теориясы және басқа математикалық пәндер туралы білімді қолданады. Барлық криптоталдау әдістерін төрт негізгі бағыт бойынша жіктеуге болады [23]:

1) Статистикалық криптоталдау: бұл әдіс ашық және шифрланған мәтіндердегі статистикалық заңдылықтарды талдауға негізделген. Ол бұзу үшін пайдаланылуы мүмкін криптожүйелердегі ықтимал осалдықтарды анықтауға тырысады. Көп жағдайда статистикалық талдау ақпаратты шифрлеуға дейін алдын ала қысу (мұрағаттау) және ұзақ псевдокездейсоқ тізбектерді (гаммаларды) пайдалану жағдайында іс жүзінде қиын [24].

2) Алгебралық криптоталдау: бұл әдіс олардың әлсіз жақтарын анықтау үшін шифрлеу алгоритмдерін математикалық талдаудан тұрады. Мысалы, алгебралық криптоталдауды жеңілдетілген эллиптикалық жүйелердегі кілттер класын анықтау үшін пайдалануға болады. Ол ашық мәтінді, шифрланған мәтінді және кілтті байланыстыратын теңдеулер жүйесін құруға және шешуге негізделген [25].

3) Дифференциалды криптоталдау: бұл әдіс бастапқы мәтіндегі өзгерістерге тәуелділікке негізделген шифрланған мәтіндегі өзгерістерді талдайды. Дифференциалды криптоталдау ашық мәтіндердің көптеген жұптарын және олардың арасындағы белгілі бір айырмашылықпен сәйкес шифрланған мәтіндерді талдауды қажет етеді. Бұл әдіс DES алгоритмін сәтті бұзу үшін қолданылды [26].

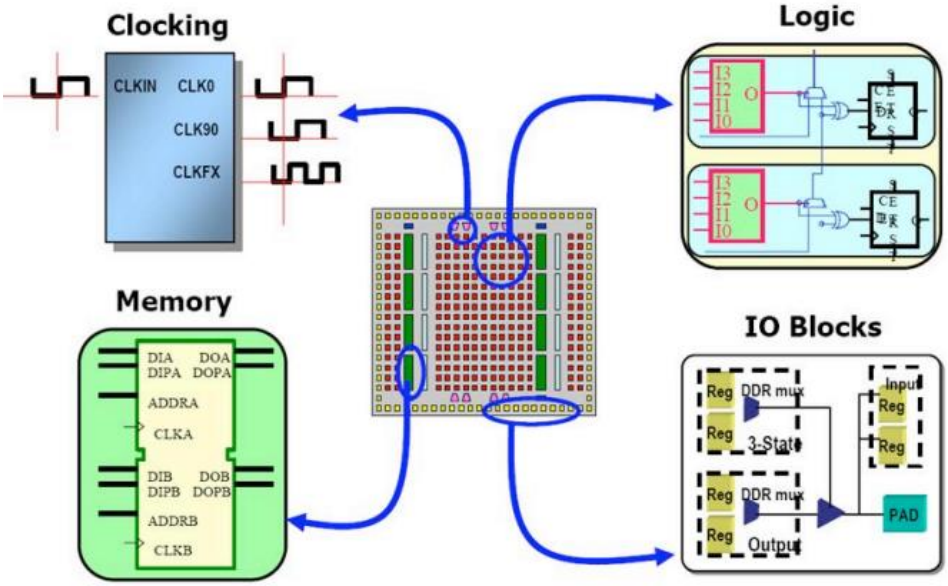
4) Сызықтық криптоталдау: бұл әдіс ашық мәтін мен шифрланған мәтін арасындағы сызықтық қатынастарды зерттейді. Ол DES шифрын бұзу үшін қолданылған және блоктық шифрларды зерттеудің кең таралған әдістерінің бірі болып табылады. Сызықтық криптоталдау блокты шифрлеу алгоритмдеріндегі алмастыру блогының кіріс және шығыс биттері арасындағы статистикалық сызықтық байланыстарды қолданады [27].

Бұл криптоталдау әдістері криптографиялық жүйелердің қауіпсіздігін бағалауда және шабуылдаушылар деректерді қорғауды бұзу үшін пайдалануы мүмкін осалдықтарды табуда маңызды рөл атқарады.

1.3 Шифрлеу алгоритмдерін аппараттық құралдармен жүзеге асыру құралы ретінде ПЛИС қолдану

Құпия ақпарат көлемінің ұдайы ұлғаюы, деректерге рұқсатсыз қол жеткізудің жаңа әдістері мен құралдарының пайда болуы ақпаратты қорғау саласының дамуына алып келеді. Криптографиялық қорғау құралдарына деректерді түрлендірудің криптографиялық алгоритмдерін аппараттық, бағдарламалық-аппараттық және бағдарламалық қамтамасыз ету кіреді, бірақ бұл аппараттық құралдардың іске асырылуы неғұрлым сенімді және жылдам.

Деректерді шифрлеуға кажетті алгоритмдерді іске асырудың стандартты емес аппараттық шешімдерін ПЛИС [28] көмегімен алуға болады, бұл жоғары деңгейлі аппараттық сипаттама тілдерін қолдана отырып, сандық құрылғыларды жобалауға мүмкіндік береді, бұл әзірлеудің күрделілігін төмендетеді және IP ядроларын қолдану арқылы кодты қайта пайдалануға мүмкіндік береді [29]. ПЛИС-тің ішкі құрылымы 1.8-суретте көрсетілген.



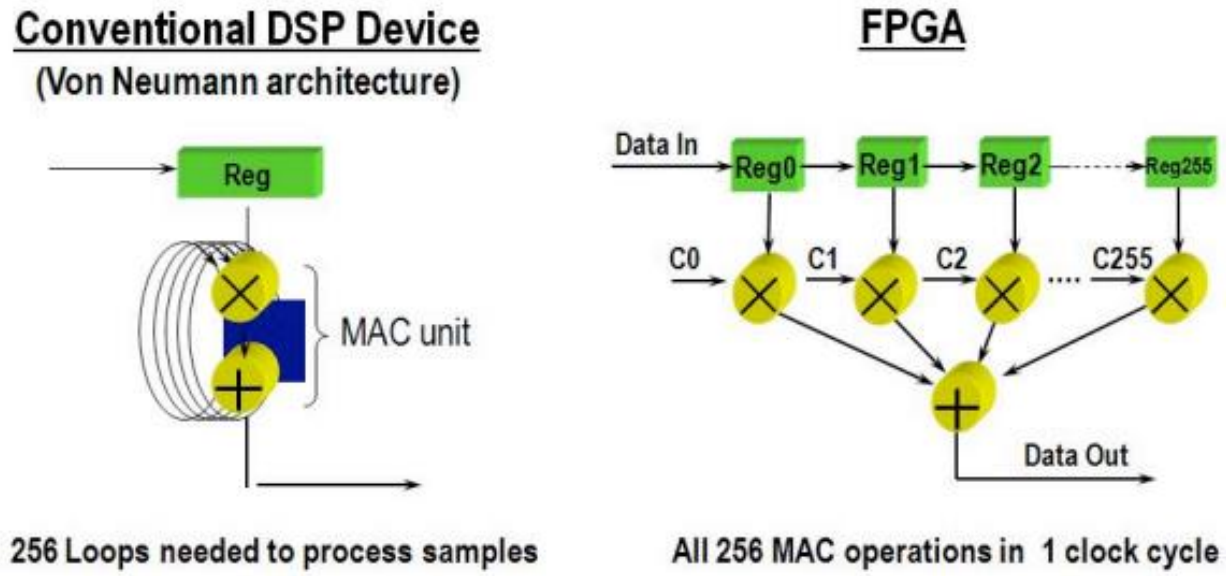
Сурет-1.8. Типтік ПЛИС ішкі құрылымы (Xilinx, Inc. рұқсатымен)

Сонымен қатар, ПЛИС-да шифрлеу алгоритмдерің іске асырудың маңызды артықшылықтарының бірі – микропроцессорларға, графикалық тапсырмалар процессорі (GPU) немесе CPU-ға негізделген шешімдерден жоғары параллель және асинхронды архитектураны құру мүмкіндігі [30]. Сонымен қатар, мұндай алгоритмдерді жүзеге асыру үшін ПЛИС-ты, соның ішінде ПЛИС-тың ең икемді түрін қолдану имплементацияланған жобаның бекітілген параметрлерімен шектеледі. Ұқсас шектеу әмбебап және мамандандырылған іске асыру арасында келу кезінде дәстүрлі болып табылады. Сондықтан, ПЛИС-та

мамандандырылған шешімдердің шекараларын бірыңғай немесе әмбебап нұсқаларға дейін кеңейтуге мүмкіндік беретін құралдардың болуы маңызды.

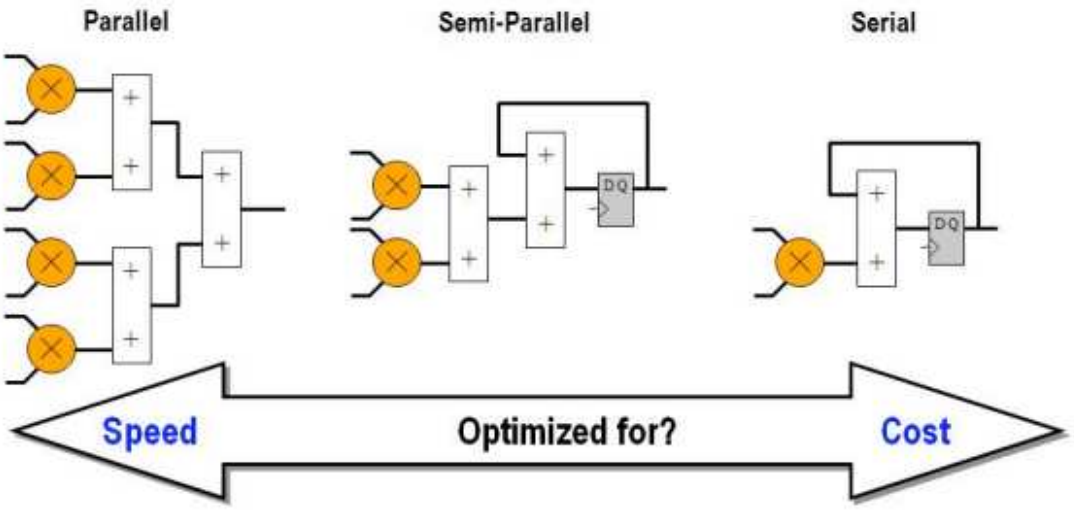
Деректерді шифрлеуды орындау үшін ПЛИС-тің әдеттегі сандық сигнал процессорларынан (ағылш. DSP – Digital Signal Processor) артықшылығы – олардың параллельді пайдалану мүмкіндігі. ПЛИС чиптің әртүрлі бөліктерінде бір уақытта жұмыс істейтін аппараттық функцияларды қайталауға мүмкіндік береді. 1.9-суретте соңғы импульстік сипаттама сүзгісін екі платформада қалай іске асыруға болатыны көрсетілген.

DSP шығыс үлгісін есептеу үшін 256 такт циклін қажет етсе, ПЛИС әрбір тактілік цикл үшін жаңа үлгіні жасайды. DSP чиптері ПЛИС-тан жылдамырақ болуы мүмкін болса да, айырмашылық ешқашан 10 еседен аспайды. Мұндай көптеген сүзгілердің бір уақытта болуы және бір ПЛИС ішінде байланысуы мүмкін екенін ерекше атап өту маңызды.



Сурет 1.9 – DSP және ПЛИС арасындағы соңғы импульстік сипаттама сүзгілерін салыстыру

ПЛИС-тің тағы бір артықшылығы жобалау циклі бойына аумақ пен жылдамдық арасындағы таңдау икемділігі болып табылады. 1.10-суретте өнім сомасының үш түрлі іске асырылуы көрсетілген. Бұл өнімділік пен аумақ арасындағы оңтайлы комбинацияны таңдай отырып, жобаның нақты талаптарына сәйкес ПЛИС-ті орнатуға мүмкіндік береді. Бұл икемділік ПЛИС-ті жоғары өнімді деректерді шифрлеу жүйелерін қоса алғанда, көптеген қосымшалар үшін тартымды таңдау болып табылады.



Сурет 1.10 – ПЛИС-тегі жылдамдық пен аумақты оңтайландыру

Кәдімгі цифрлық чиптерден айырмашылығы, ПЛИС жұмысының логикасы бастапқыда өндіріс сатысында анықталмайды, бірақ бағдарламалау арқылы беріледі [31]. ПЛИС жадында белгілі бір кодтар жазылады, олардың көмегімен әр түрлі деңгейдегі сандық құрылғыларды жасауға болады.

ПЛИС қолдану келесі артықшылықтарды ұсынады [32]: жоғары тактілік жиілік, бірнеше ГГц ретімен ПЛИС жұмыс жиілігі; бір такті ішінде көбейтуге мүмкіндік беретін мультипликаторлардың аппараттық іске асырылуы; схеманы әзірлеудің жылдам уақыты (жадқа қажетті кодты жазу жеткілікті); конфигурацияларды жеделдетілген түрлендіру; ерікті биттік деректерді беруге мүмкіндік беретін тақтаны сымдау икемділігі және т. б.

ПЛИС-та шифрлеу алгоритмдерінің іске асырылуына шолу

ПЛИС-та ең жиі іске асырылатын шифрлеу алгоритмдері RSA, AES, DES, ГОСТ 28147-89 алгоритмдері болып табылады. Бұл алгоритмдерді енгізу бойынша көптеген зерттеулердің негізгі мақсаты кристалдық ресурстарды тиімді пайдалануға және деректерді шифрлеудың ең жоғары жылдамдығына қол жеткізу болып табылады. 1.1-кестеде алгоритмдердің ерекшеліктерін салыстыру нәтижелері келтірілген: кіріс деректерінің ұзындығы, кілттердің өлшемі және қолданылатын математикалық операциялар. Асимметриялық RSA шифрлеу алгоритмі деректерді шифрлеу үшін де, электрондық цифрлық қолтаңбаны құру үшін де қолданылады [33]. Бұл алгоритмнің қауіпсіздігі үлкен бүтін сандарды факторингке бөлудің қиындығына негізделген. Ол көптеген криптографиялық қосымшаларда, соның ішінде Pretty Good Privacy (PGP) шифрлеу жүйесінде қолданылады, сонымен қатар қарастырылып отырған IDEA (ағылш. International data Encryption Algorithm, деректерді шифрлеудың халықаралық алгоритмі) алгоритмі де қолданылады [34]. RSA алгоритмі ПЛИС-та басқа асимметриялық шифрлеу алгоритмдеріне қарағанда әртүрлі мақсаттар үшін әртүрлі интерпретацияларда жиірек жүзеге асырылады [35]. Бұл іске асырулардың негізгі ерекшелігі аппараттық ресурстардың құнын төмендету және ПЛИС іске

асыруда қажетті өнімділікке жету үшін Монтгомери алгоритмін қолдану болып табылады [36].

АҚШ-тағы DES симметриялы шифрлеу алгоритмі 1977-1980 жылдар аралығында Ұлттық стандарт болды. Қазіргі уақытта ол ескірген болып саналады және көбінесе оның шифрлеуға төзімді түрі қолданылады (3DES, DESX) [37]. DES ПЛИС-та комбинациялық логикалық схема ретінде құрылуы мүмкін. Мұндай іске асырудың артықшылықтарына қосымша басқару машинасының болмауы, интерфейстің қарапайымдылығы, ПЛИС-та сақталған кілттің қауіпсіздігі жатады. Кемшіліктерге: кристалдағы үлкен аумақ және мүмкін конвейерлеудің болмауы [38].

Кесте 1.1 – ПЛИС-та шифрлеу алгоритмдерін іске асырудың ерекшеліктеріне талдау

Алгоритмнің сипаттамалары	Алгоритм				
	AES (ауыстыру-орналастыру желісі)	DES (Фейстель желісі)	ГОСТ 28147-89 (Фейстель желісі)	IDEA (Фейстель желісі модификациясы)	RSA
Шифрлеу түрі	Симметриялық, блокты				Асимметриялық
Блоктың ұзындығы, бит	128	64	64	64	
Раундтар саны	10	16	38	8	
Кілт өлшемі, бит	256	56	256	128	1024
Негізгі операциялар	Кестелік байтты ауыстыру, циклдік ауысу, бағанды араластыру, 2-модульді разрядты қосу	Ауыстыру, орын ауыстыру, разрядты алып тастау НЕМЕСЕ	2^{32} модуль бойынша қосу, ауыстыру, биттік алып тастау НЕМЕСЕ, циклдік ауысым	$2^{16} + 1$ Модулі бойынша көбейту, 2^{16} модулі бойынша қосу, биттік алып тастау НЕМЕСЕ	Үлкен модуль бойынша дәрежеге көтеру операциясы

ПЛИС-те іске асыруды қарастыратын басылымдардың мазмұнын талдау	Параллель раундтық орналасуда және бір модульде шифрлеу мен шифрды шешуде алгоритмнің тиімді орындалуына шолу. Алынған жиілікті, шифрлеу жылдамдығын және логикалық элементтердің санын талдау [8].	DES шифрлеу алгоритмін аппаратты қ іске асыруға шолу, оның мақсаты шифрлеу жылдамдығын кем дегенде алу болды қазіргі заманғы криптопроцессорлар. ПЛИС-те берілген іске асырудың артықшылықтары мен кемшіліктерін талдау [9].	GPU және ПЛИС құрылғыларында зарарсыздандыру алгоритмінің өнімділігін бағалау. Алдын ала өнімділікті болжау мен эксперимент нәтижелерін салыстыру [10].	Мысал ретінде IDEA көмегімен DSP, ПЛИС және ASIC көмегімен ұялы байланыстарды шифрлеудыжүзеге асыру мүмкіндіктерін зерттеу. Шифрлеу жылдамдығы мен өнімділігі бойынша алынған мәліметтерді салыстыру. Мұндай іске асыру үшін ПЛИС артықшылықтарының сипаттамасы [11].	Әр түрлі модульдердегі әр түрлі кілт ұзындықтары бар алгоритмнің жұмысын әр түрлі есептеу бағыты мен қосқыштар санымен талдау. Алынған шифрлеу жылдамдықтарын, жиіліктерін және логикалық элементтер санын салыстыру [6].
Кристалл ПЛИС	Xilinx Virtex 5	Xilinx Virtex – E	Altera Arria II GX EP2AGX125	Xilinx XC4000	Altera Cyclone IV EP4CE115F29C7
Жиілік	347,6 МГц	19,4 МГц	125 МГц	33 МГц	13,31 МГц
Жылдамдық	44,5 Гб/с	4,25 Гб/с	748 Мб/с	528 Мб/с	12,81 Кб/с

AES – АҚШ шифрлеу стандарты ретінде қабылданған ең танымал симметриялы блоктық шифрлеу алгоритмдерінің бірі [39]. Алгоритмді түсіну оңай және бағдарламалау тілдерінде жазудың жылдамдылығымен және жоғары шифрлеу жылдамдығымен танымал [40].

ПЛИС енгізілген қауіпсіздік механизмдерін іске асыруға арналған икемді және жоғары конфигурацияланатын платформа болғандықтан, AES алгоритмі қауіпсіз жұмысты қамтамасыз ету үшін чиптегі жүйелерде (SoC) қолданылады. Бұл жоғары есептеу күрделілігінің алгоритмі болғандықтан, оны ПЛИС-та іске асыруға арналған жұмыстарда SoC дизайнының әртүрлі бағдарламалық және аппараттық шешімдері сипатталған [41].

Бөлім бойынша қорытынды

Ақпараттық қауіпсіздік қазіргі әлемде өзекті және маңызды болып келеді, мұнда деректер барлық қызмет салаларында шешуші рөл атқарады. Деректердің құпиялылығын, тұтастығын және қол жетімділігін қорғау бірінші кезектегі міндетке айналды және криптография осы қорғауды қамтамасыз ететін негізгі ғылым болып табылады.

Бұл бөлімде ақпараттық-коммуникациялық технологиялар контекстінде ПЛИС-ті қолдана отырып, шифрлеу алгоритмдерін аппараттық іске асырудың рөлі қарастырылды. ПЛИС – бұл жоғары өнімділікті, икемділікті және криптографиялық алгоритмдерді аппараттық жолмен жүзеге асыру мүмкіндігін біріктіретін қуатты деректер қауіпсіздігі құралы.

Тұтастай алғанда, ПЛИС-тің көмегімен шифрлеу алгоритмдерін аппараттық жолмен енгізу деректердің қауіпсіздігін қамтамасыз етудің перспективалы тәсілі болып табылады және бұл зерттеу бағыты тез өзгеретін киберқауіпсіздік жағдайында өзекті болып қала береді. Мұндай жүйелерді енгізу және жетілдіру банк саласынан бастап күнделікті өмірде ендірілген құрылғыларға дейінгі әртүрлі салалардағы ақпаратты қорғау деңгейін едәуір арттыра алады.

2 КӨПМҮШЕЛІК КОЭФФИЦИЕНТТЕРГЕ НЕГІЗДЕЛГЕН СИММЕТРИЯЛЫ БЛОКТЫҚ ШИФРЛЕУ АЛГОРИТМІ

2.1 Ауыстыру–алмастыру жүйесі негізінде симметриялы блоктық шифрлеу алгоритмі

Ақпараттық және телекоммуникациялық жүйелер әлемінде симметриялық блоктық шифрлеу деректердің құпиялылығы мен тұтастығын қамтамасыз етуде басты орын алады. Шифрлеудың бұл әдісі бүгінде ақпаратты қорғаудың және әртүрлі қызмет салаларында, соның ішінде қаржы, әскери және мемлекеттік ұйымдар мен күнделікті өмірде қауіпсіздікті қамтамасыз етудің маңызды элементі болып табылады.

Симметриялық шифрлеудың [42] негізгі ерекшеліктерінің бірі – оның жоғары өнімділігі. Кілттер жұбы қолданылатын асимметриялық шифрлеудан айырмашылығы (жалпыға ортақ және жеке), симметриялық шифрлеу деректерді шифрлеу және шифрын ашу үшін бірдей кілтті пайдаланады. Бұл өнімділік маңызды рөл атқаратын қолданбалар үшін симметриялық шифрлеуды таңдаулы таңдауға айналдыратын операциялардың жоғары жылдамдығын қамтамасыз етеді.

Заманауи симметриялы блоктық шифрлар көбінесе шифрлеудың екі негізгі тәсілін қолданады: Фейстель сызықтары және SP желілері (ауыстыру желілері). Екі жағдайда да ашық мәтіннің қайтымды түрлендірулері қолданылады, бұл дұрыс кілтті пайдаланып бастапқы деректерді қалпына келтіруге мүмкіндік береді. Бұл тәсіл шифрдың құпиялылығы шифрлеу алгоритмінің құпиялылығына емес, тек кілттің құпиялылығына байланысты болуы керек деген Керхгоф [43], принципіне сәйкес келеді.

Дегенмен, симметриялық блокты шифрлеудың көптеген артықшылықтарына қарамастан, кілттерді беру механизмі осы жүйенің әлсіз жақтарының бірі болып қала беретінін атап өткен жөн.

Шифрлеу алгоритмдері қолдануларына байланысты әр елде әртүрлі болып табылады. Мысалы:

– АҚШ:

- AES (Advanced Encryption Standard) – симметриялық блоктық шифрлеу алгоритмі. Бұл шифрлеу алгоритмі әлемдегі ең кең таралған. Ол деректерді қорғау және құпиялылықты қамтамасыз ету үшін қолданылады. AES 2001 жылы сипатталған және сенімді және қауіпсіз болып саналады.
- RSA – бұл асимметриялық алгоритм шифрлеу және цифрлық қолтаңба үшін пайдаланылады. Ол үлкен сандарды факторингке бөлудің математикалық принциптеріне негізделген.

– Ресей:

- ГОСТ 28147-89 (Калинка) – бұл симметриялық блоктық шифрлеу алгоритмі КСРО-да жасалған. Ол ресейлік ақпараттық қауіпсіздік жүйелерінде қолданылады.

– Қытай:

- SM2 – бұл асимметриялық шифрлеу алгоритмі Қытайда жасалған. Ол сандық қол қою және кілт алмасу үшін қолданылады.
- SM4 Қытайда да жасалған симметриялық блок алгоритмі.

– Еуропалық Одақ:

- ECRYPT-CSA – бұл деректерді қорғау үшін Еуропалық комиссия ұсынған алгоритмдер тобы.

– Канада:

- SKIPJACK АҚШ-та жасалған симметриялы алгоритм, бірақ, Канадада да қолданылады.

– Австралия:

- AES Австралияда да кеңінен қолданылады.

Атап өткендей шифрлеу алгоритмдері қолдануларына байланысты әр елде әртүрлі болып келеді. Сондай-ақ, Қазақстанда ҚР БҒМ ҰҚК АЕТИ-да позициялық емес көпмүшені санау жүйесі (ПЕКСЖ) негізінде шифрлеу және электрондық цифрлық қолтаңба алгоритмдерін зерттеу және әзірлеу жұмыстары жүргізілді [44,45]. Бұл зерттеулердің мәні модульдік санау жүйелерін қолданатын жаңа блоктық симметриялық шифрлеу алгоритмдері әзірленіп, талданған және бағдарламалық қамтамасыз етуге енгізілген. Жұмыс отандық ақпараттық қауіпсіздік жүйелерін дамытуға және оларды практикалық қолдану мақсатында бағдарламалық-аппараттық жүйелерді құруға бағытталған.

Қазақстанда ақпараттық технологияларды және ақпаратты криптографиялық қорғау құралдарын қолдану туралы бірқатар заңдар қабылданды [46-49]. Қазақстан Республикасы Қауіпсіздік Кеңесі қабылдаған Ақпараттық қауіпсіздік тұжырымдамасы, отандық ақпараттық қауіпсіздік жүйесін құру, оның әртүрлі деңгейдегі құпиялылығы үшін ақпаратты қорғаудың әдістері, модельдері мен алгоритмдерін, содан кейін оларды аппараттық және бағдарламалық қамтамасыз етуді құру қажеттілігін көрсетеді.

Бұл саладағы зерттеулер бұрынғы Кеңес өкіметінің ғалымдары И.Я.Акушский, Д.И.Юдицкий және В.М.Амербаевтардың зерттеу еңбектері арқылы кең тарады [50, 51]. Модульдік арифметиканы одан әрі дамыту бағыттарының бірі, отандық ғалым Р.Г.Бияшевтің қателіктерді табу және түзетуге, сонымен бірге, өзін-өзі түзететін кодтар жасау үшін қолданылатын позициялық емес көпмүшелік санау жүйелерін құру және қолдану бойынша еңбегі бар [52]. Ол симметриялы блоктық шифрлеу жүйесін құруда ПЕКСЖ алгебрасының негізгі ережелерін негіздеді. Бұл жүйені әр түрлі қосымшаларда іс жүзінде қолдану үшін оны іске асыруға арналған ұсыныстар жиынтығын әзірлеу қажет. Мұндай ұсыныстарды алу бірнеше бағытта зерттеуді қажет етеді. Оның бірі, позициялық емес жүйелерді симметриялы блоктық шифрлеу жүйесінде практикада қолдану мақсатында модельдеу.

Позициялық емес көпмүше санау жүйесін құру

ПЕКСЖ – бұл көпмүшелерді қолдана отырып, сандарды бейнелеудің икемді және тиімді әдісін ұсынатын математикалық құрылым. Бұл жүйенің дәстүрлі позициялық санақ жүйесіне (ПСЖ) қарағанда бірқатар

артықшылықтары бар және әртүрлі криптографиялық қосымшаларда кеңінен қолданылады.

Көпмүшелерді көбейту алгоритмдері көптеген криптографиялық жүйелердің маңызды құрамдас бөлігі болып табылады. Бұл жүйелердің өнімділігі үшін тиімді көпмүше көбейту алгоритмдері қажет, өйткені олар деректерді жылдам әрі қауіпсіз шифрлеуды және шифрды ашуды қамтамасыз етеді. ПЕКСЖ контекстінде көпмүшелерді көбейтудің тиімді алгоритмдерін әзірлеу қазіргі криптографияны дамыту үшін өте маңызды.

ПСЖ негізіндегі шифрлеуды жүзеге асырудың бірінші қадамы ПЕКСЖ құру болып табылады.

Басында дәрежесі N-ден жоғары емес барлық келтірілмейтін көпмүшелердің жиынынан жұмыс негіздері таңдалады

$$p_1(x), p_2(x), \dots, p_s(x). \tag{2.1}$$

Санау жүйесіндегі барлық таңдалған негіздер бір-бірінен өзгеше болуы керек, яғни олар осы санау жүйесі үшін бірегей болуы керек.

Олардың дәрежелерін сәйкесінше m_1, m_2, . . . , m_S деп белгілейік. (2.1) көпмүшелері олардың орналасу ретін ескере отырып, бір негіздер жүйесін құрайды. ПЕКСЖ-нің негізгі жұмыс ауқымы – бұл дәрежесі $\sum_{i=1}^{S} m_i$ болатын $p^m(x) = p_1(x), p_2(x), \dots p_s(x)$ көпмүшелігі.

ПЕКСЖ-да дәрежесі m-нен кіші болатын кез-келген $F(x)$ көпмүшелігі, оны (2.1) жұмыс негіздері бойынша қалдықтар тізбегі түрінде жалғыз позициялық емес түрге ие:

$$F(x) = \big(\alpha_1(x), \alpha_2(x), \dots, \alpha_3(x)\big). \tag{2.2}$$

мұнда $F(x) = \big(\alpha_i(x)(mod p_i(x)\big), i = \overline{1,S}$. $F(x) -$ (2.2) түрдегі позициялық емес көрінісі арқылы бастапқы қалпына келтіруге болады [53;57;58].

$F(x) = \sum_{i=1}^{S} a_i\,(x) B_i(x)$, мұндағы:

$$B_i(x)\frac{p^m(x)}{p_i(x)} M_i(x) = 1\big(mod p_i(x)\big). \tag{2.3}$$

$M_i(x)$ көпмүшеліктері (2.3) салыстырымы орындалатындай таңдап алынады. Тек ақпаратты беру және сақтау жағдайында $F(x)$ көпмүшесін позициялық қалпына келтіру, келесі формула бойынша жүзеге асырылады:

$F(x) = \sum_{i=1}^{S} a_i\,(x) B_i(x)$, мұндағы:

$$P_i(x)\frac{p^m(x)}{p_i(x)}. \tag{2.4}$$

Әрбір жұмыс негіздерінің дәрежесі L - дің мәнінен аспайтын болуы керек (біздің жағдайда 128, 192 және 256). Жұмыс негіздері ретінде (2.4) теңдігінің

шарты орындалатындай дәрежелері m_1-ден m_S -ке дейінгі барлық келтірілмейтін көпмүшеліктер ішінен таңдалына алады:

$$k_1 m_1 + k_2 m_2 + \cdots + k_s m_s = L. \quad (2.5)$$

(2.5) теңдеуде $0 < k_i < n_i$, $i = 1, \ldots , S$ – белгісіз коэффициенттер және дәрежесі m_i болатын таңдалған келтірілмейтін көпмүшеліктердің саны, осы коэффициенттердің бір жиынтығы (2.5)-тің шешімдерінің бірі болып табылады және жұмыс негіздерінің бір жүйесін береді, ал n_i дәрежесі m_i болатын барлық келтірілмейтін көпмүшелердің саны, $1 \leq m_i \leq L$, $S = k_1 + k_2 + \cdots + k_S$ таңдалған жұмыс негіздерінің саны. m_i дәрежелі көпмүшеліктер модульдері бойынша толық қалдықтар жүйелеріне, оларды жазу үшін m_i қажет болатын, яғни дәрежесі $m_i - 1$-ден артпайтын барлық көпмүшелері кіреді.

Шифрлеу. Деректерді шифрлеуда жылдамырақ жұмыс істеу үшін, ұзындығы L биттен тұратын пайдаланылып отырған кілт тізбегін келтірілмейтін көпмүшеліктердің (жұмыс негіздерінің) дәрежесіне қарай k_1 (x), k_2 (x), . . . , k_S (x) түріндегі қалдықтарының тізбегі ретінде кесіп аламыз.

$$K(x) = \big(k_1(x), k_2(x), \ldots k_s(x)\big). \quad (2.6)$$

Онда ω_1 (x), ω_2 (x), . . . , ω_S (x) криптограммасы ретінде $H(F(x),K(x))$ шифрлеу функциясының мәнін қарастыруға болады:

$$H(x) = \big(\omega_1(x), \omega_2(x), \ldots, \omega_s(x)\big). \quad (2.7)$$

$F(x)$, $K(x)$, $H(x)$ функциялары ПЕКСЖ-де орындалатын операцияларға сәйкес, жұмыс негіздері ретінде таңдалған (2.1) көпмүшелердің модулі бойынша параллель орындалады. Егер $H(F(x),K(x))$ функциясы ретінде көбейту операциясы қолданылатын болса, онда шифрдің ω_1 (x), ω_2 (x), . . . , ω_S (x) элементтерін алу үшін α_i $(x) \cdot k_i$ (x) көбейтіндісін сәйкес p_i (x) жұмыс негіздеріне бөлгендегі қалдықтар алынады [54; 55]:

$$\alpha_i(x) k_i(x) = \omega_i(x), i = 1, \ldots, S. \quad (2.8)$$

Шифрді кері ашу. $H(x)$ криптограммасын белгілі $K(x)$ кілті бойынша кері ашу үшін k_i (x)-дің әрбір мәніне сәйкес келесі салыстырылым шарты орындалатындай, кері $k_i^{-1}(x)$ көпмүшеліктерін есептеулер жүргізу арқылы табамыз:

$$k(x) k_i^{-1}(x) = 1, i = 1, \ldots, S. \quad (2.9)$$

Нәтижесінде, $K(x)$ көпмүшелігінің керісі болып табылатын $K^{-1}(x) = k_1{}^{-1}(x)$, $k_2{}^{-1}(x)$, … , $k_s{}^{-1}(x)$ көпмүшелігін аламыз. Онда (2.2) қалдықтар тізбегінің элементтерін келесідей формуламен қалыпқа келтіруге (табуға) болады:

$$\alpha_i(x) = k_1^{-1}(x)\omega_i(x)(mod p_i(x), i = 1, \dots, S. \qquad (2.10)$$

Осылайша, берілген L ұзындықты электрондық хабарламаларды шифрлеу алгоритмінің қарастырып отырған ПЕКСЖ-ға негізделген модификациясында толық кілт ретінде $p_1(x)$, $p_2(x)$, . . . , $p_S(x)$ көпмүшелік жұмыс негіздері жүйесі $K(x)=(k_1(x), k_2(x), \dots, k_S(x))$ кілті және шифрді ашу үшін қажет оған кері $K^{-1}(x) =(k_1{}^{-1}, k_2{}^{-1}(x), \dots, k_s{}^{-1}(x)$ кілттері болып табылады.

Позициялық емес криптографиялық жүйелерде криптографиялық беріктік критерийі ретінде құпия кілттердің криптографиялық күші және толық құпия кілтпен сипатталатын таңдалған негіздер жүйесі қолданылады [56-58]. Екілік коэффициенттері бар келтірілмейтін көпмүшелердің реті өскен сайын олардың саны тез өседі.

2.2 Келтірілмейтін көпмүшелер модулі бойынша көпмүшелік көбейткіштерді құру

Айнымалылардың бүтін мәндері машиналық сөздің аппараттық қолдау ұзындығымен анықталған типтік есептеу құрылғыларының максималды диапазонынан едәуір асатын есептеулерге әкелетін есептеулер бар [59, 60]. Аппараттық құралдарды енгізу өңдеу жылдамдығы мен мүмкіндіктері тұрғысынан тиімді болып саналады, мұндай мәселелерді дәстүрлі тәсілдермен шешу мүмкін емес [61, 62]. Мысалы, ECC немесе RSA криптожүйелеріне келетін болсақ, криптографиялық түрлендірулердің негізгі қиындығы, ең алдымен, көп таңбалы сандарға сериялық модульдік көбейтуді орындау қажеттілігімен байланысты [63]. Криптожүйені осындай іске асыруда тиімді модульдік көбейтуді қамтамасыз ету маңызды міндет болып табылады [64].

Қазіргі заманғы есептеу техникасының көп бөлігі санмен жұмыс істейтінін ескермеуге болмайды. Дәстүрлі позициялық жүйеде ұсынылған көп таңбалы арифметикалық амалдар үшін есептеу жылдамдығын едәуір баяулататын және калькулятор құрылымын қиындататын разрядаралық ауысуларды ескеру қажет. Осыған байланысты есептеу құрылғыларының өнімділігін арттырудың жаңа жолдарын іздеуге арналған тиісті зерттеулер өзекті болып табылады. Сандық ақпаратты кодтаудың дәстүрлі емес әдістерін және компьютерлік арифметиканың сәйкес параллель нұсқаларын қолдануға бағытталған зерттеулердің маңызы зор.

Санды белгілеудегі әрбір сандық белгінің (цифрдың) мәні оның орнына немесе дәстүрлі санау жүйесіндегі цифрға байланысты. Сонымен қатар, "негізсіз санау жүйелері" деп аталатындар да бар, олардың бірі "қалдық санау жүйесі" (ҚСЖ) [65]. ҚСЖ қосымшасы – үлкен деректерді есептеудің тиімді әдісі. Атап айтқанда, ҚСЖ қосымшасы қосу кезінде тасымалдаудың болмауы, үлкен кіріс

блогын кішігірім қосалқы блоктарға бөлу және параллельді өңдеу арқылы жұмыс жылдамдығын арттыруға мүмкіндік береді.

Қалдық сандар жүйесі – бұл есептеу арифметикасындағы деректерді ұсыну жүйесі, мұнда бүтін сан кіші сандар жиынтығымен белгіленеді.

$$A = (\alpha_1, \alpha_2, \dots, \alpha_n). \qquad (2.11)$$

оны жүйенің негіздері деп аталатын берілген $p_1, p_2, \dots, p_n$, натурал сандарына бөлуден. α_1, сандары келесідей құрастырылады:

$$\alpha_i = A - \left[\frac{A}{p_i}\right] p_i, \ i = \underline{1, n} \ , \qquad (2.12)$$

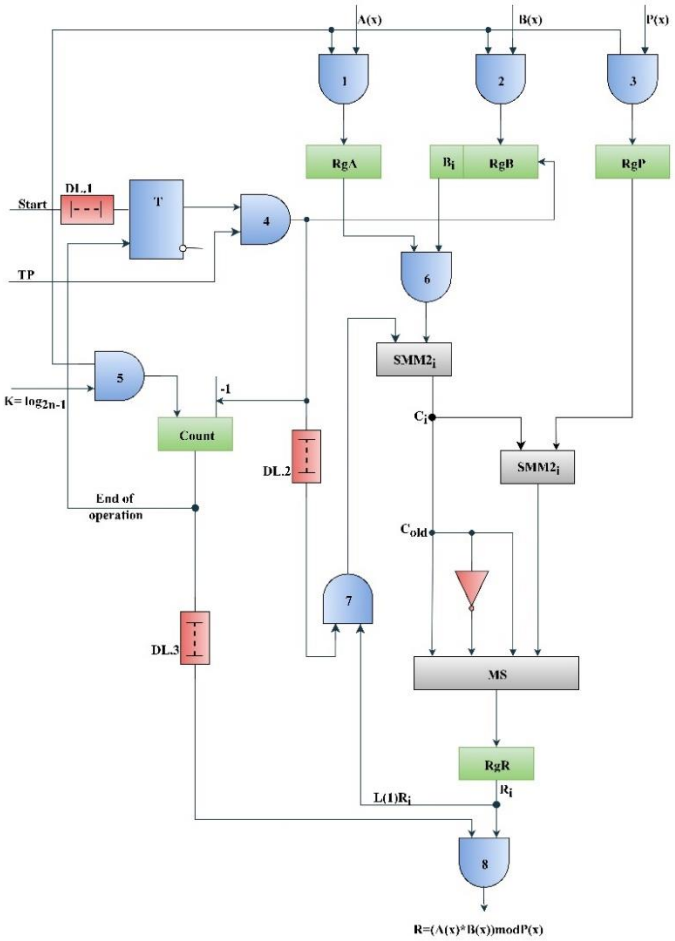
мұндағы $[A/p_i]$ A-ны α_i-ке бөлудің бүтін бөлігін білдіреді. (2.3)-тен A санының i -ші разрядының α_i санының A -ны p_i-ға және α_i<p_i-ге бөлгеннен кейінгі ең кіші оң қалдық екендігі шығады. Қытайлық қалдық теоремасына сәйкес, A санының (2.11) түрінде ұсынылуы, егер p_i сандары бір-бірінен екі рет жай болса, бірегей болады. Бұл жағдайда ұсынылатын сандар диапазонының көлемі $p = p_1, p_2, \dots, p_n$. Мұнда позициялық санау жүйесіне ұқсас, ұсынылатын сандардың диапазоны негіздердің көбейтіндісі ретінде өседі, ал сандардың сыйымдылығы сол негіздердің сыйымдылықтарының қосындысы ретінде өседі.

Есептеу техникасының кейбір салаларында модульдік арифметиканы тиімді пайдаланудың негізгі артықшылықтары: сандық жүйе деңгейіндегі табиғи параллелизмнің жоғары деңгейі, бұл қосу және көбейту кезінде цифрлардың берілмеуімен және қателіктердің таралуының болмауымен байланысты. Разрядты санау жүйесінен айырмашылығы, вектордың барлық элементтері эквивалентті және олардың біреуіндегі қате тек динамикалық диапазонның төмендеуіне әкеледі. Бұл факт апатқа қарсы және қателерді түзететін құрылғыларды жасауға мүмкіндік береді [66].

Бұл ерекшеліктер жақсы артықшылықтар береді ҚСЖ қосу, азайту және көбейтудің модульдік операцияларында негізгі санау жүйесінен. Бұл әсіресе көп таңбалы сандар операндтар ретінде әрекет етсе дұрыс. Есептеу техникасында ҚСЖ қолданудың стратегиялық бағыты ақпаратты қорғаудың криптографиялық құралдарын жасау болып табылады. Р.Г. Бияшев жетекшілік ететін зерттеу ұжымы позициялық емес көпмүшені санау жүйесі (ПЕКСЖ) негізінде цифрлық қолтаңбаны шифрлеу мен қалыптастырудың модульдік алгоритмдерін ұсынды [67-69].

Ақпараттық-коммуникациялық жүйелерде және әртүрлі мақсаттағы желілерде позициялық емес көпмүше санау жүйелері негізінде жасалған блоктық симметриялық шифрлеу алгоритмдері осы тәсілге негізделген және позициялық емес шифрлеу алгоритмін іс жүзінде қолдану мүмкіндігін талдайтын зерттеулердің нәтижесі болып табылады [70]. Сондай-ақ, бұл бағытта параллельді есептеулерге арналған маңызды жұмыстар [71-72], сондай-ақ мақалалар бар [73-74].

Қарастырылып отырған көпмүшелердің көбейткішінің функционалдық схемасы 2.1-суретте көрсетілген.



Сурет 2.1 – Көпмүшелердің көбейткішінің функционалдық схемасы модуль бойынша келтірілмейтін көпмүшелер [75]

Келтірілмейтін көпмүше модулі бойынша көпмүшелерді көбейту екі жолмен жүзеге асырылады.

Бірінші әдісте көбейту екі кезеңде жүзеге асырылады. Бірінші кезеңде екілік коэффициенттері бар көпмүшелердің көбейтіндісі екі модуль бойынша қосындылардағы жарты көбейтінділердің матрицалық жолдарын қосу арқылы есептеледі, ал келтірілмейтін көпмүшенің модулі бойынша келтірілуі жартылай қалдық қалыптастырғыштарда (ЖҚҚ) орындалады.

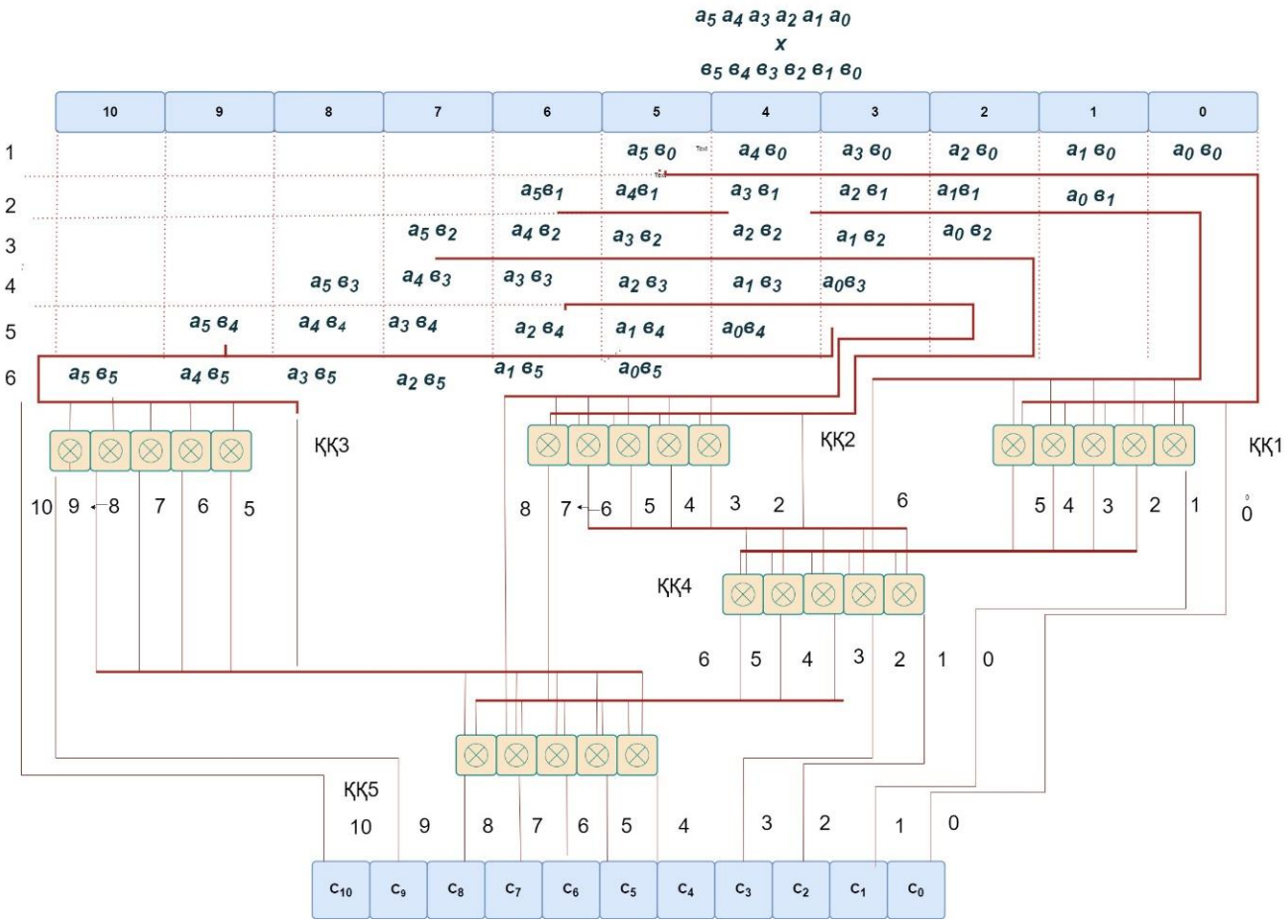
Екінші әдісте көпмүшелерді төмендетілмейтін көпмүшенің модулі бойынша көбейту құрылғысы, ол алдыңғы $r_i - 1$ қалдығын солға қарай бір разрядқа жылжыту арқылы r_i жарты қалдығын кезең-кезеңімен қалыптастыруға негізделген, оны төмендетілмейтін көпмүшенің модулі бойынша келтіріп, одан әрі модуль бойынша r_i жартылай қалдықтарын қосады, олар үшін мультипликатор көпмүшесінің сәйкес екілік коэффициенттері бір мәнді алады.

Келтірілмейтін көпмүшенің модулі бойынша көпмүшелерді көбейту $R = [A(x) * B(x)]\ mod\ P(x)$, формуласы бойынша орындалады, мұнда $A(x)$ –

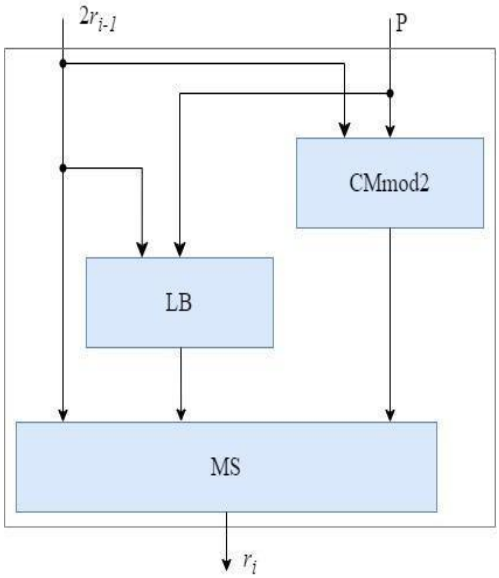
көпмүше, $B(x)$ – көпмүше-мультипликатор, $P(x)$ – модуль көпмүшесі және $A(x) \leq P(x)$ және $B(x) \leq P(x)$.

Қарастырылып отырған құрылғыларда көбейткіш көпмүшенің екілік коэффициенттерінің N-разрядты сыйымдылығы $B(x)$.

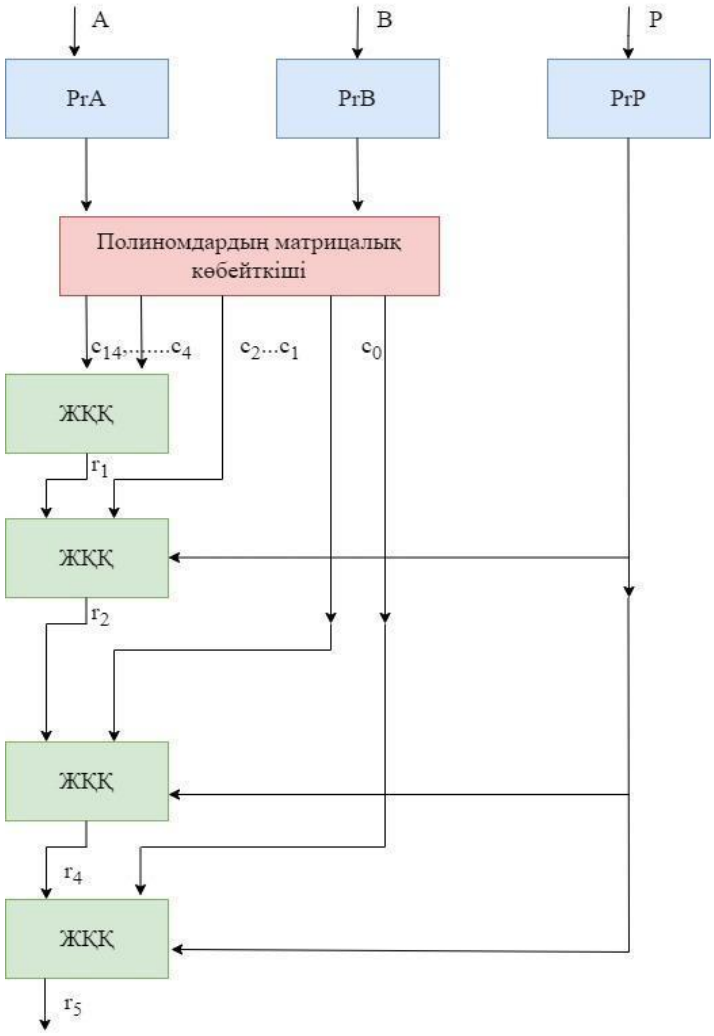
Сыйымдылығы N = 6 көпмүшедерінің жартылай көбейтінділерінің матрицасы 2.2-суретте көрсетілген, мұнда осы матрицаның екі модулінің әрбір бағанын қосу арқылы C(x) көбейтіндісінің 2_{N-1} разрядты екілік бейнесін аламыз. Екінші кезеңде C(x) өнімі ЖҚҚ көмегімен P(x) модулі бойынша төмендетіледі, оның құрылымы 2.3-суретте көрсетілген. Бұл суреттен көрініп тұрғандай, ЖҚҚ модулі екі қосқыштан (ҚҚmod2), логикалық блоктан (ЛБ) және мультиплексордан (MS) тұрады. Алдыңғы қалдық $2r_{i-1}$екі еселенген мәні CMmod2 бірінші кірісіне және MS кірісіне қолданылады. P(x) модулінің екілік коэффициенттері қосқыштың екінші кірісіне беріледі. Логикалық блогы (ЛБ) MS $2r_{i-1}$шығысын басқарады, егер $2r_{i-1} < P(x)$, немесе CMmod2 шығысынан екі қосу $2r_{i-1} \oplus P(x)$ модулінің нәтижесі.



Сурет 2.2 – A(x) және B(x) көпмүшелерының матрицалық көбейткіші



Сурет 2.3 – ЖҚҚ структурасы



Сурет 2.4 – Келтірілмейтін көпмүшеның қарастырылатын матрицалық көбейткіш модулінің схемасы

2.4-суретте қарастырылатын матрицалық көбейткіш модулінің келтірілмейтін көпмүшесінің схемасы көрсетілген.

Бірінші әдіс бойынша құрастырылған модульдік көпмүше көбейткіштердің кемшілігі, біріншіден, олардың орындалуының күрделілігі, екіншіден, C(x) қалыптастыру үшін белгілі бір уақыт қажет.

Әрі қарай, жартылай қалдықты кезең-кезеңмен құруға негізделген көпмүшелердің модульдік көбейткіштерін қарастырамыз.

2.3 Құрылған симметриялық блоктық шифрлеу алгоритмінің сенімділігін зерттеу

Шифрлеу алгоритмдерін сәтті жүзеге асыру үшін олардың криптографиялық беріктігін қамтамасыз ету қажет. Шифрлық беріктік – деректерді шифрлеу алгоритмнің шабуылдардың әртүрлі түрлеріне төтеп беру қабілеті, тіпті шабуылдаушылардың белсенді араласуымен де деректердің құпиялығын, тұтастығын және қолжетімділігін қамтамасыз етеді. Бұл тарауда шифрлардың беріктігін бағалау үшін қолданылатын негізгі ұғымдар мен әдістер қарастырылды.

Ақпараттық қауіпсіздік стандарттары мен әдістемелерін әзірлеуде NIST және ENISA (Еуропалық киберқауіпсіздік агенттігі) сияқты ұйымдар маңызды рөл атқарады [76].

Ақпараттық технологиялар қауіпсіздігін бағалау үшін NIST әзірлеген бірнеше сынақ түрлерін ұсынады:

- Frequency Test (Monobit) – бірліктер мен нөлдердің тізбек бойынша біркелкі таралуын тексереді;
- Run Test – – үзіліссіз немесе нөлдердің тізбектерінің (жұмыстарының) санын және олардың ұзындығын талдайды;
- Longest Run of Ones – ұсынылған үлгідегі ең ұзын бірліктер тізбегінің ұзындығын бағалайды;
- Binary Matrix Rank - олардың сызықтық тәуелділігін анықтау үшін бір және нөл тізбегінен құрылған кездейсоқ матрицалардың рангін өлшейді;
- Discrete Fourier Transform – тізбектің кезеңділігін талдау үшін дискретті Фурье түрлендіруін қолданады;
- FIPS 140-2/140-3(Federal Information Processing Standards) – ақпаратты өңдеудің федералдық стандарттары.

Осының ішінде FIPS 140-2/140-3 стандарттары АҚШ мемлекеттік мекемелерінде қолданылатын деректерлі шифрлеу модульдеріне қойылатын талаптарды белгілейді, ал халықаралық стандарт Common Criteria (ISO/IEC 15408) негізді қамтамасыз етеді. FIPS 140-3 бағдарламалық құралды енгізумен салыстырғанда жылдамдық пен қауіпсіздікті жақсарту үшін аппараттық құралда шифрлық алгоритмдерді енгізудің маңыздылығын атап көрсетеді [77]. Бұл стандарттар қорғаныстың төрт деңгейіне бағытталған, олардың әрқайсысы әртүрлі пайдалану жағдайлары мен қауіп деңгейлеріне арналған:

- 1-деңгей қауіпсіздіктің ең төменгі деңгейін қамтамасыз етеді және қауіптілігі төмен қолданбаларда пайдалануға арналған;

- 2-деңгей қауіпсіздіктің қалыпты деңгейін қамтамасыз ететін криптографиялық модуль үшін қауіпсіздік талаптарын сипаттайды;

- 3-деңгей криптографиялық модульдерге және құпия ақпаратқа рұқсатсыз кіруден қорғайды;

- 4-деңгей стандартта анықталған қауіпсіздіктің ең жоғары деңгейі болып табылады және қауіпсіздік қатесінің салдары ауыр болатын қолданбаларға арналған.

FIPS 140-2 және FIPS 140-3 стандарттары деректерлі шифрлеу модульдерінің шифрлық беріктігін тексеру үшін бірқатар сынақтарды қамтамасыз етеді. Бұл сынақтарға мыналар жатады:

- Криптографиялық алгоритмдерге сәйкестік сынақтары: алгоритмдердің белгіленген стандарттарға сәйкестігін тексеру;
- Физикалық қауіпсіздік сынақтары: модульдердің физикалық кедергілерге төзімділігін бағалау;
- Кездейсоқтық тестілері: болжамдылық пен таралу біркелкілігі үшін кездейсоқ сандар генераторларын сынау;
- Электромагниттік эмиссиялар/соққы сынақтары: құрылғының электромагниттік эмиссияларды талдау сияқты бүйірлік арна шабуылдарына осалдығын бағалау.

FIPS стандарттарының осы ауқымды және қатаң талаптары осы стандарттарды пайдаланатын құрылғылар мен бағдарламалық қамтамасыз етудің жоғары қауіпсіздікті және бүгінгі киберқауіпсіздік қатерлеріне төтеп бере алатынын қамтамасыз етеді.

Деректерді шифрлеу беріктігін бағалау үшін әртүрлі талдау әдістері қолданылады:

1. Әзірленген шифрлеу алгоритмінің лавиндік әсері

Лавиндік әсер ұғымы криптографияда блоктық шифрлар мен шифрлық хэш функцияларды зерттеуде қолданылады және маңыздылығы жағынан басқа да жүргізілетін талдаулардан кем түспейді [78]. Егер алгоритмде қажетті деңгейде лавиндік әсері болмаса, онда криптоталдаушылар алгоритмге кіретін ашық деректерге қарап шифрлеу нәтижесінде шығатын мәліметтер туралы болжам немесе қорытынды шығара алады. Деректерді шифрлеудағы лавин эффектісі кіріс деректеріндегі аздаған өзгерістер шығыс деректерде үлкен өзгерістерге әкелетін құбылысты білдіреді. Бұл әсер шифрлеу алгоритмдерінің криптографиялық күшін бағалауда маңызды рөл атқарады.

2. Шифрлеу алгоритміне дифференциалдық криптоталдау әдісін қолдану

Дифференциалды криптоталдау және сызықтық криптоталдау шифрлеу алгоритмдердің қауіпсіздігін бағалау үшін қолданылатын криптографиялық талдау әдістері болып табылады. [79] еңбекте зерттелген дифференциалды криптоталдау криптографиялық жүйелердегі шығыс деректерге кіріс деректерінің өзгеруінің әсерін зерттеуге бағытталған. Бұл әдіс кіріс деректері өзгерген кезде шығыс деректеріндегі айырмашылықтарды талдау арқылы

шифрлеу алгоритмдерінің ықтимал осалдықтары мен әлсіз жақтарын анықтауға мүмкіндік береді.

3. Шифрлеу алгоритміне сызықтық криптоталдау әдісін қолдану

Сызықтық криптоталдау криптографиялық алгоритмдерді талдау әдісі болып табылады, ол кіріс деректеріндегі шағын өзгерістердің шығыс деректерге әсерін зерттеуге бағытталған. [80] жұмыста сызықтық криптоталдау AES-тәрізді шифрлеу алгоритмдері контекстінде қарастырылады, мұнда XSLP шифрлары біркелкі шашырау алмастырулары мен тіркелген сызықтық шашырау коэффициенттерін ескере отырып қарастырылады. Бұл зерттеу криптографиялық алгоритмдердің беріктігін бағалау үшін маңызды болып табылатын XSLP шифр итерацияларының ерікті саны үшін сызықтық ықтималдық қатынастары жүйесіне қатысатын жергілікті алмастырулардың ең аз санының дәл бағасын алуға мүмкіндік береді.

4. Шифрмәтіндердің статистикалық қауіпсіздігі

Статикалық криптоталдау, сондай-ақ тек шифрленген мәтіндік криптоталдау ретінде белгілі, шифрды бұзу әдісі, онда талдаушы шифрланбаған бастапқы деректерге (ашық мәтін) немесе шифрлеу кілттеріне қол жеткізбестен тек шифрланған хабарламаларды (шифрлық мәтіндер) пайдаланады. Бұл әдіс таңбалар жиілігі немесе олардың комбинациялары сияқты шифрленген мәтіннің статистикалық сипаттамаларын зерттеуге негізделген және осы деректерден хабарламаның шифрын шешуге немесе пайдаланылатын кілтті анықтауға әкелуі мүмкін үлгілерді алуға әрекет жасайды [81].

5. Алгебралық криптоталдау негізіндегі зерттеулер

Алгебралық криптоталдау – шифрлеу кілттерін табу үшін алгебралық теңдеулерді қолдануға негізделген криптографиялық алгоритмдерді талдау әдісі. Бұл әдіс [82] мақаласында талқыланғандай, шифрлеу алгоритмдеріндегі осалдықтарды анықтауға және алгебралық криптоталдаудан қорғау әдістерін әзірлеуге арналған тиімді құрал болып табылады. Алгебралық криптоталдау шифрлеу және дешифрлеу операцияларын алгебралық теңдеулер ретінде көрсету және кілттерді табу үшін оларды шешу арқылы криптографиялық жүйелерді талдауға мүмкіндік береді.

Сонымен қатар, автор Алғазы К.Т. жұмысында [83] жүргізілген ПЕКСЖ негізделген алгоритм де зерттеудің бөлігі ретінде шифрлық беріктікке сыналған. Осы шифрлық беріктік сынағының нәтижелері, жұмыстың 2.1 және 2.2 кестелерінде толық көрсетілген.

Кесте 2.1 – Шифрмәтіндерді статистикалық зерттеу нәтижелері

Типі	Тесттер	Qamal	Qamal NPNS
Графикалық	1. Элементтердің үлестірілім гистограммасы	995	990
	2. Жазықтықта үлестірілу	1000	1000
	3. Сериялады тексеру	968	980

	4. Монотондылыққа тексеру	980	965
	2. Байттық автокорреляциялық функция	969	955
	3. Биттік автокорреляциялық функция	972	970
	4. Графикалық спектарльдық тест	999	976
	5. Сызықтық күрделілік профилі	1000	1000
Бағалау тесттер	1. 0 мен 1-ді тексеру	1000	1000
	2. Байланыстырылмаған сериялады тексеру	998	991
	3. Символ бойынща тексеру	987	983
	4. Интервалдарды тексеру	983	988
	5. Комбинацияларды тексеру	970	975
	6. Купондар жинаушы тесті	956	968
	7. Алмастырылуларды тексеру	960	971
	8. Монотондылыққа тексеру	959	962
	9. Корреляциялық тексеру	980	962
	10. Сызықтық күрделілікті тексеру	1000	1000
	11. Спектрлік тест	998	1000
	12. Қиысылысатын ауыстыруларды тексеру	1000	995

Қазіргі заманда стандарт ретінде ұсынылған симметриялы блокты шифрлардың басым көпшілігінде S-блоктар қолданылады. Оларды қолдану шифрлеу алгоритмдерінің беріктілігін арттыруда өте үлкен маңызға ие. Көп жағдайда сызықтылықты бұзатын да осы S-блоктар болып табылады. Салыстыру үшін 2.2-кестеде, басқада белгілі шифр алгоритмдерінде қолданған S–блоктарға жүргізілген сызықтық және дифференциалдық криптоталдаудың нәтижелері көрсетілген.

Кесте 2.2 – S блоктарға жүргізілген сызықтық криптоталдау нәтижелері

Шифрлеу алгоритмнің атауы	Ең аз кездесу саны	Ең көп кездесу саны
DES	12	48
ГОСТ 28147-89 (құрастырушылар көрсеткен S-блок үшін)	2	14
ГОСТ Р 34.13-2015	100	156
AES-128(Rijndael)	111	145
Калина	104	152
EM cipher	100	156
Qamal	112	144

Зерттеліп отырған алгоритмдерге жүргізілген зерттеулер, осы алгоритмдер бойынша алынған шифрмәтіндердің талап етілген деңгейге сай екендігін көрсетті.

Сондай-ақ, деректерді шифрлеу алгоритмдерін аппараттық жүзеге асыруға назар аудару керек. Аппараттық құрал физикалық бұзу және бүйірлік арналарды талдау мүмкіндігін болдырмайтындай етіп жасалуы керек. Осы контексте криптографиялық беріктікті қамтамасыз ететін аппараттық және бағдарламалық құралдардың құрамдас бөліктеріне арналған спецификациялар мен талаптарды қамтитын FIPS стандарты кеңінен қолданылады.

Аппараттық құрал ретінде ПЛИС пайдалану криптографиялық беріктікті жақсарту үшін зерттеушілер әртүрлі тәсілдерді зерттеді. Анандакумар және басқалар [84]көліктік деңгей қауіпсіздігі және IoT қолданбалары үшін Интернет протоколының қауіпсіздігі сияқты криптографиялық протоколдарда ПЛИС пайдалануды талқылайды. Бұл ПЛИС негізіндегі криптографиялық шешімдердің өсу тенденциясын көрсетеді. [85] жұмыс авторлары AES-GCM сияқты күшті криптографиялық алгоритмдерді қамтитын Smart Grid жүйесіндегі желілік жабдықтың IP мекенжайлары үшін қауіпсіз ПЛИС негізіндегі протоколды және IP өзегін ұсынады. Бұл маңызды инфрақұрылымдағы қауіпсіздік мәселелерін шешу үшін ПЛИС практикалық іске асырылуын көрсетеді. Сонымен қатар, Рой және т.б. [86] криптографияда, қорғаныста және аэроғарышта ПЛИС қолданбаларының дамып келе жатқан қолданбаларына тоқталып, қауіпсіздіктің маңызды салаларында ПЛИС технологиясының әмбебаптығы мен өсіп келе жатқан маңыздылығын көрсетеді. Сонымен қатар, Пан және т.б. [87] криптографиялық алгоритмдерді, әсіресе AES-ті IoT қолданбаларында пайдаланудың артуын атап өтеді, бұл өзара байланысты құрылғыларда күшті қауіпсіздік шараларының қажеттілігіне сәйкес келеді. Бұл IoT экожүйелерін қорғау үшін ПЛИС енгізулерінің өзектілігін көрсетеді. [88] жұмыс авторлары AES, RSA, DES және SHA сияқты әртүрлі криптографиялық алгоритмдерді енгізу үшін ПЛИС пайдалануды көрсетеді, әртүрлі өнімділік көрсеткіштері үшін криптографиялық шешімдерді сынау және талдау кезінде ПЛИС технологиясының практикалық қолданылуын көрсетеді. Қорытындылай келе, криптографиялық қосымшалар үшін ПЛИС пайдалану әртүрлі салалардағы қауіпсіздік шараларын жақсартуға перспективалы жол ашады, соның ішінде Интернет заттары, смарт-торлар және байланыс жүйелері. Қаралған зерттеулер деректер қауіпсіздігін арттыру және қауіпсіз байланыс арналарын қамтамасыз ету үшін ПЛИС негізіндегі криптографиялық шешімдердің маңыздылығын көрсетеді.

2.4 Кілттерді қорғауды жақсарту үшін динамикалық хаосты қолдану

Докторлық диссертацияның бұл бөлімінде кілттерді қорғауды жақсарту контекстінде динамикалық хаосты қолдану қарастырылады. Динамикалық хаос – бұл кездейсоқ немесе псевдокездейсоқ деректерді жасау үшін болжанбайтын және күрделі іс әрекет бар жүйелерді қолдануға болатын математикалық құбылыс. Динамикалық хаостың бұл қасиеті оны кілттердің қауіпсіздігін

нығайтудың пайдалы құралына айналдырады, өйткені, болжамды кілттер жасау шабуылдарға осал болуы мүмкін.

Тәжірибелік деңгейде хаостық және криптографиялық жүйелер арасында ұқсастықтар бар. Осылайша, К.Шенноннның классикалық еңбектерінде атап өткендей, хаостық сигналдар туралы еске салуға болады. Мысалы, [89]-де «хаос» сөзін айтпайды деп айта аламыз, бірақ ол аргументке тәуелді араластыру, өлшемді сақтайтын түрлендірулерді ұсынып, созылу мен бүктеу арқылы хаостың пайда болуының негізгі механизмін нақты атап өтеді.

Хаос теориясы – бастапқы шарттарға өте сезімтал динамикалық жүйелердің қозғалысын қарастыратын математиканың бір бөлімі, яғни бастапқы шарттардағы аздаған өзгерістер түбегейлі басқа нәтижелерге әкелуі мүмкін [90-91].

Кілттерді қорғауды жақсарту үшін динамикалық хаосты қолданудың маңызды әдістерінің бірі – кездейсоқ кілттерді құру үшін хаостық жүйелерді пайдалану. Псевдокездейсоқ генераторлар (ПКГ) сияқты дәстүрлі кездейсоқ сандарды генерациялау әдістерінің болжамдылығы немесе кездейсоқтықтың төмен дәрежесі сияқты кемшіліктері болуы мүмкін.

Динамикалық хаосқа негізделген кездейсоқ кілттерді құру процесі келесі қадамдарды қамтуы мүмкін:

- Сандық тізбектерді құру үшін Лоренц жүйесі немесе басқа хаостық тартқыштар сияқты хаостық жүйені қолдану;

- Бұл сандық тізбектерді негізгі таңбаларға немесе биттерге түрлендіру;

- Деректерді шифрлеу немесе қол қою үшін кілттерді қолдану.

Кездейсоқ кілттерді құру үшін динамикалық хаосты қолдану негізгі тізбектерді болжау мүмкін емес, сондықтан шабуылдарға төзімді етеді.

Кездейсоқ кілттерді құру үшін динамикалық хаосты қолдануға болады. Лоренц жүйелері [92] немесе Ресслер жүйелері [93] сияқты хаостық жүйелер бастапқы жағдайларға болжамсыздық пен сезімталдық қасиеттеріне ие. Бұл бастапқы параметрлердегі шамалы өзгерістердің өзі айтарлықтай әртүрлі нәтижелерге әкелуі мүмкін дегенді білдіреді. Бұл болжамсыздықты криптографиялық тұрғыдан сенімді кілттер жасау үшін пайдалануға болады.

Динамикалық хаосты деректерді шифрлеу және шифрын ашу процестерін нығайту үшін де пайдалануға болады. Бұған шифрлеу процедураларына хаотикалық элементтерді енгізу арқылы қол жеткізуге болады. Мысалы:

- Хаостық функцияны шифрлеу алгоритміне енгізу қорғауды күшейтетін неғұрлым күрделі және болжанбайтын шифрланған деректерге әкелуі мүмкін.

- Шифрлеу процесінде қолдануға болатын қосымша кілттерді немесе параметрлерді жасау үшін хаотикалық жүйелерді пайдалану қауіпсіздікті арттырады.

- Аутентификация процестерін басқару немесе деректерді ұрлаудан қорғау үшін динамикалық хаосты қолдану жүйенің жалпы қауіпсіздігін күшейтуі мүмкін.

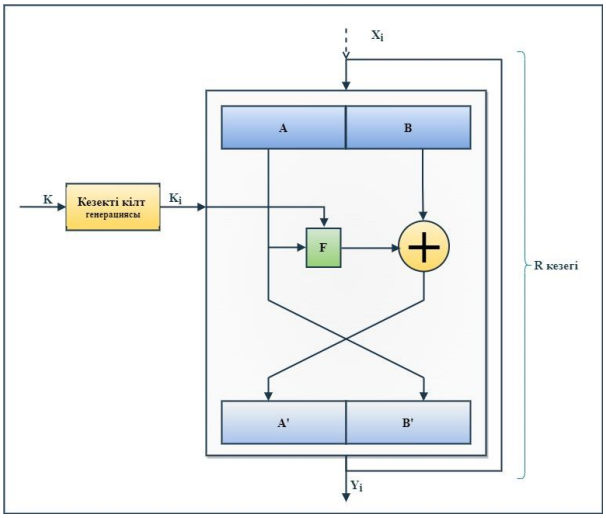
Кілттерді қорғауды жақсарту контекстінде динамикалық хаосты сәтті қолдану үшін құбылыстың негізгі аспектілерін түсіну маңызды:

Анықталмаған динамикалық хаос өзінің детерминизмімен сипатталады [94]. Бұл хаостық процестерге ұшыраған жүйелердің болжау мүмкін еместігі жоғары екенін білдіреді. Мысалы, жүйенің бастапқы жағдайындағы шамалы өзгерістер оның болашақ мінез-құлқында айтарлықтай айырмашылықтарға әкелуі мүмкін. Бұл хаосқа тән детерминизм динамикалық хаосты кездейсоқ немесе псевдокездейсоқ деректерді, соның ішінде кілттерді жасау үшін пайдалы етеді. Жақсы жасалған кілттер деректер қауіпсіздігі үшін маңызды болғандықтан, динамикалық хаосты пайдалану кілттердің болжанбау деңгейін күшейтіп, оларды шабуылдарға төзімді ете алады.

Динамикалық хаостың тағы бір маңызды аспектісі – оның бастапқы жағдайларға сезімталдығы. Бұл бастапқы параметрлердегі немесе кірістердегі ең аз өзгерістер жүйенің іс-әрекетіндегі елеулі өзгерістерге әкелуі мүмкін дегенді білдіреді. Бастапқы жағдайларға мұндай сезімталдық бастапқы деректерге негізделген жүйенің болашақ мәндерін немесе күйлерін болжауды қиындатады. Кілт қауіпсіздігі контекстінде бұл Кілт жасау үшін бастапқы деректердегі шамалы өзгерістер де мүлдем басқа кілттердің пайда болуына әкелуі мүмкін дегенді білдіреді. Бұл кілттерді бұзғысы келетін немесе шифрланған деректердің шифрын ашқысы келетін зиянкестерге қосымша кедергі бола алады.

Динамикалық хаостың осы негіздерін зерттеу оны кілттерді қорғау контекстінде қалай пайдалануға болатынын жақсырақ түсінуге және құпия ақпаратты сенімді қорғауды қамтамасыз етуге көмектеседі.

Таратқыш-қабылдағыштарды деректерді шифрлеу функциясын қамтамасыз ету үшін динамикалық хаос, атап айтқанда хаостық салыстыру принциптеріне негізделген шифрлеу алгоритмі әзірленген [95]. Осылайша, хаостық кескіндеу есептеу ресурстарына қажеттілік тұрғысынан «арзан». Бейсызық болуына байланысты мұндай салыстырулар іздеу кестелерінің орнына (дәстүрлі алгоритмдерде) қолдануға жақсы үміткер болып табылады, бұл жедел жадыға қойылатын талаптарды айтарлықтай төмендетеді (2.5-сурет).



Сурет 2.5 – Фейстель желісін қолданатын динамикалық хаосқа негізделген шифрлеу алгоритмінің схемасы

Сонымен қатар, хаостық картаны есептеуге арналған өрнек оның аргументтері анықталған жиынға қатысты инвариантты болып табылады, бұл ресурстардың қолжетімділігіне байланысты оның әртүрлі нұсқаларын жүзеге асыру үшін бірдей салыстыру құрылымын пайдалануға мүмкіндік береді. Және бұл шифрлеу алгоритміндегі блок ұзындығын өзгерту мүмкіндігіне әкеледі. Хаостық динамиканың көмегімен әзірленген блоктық симметриялық шифрлеу алгоритмінің құрылымы итерациялық Шеннон схемасына негізделген, онда негізгі түрлендіру ретінде Фейстель желісі қолданылады.Сонымен қатар, хаосқа негізделген криптографияның бастапқы жағдайларға жоғары сезімталдық, эргодизм және үлкен кілт кеңістігі сияқты артықшылықтары атап өтілді, бұл оның криптографиялық қолданбалар үшін әлеуетін арттырады. Сондай-ақ, сандық косинус хаос картасының хаосқа негізделген криптографиялық алгоритмдердің өнімділігін жақсарту мүмкіндігі көрсетілді, бұл оның криптографиялық тұрақтылықты жақсартудағы маңыздылығын көрсетеді.

Бөлім бойынша қорытынды

Бұл 2-тарауда құрылған шифрлеу алгоритмінің сенімділігі зерттеліп, оның нәтижелері алынды. Зерттеу жұмыстары шифрлеу алгоритмінің көмегімен алынған шифрмәтіндердің статистикалық қауіпсіздігін тексеру жұмыстарымен басталады. Одан кейін, криптографияда қажетті шарттардың бірі – шифрдың лавиндік әсері тексерілген. Алгоритмнің беріктілігін бағалау үшін криптоталдаудың алгебралық, дифференциалдық, сызықтық және тағы басқа да әдістерімен тексерілген. Жүргізілген криптошабуылдардың теориялық бағасы ғана емес, нәтижелері нақты мысалдар арқылы да көрініс табады. Сонымен қатар, кілтті ПЕКСЖ-де пайдаланғанда алгоритмнің беріктілігіне әсері зерттелді және анықталды. Деректерді шифрлеу үшін ПЛИС шығаратын хаостық сигналдарды қолдану туралы пікірталас та қызықты, өйткені ол шифрлеу үшін пайдаланылуы мүмкін кездейсоқ сандарды генерациялаудағы осы сигналдардың әлеуетін көрсетеді.

3 МӘЛІМЕТТЕРДІ ШИФРЛЕУ ҮШІН БАҒДАРЛАМАЛЫҚ ЖӘНЕ АППАРАТТЫҚ ҚҰРАЛДАРДЫ ӘЗІРЛЕУ

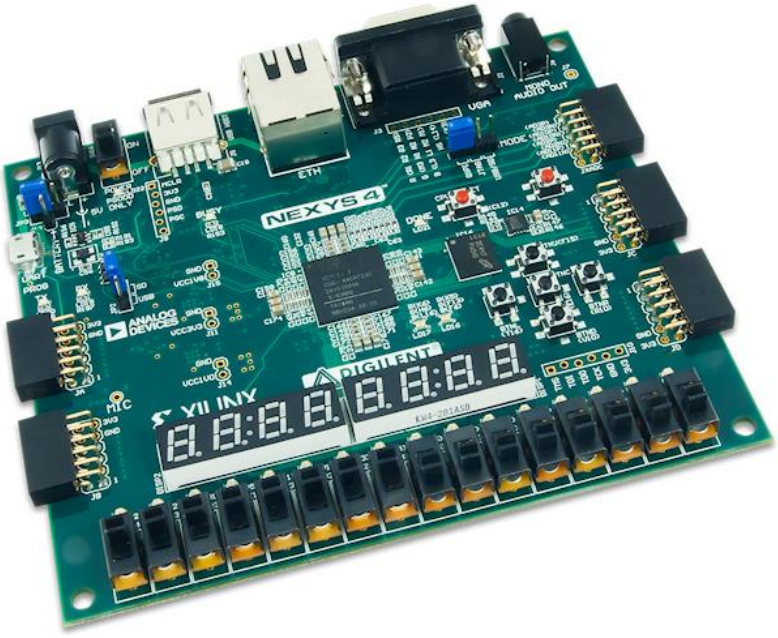
Жүйелер мен есептеуіш техниканың дамуының қазіргі тенденциялары ақпараттық қауіпсіздікті қоса алғанда, өнімділігі жоғары есептеуіш құрылғыларды әзірлеуді талап етеді. Ақпараттық-коммуникациялық желілер мен интеграциялық құрылғылар дамыған сайын криптографиялық түрлендірулер үшін тиімді аппараттық шешімдерді құру қажеттілігі артады. Қауіпсіздікті, тұтастықты және қуатты талдау шабуылдарына төзімділікті қамтамасыз ететін аппараттық криптографиялық жүйеге артықшылық беріледі. Криптографиялық алгоритмдерді іске асыруға арналған стандартты емес аппараттық шешімдерді жоғары деңгейлі аппараттық сипаттама тілдерін пайдалана отырып, цифрлық құрылғыларды жобалауға мүмкіндік беретін ПЛИС көмегімен алуға болады. Мұндай технологияларды қолдану әсіресе жаңа буын байланыс желілерін (5G) және заттар интернетін дамыту контекстінде өзекті болып табылады[96-98].

3.1 Цифрлық технологиялардағы тиімділік пен қауіпсіздікті арттыру

Бұл бөлімде GF(2) шекті өрісінде келтірілмейтін көпмүшелердің мәліметтер қорын құруға арналған Бен-Ор алгоритмінің эксперименттік талдауының нәтижелерін ұсынылды. Бұл зерттеудің негізгі мақсаты – бейсызық көпмүшелердің негізін құрудағы алгоритмнің тиімділігін бағалау [99].

Digilent Nexys 4 Artix-7 тақтасын GF(2) соңғы өрісінде келтірілмейтін көпмүшелердің дерекқорын жасау үшін қолданылды. Бағдарламалық жасақтама ретінде Vivado Design Suite HLx Edition және Verilog бағдарламалау тілі пайдаланылды.

Nexys 4 DDR тақтасы (3.1-сурет) – Artix-7 чипіне негізделген Xilinx компаниясының ПЛИС платформасы.



Сурет 3.1 – Nexys4 DDR тақтасы

Ол қуатты ПЛИС (Xilinx XC7A100T-1CSG324C үлгісі), кең сыртқы жады және көптеген порттармен, соның ішінде USB, Ethernet және басқа интерфейстермен жабдықталған. Nexys 4 DDR коммутациялық тізбектерді қоса алғанда, өндірістік жобаларға және қуатты кірістірілген процессорлармен бірге пайдалануға жарамды. Nexys 4 DDR тақтасы жаңа өнімділігі жоғары Vivado® Design Suite және Xilinx компаниясының бұрынғы ISE® пакетімен үйлесімді. Xilinx қосымша ақысыз дизайн жасауға мүмкіндік беретін WebPACK™ сияқты осы құралдардың тегін нұсқаларын ұсынады. Бұл Nexys 4 DDR тақтасын өз жобаларында пайдаланғысы келетін әзірлеушілер үшін қолжетімділік пен ыңғайлылықты қамтамасыз етеді.

Бұл жұмысты жүзеге асыру үшін Xilinx компаниясының Vivado Design Suite hlx Edition бағдарламалық құралы жобаның бағдарламалық бөлігін әзірлеу және жобалау үшін пайдаланылды (3.2-сурет). Ол барлық бағдарламаланатын чиптер мен ПЛИС жүйелері үшін «Driven Design Platforms» технологиясын қолдана отырып, өте жоғары өнімділікті қамтамасыз етеді.

Vivado® Design Suite HLx жаңа нұсқалары үш басылымды қамтиды: HL System Edition, HL Design Edition және HL WebPACK™ Edition.



Сурет 3.2– Vivado Design Suite HLx Edition белгісі

Xilinx Vivado Design Suite HLx Edition ПЛИС негізіндегі жобаларды жобалауға, модельдеуге, синтездеуге және талдауға арналған көптеген құралдарды ұсынады. Бұл жобаларды құруды және оңтайландыруды жеңілдететін, сонымен қатар әзірлеушілердің өнімділігін арттыратын және жобада жылдам әрі тиімді жұмыс істеуге мүмкіндік беретін дамудың барлық аспектілерін басқарудың қарапайым интерфейсін қамтамасыз ететін интеграцияланған даму ортасын қамтиды.

Келтірілмейтіндікке тексеру алгоритмі – берілген екілік көпмүшенің берілетінін немесе берілмейтінін анықтау әдісі. Бен-Ор алгоритмі - қарапайым рекурсивті процедураға негізделген жылдам төмендетілмейтін тексеру алгоритмі. Негізгі идея – берілген көпмүшеге қарапайым сызықтық түрлендірумен байланысты көпмүшелердің тізбегін есептеу. Содан кейін берілген көпмүшенің төмендетілмейтіндігі тізбектегі соңғы көпмүшенің дәрежесін зерттеу арқылы анықталады.

ПЛИС-та алгоритмді жүзеге асыру бірнеше кезеңдерді қамтиды. Алдымен модульдің кірістері мен шығыстарын анықтаймыз. Кіріс деректері тексерілетін

екілік көпмүше және көпмүше дәрежесі болып табылады. Шығу көпмүшенің келтірілетін немесе келтірілмейтіндігін көрсететін екілік сигнал болып табылады.

Әрі қарай, Verilog көмегімен алгоритмнің өзін жүзеге асырылды. Алгоритм бірнеше қадамдарды қамтиды, соның ішінде тізбектегі алғашқы бірнеше көпмүшелерді есептеу және олардың өкілеттіктерін тексеру. Бұл қадамдар өзара байланысты модульдер тізбегі ретінде жүзеге асырылды.

Бен-Ор алгоритмі көпмүшелердің ЕҮОБ (ең үлкен ортақ бөлгіш) есептеу үшін қолданылады. $i = 1 - \text{ден } n/2$ дейін мәндер үшін $q = \gcd(f(x), (x^{Q^i} - x) mod f(x))$ мәндерін есептеу үшін қолданылды. Егер q кем дегенде бір рет өзгеріссіз қалады, онда көпмүше келтірілмейтін болып саналады.

Көпмүше келтірілмейтін болуы үшін $i = 1 - \text{ден } n/2$ дейін мәндердің барлық шығыс мәндер үшін ақиқат болуы қажет.

Бұл алгоритм мына тұжырымдамаға негізделген:

Кіріс: $f(x) \in F_Q[x]$ – n дәрежелі нормаланған көпмүше. 3.1-формулада көрсетілгендей кірісті *[n:0] f* деп алдық, мұндағы *n* параметр ретінде берілген разрядты білдіреді, біздің жағдайымызда 16 разрядты көпмүше.

Шығыс: «Ақиқат» - $f(x)$ көпмүшесі келтірілмейтін көпмүше

«Жалған»- $f(x)$ көпмүшесі келтірілетін көпмүше

Шығыс мәніміз *[(2**((n+1)/2))+1:0]q* ретінде берілген. Шығарылатын мәннің саны кіріс көпмүшесінің санына байланысты есептеледі.

Алгоритмде орындалатын негізгі операция ЕҮОБ табу болып табылады:

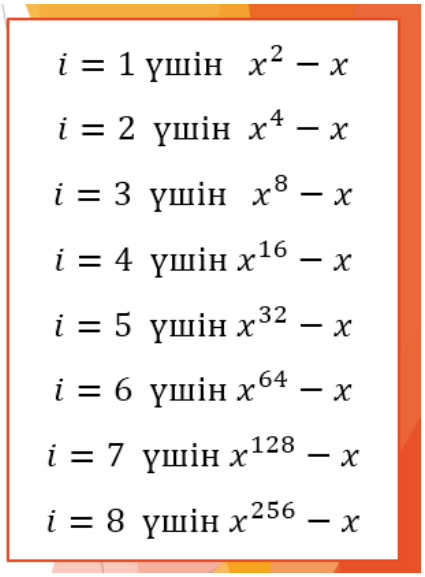
$$q = \gcd\left(f(x), \left(x^{Q^i} - x\right) mod f(x)\right), \tag{3.1}$$

мұнда кіріс көпмүше *[n:0] f* және әрбір i итерациясы үшін $\left(x^{Q^i} - x\right)$ көпмүшенің модуль бойынша кіріс көпмүшеге бөліндісі арасындағы ЕҮОБ есептеледі. Q −өріс көрсеткіші, біздің жағдайымызда 2-ге тең [100].

$\left(x^{Q^i} - x\right)$ көпмүшесі әрбір *i* үшін есептеліп алғашқы мән ретінде беріледі:

p1[1] = 6;
p1[2] = 18;
p1[3] = 258;
p1[4] = 65538;
p1[5] = 4294967298;
p1[6] = 18446744073709551618;
p1[7] = 340282366920938463463374607431768211458;
p1[8]=115792089237316195423570985008687907853269984665640564039457584007913129639938;

Бұл мәндердің көпмүше ретіндегі көрінісі 3.3-суретте көрсетілген.



Сурет 3.3 – әрбір *i* үшін есептелген $\left(x^{Q^i} - x\right)$ көпмүшелері

Бұл көпмүшеліктер модульдің *p* кірісіне беріледі. ЕҮОБ таппас бұрын $\left(x^{Q^i} - x\right) mod f(x)$ өрнегін орындауымыз керек. Бұл өрнек Verilog бағдарламалау тілінде *Assign g=p%f* ретінде ұсынылған. Мұндағы тапсырма құрылымы үздіксіз тапсырманың немесе тапсырмалар тізбегінің сипаттамасы болып табылады.

Ары қарай осы табылған *g* және кіріс көпмүшесін *f* –тің ЕҮОБ тапса болады. ЕҮОБ табу үшін Евклид алгоритмін қолданамыз. Әрі қарай бұл нәтиженің ЕҮОБ g және кіріс көпмүшесі *f* табылды. ЕҮОБ үшін Евклид алгоритмі қолданылды.

Евклид алгоритмі екі бүтін санның ЕҮОБ табудың тиімді алгоритмі болып табылады. Алгоритм келесідей түсіндіріледі:

a және *b* екі нөлдік емес бүтін сан деп есептесек, ЕҮОБ келесі амалдарды орындау арқылы табуға болады.

$$\begin{gathered} a = bq_0 + r_1, \\ b = r_1 q_1 + r_2, \\ r_1 = r_2 q_2 + r_3, \\ \dots \\ r_{n-2} = r_{n-1} q_{n-1} + r_n, \\ r_{n-1} = r_n q_n \end{gathered} \tag{3.2}$$

Сонда *a* және *b* сандарының ЕҮОБ r_n-ге тең. Қарапайым мысал ретінде a = 1071 және b = 462 сандарының ЕҮОБ табу үшін келесі есептеулер орындалды: $1071 = 2 \times 462 + 147$, $462 = 3 \times 147 + 21$, $147 = 7 \times 21 + 0$.

Нәтижесінде ЕҮОБ(1071,462) = 21.

Бұл алгоритмді жүзеге асыру үшін модуль әдісін қолданамыз. Яғни, $r_1 = a \,\%b$, бұл өрнек *a*-ны *b*-ге бөлгенде қалдықты анықтайды.

Біздің жағдайымызда a және b мәндері f және g көпмүшелері болып табылады. Мына $q = \gcd(f(x), (x^{Q^i} - x) mod f(x))$ өрнекті жүзеге асыру алгоритмінің коды Қосымша А-дан табуға болады.

Бұл әрекет әрбір i итерациясы үшін қайталанады, өйткені әрбір итерацияда p және g мәні сәйкесінше өзгереді.

GF(2) ақырғы өрісінде келтірілмейтін көпмүшелердің негізін құру үшін келесі алгоритмі қолданылды:

GF(2) өрісіндегі n дәрежелі көпмүшелердің саны 2^n болатынын және тұрақты мүшесі жоқ көпмүшелердің кертірілмейтінін (өйткені олар x-ке бөлінетінін) ескере отырып, келтірілмейтін қасиеттерге сыналуды қажет ететін көпмүшелердің жалпы саны: 2^{n-1} .

Кіріс: n - GF(2) ақырғы өрістегі көпмүше дәрежесі $n \geq 1$.

a – келтірілмейтін қасиетке зертеу алгортмі.

Шығыс: келтірілмейтін көпмүшелер базасы

егер $n = 1$

b=1; s=1;

егер $n > 1$

$b=2^n + 1$; s=2;

соңы

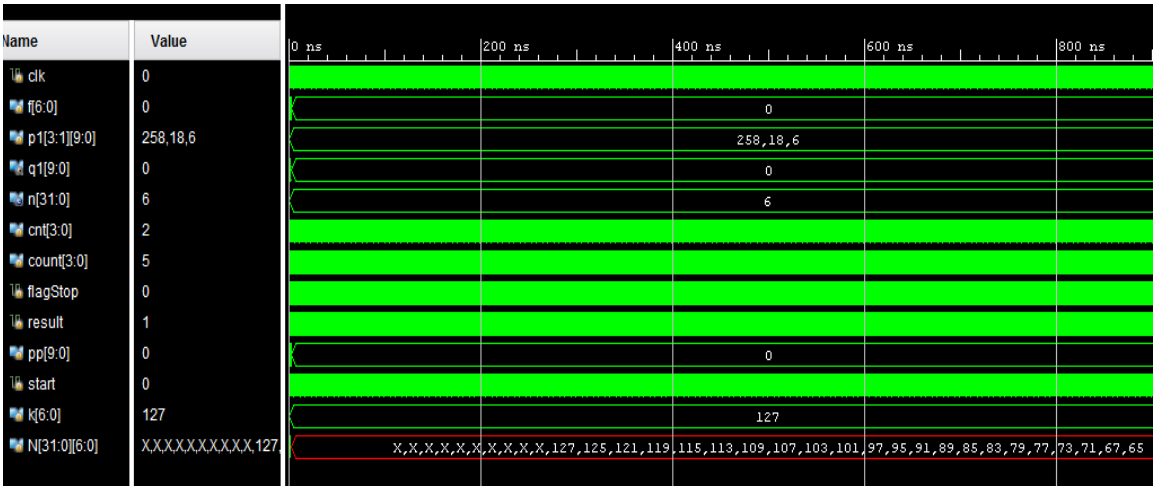
цикл: $j = b$ – дан $2^{n+1} - 1$ *дейін s қадаммен:*

егер *a(j)=”Ақиқат” болса j көпмүшесін базаға жазу*

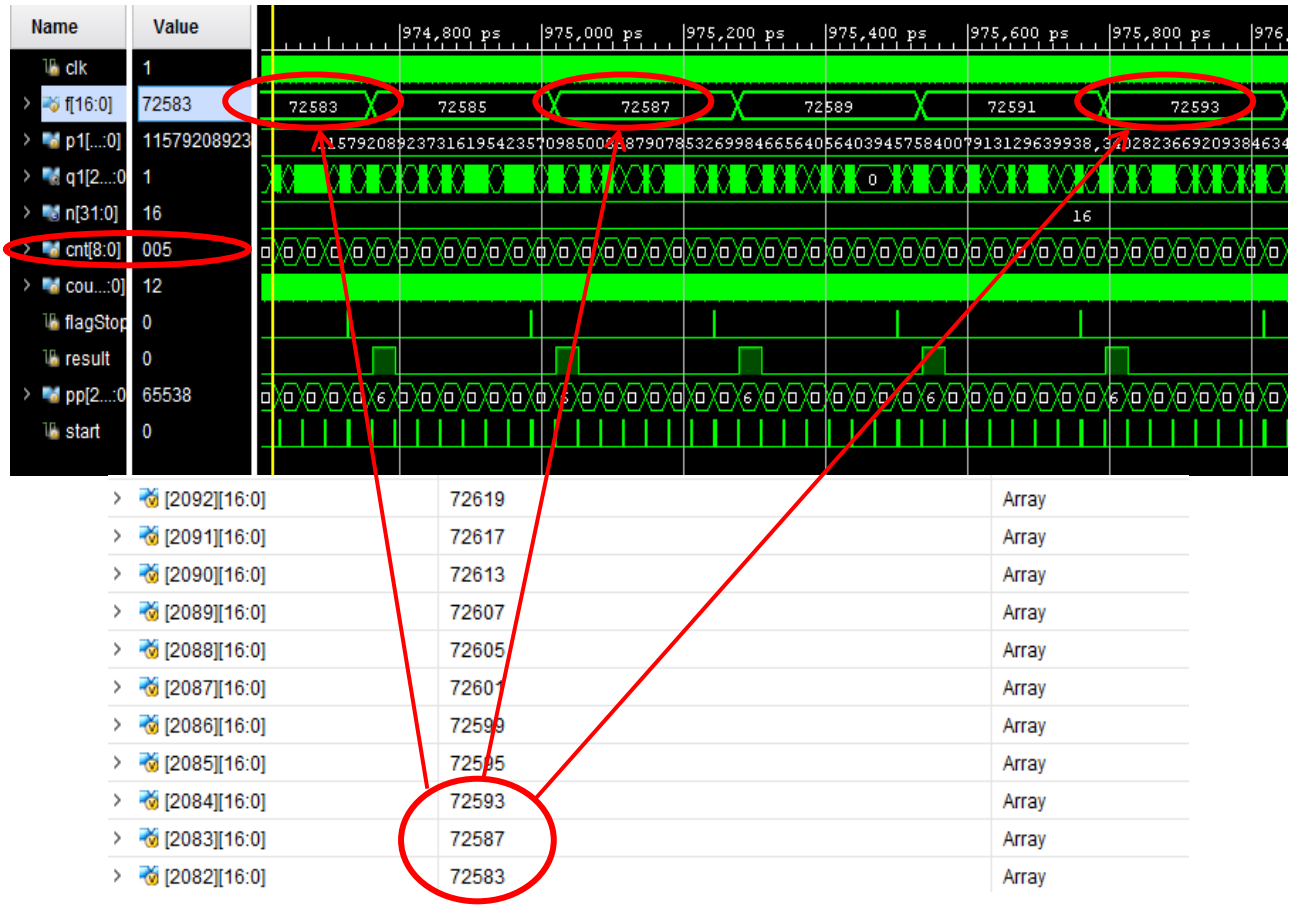
цикл соңы

Көпмүшелерді тексеру кезінде олардың барлығын бір уақытта тексермейміз, өйткені тіпті көпмүшелердің 1-ден басқа бөлгіші болатыны белгілі.Сондықтан жоғарыдағы алгоритм бойынша әрбір ранг үшін тақ сандарға сәйкес келетін көпмүшелерді жинаймыз.Verilog тілінде келтірілмейтін көпмүшелердің жинақталуын Қосымша Ә-ден табуға болады.

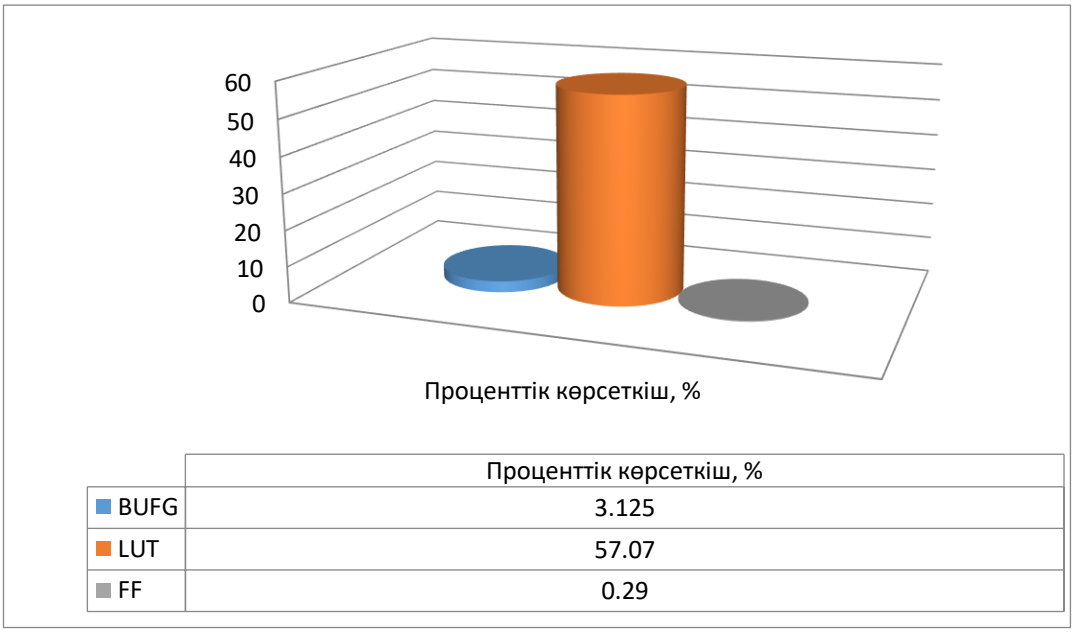
Алынған нәтижелермен уақыт кестесін талдамас бұрын, 3.4-суретті қарастырайық. Ол 6 разрядтан тұратын көпмүшелерді зерттеді, сонымен қатар 22 келтірілмейтін көпмүшені тапты.



Сурет 3.4 – 6 разрядты көпмүшелерді келтірілмейтін қасиетке тексеру нәтижесі

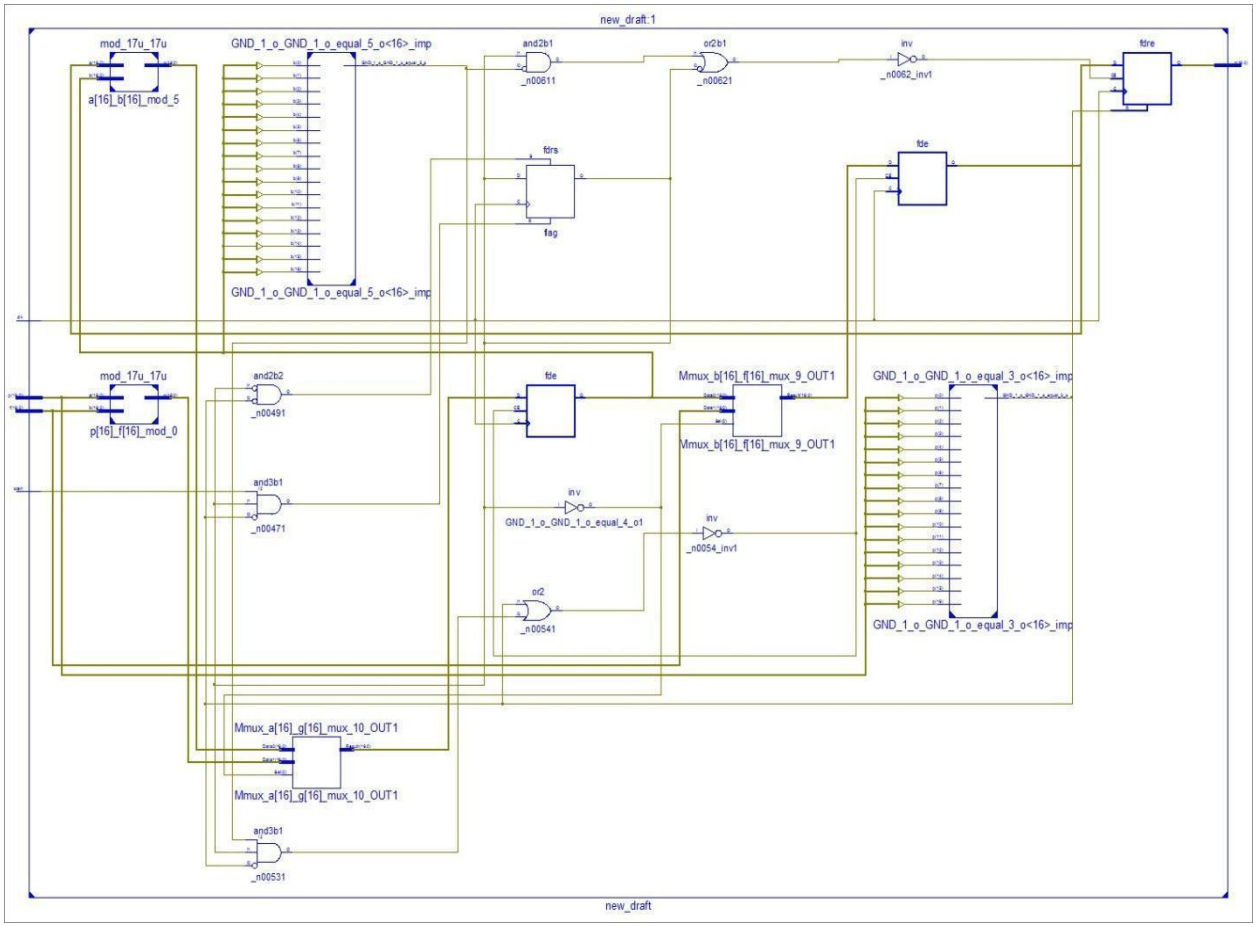


Сурет 3.5 –16 разрядты көпмүшелердің уақыттық диаграмасы және N матрицасындағы орны

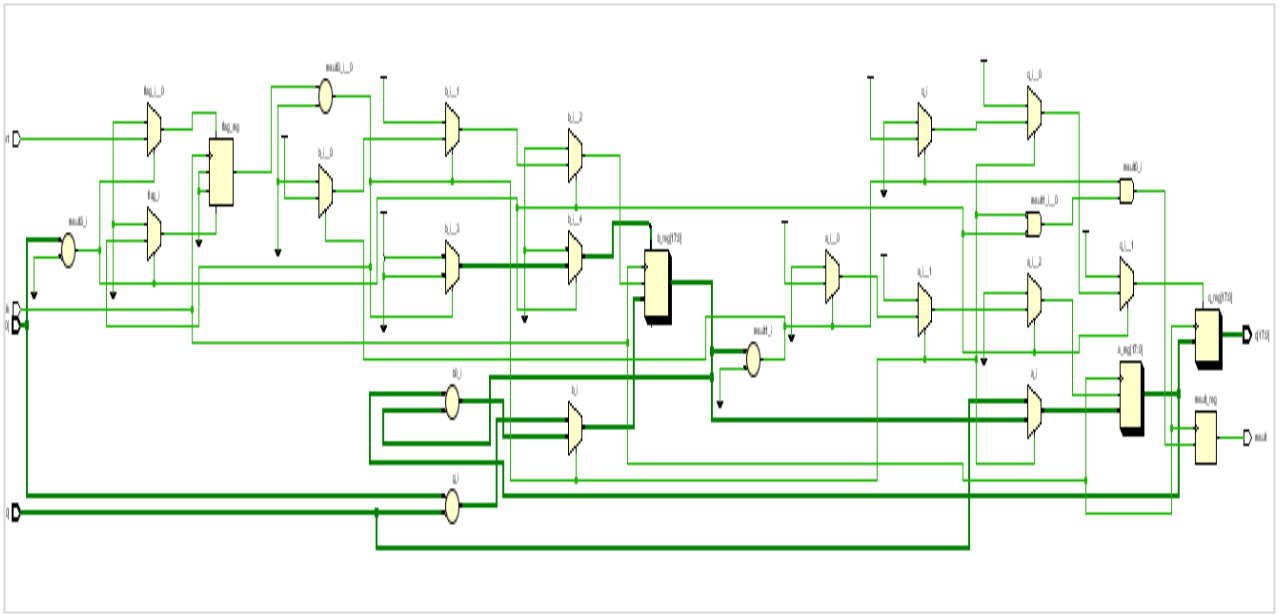


	Проценттік көрсеткіш, %
BUFG	3.125
LUT	57.07
FF	0.29

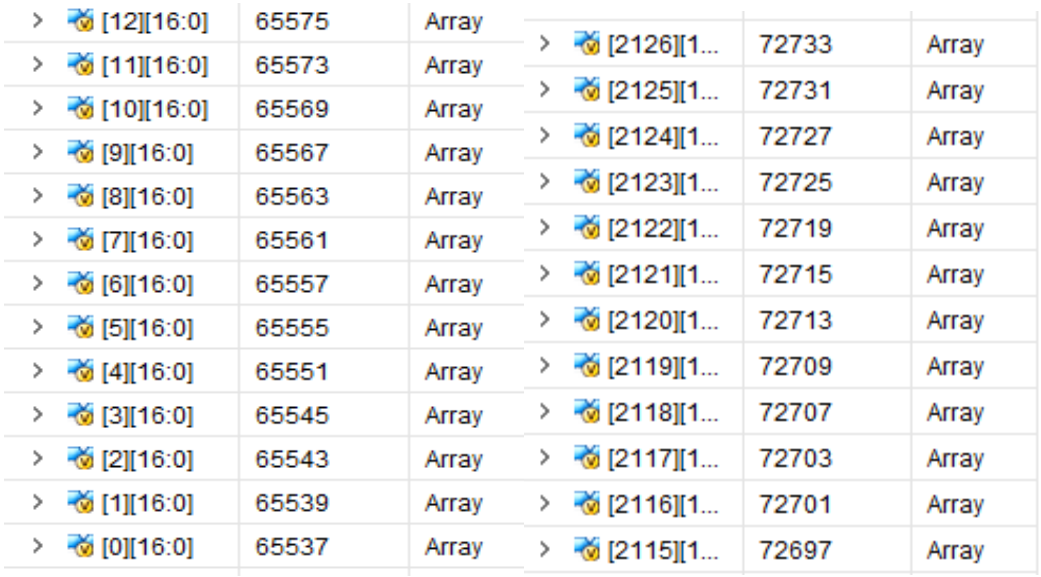
Сурет 3.6 –16 разрядты көпмүшелерді зерттеу кезіндегі ресурстар қолданысының гистограммасы



Сурет 3.7 – ПЛИС-та іске асырылған Бен-Ор алгоритмінің RTL диаграммасы



Сурет 3.8 – GF(2) ақырғы өрістегі келтірілмейтін көпмүшелер базасы алгортимінің RTL схемасы



Сурет 3.9 – Келтірілмейтін көпмүшелердің *N* матрицасы

Бұл жұмыста 16 разрядқа дейінгі көпмүшелер зерттелінді. Алынған келтірілмейтін көпмүшелер N матрицада жинақталады. 3.5-суретте 16-разрядты көпмүшелердің уақыт диаграммасы және олардың N матрицадағы орны көрсетілген. 16-разрядты көпмүшелерден барлығы 2139 көпмүше келтірілмейтін болып шықты. 16-разрядты көпмүшелерді тексеру кезінде LUT логикалық блогының 57,07%, FF тригерлердің 0,29% және BUFG ішкі жадының 3,13% пайдаланылды. Бұл есептеулер 258 биттік мәндерді пайдаланды. Бұл есептеулер кезінде 258 разрядты мәндер қолданылды. Бұл дегеніміз 134 600 LUT-тан 76 822 LUT, 269 200 FF-тен 789 FF, 32 BUFG-дан 1 BUFG пайдаланылуда дегенді білдіреді.

3.2 ПЛИС негізіндегі хаостық генераторларды пайдалана отырып, псевдокездейсоқ сандарды алу

Бұл бөлімде ПЛИС негізіндегі хаостық генераторларды пайдалана отырып, псевдокездейсоқ сандарды алу әдісін, сондай-ақ олардың тиімділігін және криптографиялық жүйелерде пайдалану мүмкіндіктерін зерттеу қарастырылды. Криптографияда псевдокездейсоқ сандарды генерациялау үшін ПЛИС негізіндегі хаостық генераторларды пайдалану мүмкіндігін зерттеу үшін келесі тапсырмалар орындалды:

1. ПЛИС негізіндегі псевдокездейсоқ тізбектердің хаостық генераторларының жұмысының негізгі принциптері қарастырылады.
2. Чуа және Росслер жүйелері [101] сияқты бейсызық динамикалық жүйелерде мұндай генераторларды жүзеге асыру мысалдары талданған.
3. Жүзеге асырылған хаостық генераторлар [102] негізіндегі PSS генерациясының тиімділігі мен қасиеттері бағаланады.

4. Алынған нәтижелерді криптографиялық жүйелерде қолдану мүмкіндігі қарастырылып, тиісті ұсыныстар берілген [103].

Осы жұмыстың шеңберінде ПЛИС негізіндегі псевдокездейсоқ реттіліктердің енгізілген хаостық генераторлары негізінде құрылған SRP тиімділігі мен қасиеттерін бағалау бойынша нақты нәтижелер алу күтілуде. Жасалған PRP сапасын, оның ішінде олардың статистикалық қасиеттерін бағалау үшін эксперименттер жүргізіледі және криптографиялық жүйелерде осы реттіліктерді пайдалану бойынша ұсыныстар беріледі [104].

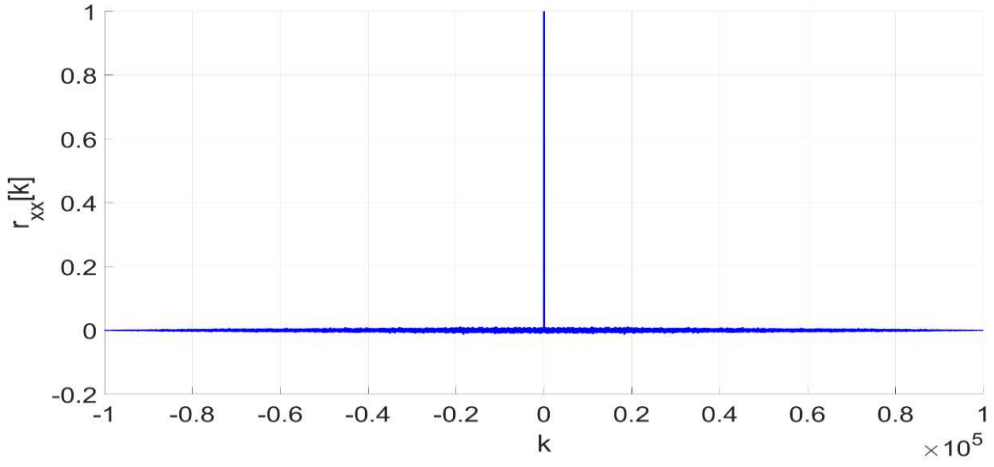
Бұл зерттеудің материалдары мен әдістеріне зерттеу объектілерін таңдау кірді – Росслер және Чуа жүйелері сияқты бейсызық дифференциалдық теңдеулер немесе бейнелеулер арқылы сипатталған белгілі динамикалық жүйелер. Сандық және графикалық мәліметтерді алу үшін MATLAB Simulink бағдарламалық пакетін қолданылды, осы жүйелерді сипаттайтын теңдеулерді пайдалана отырып, әрбір жүйенің блок-схемалары жасалынды. Содан кейін әр айнымалы үшін уақыт қатарын алу үшін Simulink арқылы сандық модельдеу орындалды.

Бұл жүйелердегі хаосты талдау үшін қолданылған әдістердің бірі автокорреляциялық функцияны есептеу болды. Кездейсоқ сигналдың автокорреляциялық функциясы мына формула арқылы есептелінді:

$$r_{xx}[k] = \frac{1}{N}\sum_{n=1}^{N} x[n]x[n+k]. \qquad (3.3)$$

Уақыт сериясының әртүрлі нүктелерінде сигнал мәндерінің қалай корреляцияланатынын көрсетеді (3.10-сурет).

Аппараттық құралдарға динамикалық хаос генераторларын енгізу үшін Xilinx Artix 7 ПЛИС негізіндегі, атап айтқанда Nexys 4 ddr және XC7A100T-1CSG324C тақтасы қолданылды. MATLAB жүйесінде блок-схемалар жиналды және оларды Verilog-қа түрлендірілді. Осыдан кейін алынған кодты ПЛИС-та жарқыратылып жіберілді, бұл динамикалық хаос генераторларын енгізуге мүмкіндік берді.

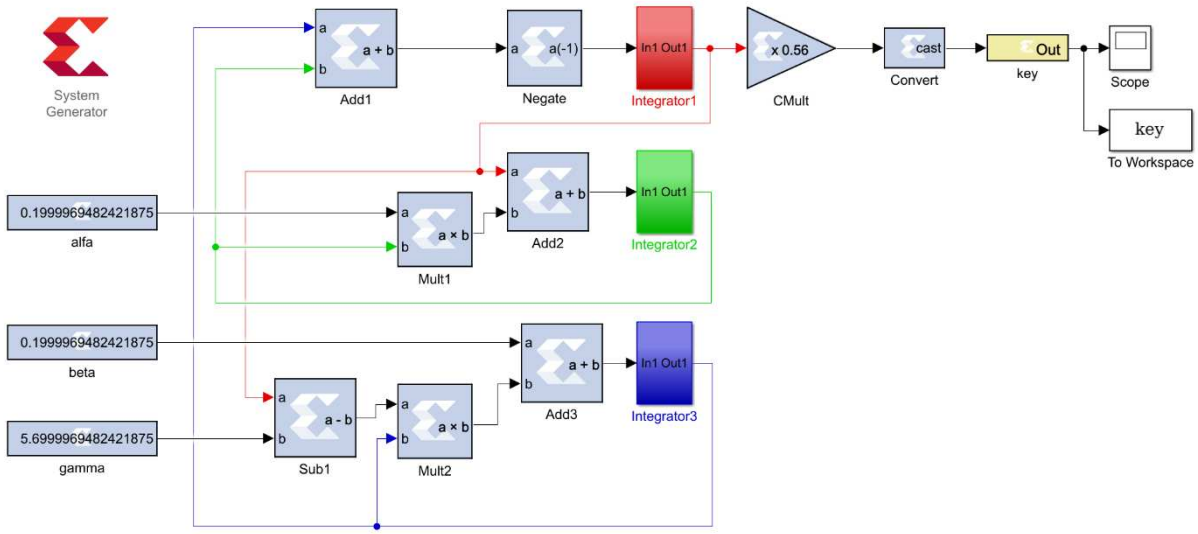


Сурет 3.10 – Дискретті сигнал үшін автокорреляция формуласы

Ресслер жүйесін сандық модельдеу үшін біз MATLAB Simulink жүйесінде осы жүйені сипаттайтын теңдеулерді пайдалана отырып, блок-схеманы (3.11-сурет) құрдық:

$$\begin{aligned} \dot{x} &= -(y+z), \\ \dot{y} &= x + ay, \\ \dot{z} &= \beta + z(x-\gamma), \end{aligned} \tag{3.4}$$

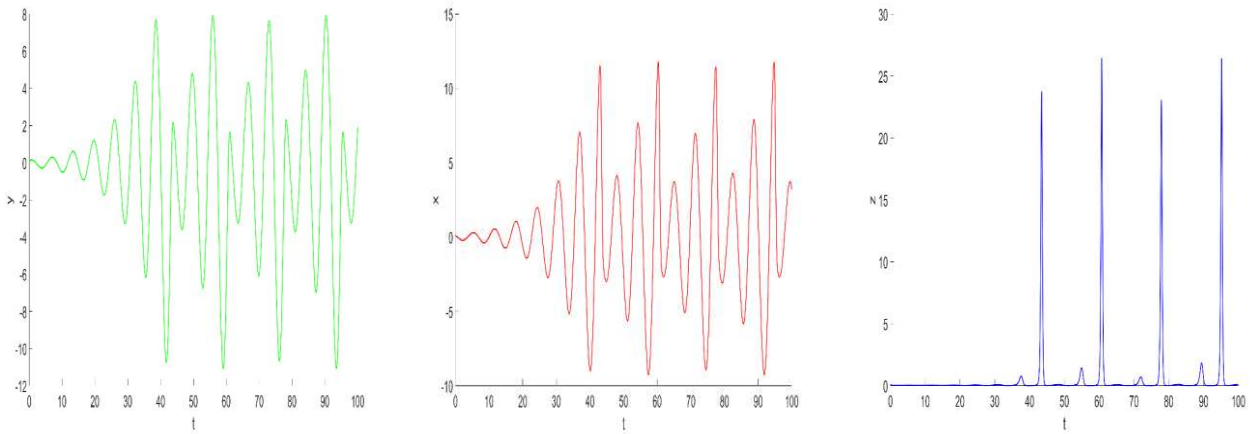
α = 0.2, β = 0.2, γ = 5.7 болған кезде



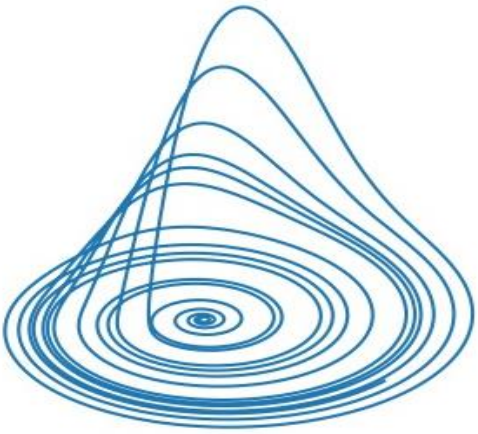
Сурет 3.11 – MATLAB ортасында Ресслер жүйесі

Блок-схема x, y және z күй айнымалыларына сәйкес үш блокты, сондай-ақ α, β және γ параметрлеріне сәйкес блоктарды қамтиды.

Блок-схеманы жасағаннан кейін әр айнымалы үшін уақыт қатарын алу үшін Simulink көмегімен Rossler жүйесінің сандық модельдеуі орындалды.



Сурет 3.12 – y, x және z (a) айнымалылары бар Matlab жүйесінде Ресслер генераторын уақытша енгізу



Сурет 3.13 – Росслер жүйесінің аттарокторы



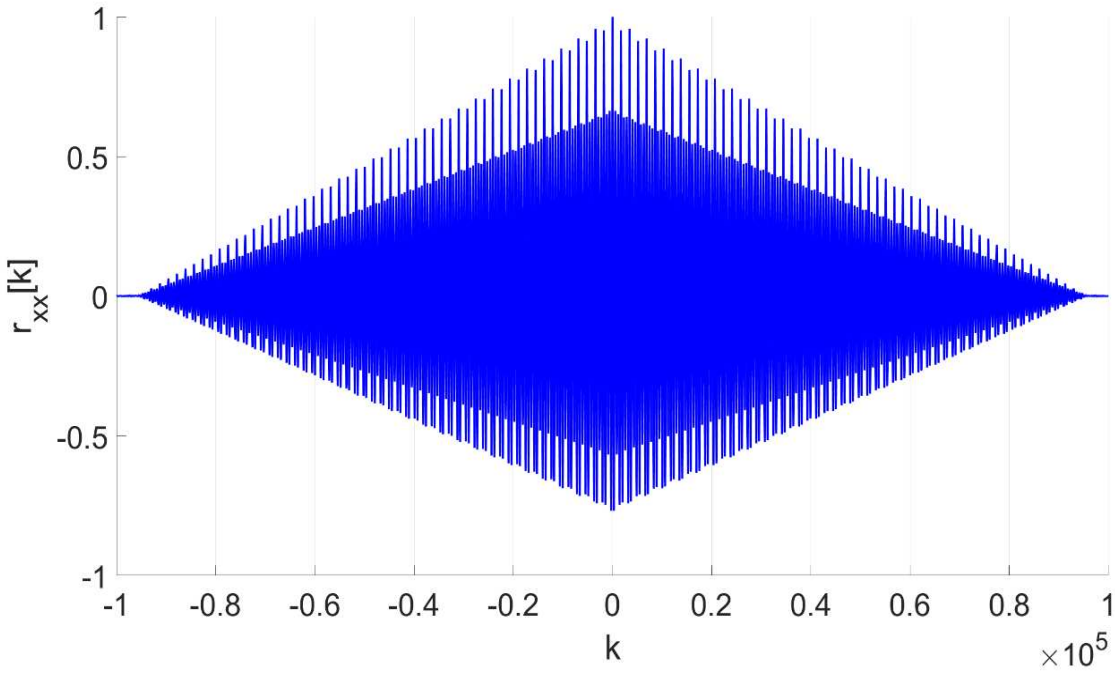
Сурет 3.14 – Verilog-те Росслер генераторының уақытша іске асырылуын сынау нәтижесі



Сурет 3.15 – ПЛИС негізіндегі Росслер генераторын нақты уақытта енгізу

Содан кейін алынған нәтижелерді автокорреляциялық функция әдісі пайдаланылып, Росслер жүйесінде хаос кездейсоқтықтың болуы анықталды (3.16-сурет).

Осылайша, MATLAB Simulink-те Росслер жүйесінің блок-схемасы және сандық модельдеу осы жүйенің динамикасын зерттеуге және ондағы хаос кездейсоқтықтың болуын анықтауға мүмкіндік береді.



Сурет 3.16 – Росслер генераторының автокорреляция функциясы

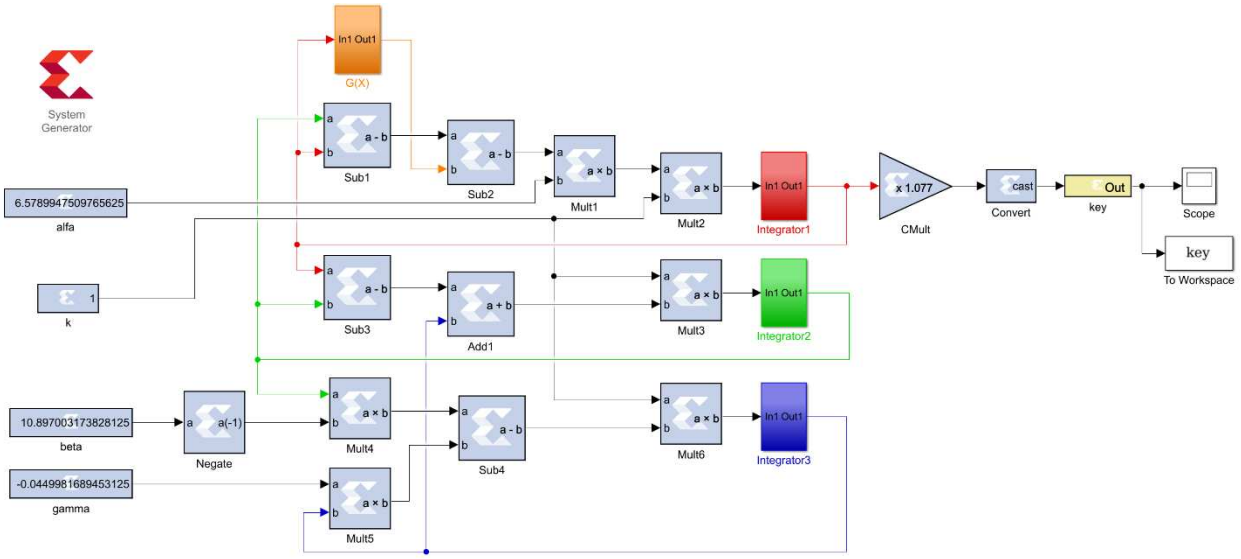
Модельдеу нәтижелеріне сәйкес, Росслер жүйесіне негізделген генератор периодты сандық мәндерді жасайтыны анықталды. Бұл жүйе жасайтын сандар тізбегін болжауға болатынын және кездейсоқ емес екенін білдіреді.

Осыған байланысты кездейсоқ сандарды генерациялау үшін бұл жүйені пайдалану тиімсіз, өйткені ол кездейсоқтықтың жеткілікті жоғары дәрежесін қамтамасыз етпейді.

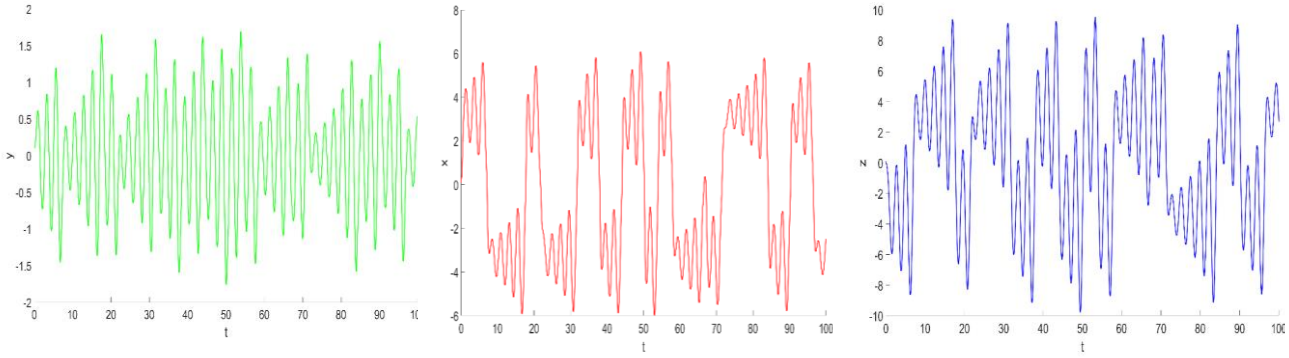
Бұдан басқа, Чуа жүйесі де қарастырылды (3.17-сурет), оның келесі теңдеуі бар:

$$\begin{gathered} \dot{x} = k\alpha(y - x - g(x)), \\ \dot{y} = k(x - y + z), \\ \dot{z} = \beta + z(x - \gamma), \\ g(x) = m_0 x + 0.5(m_1 - m_0)(|x + 1| - |x - 1|). \end{gathered} \tag{3.5}$$

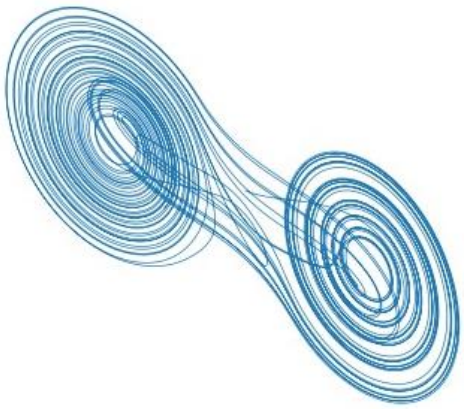
Егер: $\alpha = 6.579,\ k = 1, \beta = 10{,}879, m_0 = -0.652, m_1 = -1.812, \gamma = -0.045$



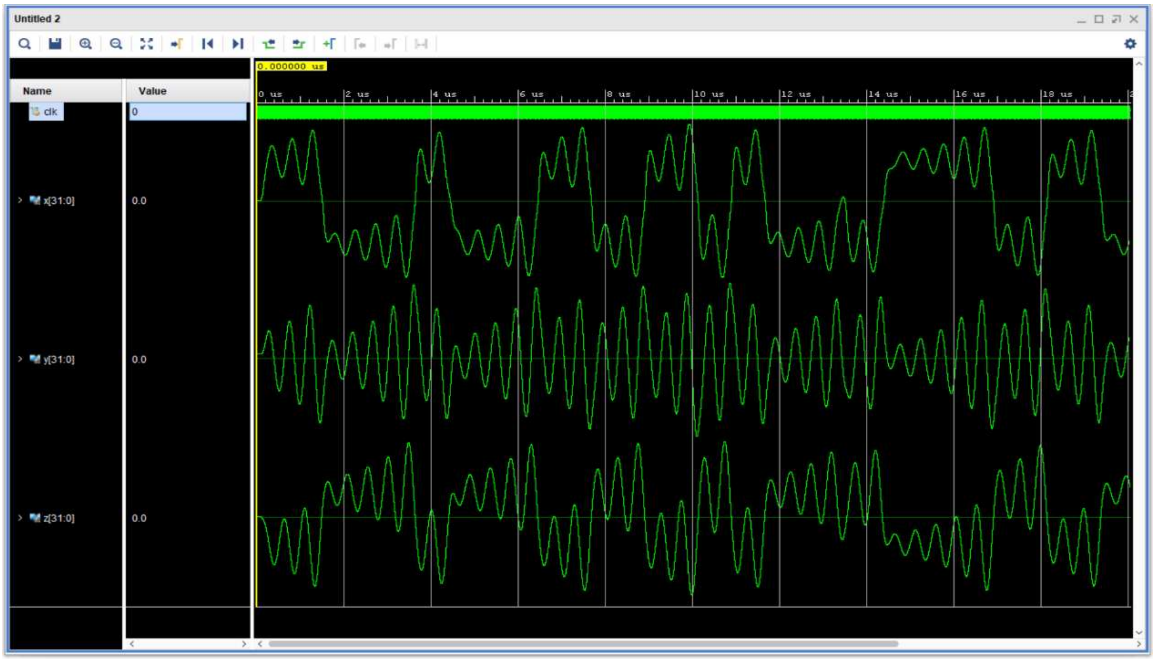
Сурет 3.17 – Чуа генераторының құрылымдық сұлбасы



Сурет 3.18 – Чуа генераторын Matlab жүйесінде x, y және z (a) айнымалыларымен уақытша енгізу

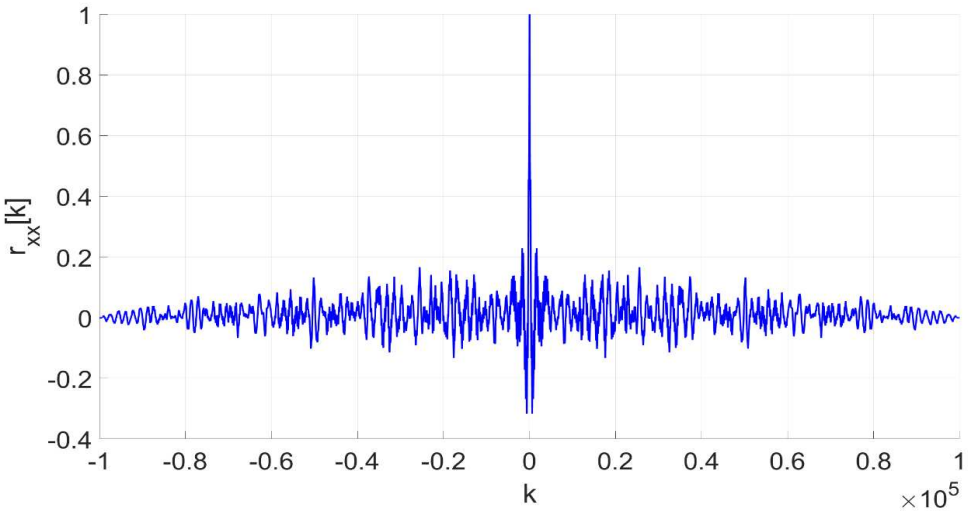


Сурет 3.19 – Чуа жүйесінің аттарокторы



Сурет 3.20 – Verilog-те Чуа генераторының уақытша іске асырылуын сынау нәтижесі

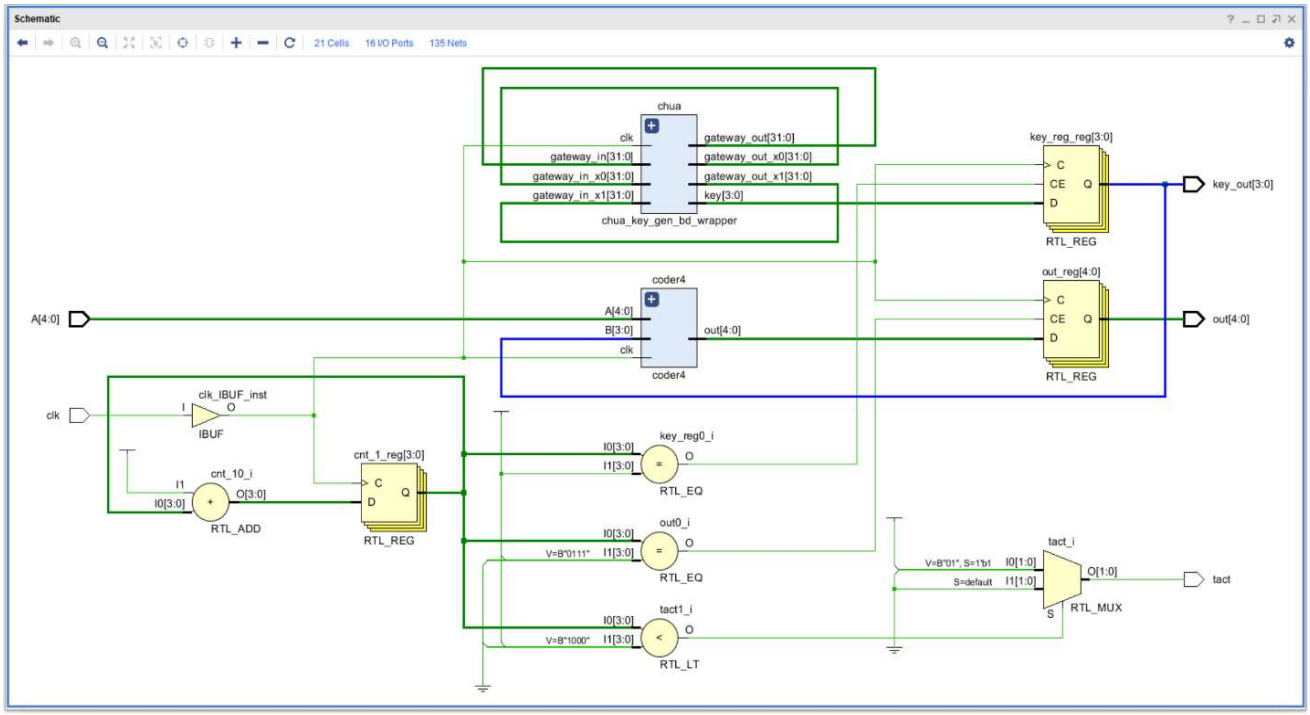
Осыдан кейін генерацияланған сигналдар үшін автокорреляция функциясы құрылды және оның нәтижелерін талқыланды (3.21-сурет).



Сурет 3.21– Чуа генераторының автокорреляциялық функциясы

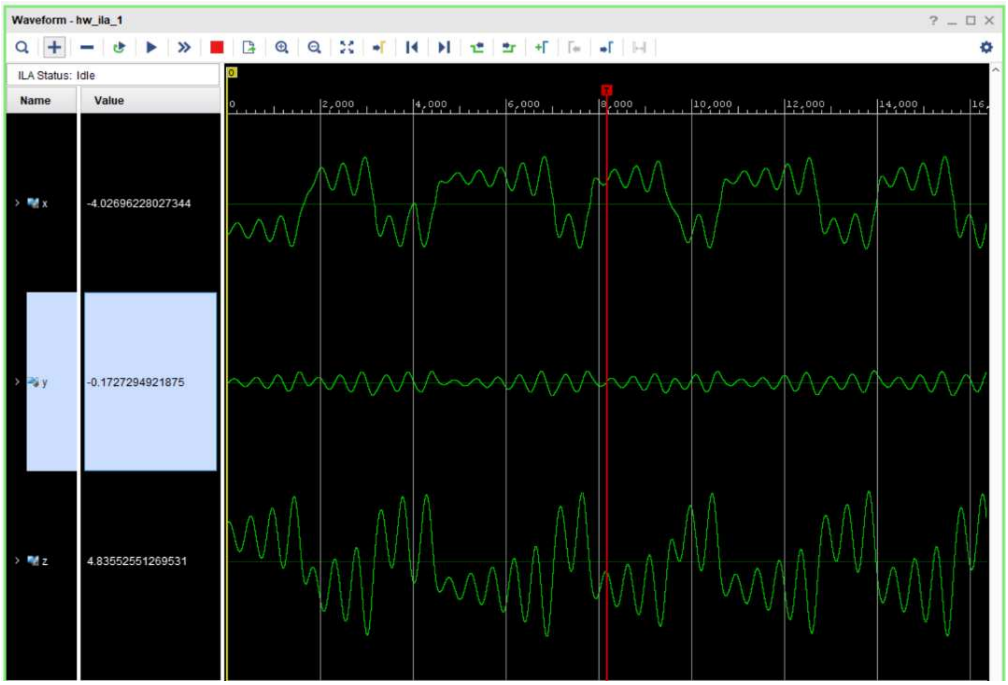
Алынған мәліметтерге сәйкес, Чуа жүйесінде сигнал генерациясы кездейсоқ сигналға жақын, бұл оны кездейсоқ сандарды генерациялауға қолайлы етеді. Атап айтқанда, осы генератормен құрылған кездейсоқ санды генерациялау арқылы шифрлеу кілті ретінде пайдалануға болады. Чуа генераторының криптотөзімділігі [105] жұмыста да зерттелінген.

Төменде құпия кілттерді жасау үшін пайдаланылған Чуа жүйесінің RTL схемалары және Debug схемалары берілген.

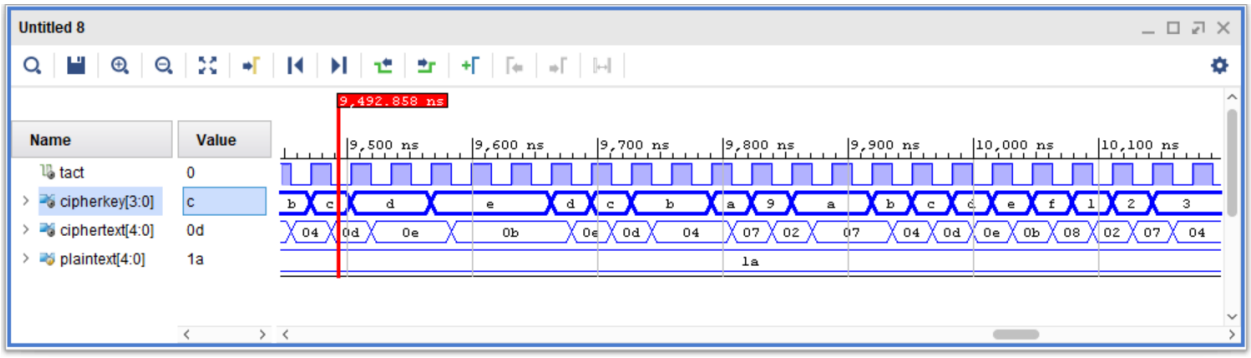


Сурет 3.22 – Шифрлеу блогының (RTL) құрылымдық диаграммасы

Чуа RTL схемалары чиптегі ПЛИС жүйесін енгізуге арналған. Олар кездейсоқ сандарды генерациялауды басқаруға және шифрлеу кілттері ретінде пайдалану үшін құпия кілттерді жасауға мүмкіндік береді.



3.23– а-сурет. ПЛИС негізіндегі Росслер генераторын нақты уақытта енгізу



3.23 – б-сурет . Шифрлеу алгоритмін тексеру нәтижесі

Чуа жүйені жөндеу диаграммалары жүйедегі ПЛИС микросхемасын жөндеу және тексеру үшін пайдаланылды. Олар кездейсоқ сандарды генерациялау процесін басқаруға және жүйедегі ықтимал қателер мен ақауларды анықтауға мүмкіндік береді.

Зерттеу барысында екі жүйеге – Росслер мен Чуаға салыстырмалы талдау жасалды. Әрбір жүйе үшін автокорреляциялық функциялар құрастырылды және Чуа жүйесіндегі сигналдың генерациясы кездейсоқ сигналға жақын, бұл кездейсоқ сандарды генерациялау үшін қолайлы деген қорытынды береді. Сондай-ақ, шифрлеу кілттерін генерациялау үшін Чуа жүйесін пайдалануды қарастырылды және ПЛИС негізіндегі шифрлеу алгоритмі енгізілді.

Зерттеулеріміздің нәтижесінде Чуа жүйесі кездейсоқ сандарды генерациялау және шифрлеу кілттерін жасау үшін пайдалануға болатыны расталды. Чуа жүйесін талданып, генерацияланған сигналдың кездейсоқ сигналға жақындығын көрсететін автокорреляция функциясы алдынд. Шифрлеу кілттерін жасау үшін Чуа жүйесін пайдалану құпия ақпаратты қорғаудың тиімді әдісі болуы мүмкін. Дегенмен, бұл жүйенің ықтимал шектеулері мен осалдықтарын ескеру қажет, оны шабуылдаушылар шифрланған хабарламалардың шифрын ашу үшін пайдалана алады.

Бұл жұмысты одан әрі дамыту кездейсоқ сандарды генерациялау және шифрлеу кілттерін жасау үшін пайдаланылуы мүмкін басқа жүйелерді тереңірек зерттеуді қамтуы мүмкін. Сондай-ақ, ПЛИС негізіндегі шифрлеу алгоритмін күрделірек математикалық әдістер мен алгоритмдерді пайдалана отырып жақсартуға болады.

3.3 ПЛИС негізіндегі ашық кілтпен шифрлеу жүйелерін жеделдету бойынша жұмыс

ПЛИС шифрлеудың ең танымал тәсілдерінің бірі AES (Advanced Encryption Standard) немесе DES (Data Encryption Standard) сияқты симметриялық кілт алгоритмдерін пайдалану болып табылады. Дегенмен, шифрлеудың басқа әдістері бар, мысалы, RSA алгоритміне негізделген асимметриялық шифрлеу (Rivest-Shamir-Adleman) немесе эллиптикалық қисық шифрлеу (ECC).

Сонымен қатар, деректер қауіпсіздігін жақсарту үшін кейбір әзірлеушілер бірнеше шифрлеу әдістерінің тіркесімін пайдаланады. Мысалы, деректерді жылдам өңдеу үшін AES симметриялық шифрлеуын және кілттерді қауіпсіз тасымалдау үшін RSA асимметриялық шифрлеуын пайдалануға болады.

ПЛИС-та шифрлеу маңызды міндет болып табылады, мұнда негізгі аспектілердің бірі қолайлы кілт пішімін таңдау және деректерді шифрлеу/шифрды шешудің жоғары жылдамдығын қамтамасыз ету үшін өнімділікті оңтайландыру болып табылады. ПЛИС-та шифрлеу алгоритмдерінің кең ауқымын іске асыруға болатынымен, кілт пішімін таңдау өнімділікке айтарлықтай әсер етуі мүмкін. Кейбір алгоритмдер ПЛИС енгізу үшін тиімді болмауы мүмкін ұзын пернелерді пайдалануы мүмкін. Сонымен қатар өнімділікті оңтайландыру параллельді есептеу және ПЛИС ресурстарын оңтайландыру сияқты әртүрлі әдістерді пайдалануды қамтуы мүмкін.

Бұл жұмыста CAD Quartus Prime Lite Edition жүйесінде жабдықтың оңтайлы құны бар модульдік беріліс қорабының әрекеттік моделі әзірленді. Операция алгоритмі Verilog HDL тілінде жүзеге асырылады [106]. Әрбір есептеу қадамында модульдің үштік немесе екі еселенген мәні немесе бірлік мәні солға екіге жылжыған ең маңызды биттерден шегерілетін әдіс қолданылады. Әрекеттік моделінің алгоритмін функционалдық және уақытша модельдеу мысалдар арқылы жүзеге асырылды және алгоритмнің дұрыстығы расталды. Төмен бюджеттік ПЛИС Cyclone VE 5CEBA4F23C7 үшін Altera компаниясынан регистрді тасымалдау деңгейі (RTL) құрылғысының диаграммасы алынды. Уақытты талдау әр түрлі жұмыс жағдайларындағы негізгі және әрекеттік модель үлгілері үшін максималды тактілік жылдамдығын анықтау үшін уақыт анализаторының көмегімен орындалды. Мұнда өнімділікті арттыру көптеген шешімдерде берілген сандардың өткізу қабілетіне тікелей пропорционалды аппараттық құралдардың құнын арттыру арқылы қол жеткізіледі. Сондықтан, оларды көп таңбалы сандарды азайту үшін пайдалану, егер аппараттық шығындарды оңтайландыруға қойылатын талаптар болмаса, ақталған. Аппараттық құралдар құнының артуы қуат тұтынуды арттырып, сенімділікті төмендететінін есте сақтаңыз. Аппараттық құралдар шығындарын оңтайландыру және төмен бағамен қалдық модульді ерікті санды (модульді азайту) жеделдетіп анықтау өзекті мәселе болып табылады.

Аппараттық шығындарды оңтайландыру және төмен құны бар екілік бүтін сандарды модульдік қысқарту құрылғысының функционалды блок-схемасы авторларымен әзірленген және ұсынылған. Бұл жұмыстың мақсаты негізгі модельмен салыстырғанда оның артықшылықтары мен кемшіліктерін анықтау үшін осы модульдік қысқарту құрылғысының әрекеттік моделін жобалау және зерттеу болып табылады. Алдыңғы зерттеулерде құрылғы блоктарының схемаларын және тұтастай алғанда бүкіл құрылғының принципті моделін жобалау жүзеге асырылды [107]. Негізгі үлгіні жобалау және жөндеу үшін CAD Quartus Prime Lite Edition 16.0 (Altera) нұсқасы қолданылды. Құрылғыны іске асыру Altera (Intel компаниясының бас ұйымы) - 5CEBA4F23C7 шығарған

базалық Cyclone VE отбасының ПЛИС интегралды схемасы бар төмен бюджеттік DE0-CV тақтасына бағытталған.

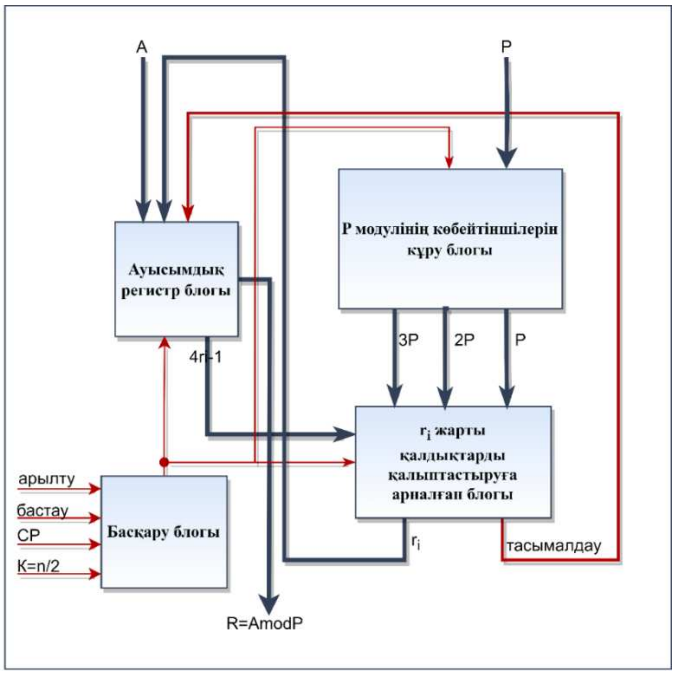
Бұл зерттеудің міндеттері Verilog HDL тілінде әрекеттік моделін жасау, оны зерттеу және екілік бүтін сандарды модульден жылдам азайту үшін құрылғы схемаларының мәтіндік сипаттамасын пайдалана отырып, құрылғының негізгі моделі мен әрекеттік моделінің салыстырмалы талдауын жүргізу болып табылады[108-109].

5CEBA4F23C7 Cyclone VE базалық тобының төмен бюджеттік ПЛИС құрылғысын енгізу кезінде аппараттық және уақыт шығындарының көзқарасы.

Құны төмен Cyclone VE сериялы тақталар аз қуат тұтыну және төмен баға маңызды болып табылатын әртүрлі қолданбаларда пайдалануға арналған. Осылайша, Cyclone VE отбасы жаппай және үнемді қолдану үшін оңтайлы шешім болып табылады [110].

Модульдік қысқарту үшін жоғары жылдамдықты құрылғының әрекеттік моделін әзірлеу.

Модуль сандарын жылдам азайту үшін қарастырылатын құрылғының құрылымы басқару блогынан, ауысымдық регистр блогынан, P модулінің көбейткіштерін генерациялау қондырғысынан және r_i жарты қалдықтарды қалыптастыруға арналған қондырғыдан тұрады (сурет-3.24). Бұл құрылғыда 2n-разрядты A санын n-разрядты P модуліне бөлу кезінде қалдықты есептеуді екі есе жылдамдату үшін біз әрбір есептеу қадамында P мәні не үш есе (3P) өсетін әдісті қолдандық немесе екі еселенген (2P), немесе бір рет (P), модульдің мәні P алдыңғы қалдықтың жоғары ретті биттерінен алынады (4ri-1), екі бит солға жылжиды.



Сурет 3.24 – Сандарды модуль бойынша жылдам қысқартуға арналған құрылғының құрылымдық схемасы

Басқару блогы қалпына келтіру, іске қосу, тактілік импульстарды (CP) қабылдайды, K = n/2 (n - модульдің сыйымдылығы P, n/2 - модульді азайту операциясын орындау үшін қажетті тактілік импульстардың санын анықтайды).

P модулінің көбейткішінде екі еселенген және үш еселенген модульдің (2P және 3P) екілік көрінісі және бірліктер толықтауышы (бұдан әрі азайту амалдары үшін) алдын ала есептеледі.

Ауысым регистрінің блогы бастапқыда азайтылған A санын, содан кейін жарты қалдықтарды сақтауға және оларды екі битке солға жылжытуға, содан кейін ең маңызды разрядтарды $(4r_{i-1})$ жарты қалдық генераторға беру үшін қолданылады.

Жарты қалдық r_i генерациялау блогы әрбір бөлу циклінде жарты қалдық r_i алады. Тасымалдау сигналы ауысу регистрінің жоғары ретті разрядтарына r_i кіруін басқарады. Соңғы бөлу циклінде алынған жарты қалдық R нәтижесі болып табылады. Құрылғының және оның блоктарының жұмысының толық сипаттамасы берілген.

CAD Quartus Prime жүйесінде әрекеттік үлгісін жасау үшін Verilog HDL схемаларының мәтіндік сипаттамасын пайдалана отырып, модуль сандарын азайту модулінің жұмысын сипаттайтын жоба жасалды. Б қосымшасында n = 6 екілік сандар үшін әрекеттік моделдің алгоритмі көрсетілген деп жазамыз. Бұл бағдарламаның негізгі құрылымдық бірлігі redmod модулі болып табылады. Модуль енгізу және шығару порттарының атауларын хабарлайды: R шығысы нәтижені (соңғы қалдық) құрылғының шығысына шығару үшін қолданылады; A және P_1 кірістері сәйкесінше 12-биттік азайтылатын екілік A санының және 6-биттік P модулінің мәндерін алады (сандардың екілік көрінісі пайдаланылады). Солға жылжыту операциясын биттерді жоғалтпай орындауға мүмкіндік беру үшін ең маңызды екі бит қосымша енгізілген. Құрылғының жұмысын басқаратын сағат импульсі clk кірісі арқылы беріледі.

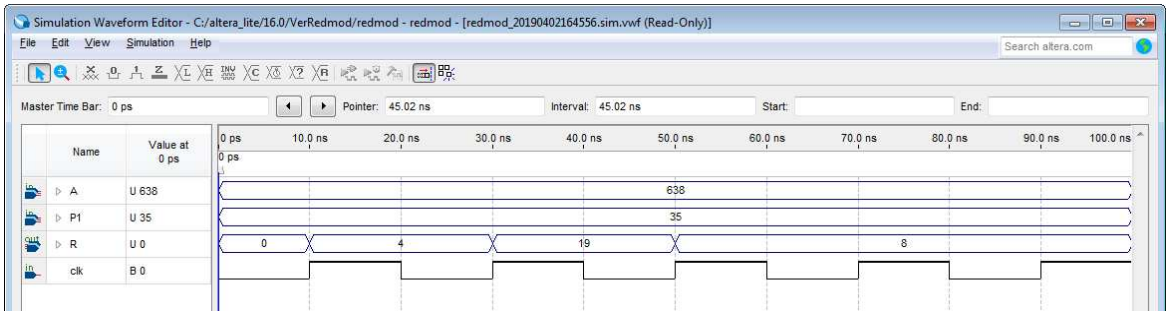
Verilog HDL-де әрекеттік моделін сипаттаудың құрылған алгоритмін тексеру A санының және P модулінің әртүрлі мәндері үшін уақыт диаграммаларын қолдану арқылы жүзеге асырылды. Verilog тілінде орындалған осы модульдің жұмыс сипаттамасы Қосымша Б-да. Тестілеу әрекеттік моделінің алгоритмінің дұрыстығын көрсетті. Модульдік қысқартудың екі мысалы үшін уақыт диаграммалары 3.25 және 3.26-суреттерде көрсетілген. Түсіну және тексеру жеңілдігі үшін диаграммаларда барлық сандар ондық жүйеде берілген: келтірілген сан A_{10}, модуль $P1_{10}$ және қалған R_{10}.

$A=638_{10}=001001111110_2$ и $P1=35_{10}=100011_2$ мәні кезінде, модульді азайту операциясының нәтижесі соңғы қалдық болады $R=8_{10}=00.001000_2$ (Екі битке солға жылжу үшін ауыстыру регистрінде екі қосымша бит қажет).

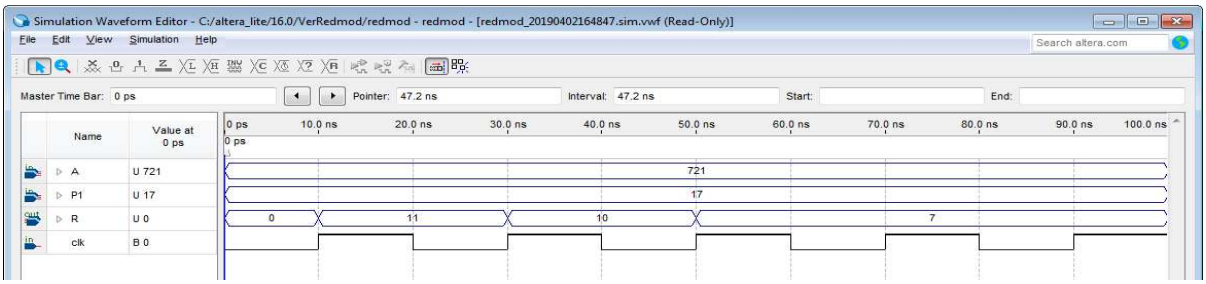
Тексеріс: $R = 638 -]\frac{638}{35}[35 = 8_{10} = 1000_2$.

$A=721_{10}=001011010001_2$ және $P1=17_{10}=010001_2$ мәні кезінде, сандар модулін азайтудың соңғы нәтижесі қалдық болып табылады $R=8_{10}=00.001000_2$.

Тексеріс: $R = 721 -]\frac{721}{17}[17 = 7_{10} = 0111_2$.

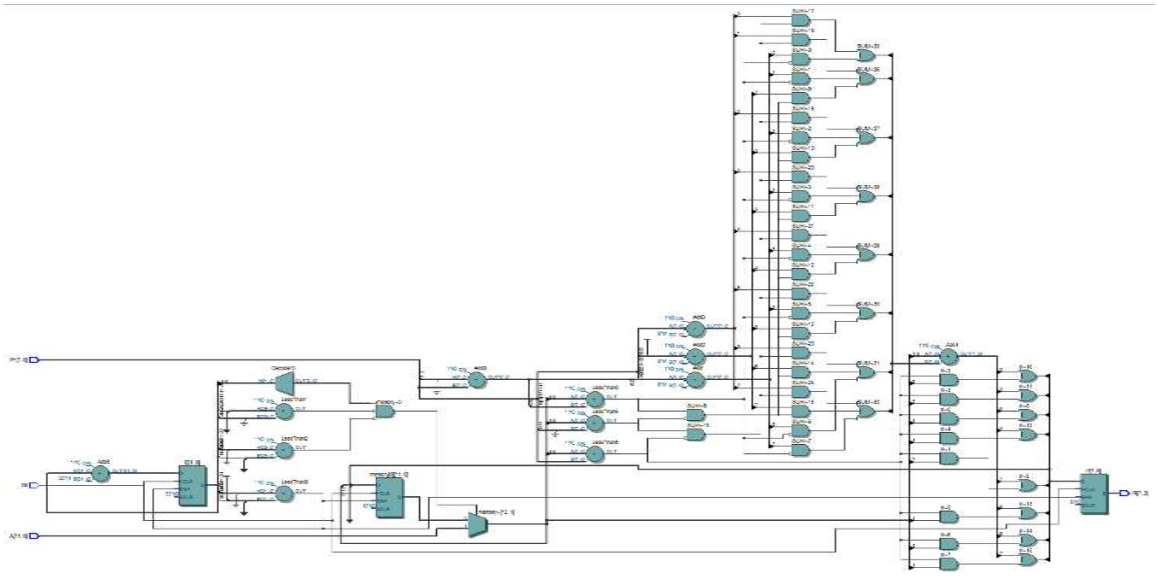


Сурет 3.25 – Келтірілген санның мәні A = 638 және модулі P1 = 35 болатын уақыт диаграммалары



Сурет 3.26 – A=721 санының мәні және P1=17 модулі келтірілетін уақыт диаграммалары

RTL Viewer утилитасы жұмыс алгоритмі Verilog HDL-те сипатталған құрылғының логикалық орындалуын графикалық түрде көрсетуге мүмкіндік береді. 3.27-суретте әрекеттік моделінен алынған регистрді беру деңгейіндегі модульдік редуктордың құрылымдық сұлбасы (RTL көрінісі, регистрді тасымалдау қабаты) көрсетілген: енгізу/шығару түйреуіштері, бес қосқыш және алты салыстыру схемасы, кіріс және шығыс регистрлері бар. Іске асыру төмен бюджеттік ПЛИС Altera Cyclone VE 5CEBA4F23C7-ге бағытталған.

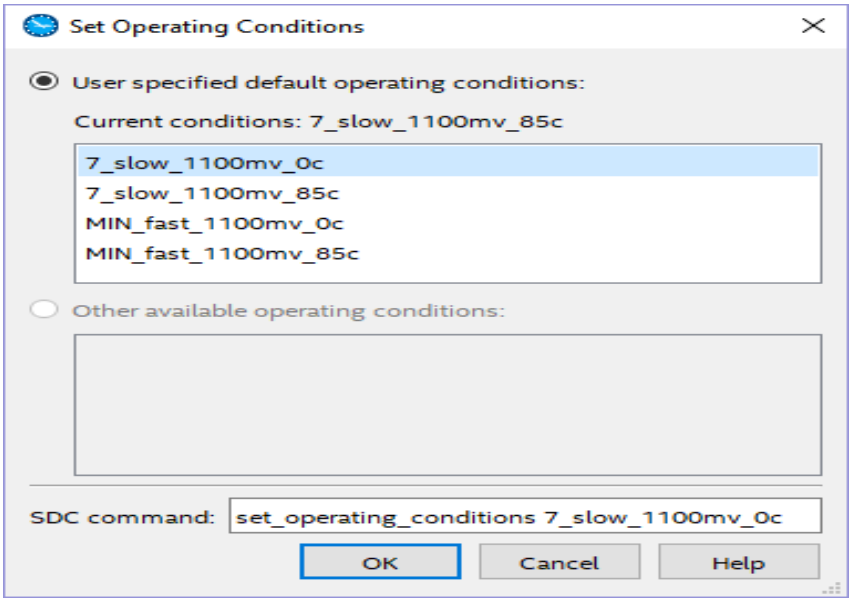


Сурет 3.27 – Құрылғының әрекеттік моделінің RTL көрінісі

Сандарды жылдам қысқартуға арналған модульдің негізгі және әрекеттік үлгілерінің өнімділігін талдау Quartus Prime TimeQuest уақыт анализаторының көмегімен жүзеге асырылды. Жобаның дұрыс жұмыс істеуі үшін сақталуы қажет уақыт қатынастары анықталды және сигналдың өту уақыты жобаның тұрақты жұмыс істеуі үшін қажетті уақытпен салыстырылды.

Кешенді статикалық уақыт талдауы барлық сигнал жолдарын талдауды қамтиды. TimeQuest уақыт анализаторы деректерді алуға кететін уақытты, деректердің келген уақытын және сағаттың келген уақытын көрсетеді. Оның көмегімен сіз схеманың функционалдығын тексере аласыз және мүмкін уақытты бұзуды анықтай аласыз.

Логикалық тізбектегі уақытты талдау схеманың сенімді жұмыс істеу шарттарын анықтау үшін орындалады. Бұл шарттар тізбек дұрыс шығысты беретін максималды тактілік жиілікті (FMAX) қамтиды. Жобадағы барлық маңызды жолдар үшін уақыт есептері жасалды. Көпжақты талдау конструкцияны (негізгі және әрекеттік үлгілері) құрылымның статикалық уақыттық талдауын орындау кезінде әртүрлі жұмыс жағдайларында, кернеудің, жылдамдықтың және температураның өзгеруінде сынақтан өткізуге мүмкіндік берді (3.28-сурет).



Сурет 3.28 – Көпжақты талдау үшін жұмыс жағдайларын таңдау

Тізбектегі максималды кідіріс ең ұзақ кезеңді және сәйкесінше құрылғының максималды жиілігін (FMAX) анықтайтын критикалық жолға сәйкес келеді.

3.29-суретте әртүрлі жұмыс жағдайларындағы негізгі модельдің максималды тактілік жиілігін анықтау үшін талдау нәтижесінің фрагменті көрсетілген. 3.30-суретте әртүрлі жұмыс жағдайларындағы әрекеттік моделінің максималды тактілік жиілігін анықтау үшін талдау нәтижесінің фрагменті көрсетілген.



Slow 1100mV 85C Model

	Fmax	Restricted Fmax	Clock Name	Note
1	27.43 MHz	27.43 MHz	Start	
2	30.8 MHz	30.8 MHz	Reset	
3	33.25 MHz	33.25 MHz	Clk	

Slow 1100mV 0C Model

	Fmax	Restricted Fmax	Clock Name	Note
1	27.6 MHz	27.6 MHz	Start	
2	30.83 MHz	30.83 MHz	Reset	
3	32.91 MHz	32.91 MHz	Clk	

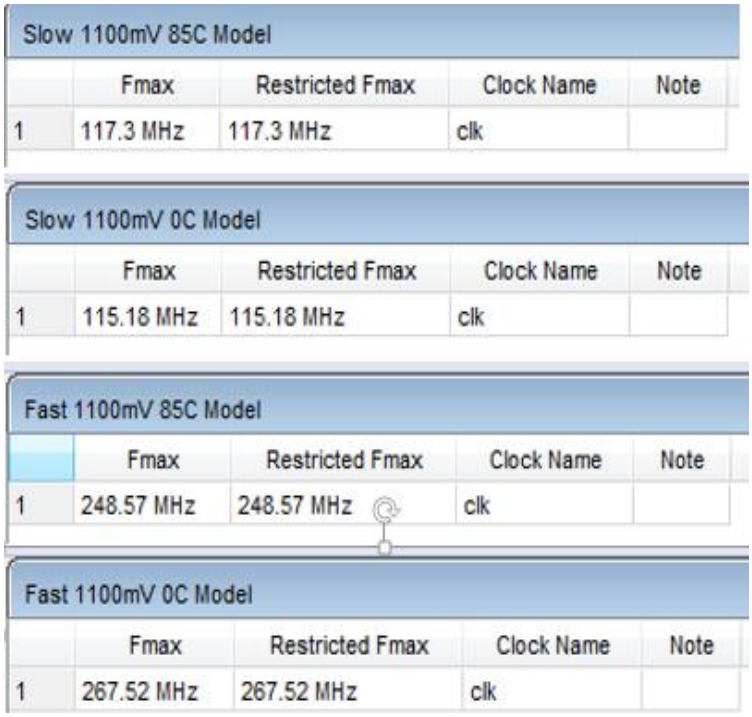
Fast 1100mV 85C Model

	Fmax	Restricted Fmax	Clock Name	Note
1	52.21 MHz	52.21 MHz	Start	
2	63.0 MHz	63.0 MHz	Clk	
3	66.54 MHz	66.54 MHz	Reset	

Fast 1100mV 0C Model

	Fmax	Restricted Fmax	Clock Name	Note
1	56.54 MHz	56.54 MHz	Start	
2	68.77 MHz	68.77 MHz	Clk	
3	72.53 MHz	72.53 MHz	Reset	

Сурет 3.29 – Негізгі модель үшін модуль сандарын жылдам азайту үшін құрылғының әртүрлі жұмыс жағдайларында максималды тактілік жиілігін анықтау



Slow 1100mV 85C Model

	Fmax	Restricted Fmax	Clock Name	Note
1	117.3 MHz	117.3 MHz	clk	

Slow 1100mV 0C Model

	Fmax	Restricted Fmax	Clock Name	Note
1	115.18 MHz	115.18 MHz	clk	

Fast 1100mV 85C Model

	Fmax	Restricted Fmax	Clock Name	Note
1	248.57 MHz	248.57 MHz	clk	

Fast 1100mV 0C Model

	Fmax	Restricted Fmax	Clock Name	Note
1	267.52 MHz	267.52 MHz	clk	

Сурет 3.30 – Әрекеттік моделі үшін модуль сандарын жылдам азайту үшін құрылғының әртүрлі жұмыс жағдайларында максималды тактілік жиілігін анықтау

Компилятор есебі жобаны және оның құрамдас бөліктерін өңдеу нәтижелері туралы толық ақпаратты қамтиды. Құрастырылған жобалардың аппараттық шығындары туралы толық ақпаратты қамтитын жиынтық есептерден алынған негізгі сипаттамалар 3.2-кестеде берілген.

Кесте 3.2. ПЛИС пайдаланатын ресурстар

Ресурс атауы	**Негізгі модель**	**Мінез-құлық үлгісі**	**ПЛИС ресурстарының жалпы көлемі**
Adaptive Logic Module (ALM)	72	74	18480
Registers	22	65	73920
I/O Pins	48	29	224
Logic Array Block (LAB)	12	11	1848

3.2-кесте Altera ПЛИС - 5CEBA4F23C7 жүйесінде қанша ортақ ресурстар (ALM, регистрлер, енгізу/шығару түйреуіштері, LAB) қолжетімді және 6-биттік екілік модуль арқылы 12-биттік екілік азайтуды жүзеге асыру үшін қанша ресурстар қажет екені көрсетілген.

Кесте біріккен бейімделген іздеу кестесінің (ALUT) деректерін көрсетпейді, себебі, ALUTS - ALM аппараттық логикалық құрылымдары және тәуелсіз ПЛИС ресурсы емес.

Компилятор есебіндегі ALUT туралы ақпарат: принциптік үлгі үшін – 116, әрекеттік үлгісі үшін – 122. Компилятор есебінде шығыс деректерінің орташа тармақталу коэффициенті туралы да ақпарат бар: принциптік модель үшін – 2,55, әрекеттік моделі үшін – 2.56.

Уақыт сипаттамаларын талдау құрылғының жұмыс жағдайына қарамастан әрекеттік модель үлгісі негізгі модельге қарағанда жылдамырақ екенін көрсетеді. Негізгі үлгі үшін де, әрекеттік модель үлгісі үшін де құрылғы ең нашар жағдайларда жұмыс істеген кездегі шекті максималды жиілік (кернеу 1100 мВ және температура 85) қалыпты жағдайда құрылғының шекті максималды тактілік жиілігінен дерлік екі есе төмен. Сонымен қатар, қалыпты жұмыс жағдайында негізгі модель үшін максималды тактілік жиілік FMAX = 68,77 МГц, әрекеттік үлгісі үшін FMAX = 267,52 МГц, бұл қолданыстағы арнайы RSA процессорларының тактілік жиілігінен жоғары (5 МГц-тен 30 МГц-ке дейін) VLSI құрылғыларының өте кең ауқымды интеграциясымен жүзеге асырылады.

Пайдаланылған және қолда бар ресурстарды талдау n = 6 үшін ұсынылған жоғары жылдамдықты модульдік қысқарту құрылғысын іске асыру үшін принциптік үлгіде де, әрекеттік үлгісінде де төмен бюджеттік ресурстардың 0,6% -дан аспайтынын көрсетеді. ПЛИС Cyclone VE 5CEBA4F23C7 пайдаланылды. Салыстырмалы талдау көрсеткендей, схеманың графикалық сипаттамасында (негізгі үлгі) жоғары жылдамдықты модульдік қысқарту құрылғысының аппараттық құралдары 10,5% аз болады, өйткені құрылғы дизайнері жобалау кезеңінде қажетті ресурстарды өзі таңдайды. Схемаларды сипаттаудың мәтіндік әдістерін пайдаланған кезде, АЖЖ ресурстарды берілген логикаға сәйкес дербес бөледі, бұл ресурстарды оңтайлы емес тұтынуға әкеледі.

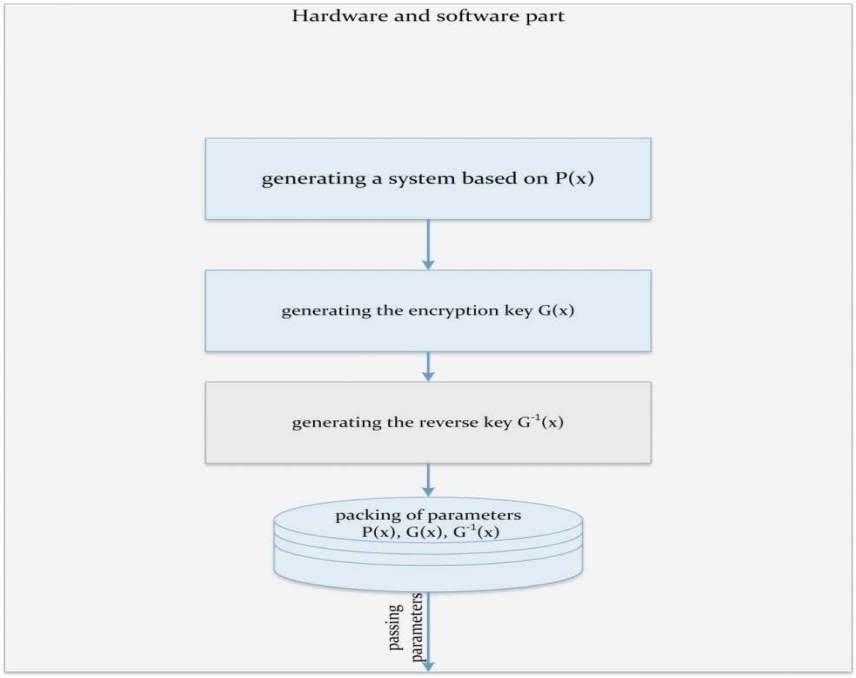
3.4 ПЛИС негізінде деректерді шифрлеу үшін бағдарламалық және аппараттық қамтамасыз ету

Заманауи қауіпсіздік протоколдары алгоритмге тәуелсіз бола бастаған сайын, криптографиялық алгоритмдерге қатысты икемділіктің жоғары дәрежесі қажет. Жоғары икемділікті дәстүрлі аппараттық құралдардың жылдамдығымен және физикалық қауіпсіздігімен біріктіретін перспективалы шешім ПЛИС сияқты қайта конфигурацияланатын құрылғыларда криптографиялық алгоритмдерді енгізу болып табылады [111].

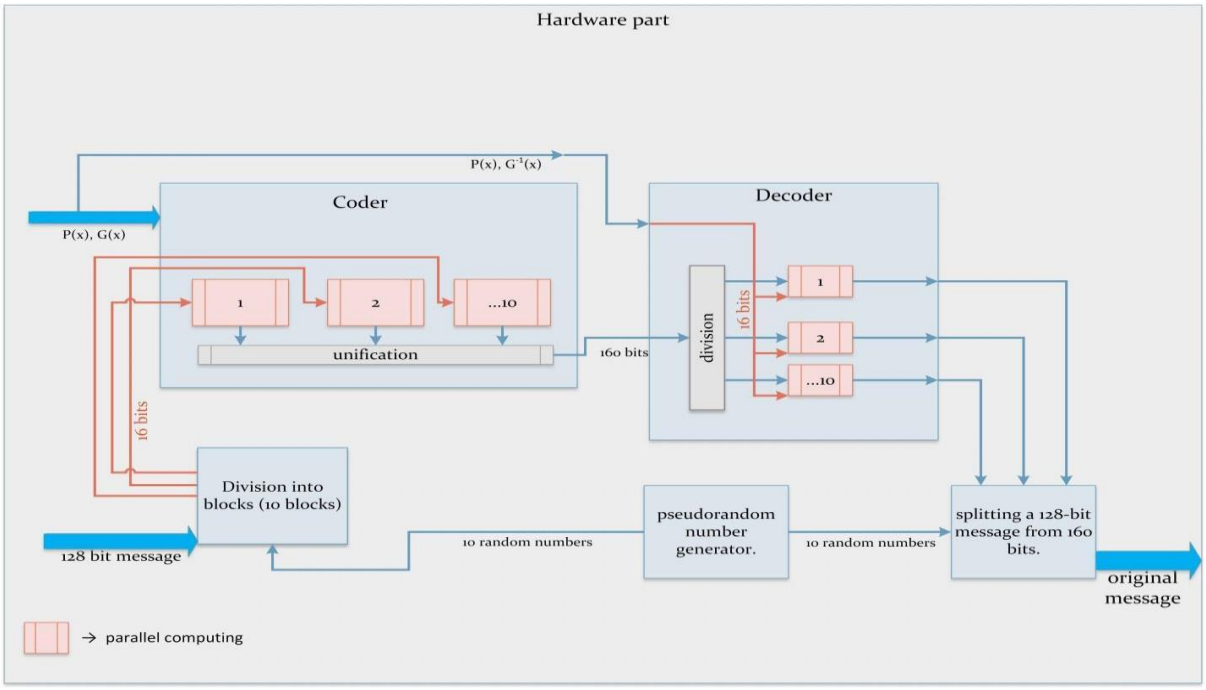
Xilinx Nexys 4 DDR әзірлеу тақтасы, толық пайдалануға дайын ПЛИС негізіндегі сандық схемаларды әзірлеу платформасы жобалау және тестілеу үшін пайдаланылды. Nexys 4 DDR платформасында үлкен және жоғары өнімді ПЛИС (Xilinx бөлігінің нөмірі XC7A100T-1CSG324C), кең сыртқы жады және кіріспе комбинациялық схемалардан бастап қуатты ендірілген процессорларға дейінгі конструкцияларды орналастыру үшін әртүрлі порттар бар.

Деректерді шифрлеу және шифрды шешу алгоритмі Verilog және VHDL аппараттық сипаттама тілдерін пайдалана отырып, Xilinx ISE Design Suite 14.4 әзірлеу ортасында әзірленді және сынақтан өтті. Бұл жұмыста көбейткіштің ең аз мәнді цифрларын талдау арқылы келтірілмейтін көпмүшелердің модулі бойынша көпмүшелерді көбейту қарастырылды.

Шифрлеу және шифрды шешу алгоритмі ISE біріктірілген ISim толық жұмыс істейтін HDL симуляторында сыналған. Шифрлеу құрылғысының жалпы құрылымдық схемасы 3.31-суретте көрсетілген. Суретте p(x) функциясы негізінде жүйені құрудан бастап криптографиялық жүйені дамытудың блок-схемасы көрсетілген. Бұл процесс G(x) шифрлау кілтін және оған сәйкес келетін G^(-1)(x) кері кілтін құруды қамтиды. Кілттер пайда болғаннан кейін P(x), G(x), және G^(-1)(x) параметрлерін мәліметтер базасына орап, осы параметрлерді берумен аяқталады.



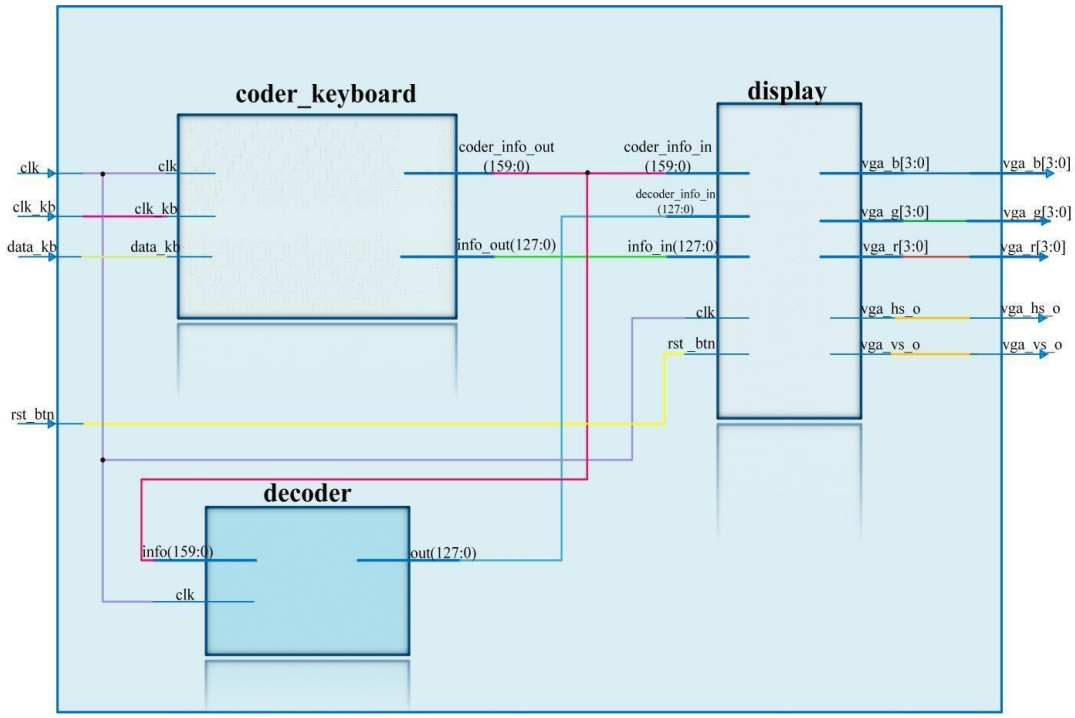
а)



б)

Сурет 3.31 – Шифрлеу құрылғысының жалпы құрылымдық схемасы
а) Аппараттық және бағдарламалық қамтамасыз ету бөлімі; б) аппараттық бөлім

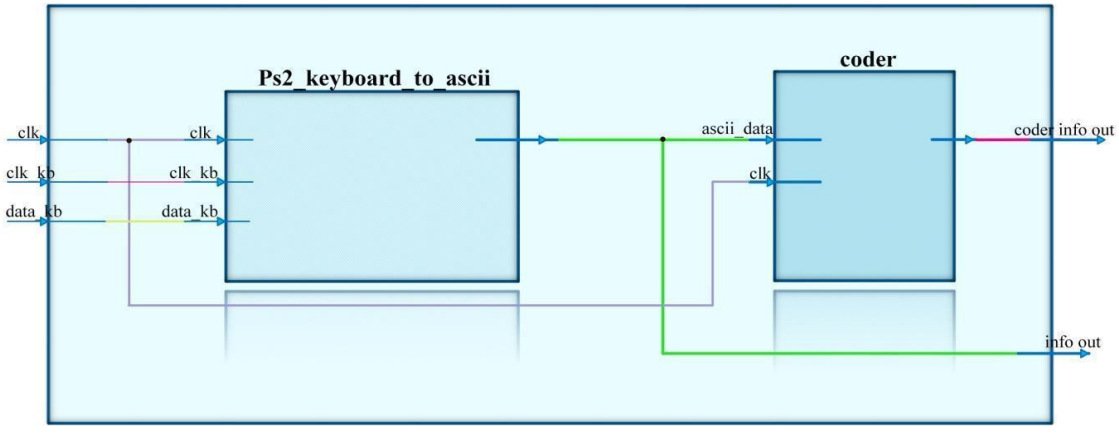
Негізгі деректерді шифрлеу және дешифрлеу блогының RTL диаграммасы 3.32-суретте көрсетілген.



Сурет 3.32 – Шифрлеуды және шифрды шешуді жүзеге асыруға арналған RTL схемасы

Негізгі блок үш негізгі қосалқы блоктан тұрады, атап айтқанда coder keyboard, декодер дешифраторы және дисплей. Coder keyboard қосалқы блогы ашық мәтінді немесе өңделмеген ақпаратты жасау және жасалған деректерді шифрлеу сияқты маңызды тапсырманы орындайды. Көмекші блок ретінде әрекет ететін декодер қосалқы блогы деректерді шифрдан шығару үшін қолданылады. Сол сияқты, дисплейдің қосалқы блогы деректерді шифрлеу және шифрды шешу процесін визуализациялау үшін пайдаланылады. Негізгі блок 4 кіріс және 5 шығыспен жабдықталған. Бірінші кіріс, clk, блок жұмысы үшін сілтеме ретінде қызмет ететін 100 МГц тактілік сигнал болып табылады. Clk_kb және data_kb кірістері сканерлеу кодын пернетақтадан оқуға жауапты. rst_btn кірісі мониторда көрсетілген ақпаратты қалпына келтіру үшін пайдаланылады. vga_b, vga_g және vga_r шығыстары мониторды оқшаулауға жауапты. Көлденең және тік синхрондау сигналдары vga_hs_o және vga_vs_o монитордағы әрбір пикселдің реңктерін беру үшін пайдаланылады.

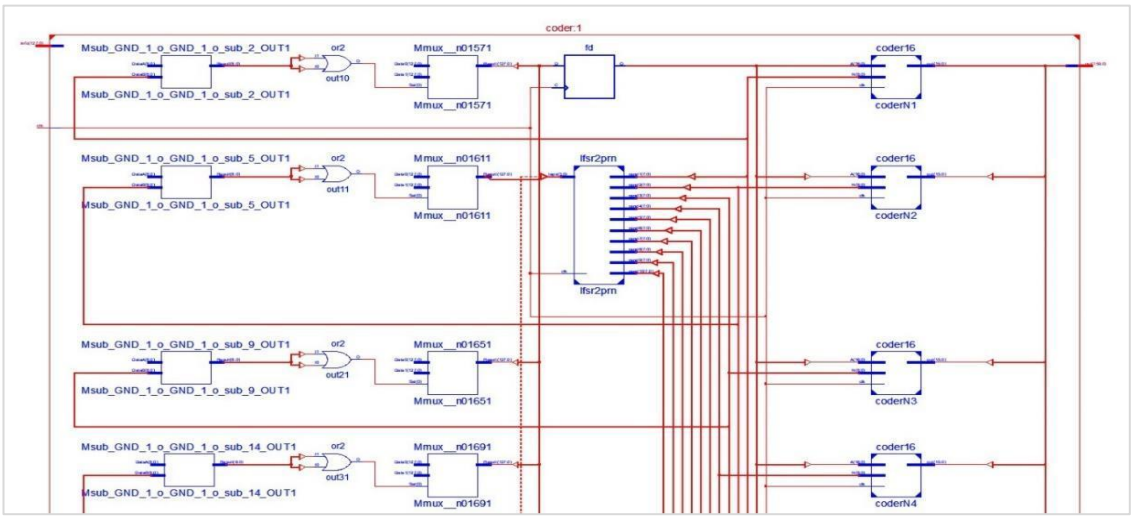
3.33-суретте буферге деректерді енгізу және буферден деректерді шифрлеу үшін негізгі coder_keyboard ішкі блогының диаграммасы көрсетілген. Негізгі coder_keyboard ішкі бөлігі бірнеше бөліктерден тұрады: кодер, ps2_keyboard_to_ascii және буферленген деректерді генерациялау. Пернетақтадан деректер сканерлеу кодын ascii кодына түрлендіретін ps2_keyboard_to_ascii мекенжайына өтеді. Ашық мәтінді жасау USB пернетақтасынан деректерді сканерлеу арқылы орындалады. Деректер 128-биттік немесе 16-байттық буферде сақталады. Егер буфер пернетақтадағы деректермен толтырылмаса, ол нөлдермен толтырылады. Әрбір байт әрбір пернетақта таңбасына сәйкес келеді, яғни. максималды буфер өлшемі 16 таңба. Бастапқы ақпараттың өлшемін өзгертуге болады. Пернетақта таңбасы басылғанда, (ascii_new) ps2_keyboard_to_ascii шығысы басу сигналын жасайды. Бұл сигнал қабылданған кезде деректер буфері ascii_data шығысындағы деректерді қосады. Enter пернесін басу ақпаратты шифрлеу үшін буферден кодтау блогына жібереді. BackSpace пернесі деректерді әр басқан сайын буфер таңбасынан таңба бойынша тазартады. VHDL тілінде орындалған және scan кодын ASCII-ге түрлендіруге арналған бағдарламалық код Қосымша В-дан табуға болады.



Сурет 3.33 – Негізгі шифрлеу қосалқы блогының RTL схемасы

Пернетақта таңбасы басылғанда, ps2_keyboard_to_ascii блогының шығысы (ascii_new) басу сигналын жасайды. Бұл сигнал келгенде, деректер буфері ascii_data шығысындағы деректерді қосады. Егер Enter таңбасын бассақ, онда буфер ақпараты деректерді шифрлеу үшін кодтау блогына жіберіледі. BackSpace пернесі деректерді әр басқан сайын буфер таңбасынан таңба бойынша тазартады.

Кодер блогы деректерді шифрлеу үшін буферден 128 бит ақпарат алады. Кодер параллельді шифрлайтын 10 кодер 16 ішкі блоктарынан тұрады (3.34-сурет). Әрбір кодер16 қосалқы блогы үшін ең үлкен кіріс деректер өлшемі 16 бит. Сондықтан бастапқы деректерді 10 бөлікке бөлу қажет, олардың әрқайсысы сәйкес кодер16 қосалқы блогымен өңделеді.

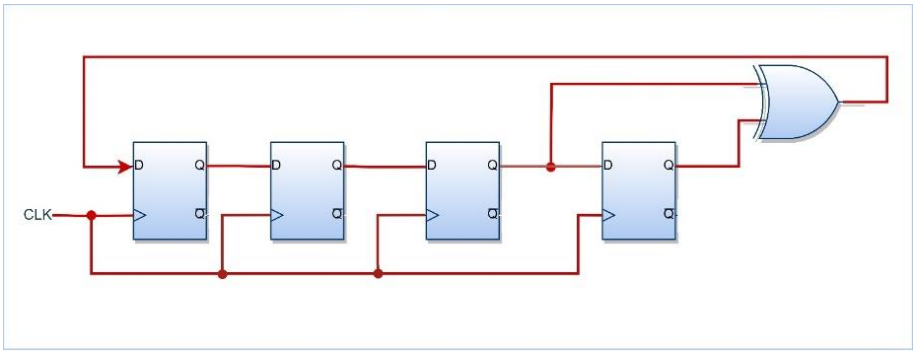


Сурет 3.34 – Енгізілген деректерді бөлу функциясы бар шифрлеу блогы

Бұған қоса, әрбір coder16 қосалқы блогы бастапқы деректердің өз бөлігін өңдегеннен кейін нәтижелер соңғы шифрланған хабарламаны қалыптастыру үшін біріктіріледі.

Әрбір қосалқы блок өз бөлігін дұрыс және қатесіз өңдеуі үшін деректердің 10 бөлікке дұрыс бөлінгеніне көз жеткізу маңызды. Сондай-ақ, деректерді жоғалтпай соңғы шифрланған хабарды қалыптастыру үшін әрбір ішкі блоктың нәтижелері дұрыс біріктірілгеніне көз жеткізу керек. Бұл деректердің тұтастығы мен құпиялылығы сақталуы қажет желі арқылы деректерді тасымалдау немесе деректерді дискіде сақтау кезінде әсіресе маңызды болуы мүмкін.

Кодер блогы псевдокездейсоқ сандарды генерациялау арқылы 128 биттік деректерден 10 шифрлеу қосалқы блоктарын жасайды. Псевдокездейсоқ сандар генераторы 10 санды жасайды және барлық сандардың қосындысы 128-ге тең болуы керек. Псевдокездейсоқ сандарды генерациялау сызықтық кері байланыстың ауысу регистрінің алгоритмі бойынша жүзеге асырылады. Бұл алгоритмнің схемасы 3.35-суретте көрсетілген.



Сурет 3.35 – Сызықтық кері байланысы бар ауысым регистрінің схемасы

Схема XOR шлюзінен және ауысым регистрінен тұрады. Сызықтық кері байланысты ауыстыру регистрінің алгоритмі 1-ден 15-ке дейінгі санды құра алады және 15 есептен кейін қайталанады. Бұл жағдайда сандар бір есепте қайталанбайды. Мысалы, 1, 8, 4, 2, 9, 12, 6, 11, 5,10, 13, 14, 15, 7, 3, бірақ 10 санның қосындысы 128-ге тең болмайды. Сондықтан өзгерту керек. кездейсоқ сандар диапазоны 6 модулін алу және 11 қосу арқылы. Нәтижесінде 11-ден 16-ға дейінгі аралықта 9 сан, ал алдыңғы сандардың қосындысынан 128 айырмашылығынан онынші рет кездейсоқ түрде жасалады. Кіріс деректерін құруға, сызықтық кері байланысты ауыстыру регистрін жүзеге асыруға және псевдокездейсоқ сандарды құру үшін Verilog тілінде жазылған бағдарламалық кодты Қосымша Г-дан табуға болады.

128 битті әр түрлі биттермен 10 қосалқы блокқа бөлу үшін олардың әрқайсысы кездейсоқ сандар генераторының сандарына сәйкес келетін өлшемге ие болуы керек. Егер деректер блогының өлшемі 16-дан аз болса, үлкен биттер нөлдермен толтырылады. Әр түрлі биттерге арналған шифрлеу кілттері мен негіз жүйелері криптожүйенің бағдарламалық бөлігі арқылы жасалды. Сипатталған шифрлеу процесін келесідей формуламен көруге болады:

$$L(x) = [A(x) \cdot G(x)] mod P(x), \tag{3.6}$$

мұндағы, $A(x)$ – бастапқы деректер, $G(x)$ – шифрлеу кілті және $P(x)$ – базалар жүйесі.

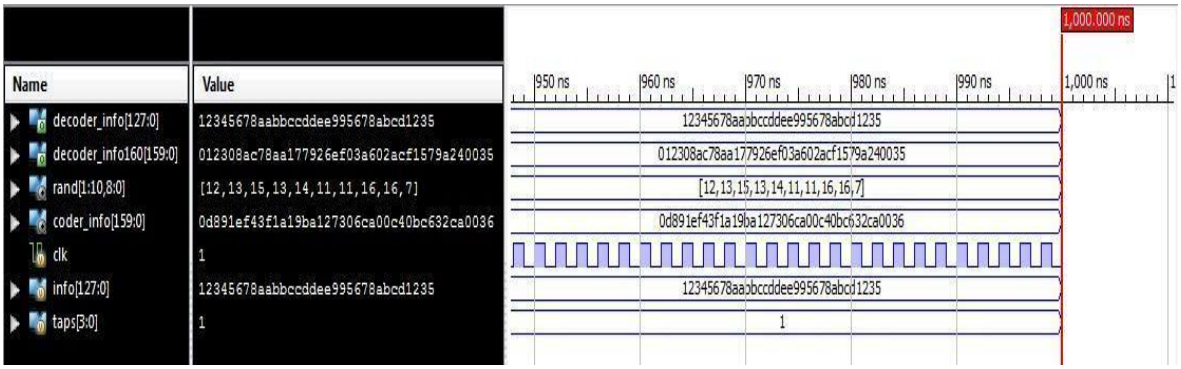
10 қосалқы блоктар параллель бөлінген деректерді шифрлайды. Шифрлеудан кейін әрбір ішкі блоктан алынған деректер каскадты түрде қосылып, 160 биттік деректерді құрайды. Кілттер деректерді шифрлеу үшін қолданылады. Шифрлеудан кейін әр блоктан алынған мәліметтер каскадты түрде қосылады. Бұл өлшемі 160 бит болатын $L(x)$ түзілуіне әкеледі. 128 битті 10 ішкі блокқа бөлу үшін, сондай-ақ шифрлеу кілті $G(x)$ мен негіз жүйесін $P(x)$ және шифрлеуға арналған бағдарламалық коды Қосымша Ғ-да да келтірілген.

decoder блогы $L(x)$ 160 бит шифрланған деректерді 16 битке бөледі және параллельді шифрды декодтайды. Кері кілтті $G^{-1}(x)$ декодтау кезінде криптожүйелердің бағдарламалық $P(x)$ бөлігінің көмегімен әртүрлі биттерге арналған кері кілт және негізгі жүйелер жасалады. Бұл параметрлер

жинақталады және ПЛИС жадына жазылады. Деректерді декодтау кезінде біз он 16-биттік қосалқы блоктарды аламыз. Қажетті бөлікті қиып алу керек және өлшемі 128 бит болатын бастапқы деректерді алу керек. *decoder* блогында шифрлеуға сәйкес 128 биттік кіріс деректерінің сандарды бөлу ретін қайталайтын псевдокездейсоқ сандарды генерациялау қосалқы блогы бар. Нәтижесінде біз бастапқы ақпаратты аламыз.

Шифрлеу және дешифрлеу нәтижелері 3.36 суретте уақыт диаграммалар түрінде көрсетілген. 3.36(а)-суретте info кірісіне блоктарда кездейсоқ бөлінген шифрлеуға жататын деректер беріледі: 12, 13, 15, 13, 14, 11, 11, 16, 16, 7 (бастапқы 128 биттік шифрланатын мәтіннің ішкі блоктарының ұзындығы). Және де осы блоктарда кіріс деректердің көпмүшелерін көбейтуге қатысатын бірегей төмендетілмейтін көпмүшелер мен тура кілттер, модульді келтірілмейтін көпмүшелердің кілттері кездейсоқ бекітілген. 3.36 (а) суретінен көрініп тұрғандай, шифрланған мән coder_info шығысында: 0d891ef43f1a19ba127306ca00c40bc632ca0036 он алтылық санауда қалыптасады.

Бастапқы 128 биттік шифрланатын мәтіннің ішкі блоктарының ұзындығын өзгерту кездейсоқ реттілік (13, 14, 11, 11, 16, 16, 15, 12, 13, 7), coder_info шығысында біз он алтылық санау жүйесінде шифрланған деректерді аламыз: 1f9e24ff02a005f7427a4c0d32ba01ff1b590036 (3.36 (б)-сурет). Шифрды шешудің екі жағдайында да біз decoder_info шығысында аламыз.



а)

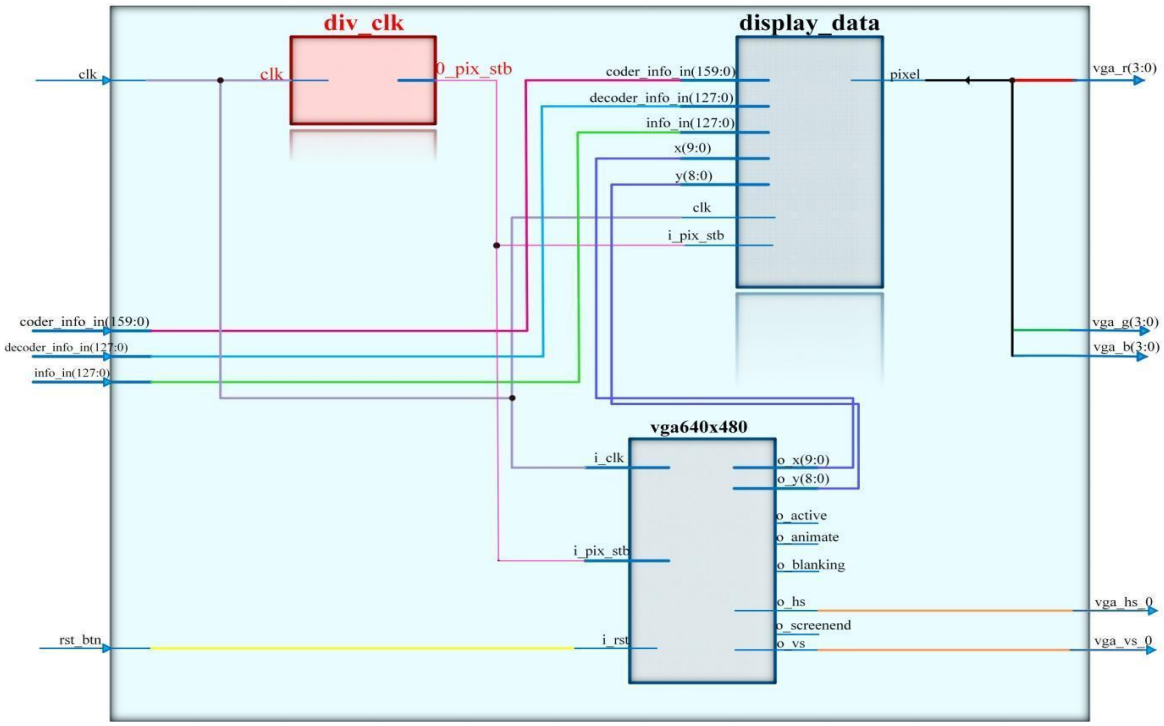


б)

Сурет 3.36 – Түпнұсқа ақпаратты бөлу нөмірлерінің басқа тізбегі үшін шифрлеу және дешифрлеудің уақыттық диаграммасы

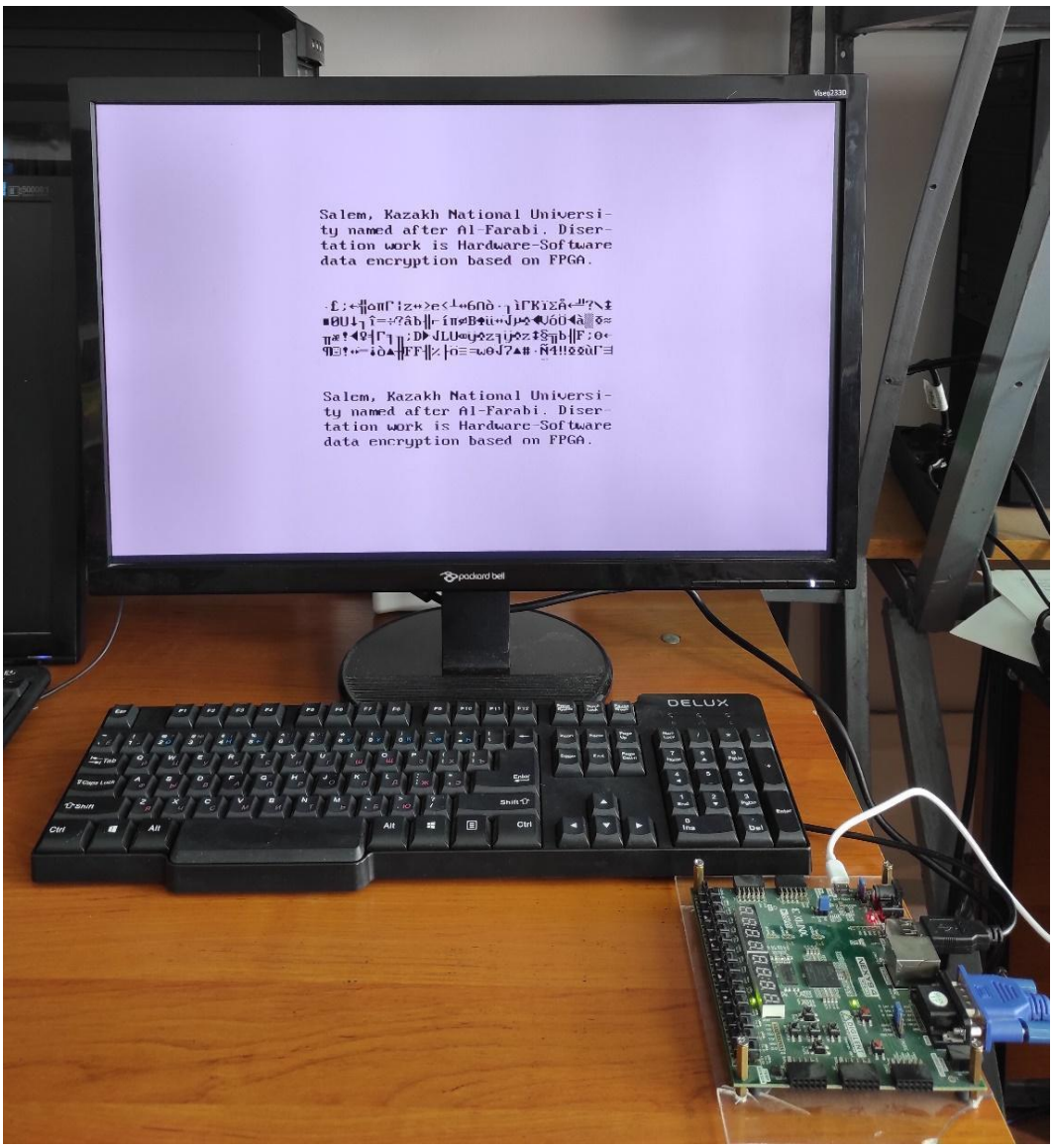
Төрт блок үшін аппараттық-бағдарламалық шифрлеу кешенін тексеру үшін латын әріптерімен төрт жолдан тұратын мәтін алынды (3.36(а)-сурет), екінші жағдайда (3.36 (б)-сурет) 128-биттік ақпарат параллель кодталады. 16 бит бойымен бірдей ұзындықтағы 10 қосалқы блокта, олардың ішіндегі ақпарат кездейсоқ сандар генераторының көмегімен кездейсоқ таратылады. Қабылдаушы жағында ақпарат 16 блоктан жиналады және қажет емес, бұрын толтырылған биттер алынып тасталады.

Визуализация блогы үш ішкі блоктан тұрады: белгілі бір жиіліктегі тактілік сигналды генерациялау, VGA үшін драйвер және экранда көрсету үшін символды генерациялау (3.37-сурет). 128 биттік деректерді қалпына келтіру функциясы бар шифрды декодтау блогының бағдарламалық кодының үзіндісі, 160 битті он 16 битке бөлу функциясы бар шифрды ашу блогының бағдарламалық кодының үзіндісі және Визуализацияу блогының коды Қосымша Д-да берілген.



Сурет 3.37 – Дисплейдегі ақпаратты визуализация блогы

VGA драйвер блогы монитордың тік және көлденең орналасуына байланысты синхронды сигналдарды жасайды. Блокта шифрланған, шифры шешілген және өңделмеген деректерге сәйкес келетін coder_info_in, decoder_info_in және info_in кірістері бар. Шифрлеу мен шифрды шешудің дұрыс жұмыс істеуі оны Artix 7 тобының Xilinx ПЛИС негізіндегі мониторда көрсету арқылы тексерілді (3.38-сурет).



Сурет 3.38 – Artix 7 тобының Xilinx ПЛИС негізіндегі шифрлеу және дешифрлеу алгоритмінің аппараттық орындалуы

Жұмыста ПЛИС-дағы шифрлеу блогының прототипі көпмүшелерді модульге келтірілмейтін көпмүшелерді көбейту арқылы жүзеге асырылады. Орындалған блоктың дұрыс жұмыс істеуі бастапқы мәтінді шифрлеу және қабылданған шифрлық мәтінді шешу мысалында көрсетілген.

Шифрлеу алгоритмі Xilinx компаниясының Artix-7 ПЛИС ядросы (XC7A100T-1CSG324C) бар Nexys 4 DDR әзірлеу тақтасында жүзеге асырылды. Құрылғы осы жұмысқа жарамды оңтайлы өнімділігі үшін таңдалды. 3.3-кестеде кесінділер регистрлерінің саны, LUT кесінділерінің саны, толық пайдаланылған LUT-FF жұптарының саны, қосылған IOB саны, RAM/FIFO бірліктерінің саны, BUFG/BUFGCTRL саны және т.б. қажет. чип дизайны. 3.4-кестеде сонымен қатар максималды жиілік, ең аз кезең, clk алдында қажетті ең аз уақыт, clk кейінгі ең көп уақыт сияқты уақытты синхрондауға қатысты параметрлер туралы толық ақпарат бар.

Кесте 3.3 – ПЛИС синтезі есебі түріндегі аппараттық құрал параметрлерінің қысқаша сипаттамасы.

Device Utilization Summary			
Slice Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	8,599	126,800	6%
Number used as Flip Flops	8,599		
Number of Slice LUTs	8,761	63,400	13%
Number used as logic	5,278	63,400	8%
Number using O6 output only	3,567		
Number using O5 output only	38		
Number using O5 and O6	1,673		
Number used as Memory	2,946	19,000	15%
Number used as Shift Register	2,946		
Number using O6 output only	2,946		
Number used exclusively as route-thrus	537		
Number with same-slice register load	164		
Number with same-slice carry load	5		
Number with other load	368		
Number of occupied Slices	2,527	15,850	15%
Number of LUT Flip Flop pairs used	9,236		
Number with an unused Flip Flop	2,281	9,236	24%
Number with an unused LUT	475	9,236	5%
Number of fully used LUT-FF pairs	6,480	9,236	70%
Number of unique control sets	29		
Number of slice register sites lost to control set restrictions	119	126,800	1%
Number of bonded IOBs	26	210	12%
IOB Flip Flops	7		
Number of RAMB18E1/FIFO18E1s	2	270	1%

Number using RAMB18E1 only	2		
Number of BUFG/BUFGCTRLs	4	32	12%
Number used as BUFGs	4		
Number of OLOGICE2/OLOGICE3/OSERDESE2s	7	300	2%
Number used as OLOGICE2s	7		
Average Fanout of Non-Clock Nets	3.41		

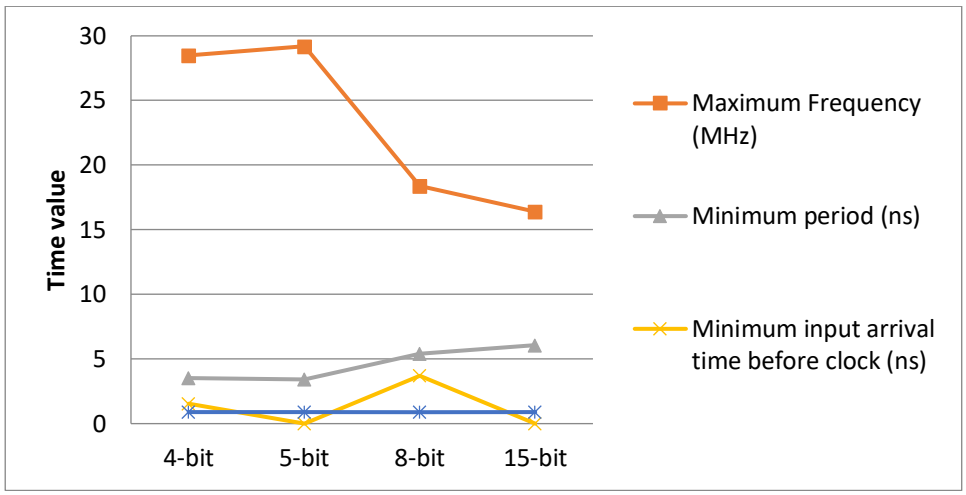
3.4-кесте. ПЛИС синтезінің есебі ретіндегі уақыт қорытындысы

Timing Summary	
Maximum Frequency	160.270MHz
Minimum period	6.239ns
Minimum input arrival time before clock	1.504ns
Maximum output required time after clock	3.585ns
Speed Grade	-1

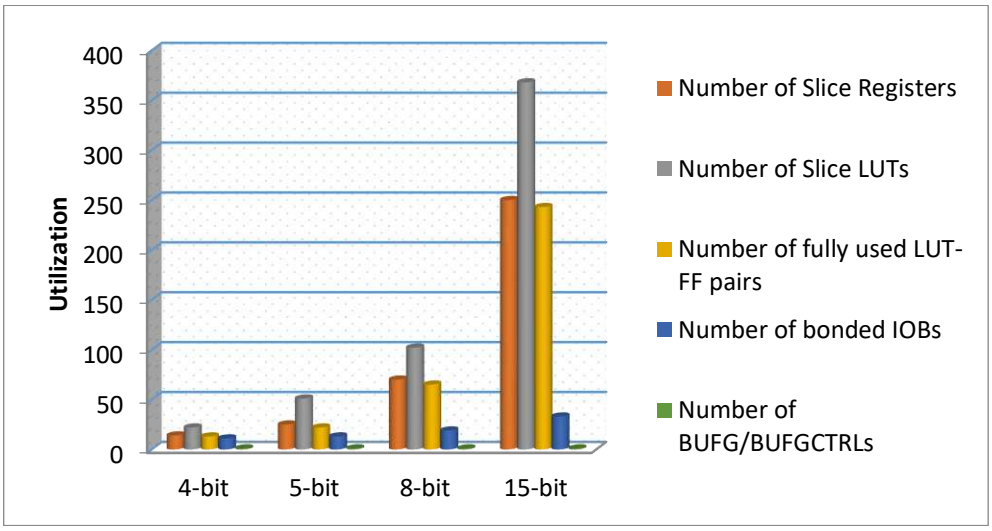
Ұсынылған ПЛИС негізіндегі деректерді шифрлеу және шифрды шешу алгоритмі ПЛИС аппараттық құралында іске асырылған криптографиялық алгоритмнің басқа деректер талдауларымен салыстырылды [112-117]. ПЛИС-та шифрлау алгоритмінің кешігу уақыты қолданылатын ПЛИС чипіне байланысты өзгеруі мүмкін. Бұл айырмашылық бірнеше факторларға байланысты болуы мүмкін:

1. Аппараттық Конфигурация;
2. Verilog кодының тиімділігі;
3. Алгоритмдердің параметрлері.

Осы факторларға байланысты бір ПЛИС құрылғысы үшін өлшенетін немесе есептелетін кідіріс уақыты басқа конфигурациясы немесе коды бар басқа құрылғыдағы кідіріс уақытына сәйкес келмеуі мүмкін. Сондықтан кез-келген кідіріс уақытының мәндері дәл емес, жуықтау ретінде қарастырылуы керек.



а)



б)

Сурет 3.39 – а) ПЛИС синтезінің есеп уақыты туралы қысқаша мәлімет б) ПЛИС аппараттық ресурстарын пайдалану

3.39 суреттің а және б суреттері әртүрлі өлшемдегі деректерді шифрлеу кезінде пайдаланылатын уақыт пен аппараттық ресурсты көрсетеді. Аппараттық құрал ресурстарын пайдалану шифр блогының ұзындығына қарай артады, пайдаланылған жадтан басқа, өйткені есептеу n-бит үшін итеративті түрде орындалады. Үлкенірек кілт өлшемі әзірленген шифрлеу және дешифрлеу чипі үшін түрлендіру циклдерінің санын көбейтеді, бұл комбинациялық жолдың кешігуін, ең аз және максималды сағат синхрондауын арттырады. Чиптің өнімділігі ПЛИС аппараттық құралы қолдайтын максималды жиілік негізінде бағаланады.

3.5 Бөлім бойынша қорытынды

Жұмыстың нәтижесінде ПЕКСЖ негізіндегі ПЛИС негізіндегі шифрлеу құрылғысы әзірленді және сынақтан өтті. Көп ағынды параллельді есептеу әдістерін және кездейсоқ сандар генераторын қолдану шифрлеу

процесін жылдамдатып қана қоймай, шифрлеу мәтінінің сенімділігін арттыратын шифрлеу құрылғысын жасауға мүмкіндік берді.

Алынған құрылғы дербес компьютердің архитектурасына біріктірілуі мүмкін немесе жоғары өнімді дербес цифрлық шифрлеу құрылғыларын жасау үшін пайдаланылуы мүмкін. Көпмүшелердің келтірілмейтіндігін тексеру, тура және кері кілттердің генераторлары, сондай-ақ матрицалық-құбырлық схемалар бойынша көпмүшелердің жоғары өнімді көбейткіштері сияқты блоктарды аппараттық іске асыру кезінде деректерді шифрлеудың жоғары жылдамдықты құрылғыларын құруға болады.

Айта кету керек, шифрлеу алгоритмдерінде ПЛИС қолдану олардың параллельді өңдеу мүмкіндіктері мен икемділігіне байланысты перспективалық әлеуетті көрсетеді.

Осылайша, әзірленген құрылғы жоғары шифрлеу жылдамдығы мен жоғары криптографиялық сенімділік қажет болатын заманауи ақпараттық қауіпсіздік жүйелерінде пайдалану үшін перспективалы болып табылады.

ҚОРЫТЫНДЫ

Осы диссертациялық жұмыстың қорытындысында ПЛИС-та аппараттық шифрлеу алгоритмін енгізу деректерді шифрлеу жылдамдығын едәуір арттырып, бір тактіде 6.239 наносекундта әсерлі өнімділікке қол жеткізген. Бұл жылдамдық телекоммуникация жүйелері, заттар интернеті және арнайы байланыс жүйелері үшін өте маңызды болып табылатын нақты уақыттағы жоғары тиімді шифрлеуды қамтамасыз етеді.

Бұл жұмыстағы зерттеу сонымен қатар шифрлеу кілттері ретінде қызмет ететін псевдокездейсоқ сандардың көзі ретінде пайдалану үшін әртүрлі хаос генераторларын талдауды қамтыды. Ресслер генераторын енгізу оның жиілігін көрсетті, бұл оны деректерді шифрлеу мақсаттары үшін жарамсыз етті. Одан айырмашылығы, Чуа генераторы деректерді шифрлеу кілттерінің сенімді көзі ретінде жарамдылығын растайтын псевдокездейсоқ әрекеттерді көрсетті.

Сонымен қатар, диссертация көпмүшелердің келтірілмейтіндігін тексеру сияқты күрделі алгоритмдік процестерді жүзеге асырудағы ПЛИС мүмкіндіктерін көрсетеді. Атап айтқанда, 16 биттік келтірілмейтіндікті тексеру сәтті жүзеге асырылды, бұл ретте өтпелі кесте логикалық ұяшықтарды пайдалану 57.07% құрады. Бұл ПЛИС арқылы шифрлеудың жоғары жылдамдығы мен сенімділігін қамтамасыз етіп қана қоймай, сонымен қатар криптографиялық жүйенің тұтастығын сақтау үшін қажетті күрделі математикалық операцияларды тиімді жүзеге асыруға мүмкіндік беретіндігін растайды.

Жалпы, осы диссертациялық жұмыстың нәтижелері қазіргі телекоммуникациялық жүйелердегі деректерді шифрлеудың тиімділігі мен қауіпсіздігін арттыру үшін жаңа перспективалар ашады. ПЛИС-те ұсынылған әдістер мен алгоритмдерді енгізу қазіргі цифрлық әлемнің қиындықтарына жауап бере алатын жоғары өнімді және сенімді шифрлеу жүйелерін құрудың практикалық шешімін ұсынады.

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1. Shi-hai Zhu. Hardware Implementation Based on FPGA of AES Encryption and Decryption System // Scholars Journal of Engineering and Technology (SJET). – Vol.2. – P. 352–357.

2. Bian D., Pan J., Wang Y. Study of Encrypted Transmission of Private Data During Network Communication: Performance Comparison of Advanced Encryption Standard and Data Encryption Standard Algorithms//JCSANDM 2022. – Vol 11, –No. 5. –P.714-726.

3. Zodpe H., Sapkal A. An efficient AES implementation using FPGA with enhanced security features //Journal of King Saud University-Engineering Sciences. – 2020. – Vol. 32, No. 2. – P. 115-122.

4. He Y., Yang N., Zhang R. Information Security Protection Technology in Industrial Internet of Things//Wireless Communications and Mobile Computing. – 2021. –Vol. 2295130. –P.1-19.

5. Lionel G. De Lazo, Dr. Pasupuleti V. Role and Importance of Cryptography Techniques in Cloud Computing//Technoarete Transactions on Internet of Things and Cloud Computing Research. –2022. Vol. 2. –No.1. – P. 27-32.

6. Geng Q., Yan H..Application of Image Encryption Algorithm for Wireless Sensor Network in the Security Analysis of Public Big Data//Wireless Communications using Embedded Microprocessors. –2023.–Vol. 6186275. –P.1-12.

7. Pérez-Resa A., Garcia-Bosque M., Sánchez-Azqueta C., Celma S. A New Method for Format Preserving Encryption in High-Data Rate Communications//IEEE Access. –2022. –Vol.8. –P.21003-21016.

8. PLEȘA M., PLEȘCA C. Privacy-Preserving Clustering: A New Approach Based on Invariant Order Encryption//Journal of Military Technology. –2020. – Vol. 3, No. 2. –P.65-72

9. Pérez-Resa A., Garcia-Bosque M., Sánchez-Azqueta C., Celma S. Physical Layer Encryption for Industrial Ethernet in Gigabit Optical Links//IEEE Transactions on Industrial Electronics. –2019. – Vol. 66. –No. 4. –P. 3287 – 3295

10. Adee R., Mouratidis H. A Dynamic Four-Step Data Security Model for Data in Cloud Computing Based on Cryptography and Steganography//Sensors.–2022. – Vol.22. – No.3. – P. 1-23.

11. Jahanbakht M., Wang X., Hanzo L., Azghadi M.. Internet of Underwater Things and Big Marine Data Analytics—A Comprehensive Survey//IEEE Communications Surveys & Tutorials. –2021. – Vol. 23. –No. 2. –P. 904 - 956

12. Mayes M. Hoobi. Efficient Hybrid Cryptography Algorithm. Journal of Southwest Jiaotong University. –2020. – Vol.55. – No. 3. –P.1-9.

13. Geng Q., Yan H. Application of Image Encryption Algorithm for Wireless Sensor Network in the Security Analysis of Public Big Data//Wireless Communications using Embedded Microprocessor. –2022. –P.1-12.

14. Abikoye Ch., Haruna A., Abubakar A., Oluwatobi N., Oluwatobi E. Modified Advanced Encryption Standard Algorithm for Information Security//Symmetry. – 2019. – Vol.1. – No.12. –P.1-16.

15. Xu Q., Sun K., Zhu C.. A visually secure asymmetric image encryption scheme based on RSA algorithm and hyperchaotic map//Physica Scripta. –2020. – Vol. 95. – No.3.

16. Cheng C., Cantu M., Kumar S. Analyzing. Computational Components of Standard Block Encryption Schemes//Journal of Computer and Communication. – 2022. – Vol.10 – No.6. – P.81-89.

17. Mohamed N., Ashraf A M. Khalaf., Sara M. Hassan. Design and Implementation of Pipelined AES Encryption System using FPGA// IEEE. – Vol.8. – No.5. – P. 2565-2571.

18. Muhajer S. SA Review of DES and AES algorithms for image Encryption//Al-Qadisiyah journal of pure science. – 2022. – Vol.27. – No 1. – P. 48-56.

19. Yu H., Kim Y. New RSA Encryption Mechanism Using One-Time Encryption Keys and Unpredictable Bio-Signal for Wireless Communication Devices//Electronics. – 2020. –Vol. 9. – No.2. – P.1-10.

20. Martin K. Everyday Cryptography:Fundamental Principles and Applications//Oxford University Press. – 2012. – P.560.

21. Cheng Z., Dai W., Li L. A High-Security Privacy Image Encryption Algorithm Based on Chaos and Double Encryption Strategy// Journal of Applied Mathematics. – 2022. –P.1-14.

22. Dong J. Design of network security optimization encryption algorithm//Proc. SPIE 12718, International Conference on Cyber Security, Artificial Intelligence, and Digital Economy. – 2023. – Vol. 1271807.

23. Xia T., Zhao Z., Li W., Fan H. , Cao C. A New Kind of Integral Cryptanalysis for the Round-reduced AES//Scientific Conference on Network, Power Systems and Computing. – 2019. – P. 6-9.

24. Fan T., Liht L, Wei Y., Pasalic E. Differential cryptanalysis of full-round ANU-II ultra-lightweight block cipher// International Journal of Distributed Sensor Networks. – 2022 – Vol. 18. – No. 9.

25. Biyashev R., Dyusenbayev D., Algazy K., Kapalova N. Algebraic cryptanalysis of block ciphers // International Conference on Wireless Communication, Network and Multimedia Engineering. Advances in Computer Science Research. Atlantis press. – 2019. – Vol. 89. – P. 129-132

26. Fan T., Liht L, Wei Y., Pasalic E.. Differential cryptanalysis of full-round ANU-II ultra-lightweight block cipher// International Journal of Distributed Sensor Networks. – 2022 – Vol.18. – No. 9. – P.1-10.

27. Liu Zh., Han Sh., Wang Q., Li W., Liu Ya, Gu D. New insights on linear cryptanalysis//Science China Information Sciences. – 2020. – Vol.63. – P.112104.

28. Petrescu I., Păvăloiu I., Drăgoi G. Digital Logic Introduction Using FPGAs// Procedia - Social and Behavioral Sciences. – 2015 – Vol.180. – P.1507 – 1513.

29. Kulanov, V. Parameterized IP Infrastructures for Fault-Tolerant FPGA-Based Systems: Development, Assessment, Case-Study / V. Kulanov, V. Kharchenko, A. Perepelitsyn // Proceedings of IEEE East-West Design & Test Symposium (EWDTS 2019). – 2009. – P. 322–325.

30. Perepelitsyn, A. FPGA Technologies in Medical Equipment: Electrical Impedance Tomography / A. Perepelitsyn, D. Shulga // Proceedings of IEEE East-West Design & Test Symposium (EWDTS 2012). – 2012. – P. 437-440.

31. Ahmed I., Linda L. Shen, V. Betz. Optimizing FPGA Logic Circuitry for Variable Voltage Supplies// IEEE Transactions on Very Large Scale Integration (VLSI) Systems. – 2020. – Vol. 28. – No. 4. – P. 890 – 903.

32. Haigang Y., Jia Zh., Jiabin S., Yu Le. Review of advanced fpga architectures and Technologies// Journal of electronics. – 2014. – Vol. 31. – No.5. – P.371-393.

33. Yu H., Kim Y. New RSA Encryption Mechanism Using One-Time Encryption Keys and Unpredictable Bio-Signal for Wireless Communication Devices// Electronics. – 2020. – Vol. 9. – №2. – P. 246.

34. Yang Ch., Wu H., Su Sh. Implementation of Encryption Algorithm and Wireless Image Transmission System on FPGA//IEEE Access. – 2019. – Vol.7. – P. 50513 – 50523.

35. Škobić, V. Hardware Modules of the RSA Algorithm [Text] / V. Škobić, B. Dokić, Ž. Ivanović // Serbian Journal of Electrical Engineering. – 2014. – Vol. 11, No. 1. – P. 121 – 131.

36. Ghayoula R., Hajlaoui E., Korkobi T., Traii M., Trabelsi H. FPGA Implementation of RSA//World Academy of Science, Engineering and Technology.– 2008. – Vol. 2. – No. 8. – P. 848 – 852.

37. Subhi R., Zeebaree M. DES encryption and decryption algorithm implementation based on FPGA// Indonesian Journal of Electrical Engineering and Computer Science. – 2020. – Vol.18. – No.2. – P. 774-781.

38. Silva D., Oliveira D., Batista G. Design of DES encryption algorithm as bundleddata asynchronous pipeline using FPGA// Rev. Bras. Apl. Vac., Campinas. – 2020. – Vol. 39. – No. 3. – P. 260-270.

39. Gurmail, S. High Throughput AES Encryption Algorithm Implementation on FPGA / S. Gurmail,M. Rajesh // International Journal of Computer Technology and Applications. – 2011. – P. 1993-1996.

40. Pirzada S., Murtaza A., Xu T., Jianwei L.The Compatibility Analysis of AES Algorithm for Design Portability on FPGA//IJCTE. – 2019. – Vol.11. – No. 6. – P. 112-115.

41. Visconti P., Velazquez R., Capoccia S., Fazio R. High-performance AES-128 algorithm implementation by FPGA-based SoC for 5G communications//International Journal of Electrical and Computer Engineering (IJECE). – 2021. – Vol.11. – No.5. – P. 4221-4232.

42. Heinrich E., Staamann S., Joost R., Salomon R. Comparison of FPGA-based Implementation Alternatives for Complex Algorithms in Networked Embedded Systems - the Encryption Example//Emerging Technologies and Factory Automation (ETFA 2008). – 2008.– P. 1449-1456.

43. Yang Ch., Chien Yu-Sheng. FPGA Implementation and Design of a Hybrid Chaos-AES Color Image Encryption Algorithm//Symmetry. – 2020. – Vol. 12. – No.2. – P. 189.

44. Закон Республики Казахстан «О государственных секретах»: утв. 15 марта 1999 г. № 349-I.

45. Закон Республики Казахстан «Об электронном документе и электронной цифровой подписи». / Утв. 7 января 2003 г. № 370-II.

46. Постановление Правительства Республики Казахстан. Правила электронного документооборота государственных органов Республики Казахстан / Утв. 17 апреля 2004 г. № 430.

47. СТ РК 1073-2007. Средства криптографической защиты информации / Общие технические требования. / Утв. 2009. 01.01. – Астана: 2009.

48. Концепции информационной безопасности Республики Казахстан/ Утв. 10 октября 2006 года N 199.

49. Киберқауіпсіздік тұжырымдамасы («Қазақстанның киберқалқаны») / Қазақстан Республикасы Үкіметінің 2017 жылғы 30 маусымдағы № 407 қаулысымен бекітілген.

50. Akushsky I.Ya., Yuditsky D.I. Machine arithmetic in residual classes. – M.: Soviet radio. – 1968. – P. 439

51. Amerbaev V.M., Biyashev R.G. Interpolation and error-correcting codes // Coding theory and information modeling. –1973. – P. 51-64.

52. Biyashev R.G. Development and research of methods for end-to-end reliability improvement in data exchange systems of distributed automated control systems: dissertation. Doctor of Technical Sciences: 13/05/06: protected 10/09. 1985: approved. 03/28/1986. – M. – P.1985. – 328.

53. Amerbaev V.M., Biyashev R.G., Nysanbaeva S.E. Application of non-positional number systems in cryptographic protection // News of the National Academy of Sciences of the Republic of Kazakhstan. Physics and mathematics series. – Almaty: Gylym. – 2005. – No. 3. – P. 84-89.

54. Biyashev R.G., Nysanbaeva S.E. Algorithm for generating an electronic digital signature with the ability to detect and correct errors // Cybernetics and system analysis. – 2012. – Vol. 48, No. 4. – P. 14-23.

55. Biyashev R., Nyssanbayeva S., Kapalova N., Khakimov R. Modular models of the cryptographic protection of information // International Conference on Computer Networks and Information. – 2014. – P. 49-54.

56. Kapalova N., Dyusenbayev D. Security analysis of an encryption scheme based on nonpositional polynomial notations // Open Engineering – 2016. – No.6. – P. 250-258.

57. Biyashev R.G., Nyssanbayeva S.E., Begimbayeva Ye.Ye., Magzom M.M. Modification of the Cryptographic Algorithms, Developed on the Basis of Nonpositional Polynomial Notations // Proceedings of the International Conference on Circuits, Systems, Signal Processing, Communications and Computers (CSSCC 2015). Vienna. – 2015. – P.170-176.

58. Biyashev R., Kalimoldayev M., Nyssanbayeva S., Magzom M., Development of an encryption algorithm based on nonpositional polynomial notations // Proceedings of the International Conference on Advanced Materials Science and Environmental Engineering (AMSEE 2016). - Chiang Mai; Thailand. – 2016. – P.243-245.

59. Poschmann A. Lightweight Cryptography – Cryptographic Engineering for a Pervasive World. IACR ePrint archive. –2009. – P.516.

60. Kalimoldayev M., Tynymbayev S., Gnatyuk S., Ibraimov M., Magzom M. The device for multiplying polynomials modulo an irreducible polynomial News of the National Academy of Sciences of the Republic of Kazakhstan, Series of Geology and Technical Sciences. – 2019. – Vol.2. – No.434. – P. 199-205.

61. Tynymbayev S., Gnatyuk S.A., Aitkhozhayeva Y.Z., Berdibayev R.S., Namazbayev T.A. Modular reduction based on the divider by blocking negative remainders, News of the National Academy of Sciences of the Republic of Kazakhstan, Series of Geology and Technical Sciences, – 2019. – Vol.2. – No.434. – P. 238-248.

62. Gura N. Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs''Proc. 6th Int'l Workshop Cryptographic Hardware and Embedded Systems (CHES 04) LNCS 3156//Springer. – 2004. – Vol. 3156. – P. 119-132.

63. Al-Kaabi Sh., Belhaouari S. Methods toward enhancing rsa algorith a survey// International Journal of Network Security & Its Applications (IJNSA). – 2019. –Vol. 11. – No.3. – P.53-70.

64. Hu Z., Gnatyuk S., Kovtun M., Seilova N. Method of searching birationally equivalent Edwards curves over binary fields//Advances in Intelligent Systems and Computing. – 2018. – Vol. 754. – P. 309-319.

65. Gnatyuk S., Kovtun M., Kovtun V., Okhrimenko A. Search method development of birationally equivalent binary Edwards curves for binary Weierstrass curves from DSTU 4145- 2002//Proceedings of 2nd Intern. Scientific-Practical Conf. on the Problems of Infocommunications. Science and Technology (PIC S&T 2015), Kharkiv, Ukraine. – 2015. – P. 5-8.

66. Kalimoldayev M., Biyashev R., Nyssanbayeva S., Begimbayeva Y. Modification of the digital signature, developed on the nonpositional polynomial notations// Eurasian Journal of Mathematical and Computer Applications. – 2016. – Vol.4 . – No. 2. – No. 33-38.

67. Biyashev R. Kalimoldayev, M., Nyssanbayeva S., Magzom M. Development of an encryption algorithm based on nonpositional polynomial notations// Proceedings of the International Conference on Advanced Materials Science and Environmental Engineering. –2016. – P.112-118.

68. Gnatyuk S., Okhrimenko A., Kovtun M., Gancarczyk T., Karpinskyi V. Method of Algorithm Building for Modular Reducing by Irreducible Polynomial// Proceedings of the 16th International Conference on Control, Automation and Systems, Gyeongju, Korea. – 2016. – P. 1476-1479.

69. Gnatyuk S., Kinzeryavyy V., Iavich M., Prysiazhnyi D., Yubuzova Kh. High-Performance Reliable Block Encryption Algorithms Secured against Linear and Differential Cryptanalytic Attack//CEUR Workshop Proceedings (Proceedings of the 14th International Conference on ICT in Education, Research and Industrial Applications. Integration, Harmonization and Knowledge Transfer, Kyiv, Ukraine. – 2018. – Vol. 2104. – P. 657- 668.

70. Tynymbayev S., Ibraimov M., Namazbayev T., Gnatyuk S. Development of pipelined polynomial multiplier modulo irreducible polynomials for

cryptosystems//Eastern-European Journal of Enterprise Technologies. – 2022. – Vol.1 – No.4. – P. 37–43.

71. Reis L., Wang Q. The dynamics of permutations on irreducible polynomials// Finite Fields and Their Applications. – 2020. – Vol. 64. – P. 101664

72. Krasnobayev V., Koshman S., Yanko A., Martynenko A. Method of Error Control of the Information Presented in the Modular Number System, Problems of Infocommunications//Science and Technology: International Scientific-Practical Conference. – 2018. – P. 35-42.

73. Krasnobayev V., Koshman S. A Method for Operational Diagnosis of Data Represented in a Residue Number System//Cybernetics and Systems Analysis. – 2018. – Vol. 54. – No.2. – P. 336-344.

74. Gohar M., Kyureghyan K. A recurrent construction of irreducible polynomials of fixed degree over finite fields// Original Paper. – 2020. – Vol. 33. – P. 163–171.

75. Gupta S. Irreducible Polynomials in Z[x] That Are Reducible Modulo All Primes//Open Journal of Discrete Mathematics. – 2021. – Vol. 9. – P. 52-61.

76. Newhouse W., Keith S., Scribner B., Witte G. National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework// National Institute of Standards and Technology Special Publication 800-181 Natl. Inst. Stand. Technol. Spec. Publ. – 2017. Vol. – 800. – No. 181. – P. 144.

77. Hurley-Smith D., Constantinos Patsakis. Julio Hernandez-Castro. On the Unbearable Lightness of FIPS 140–2 Randomness Tests// IEEE. – 2020. P.– 3946 – 3958.

78. Gorbenko I., Kuznetsov A., Gorbenko Y., Vdovenko S., Tymchenko V., Lutsenko M. Studies on statistical analysis and performance evaluation for some stream ciphers//International Journal of Computing. – 2019. – Vol. 18. – No.1. – P. 82-88.

79. Cao M., Zhang W. Related-Key Differential Cryptanalysis of the Reduced-Round Block Cipher GIFT// IEEE Access. – 2019. – Vol.7. – P. 175769 – 175778.

80. Kaplan M., Leurent G., Leverrier A., Naya-Plasencia M. Quantum Differential and Linear Cryptanalysis//Transactions on Symmetric Cryptology. – 2017. – Issue 1. – P. 71-94

81. Klyucharyov P.G. On statistical testing of block ciphers // Mathematics and mathematical modeling. – 2018. – No. 5. – P. 35-56.

82. Cid, C., Grassi, L., Gunsing, A., Lüftenegger, R., Rechberger, C., & Schofnegger, M. Influence of the Linear Layer on the Algebraic Degree in SP-Networks//IACR Transactions on Symmetric Cryptology. – 2022. - Vol.1. – P. 110–137.

83. Biyashev R.G., Algazy K.T., Dyusenbaev D.S., Sakan K.S. Results of linear cryptanalysis of the Qamal cipher // Bulletin of AUES. – Almaty, 2020. – No. 2. – P. 96-105.

84. Anandakumar N., Hashmi M., Sanadhya S. Field programmable gate array based elliptic curve menezes-qu-vanstone key agreement protocol realization using

physical unclonable function and true random number generator primitives//Iet Circuits Devices & Systems. – 2020. – Vol. 16. – No. 5. – P. 382-398.

85. Urbina M., Moreira N., Rodriguez M., Acosta T., Lázaro J., Astarloa, A. Secure protocol and ip core for configuration of networking hardware ips in the smart grid//Energies. – 2018. – Vol. 11. – No. 3. – P. 510.

86. Roy D., Bhasin S., Danger J., Guilley,S., He W., Mukhopadhyay, D., Ngo, X. The conflicted usage of rluts for security-critical applications on fpga//Journal of Hardware and Systems Security,. – 2018. – Vol.2. – No. 2. – P.162-178.

87. Pan L., Tu G., Liu S., Cai Z., Xiong, X. A lightweight aes coprocessor based on risc-v custom instructions// Security and Communication Networks. 2021. – P. 1-13.

88. Abu-Tair M., Djahel S., Perry P., Scotney B., Zia U., Carracedo J., Sajjad, A. Towards secure and privacy-preserving iot enabled smart home: architecture and experimental study//Sensors. – 2020. – Vol. 20. – No. 21. – P. 6131.

89. Boykuziev I., Angshuman Kh.,∗ , Abdurakhimov B., Das R., Khudoykulov Z. Integral cryptanalysis: a new key determination technique for 3-phase Kuznyechik encryption// Eng. Res. Express. – 2023. – Vol.5. – P.035018.

90. Miguel G., Adrián P., Carlos S., Concepción A., Santiago C. Chaos-Based Bitwise Dynamical Pseudorandom Number Generator On FPGA// IEEE Transactions on Instrumentation and Measurement. – 2018. – Vol.68. – No.1. – P.291-293.

91. Obaida M., Ashraf A., Malek M., Nasr Y. Chaotic based multimedia encryption: asurvey for network and internet security// Bulletin of Electrical Engineering and Informatics. – 2021. – Vol. 11. – No. 4. – P. 2151-2159.

92. Pchelintsev A.N. Numerical and physical modeling of the dynamics of the Lorenz system // Numerical Analysis and Applications. – 2014. – Vol.7. – No.2. – P. 159-167.

93. V. Messager. Influences on Otto E. Rössler's earliest paper on chaos // International Journal of Bifurcation & Chaos journal. — 2010. — Vol. 20, no. 11. — P. 3585—3616.

94. Parikibandla S., Alluri S. Lorentz Chaotic System key generation with Low Area FPGA Implementation using Present Security Algorithm// International Journal of Recent Technology and Engineering (IJRTE). – 2019. – Vol.8. – No.2 . – P. 2104-2110.

95. Chandra Mouli.R, K.R.K.Sastry. Hardware Implementation of High Speed RC4 Algorithm in FPGA // International Journal of Computer Applications (0975 – 8887). – 2013. – Vol. 83. – No.4. – P. 20–23.

96. Сқабылов Ә.Ә., Жексебай Д.М., Әлмен Д.Б., Ихсан Г.Б. FPGA негізінде qpsk бағдарламалық-ақпараттық комплекс //Международная научная конференция студентов и молодых ученых, «ФАРАБИ ӘЛЕМІ», 2022. – С. 215.

97. Скабылов А.А., Ибраимов М.К., Жексебай Д.М., Кожагулов Е.Т. Программно-аппаратная реализация qpsk на базе zynq и matlab // Вестник КазНПУ им. Абая, серия «Физико-математические науки», №1(77), 2022

98. Сқабылов Ә.Ә., Әлмен Д.Б., Жексебай Д.М., Азамат Р.М. ПЛИС негізінде шифрлеу және дешифрлеу алгоритмін аппараттық жүзеге асыру //Международная научная конференция студентов и молодых ученых, «ФАРАБИ ӘЛЕМІ», 2021. – С. 218.

99. Kalimoldayev M., Tynymbayev S., Almen D., Skabylov A. Implementation of the nonconductivity check algorithm for a given binary polynomial on the FPGA// Vestnik KazNRTU. – 2020. – Vol. 141. – No. 5. – P. 629-635.

100. Әлмен Д.Б., Сқабылов Ә.Ә. ПЛИС негізінде GF(2) ақырғы өрістегі келтірілмейтін көпмүшелер базасын құру //Международная научная конференция студентов и молодых ученых, «ФАРАБИ ӘЛЕМІ», 2020. – С. 269.

101. Скабылов А.А., Жексебай Д.М., Ибраимов М.К., Налибаев Е.Д. Применение хаотических генераторов для секретного ключа в приемо-передающих электронных устройствах // Вестник КазНПУ. – 2023. – Т. 82, № 2. – С. 194-201

102. Сқабылов Ә.Ә., Налибаев Е.Д., Ибраимов М.К., Данабеков Б., Азамат Р.М //«Бейсызық жүйелердегі хаос және құрылымдар. Теория және тәжірибе» XII Халықаралық ғылыми конференциясы, 2022. – С. 155.

103. Cao, H. et al. (2017). A New Pseudorandom Number Generator Based on Multimodal Chaotic Map. Complexity. – 2017. – P. 1-7.

104. Yang, H. et al. (2016). A Chaotic Stream Cipher Using the Internal States of a Three-Dimensional Autonomous Chaotic System. IEEE Transactions on Circuits and Systems II: Express Briefs. – 2016. – Vol. 63. – No.12. – No. 1149-1153.

105. Merah1 , Adda Ali-Pacha2 , Naima Hadj Said3 and Mustafa Mamat4A Pseudo Random Number Generator Based on the Chaotic System of Chua's Circuit, and its Real Time FPGA Implementation

106. Aitkhozhayeva, Y.Zh., Tynymbayev, S., Adilbekkyzy, S., Skabylov, A., Ibraimov, M. Design and research of the behavioral model for the modular reduction device// Eurasian Physical Technical Journal. – 2020. – Vol.17. – No.1. – P. 151-156.

107. Сқабылов Ә.Ә., Әлмен Д.Б., Жексебай Д.М., Азамат Р.М. Модуль бойынша бөлу құрылғысының әрекеттік моделін жобалау және зерттеу //Международная научная конференция студентов и молодых ученых, «ФАРАБИ ӘЛЕМІ», 2021. – С. 217.

108. S. Adilbekkyzy, Y. Zh. Aitkhozhayeva, S. Tynymbayev. Modeling of the partial reminder former of the modular reduction device. Eurasian Union of Scientists. – 2019. – Vol. 6. – No. 63. – P. 47-51.

109. S. Tynymbayev, Y. Zh. Aitkhozhayeva, S. Adilbekkyzy, R. Sh. Berdibaev. 2019. Development and modeling of schematic diagram for the modular reduction device. Problems of Informatics. – 2019. – Vol. 4. – P. 42-52.

110. H. Yu, G. Bai, H. Hao, J. Wang, C. Yap. 2015. Efficient Modular Reduction Algorithm Without Correction Phase. Frontiers in Algorithmics. FAW 2015. Lecture Notes in Computer Science. – 2015. – Vol. 9130. – P. 304-313.

111. Ibraimov, S.T. Tynymbayev, Skabylov A.A., Kozhagulov Y., Zhexebay D.M. Development and design of an FPGA-based encoder for NPN//Cogent Engineering. – 2022. – Vol. 9. – P. 2008847.

112. Kumar, A., Vishnoi, P., & Shimi, S. L. Smart grid security with cryptographic chip integration//EAI Endorsed Transactions on Energy Web. – 2019. – Vol. 19. – No. 23. – P. 6.

113. Kumar, N., Mishra, V. M., & Kumar, A. Smart grid and nuclear power plant security by integrating cryptographic hardware chip. Nuclear Engineering and Technology. – 2021. – No. 53. – Vol.10. – P. 3327–3334.

114. Augusto D., Oliveira D., Batista G. Design of DES encryption algorithm as bundleddata asynchronous pipeline using FPGA//Rev. Bras. Apl. Vac., Campinas. – 2020. – Vol. 39. – No.3. – P. 260-270.

115. Sakan K.S., Algazy K.T., Dyusenbayev D.S. About some ways to improve the performance of calculating the block Mixer2 of the Qamal encryption algorithm// Vestnik KazNRTU. – 2020. – No.4. – P.492-499.

116. Hussain Pirzada S., Murtaza A., Xu N., Jianwei L. Architectural Optimization of Parallel Authenticated Encryption Algorithm for Satellite Application//IEEE. – 2020. – Vol.8. – No. 48543-48556.

117. Galimberti A. FPGA-Based Design and Implementation of a Code-Based Post-quantum KEM//Special Topics in Information Technology. – 2024. – P.27-40.

ҚОСЫМША А
Verilog тілінде Ben-Or алгоритмін енгізу

```
module promo(
    input clk,
    input [n:0] f,
    input [(2**((n+1)/2))+1:0] p,
    input start,
    output reg [(2**((n+1)/2))+1:0] q,
    output reg result
);
parameter n = 16;
wire [(2**((n+1)/2))+1:0] g;
assign g = p & f;
reg [(2**((n+1)/2))+1:0] a;
reg [(2**((n+1)/2))+1:0] b;
reg flag = 0;
endmodule
```

Модульдің кіріс/шығыс мәндері және айнымалылары

```
always @(posedge clk)
begin
    result <= 0;
    q <= 0;
    if (p == 0)
        q <= 0;
    else
    begin
        if (start)
            flag <= 1'b0;
        if (flag == 0)
        begin
            a <= f;
            b <= g;
            flag <= 1'b1;
        end
        else
        begin
            if (b == 0)
            begin
                q <= a;
                result <= 1;
            end
```

```
            else
            begin
                a <= b;
                b <= a % b;
            end
        end
    end
end
```

ЕҮОБ табу алгоритмі

ҚОСЫМША Ә
GF(2) ақырғы өрістегі келтірілмейтін көпмүшелер базасын құру үшін қолданатын Verilog код

```
always @ (posedge clk)
begin
   if (n == 1)
   begin
      b <= 1;
      s <= 1;
      k <= 1;
   end
   else
   begin
      b <= (2 * n) + 1;
      s <= 2 * n;
      k <= 2 * (n + 1) - 1;
   end

   if (b < k)
   begin
      fl[i] <= b[n:0];
      b <= b + s;
      i <= i + 1;
   end

   if (b == k)
   begin
      fl[i] <= b[n:0];
      b <= k;
   end
end
```

Тексеруге қажетті көпмүшелерді *fl* регистріне жинау

```
always @(posedge clk)
begin
   if (i == 2**(n-1) - 1)
   begin
      if (count == 15)
      begin
         count <= 0;
         start <= 1'b1;
```

```
            if (j <= 2**(n-1) - 1)
            begin
               pp <= p[cnt];
               f <= f[j];
            end
            else
            begin
               pp <= 0;
               f <= 0;
            end
            if (cnt == (n+1)/2)
               flagStop <= 1'b1;
            else
               cnt <= cnt + 1'b1;
         end
         else if (flagStop == 0)
         begin
            count <= count + 1'b1;
            start <= 1'b0;
         end
      end
   end
```

Кіріс/шығыс мәндерін байланыстыру модулі

```
if (r1)
   y <= q1;
else
   y <= 0;

if (cnt == 2)
   result <= 1;
else
   result <= 0;
```

$q1$ мәнінің y-ке жинақталуы

```
always @(posedge result)
begin
   if (y == 1)
   begin
      N[w] <= f1[j-1];
      w <= w + 1;
      $display("N = %d", f1[j-1]);
   end
end
```

Келтірілмейтін көпмүшелердің жинақталуы

ҚОСЫМША Б
Verilog HDL кодтар тізімі

```
module redmod (
   output [7:0] R,
   input [11:0] A,
   input [7:0] P1,
   input clk);
      integer i = 0;
      reg [13:0] temp;
      reg [7:0] P2;
      reg [7:0] P3;
      reg [7:0] P33;
      reg [7:0] P22;
      reg [7:0] P11;
      reg SS3;
      reg SS2;
      reg SS1;
      reg [7:0] SUM;
      reg [7:0] ri;
      reg [13:0] memory [0:1];
      always @ (posedge clk)
      begin
         P2 = 2*P1;  P3 = 3*P1;
         P33 = ~P3 + 1;
         P22 = ~P2 + 1;
         P11 = ~P1 + 1;
         If (i < 3)
         begin
            memory[i] = A;
            temp = memory [0];
            temp = temp << 2;
       if (temp [13:6] >= P1 && temp [13:6] < P2)
               ri = temp [13:6] + P11;
  else if (temp [13:6] >= P2 && temp [13:6] < P3)
               ri = temp [13:6] + P22;
            else if (temp [13:6] >= P3)
               ri = temp [13:6] + P33;
            else
               ri = temp [13:6];
            SS3 = (P3 > temp [13:6])? 1: 0;
            SS2 = (P2 > temp [13:6])? 1: 0;
            SS1 = (P1 > temp [13:6])? 1: 0;
            SUM = temp [13:6] + ((P33 & {8{~SS3}}) | (P22 & {8{SS3 &
```

```
(~SS2)}}) | (P11 & {8{SS2 & (~SS1)}}));
                    ri = SUM | (temp [13:6] & {8{SS1}});
                    temp [13:6] = ri;
                    memory [0] = temp;
                    i = i + 1;
                end
            end
            assign R = ri;
        endmodule
```

ҚОСЫМША В
Scan кодын ascii-ге түрлендіру үшін VHDL тілінде жазылған бағдарламалық код

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY ps2_keyboard_to_ascii IS
  GENERIC(
    clk_freq                  : INTEGER := 50_000_000; --system clock frequency
in Hz
    ps2_debounce_counter_size : INTEGER := 8);        --set such that
2^size/clk_freq = 5us (size = 8 for 50MHz)
  PORT(
    clk        : IN  STD_LOGIC;                    --system clock input
    ps2_clk    : IN  STD_LOGIC;                    --clock signal from PS2
keyboard
    ps2_data   : IN  STD_LOGIC;                    --data signal from PS2 keyboard
    ascii_new  : OUT STD_LOGIC;                    --output flag indicating new
ASCII value
    ascii_code : OUT STD_LOGIC_VECTOR(6 DOWNTO 0)); --ASCII
value
END ps2_keyboard_to_ascii;

ARCHITECTURE behavior OF ps2_keyboard_to_ascii IS
  TYPE machine IS(ready, new_code, translate, output);              --needed
states
  SIGNAL state             : machine;                        --state machine
  SIGNAL ps2_code_new      : STD_LOGIC;                      --new PS2 code
flag from ps2_keyboard component
  SIGNAL ps2_code          : STD_LOGIC_VECTOR(7 DOWNTO 0);         --
PS2 code input form ps2_keyboard component
  SIGNAL prev_ps2_code_new : STD_LOGIC := '1';                 --value of
ps2_code_new flag on previous clock
  SIGNAL break             : STD_LOGIC := '0';                 --'1' for break code,
'0' for make code
  SIGNAL e0_code           : STD_LOGIC := '0';                 --'1' for multi-
code commands, '0' for single code commands
  SIGNAL caps_lock         : STD_LOGIC := '0';                 --'1' if caps lock
is active, '0' if caps lock is inactive
  SIGNAL control_r         : STD_LOGIC := '0';                 --'1' if right
control key is held down, else '0'
  SIGNAL control_l         : STD_LOGIC := '0';                 --'1' if left control
key is held down, else '0'
```

```
        SIGNAL shift_r          : STD_LOGIC := '0';                 --'1' if right shift is
held down, else '0'
        SIGNAL shift_l          : STD_LOGIC := '0';                 --'1' if left shift is
held down, else '0'
        SIGNAL ascii            : STD_LOGIC_VECTOR(7 DOWNTO 0) := x"FF"; --
internal value of ASCII translation

        --declare PS2 keyboard interface component
        COMPONENT ps2_keyboard IS
          GENERIC(
            clk_freq              : INTEGER;  --system clock frequency in Hz
            debounce_counter_size : INTEGER); --set such that 2^size/clk_freq = 5us
(size = 8 for 50MHz)
          PORT(
            clk          : IN  STD_LOGIC;                   --system clock
            ps2_clk      : IN  STD_LOGIC;                   --clock signal from PS2
keyboard
            ps2_data     : IN  STD_LOGIC;                   --data signal from PS2
keyboard
            ps2_code_new : OUT STD_LOGIC;                   --flag that new PS/2 code
is available on ps2_code bus
            ps2_code     : OUT STD_LOGIC_VECTOR(7 DOWNTO 0)); --code
received from PS/2
        END COMPONENT;

      BEGIN

        --instantiate PS2 keyboard interface logic
        ps2_keyboard_0:  ps2_keyboard
          GENERIC MAP(clk_freq => clk_freq, debounce_counter_size =>
ps2_debounce_counter_size)
          PORT MAP(clk => clk, ps2_clk => ps2_clk, ps2_data => ps2_data,
ps2_code_new => ps2_code_new, ps2_code => ps2_code);

        PROCESS(clk)
        BEGIN
          IF(clk'EVENT AND clk = '1') THEN
            prev_ps2_code_new <= ps2_code_new; --keep track of previous
ps2_code_new values to determine low-to-high transitions
            CASE state IS

              --ready state: wait for a new PS2 code to be received
              WHEN ready =>
```

```
            IF(prev_ps2_code_new = '0' AND ps2_code_new = '1') THEN --new
PS2 code received
              ascii_new <= '0';                          --reset new ASCII code
indicator
              state <= new_code;                          --proceed to new_code state
            ELSE                                        --no new PS2 code received yet
              state <= ready;                           --remain in ready state
            END IF;

          --new_code state: determine what to do with the new PS2 code
          WHEN new_code =>
            IF(ps2_code = x"F0") THEN    --code indicates that next command is
break
              break <= '1';              --set break flag
              state <= ready;            --return to ready state to await next PS2 code
            ELSIF(ps2_code = x"E0") THEN --code indicates multi-key command
              e0_code <= '1';            --set multi-code command flag
              state <= ready;            --return to ready state to await next PS2 code
            ELSE                         --code is the last PS2 code in the make/break code
              ascii(7) <= '1';           --set internal ascii value to unsupported code (for
verification)
              state <= translate;        --proceed to translate state
            END IF;

          --translate state: translate PS2 code to ASCII value
          WHEN translate =>
            break <= '0';    --reset break flag
            e0_code <= '0';  --reset multi-code command flag

            --handle codes for control, shift, and caps lock
            CASE ps2_code IS
              WHEN x"58" =>                 --caps lock code
                IF(break = '0') THEN        --if make command
                  caps_lock <= NOT caps_lock;     --toggle caps lock
                END IF;
              WHEN x"14" =>                 --code for the control keys
                IF(e0_code = '1') THEN      --code for right control
                  control_r <= NOT break;       --update right control flag
                ELSE                        --code for left control
                  control_l <= NOT break;       --update left control flag
                END IF;
              WHEN x"12" =>                 --left shift code
                shift_l <= NOT break;       --update left shift flag
              WHEN x"59" =>                 --right shift code
```

```
    shift_r <= NOT break;           --update right shift flag
  WHEN OTHERS => NULL;
END CASE;

--translate control codes (these do not depend on shift or caps lock)
IF(control_l = '1' OR control_r = '1') THEN
  CASE ps2_code IS
    WHEN x"1E" => ascii <= x"00"; --^@  NUL
    WHEN x"1C" => ascii <= x"01"; --^A  SOH
    WHEN x"32" => ascii <= x"02"; --^B  STX
    WHEN x"21" => ascii <= x"03"; --^C  ETX
    WHEN x"23" => ascii <= x"04"; --^D  EOT
    WHEN x"24" => ascii <= x"05"; --^E  ENQ
    WHEN x"2B" => ascii <= x"06"; --^F  ACK
    WHEN x"34" => ascii <= x"07"; --^G  BEL
    WHEN x"33" => ascii <= x"08"; --^H  BS
    WHEN x"43" => ascii <= x"09"; --^I  HT
    WHEN x"3B" => ascii <= x"0A"; --^J  LF
    WHEN x"42" => ascii <= x"0B"; --^K  VT
    WHEN x"4B" => ascii <= x"0C"; --^L  FF
    WHEN x"3A" => ascii <= x"0D"; --^M  CR
    WHEN x"31" => ascii <= x"0E"; --^N  SO
    WHEN x"44" => ascii <= x"0F"; --^O  SI
    WHEN x"4D" => ascii <= x"10"; --^P  DLE
    WHEN x"15" => ascii <= x"11"; --^Q  DC1
    WHEN x"2D" => ascii <= x"12"; --^R  DC2
    WHEN x"1B" => ascii <= x"13"; --^S  DC3
    WHEN x"2C" => ascii <= x"14"; --^T  DC4
    WHEN x"3C" => ascii <= x"15"; --^U  NAK
    WHEN x"2A" => ascii <= x"16"; --^V  SYN
    WHEN x"1D" => ascii <= x"17"; --^W  ETB
    WHEN x"22" => ascii <= x"18"; --^X  CAN
    WHEN x"35" => ascii <= x"19"; --^Y  EM
    WHEN x"1A" => ascii <= x"1A"; --^Z  SUB
    WHEN x"54" => ascii <= x"1B"; --^[  ESC
    WHEN x"5D" => ascii <= x"1C"; --^\  FS
    WHEN x"5B" => ascii <= x"1D"; --^]  GS
    WHEN x"36" => ascii <= x"1E"; --^^  RS
    WHEN x"4E" => ascii <= x"1F"; --^_  US
    WHEN x"4A" => ascii <= x"7F"; --^?  DEL
    WHEN OTHERS => NULL;
  END CASE;
ELSE --if control keys are not pressed
```

```
                --translate characters that do not depend on shift, or caps lock
                CASE ps2_code IS
                  WHEN x"29" => ascii <= x"20"; --space
                  WHEN x"66" => ascii <= x"08"; --backspace (BS control code)
                  WHEN x"0D" => ascii <= x"09"; --tab (HT control code)
                  WHEN x"5A" => ascii <= x"0D"; --enter (CR control code)
                  WHEN x"76" => ascii <= x"1B"; --escape (ESC control code)
                  WHEN x"71" =>
                    IF(e0_code = '1') THEN      --ps2 code for delete is a multi-key code
                      ascii <= x"7F";           --delete
                    END IF;
                  WHEN OTHERS => NULL;
                END CASE;

                --translate letters (these depend on both shift and caps lock)
                IF((shift_r = '0' AND shift_l = '0' AND caps_lock = '0') OR
                  ((shift_r = '1' OR shift_l = '1') AND caps_lock = '1')) THEN  --letter
is lowercase
                  CASE ps2_code IS
                    WHEN x"1C" => ascii <= x"61"; --a
                    WHEN x"32" => ascii <= x"62"; --b
                    WHEN x"21" => ascii <= x"63"; --c
                    WHEN x"23" => ascii <= x"64"; --d
                    WHEN x"24" => ascii <= x"65"; --e
                    WHEN x"2B" => ascii <= x"66"; --f
                    WHEN x"34" => ascii <= x"67"; --g
                    WHEN x"33" => ascii <= x"68"; --h
                    WHEN x"43" => ascii <= x"69"; --i
                    WHEN x"3B" => ascii <= x"6A"; --j
                    WHEN x"42" => ascii <= x"6B"; --k
                    WHEN x"4B" => ascii <= x"6C"; --l
                    WHEN x"3A" => ascii <= x"6D"; --m
                    WHEN x"31" => ascii <= x"6E"; --n
                    WHEN x"44" => ascii <= x"6F"; --o
                    WHEN x"4D" => ascii <= x"70"; --p
                    WHEN x"15" => ascii <= x"71"; --q
                    WHEN x"2D" => ascii <= x"72"; --r
                    WHEN x"1B" => ascii <= x"73"; --s
                    WHEN x"2C" => ascii <= x"74"; --t
                    WHEN x"3C" => ascii <= x"75"; --u
                    WHEN x"2A" => ascii <= x"76"; --v
                    WHEN x"1D" => ascii <= x"77"; --w
                    WHEN x"22" => ascii <= x"78"; --x
                    WHEN x"35" => ascii <= x"79"; --y
```

```
      WHEN x"1A" => ascii <= x"7A"; --z
      WHEN OTHERS => NULL;
    END CASE;
  ELSE                                --letter is uppercase
    CASE ps2_code IS
      WHEN x"1C" => ascii <= x"41"; --A
      WHEN x"32" => ascii <= x"42"; --B
      WHEN x"21" => ascii <= x"43"; --C
      WHEN x"23" => ascii <= x"44"; --D
      WHEN x"24" => ascii <= x"45"; --E
      WHEN x"2B" => ascii <= x"46"; --F
      WHEN x"34" => ascii <= x"47"; --G
      WHEN x"33" => ascii <= x"48"; --H
      WHEN x"43" => ascii <= x"49"; --I
      WHEN x"3B" => ascii <= x"4A"; --J
      WHEN x"42" => ascii <= x"4B"; --K
      WHEN x"4B" => ascii <= x"4C"; --L
      WHEN x"3A" => ascii <= x"4D"; --M
      WHEN x"31" => ascii <= x"4E"; --N
      WHEN x"44" => ascii <= x"4F"; --O
      WHEN x"4D" => ascii <= x"50"; --P
      WHEN x"15" => ascii <= x"51"; --Q
      WHEN x"2D" => ascii <= x"52"; --R
      WHEN x"1B" => ascii <= x"53"; --S
      WHEN x"2C" => ascii <= x"54"; --T
      WHEN x"3C" => ascii <= x"55"; --U
      WHEN x"2A" => ascii <= x"56"; --V
      WHEN x"1D" => ascii <= x"57"; --W
      WHEN x"22" => ascii <= x"58"; --X
      WHEN x"35" => ascii <= x"59"; --Y
      WHEN x"1A" => ascii <= x"5A"; --Z
      WHEN OTHERS => NULL;
    END CASE;
  END IF;

  --translate numbers and symbols (these depend on shift but not caps
lock)
  IF(shift_l = '1' OR shift_r = '1') THEN  --key's secondary character is
desired
    CASE ps2_code IS
      WHEN x"16" => ascii <= x"21"; --!
      WHEN x"52" => ascii <= x"22"; --"
      WHEN x"26" => ascii <= x"23"; --#
      WHEN x"25" => ascii <= x"24"; --$
```

```
    WHEN x"2E" => ascii <= x"25"; --%
    WHEN x"3D" => ascii <= x"26"; --&
    WHEN x"46" => ascii <= x"28"; --(
    WHEN x"45" => ascii <= x"29"; --)
    WHEN x"3E" => ascii <= x"2A"; --*
    WHEN x"55" => ascii <= x"2B"; --+
    WHEN x"4C" => ascii <= x"3A"; --:
    WHEN x"41" => ascii <= x"3C"; --<
    WHEN x"49" => ascii <= x"3E"; -->
    WHEN x"4A" => ascii <= x"3F"; --?
    WHEN x"1E" => ascii <= x"40"; --@
    WHEN x"36" => ascii <= x"5E"; --^
    WHEN x"4E" => ascii <= x"5F"; --_
    WHEN x"54" => ascii <= x"7B"; --{
    WHEN x"5D" => ascii <= x"7C"; --|
    WHEN x"5B" => ascii <= x"7D"; --}
    WHEN x"0E" => ascii <= x"7E"; --~
    WHEN OTHERS => NULL;
  END CASE;
ELSE                                --key's primary character is desired
  CASE ps2_code IS
    WHEN x"45" => ascii <= x"30"; --0
    WHEN x"16" => ascii <= x"31"; --1
    WHEN x"1E" => ascii <= x"32"; --2
    WHEN x"26" => ascii <= x"33"; --3
    WHEN x"25" => ascii <= x"34"; --4
    WHEN x"2E" => ascii <= x"35"; --5
    WHEN x"36" => ascii <= x"36"; --6
    WHEN x"3D" => ascii <= x"37"; --7
    WHEN x"3E" => ascii <= x"38"; --8
    WHEN x"46" => ascii <= x"39"; --9
    WHEN x"52" => ascii <= x"27"; --'
    WHEN x"41" => ascii <= x"2C"; --,
    WHEN x"4E" => ascii <= x"2D"; ---
    WHEN x"49" => ascii <= x"2E"; --.
    WHEN x"4A" => ascii <= x"2F"; --/
    WHEN x"4C" => ascii <= x"3B"; --;
    WHEN x"55" => ascii <= x"3D"; --=
    WHEN x"54" => ascii <= x"5B"; --[
    WHEN x"5D" => ascii <= x"5C"; --\
    WHEN x"5B" => ascii <= x"5D"; --]
    WHEN x"0E" => ascii <= x"60"; --`
    WHEN OTHERS => NULL;
  END CASE;
```

```
            END IF;

          END IF;

          IF(break = '0') THEN  --the code is a make
            state <= output;       --proceed to output state
          ELSE                   --code is a break
            state <= ready;        --return to ready state to await next PS2 code
          END IF;

        --output state: verify the code is valid and output the ASCII value
        WHEN output =>
          IF(ascii(7) = '0') THEN           --the PS2 code has an ASCII output
            ascii_new <= '1';               --set flag indicating new ASCII output
            ascii_code <= ascii(6 DOWNTO 0);   --output the ASCII value
          END IF;
          state <= ready;                   --return to ready state to await next PS2 code

      END CASE;
    END IF;
  END PROCESS;

END behavior;
```

ҚОСЫМША Г
Кіріс деректерін құруға арналған бағдарлама коды

```
always@ (negedgeascii_new)
begin
if(ascii_code == 7*hOD)
      info <= info_ram;
   else if(ascii_code == 7*h08)
info_ram<= info_ram>>8;
   else
info_ram<= {info_ram,{1'b0, ascii_code}};
end
assign info_out = info_ram;
always @ (posedgeclk)
   clock <= clock + 1'b1;
assign clk2 = clock;
endmodule
```

Сызықтық кері байланысты ауыстыру регистрінің Verilog тілінде жазылған алгоритмі

```
module lfsr(
  input rst,
  input clk,
  input enable,
  input [3:0] taps,
  output [3:0] out,
 );
reg [3:0] shift_reg;
assign out = shift_reg;
wire next_bit;
assign next_bit = shift_reg[1] ^  shift_reg[0];
always @(posedgeclk)
  if (rst)
shift_reg<= taps;
  else
  if (enable)
shift_reg<= {next_bit, shift_reg[3:1] };
endmodule
```

10 кездейсоқ сандарды құруға арналған код үзіндісі

```
always @ (posedge clk)
  begin
    if (cntl < 2)
```

```
            cntl <= cntl +1;
          else
          begin
            rst <=0;
            enable <=1;
        end
          if (cntl ==1)
            rst <= 1;

        if(enable)
          begin
            if(cnt2 <11)
            begin
              if(cnt2 <10)
                rand[cnt2] <=11+(lfsr_out  % 6);
              cnt2 <= cnt2 +1;
            end
          end
        end


          assign rand1 = rand[1];
          assign rand2 = rand[2];
          assign rand3 = rand[3];
          assign rand4 = rand[4];
          assign rand5 = rand[5];
          assign rand6 = rand[6];
          assign rand7 = rand[7];
          assign rand8 = rand[8];
          assign rand9 = rand[9];
          assign rand10 = 128 - (rand [1] + rand [2] + rand [3] + rand [4] + rand
[5] + rand [6] + rand [7] + rand [8] + rand [9] );
        endmodule
```

ҚОСЫМША Ғ

128 битті 10 ішкі блокқа бөлуге арналған бағдарлама коды

```
always @ (posedge clk)
begin
   in_info[1] <= info>>m->rand[1];
   in_info[2] <= (info<<(rand[1]))>>m->rand[2];
   in_info[3] <= (info<<(rand[1]+rand[2]))>>m->rand[3];
   in_info[4] <= (info<<(rand[1]+rand[2]+rand[3]))>>m->rand[4];
   in_info[5] <= (info<<(rand[1]+rand[2]+rand[3]+ rand [4]))>>m-> rand[5];
   in_info[6] <= (info<<(rand[1]+rand[2]+rand[3]+rand[4]+rand[5]))>m-
>rand[6];
   in_info[7] <=
(info<<(rand[1]+rand[2]+rand[3]+rand[4]+rand[5]+rand[6]))>m->rand[7];
in_info[8] <= (info<<(rand[1]+rand[2]+rand[3]+rand[4]+rand[5]+rand[6]
+rand[7]))>m->rand[8];
in_info[9] <= (info<<(rand[1]+rand[2]+rand[3]+rand[4]+rand[5]+rand[6]
+rand[7] +rand[8]))>m->rand[9];
in_info[10] <= (info<<(rand[1]+rand[2]+rand[3]+rand[4]+rand[5]+rand[6]
+rand[7] +rand[8] +rand[9]))>m->rand[10];
end
```

Кілттерді таңдау $G(x)$ және негізгі жүйе $P(x)$ үшін бағдарлама коды:

```
always @ (posedge clk)
begin
  case (N)
    4: begin B <= 16`o6; P<= 17`o31: end
    5: begin B <= 16`o37; P<= 17`o45: end
    6: begin B <= 16`o57; P<= 17`o103: end
    7: begin B <= 16`o147; P<= 17`o217: end
    8: begin B <= 16`o355; P<= 17`o453: end
    9: begin B <= 16`o763; P<= 17`o1003: end
    10: begin B <= 16`o1117; P<= 17`o2011: end
    11: begin B <= 16`o3511; P<= 17`o4053: end
    12: begin B <= 16`o5171; P<= 17`o10011: end
    13: begin B <= 16`o10063; P<= 17`o20033: end
    14: begin B <= 16`o35613; P<= 17`o40041: end
    15: begin B <= 16`o76203; P<= 17`o10003: end
    16: begin B <= 16`o161201; P<= 17`o20053: end
endcase
end
```

Шифрлеуға арналған программалық код

```
always @ (posedge clk)
   begin
        if (i>n) i=2;
        else begin
                C[1][n:0]=A;
                R[1]=( B[0]) ? (C[1][n:0]):0;
                C[i][n:0]=(C[i-1])?((C[i-1][n:0]<<1)^P[n:0]):(C[i-
1][n:0]<<1);
                R[i][n:0]=(B[i-1]) ? (C[i][n:0] ^R[i-1] [n:0]):R[i-1][n:0];
                Out = R[N] [n-1:0];
                i=i+1;
        end
   end
assign out = R[n][n:0];
endmodule
```

ҚОСЫМША Д

160 битті он 16 битке бөлу функциясы бар шифрды ашу блогының бағдарламалық кодының үзіндісі

```
always @ (posedgeclk clk)
begin
in_info [1] <= {1`b0, info [159:144]};
in_info [2] <= {1`b0, info [143:128]};
in_info [3] <= {1`b0, info [127:112]};
in_info [4] <= {1`b0, info [111:96]};
in_info [5] <= {1`b0, info [95:80]};
in_info [6] <= {1`b0, info [79:64]};
in_info [7] <= {1`b0, info [63:48]};
in_info [8] <= {1`b0, info [47:32]};
in_info [9] <= {1`b0, info [31:16]};
in_info [10] <= {1`b0, info [15:0]};
end

decoder16 decoderN1(
.A (in_info [1]),
.clk (clk),
.N (rand [1]),
            .out (out_decoder [1])

);
```

128 биттік деректерді қалпына келтіру функциясы бар шифрды декодтау блогының бағдарламалық кодының үзіндісі

```
Case (rand[1])
4: decoder = {decoder, out_decoder [1] [3:0]};
5: decoder = {decoder, out_decoder [1] [4:0]};
6: decoder = {decoder, out_decoder [1] [5:0]};
7: decoder = {decoder, out_decoder [1] [6:0]};
8: decoder = {decoder, out_decoder [1] [7:0]};
9: decoder = {decoder, out_decoder [1] [8:0]};
10: decoder = {decoder, out_decoder [1] [9:0]};
11: decoder = {decoder, out_decoder [1] [10:0]};
12: decoder = {decoder, out_decoder [1] [11:0]};
13: decoder = {decoder, out_decoder [1] [12:0]};
14 decoder = {decoder, out_decoder [1] [13:0]};
15: decoder = {decoder, out_decoder [1] [14:0]};
16: decoder = {decoder, out_decoder [1] [15:0]};
endcase
```

VGA үшін драйвер және экранда көрсету үшін символды визуализациялау коды

```
module displayData(
input wire CLK,
input wire i_pix_stb,
input wire [9:0] x,
input wire [8:0] y,
input wire [255:0] xoutCoder0,
input wire [255:0] xoutCoder1,
input wire [255:0] xoutCoder2,
input wire [255:0] xoutCoder3,
input wire [255:0] xoutDecoder0,
input wire [255:0] xoutDecoder1,
input wire [255:0] xoutDecoder2,
input wire [255:0] xoutDecoder3,
input wire [255:0] xout0,
input wire [255:0] xout1,
input wire [255:0] xout2,
input wire [255:0] xout3,
output wire pixel
    );


reg [9:0] xCoord1 = 5;
reg [8:0] yCoord1 = 5;

reg [9:0] xCoord2 = 5;
reg [8:0] yCoord2 = 105;

reg [9:0] xCoord10 = 5;
reg [8:0] yCoord10 = 205;

reg [2:0] col;
reg [3:0] row;
reg [4:0] cnt1;
reg [7:0] cnt2;
reg [7:0] cnt3;
reg [1:0] cnt4;
initial
begin
    col <= 7;
    row <= 15;
```

```
            cnt1 <= 0;
            cnt2 <= 0;
            cnt3 <= 248;
            cnt4 <= 3;
        end

        reg [7:0] ascii;
        reg [255:0] xoutCoder [3:0];
        reg [255:0] xoutDecoder [3:0];
        reg [255:0] xout [3:0];

        always @ (posedge CLK)
        begin
            xoutCoder[0] <= xoutCoder0;
            xoutCoder[1] <= xoutCoder1;
            xoutCoder[2] <= xoutCoder2;
            xoutCoder[3] <= xoutCoder3;

            xoutDecoder[0] <= xoutDecoder0;
            xoutDecoder[1] <= xoutDecoder1;
            xoutDecoder[2] <= xoutDecoder2;
            xoutDecoder[3] <= xoutDecoder3;

            xout[0] <= xout0;
            xout[1] <= xout1;
            xout[2] <= xout2;
            xout[3] <= xout3;
        end

        always @ (posedge i_pix_stb)
        begin
            if(x >= xCoord1 && y >= yCoord1 && x < xCoord1 + 256 && y <
yCoord1 + 64)
            begin

                ascii[0] <= xout[cnt4][cnt3];
                ascii[1] <= xout[cnt4][cnt3+1];
                ascii[2] <= xout[cnt4][cnt3+2];
                ascii[3] <= xout[cnt4][cnt3+3];

                ascii[4] <= xout[cnt4][cnt3+4];
                ascii[5] <= xout[cnt4][cnt3+5];
                ascii[6] <= xout[cnt4][cnt3+6];
                ascii[7] <= xout[cnt4][cnt3+7];
```

```
            col <= col + 1;
            cnt2 <= cnt2 + 1;
            if(cnt2 == 0)
            begin
                row <= row + 1;
                if(row == 0)
                    cnt4 <= cnt4 + 1;
            end
            if(col == 6)
            begin
                cnt1 <= cnt1 + 1;
                cnt3 <= cnt3 - 8;
            end
        end
        else if(x >= xCoord2 && y >= yCoord2 && x < xCoord2 + 256 && y <
yCoord2 + 64)
        begin
            ascii[0] <= xoutCoder[cnt4][cnt3];
            ascii[1] <= xoutCoder[cnt4][cnt3+1];
            ascii[2] <= xoutCoder[cnt4][cnt3+2];
            ascii[3] <= xoutCoder[cnt4][cnt3+3];

            ascii[4] <= xoutCoder[cnt4][cnt3+4];
            ascii[5] <= xoutCoder[cnt4][cnt3+5];
            ascii[6] <= xoutCoder[cnt4][cnt3+6];
            ascii[7] <= xoutCoder[cnt4][cnt3+7];

            col <= col + 1;
            cnt2 <= cnt2 + 1;
            if(cnt2 == 0)
            begin
                row <= row + 1;
                if(row == 0)
                    cnt4 <= cnt4 + 1;
            end
            if(col == 6)
            begin
                cnt1 <= cnt1 + 1;
                cnt3 <= cnt3 - 8;
            end
        end
        else if(x >= xCoord10 && y >= yCoord10 && x < xCoord10 + 256 && y <
yCoord10 + 64)
```

```
    begin
        ascii[0] <= xoutDecoder[cnt4][cnt3];
        ascii[1] <= xoutDecoder[cnt4][cnt3+1];
        ascii[2] <= xoutDecoder[cnt4][cnt3+2];
        ascii[3] <= xoutDecoder[cnt4][cnt3+3];

        ascii[4] <= xoutDecoder[cnt4][cnt3+4];
        ascii[5] <= xoutDecoder[cnt4][cnt3+5];
        ascii[6] <= xoutDecoder[cnt4][cnt3+6];
        ascii[7] <= xoutDecoder[cnt4][cnt3+7];

        col <= col + 1;
        cnt2 <= cnt2 + 1;
        if(cnt2 == 0)
        begin
                row <= row + 1;
                if(row == 0)
                        cnt4 <= cnt4 + 1;
        end
        if(col == 6)
        begin
                cnt1 <= cnt1 + 1;
                cnt3 <= cnt3 - 8;
        end
    end
    else
        ascii <= 0;
end

pc_vga_8x16 pc_vga_8x16 (
    .clk(CLK),
    .col(col),
    .row(row),
    .ascii(ascii),
    .pixel(pixel)
);
endmodule
```

ҚОСЫМША Е



ҚАЗАҚСТАН РЕСПУБЛИКАСЫ

РЕСПУБЛИКА КАЗАХСТАН

АВТОРЛЫҚ ҚҰҚЫҚПЕН ҚОРҒАЛАТЫН ОБЪЕКТІЛЕРГЕ ҚҰҚЫҚТАРДЫҢ МЕМЛЕКЕТТІК ТІЗІЛІМГЕ МӘЛІМЕТТЕРДІ ЕНГІЗУ ТУРАЛЫ

КУӘЛІК

2022 жылғы «19» қаңтар № 22917

Автордың (лардың) жөні, аты, әкесінің аты (егер ол жеке басын куәландыратын құжатта көрсетілсе):
СҚАБЫЛОВ ӘЛІШЕР ӘЛИҰЛЫ, Ибраимов Маргулан Касенович, Сахыбай Тынымбаев, Жексебай Дауреи Мурзатулы, Кожагулов Елдос Токкожаулы

Авторлық құқық объектісі: **ЭЕМ-ге арналған бағдарлама**

Объектінің атауы: **Аппаратный комплекс шифрования данных на основе FPGA**

Объектіні жасаған күні: **04.11.2019**

Құжат түпнұсқалығын http://www.kazpatent.kz/ru сайтының "Авторлық құқық" бөлімінде тексеруге болады https://copyright.kazpatent.kz

Подлинность документа возможно проверить на сайте kazpatent.kz в разделе «Авторское право» https://copyright.kazpatent.kz

ЭЦҚ қол қойылды

Е. Куантыров

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

КАЗАХСКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ ИМ. АЛЬ-ФАРАБИ

УТВЕРЖДАЮ
Член Правления - Проректор
по академическим вопросам
Л.К. Еркинбаева
«13» 09 2023 г.

АКТ
о внедрении завершенной научно-исследовательской работы (этапа) в учебный процесс

Комиссия Казахского национального университета им. аль-Фараби в составе Члена Правления - проректора по научно-инновационной деятельности Айтжановой Ж.Н. и членов: и.о. директора департамента по академическим вопросам Нұрмуханқызы Д., декана физико-технического факультета Бейсен Н.Ә. составила настоящий акт о том что, в 2023-2024 учебном году в физико-технического факультета внедрены результаты научных исследовании на тему «Аппаратно-программное обеспечение шифрования данных на основе программируемых логических интегральных схем» авторов **Тынымбаева С.Т., Ибраимова М.К., Скабылова А.А., Азамат Р.М.** в содержание дисциплины «Защита информации в телекоммуникациях» для образовательной программы «6В06201-Радиотехника, электроника и телекоммуникации»

Форма внедрения (наименование нового курса, спецкурса, раздела лекций, лабораторных занятий, установки, учебную пособия и т.д.): курс специальность	Объем внедрения (количество работ, семинарских часов)	Наименование внедрения (краткое содержание внедренной работы)
Создание нового раздела в дисциплине Наименование нового раздела: "Аппаратно-	Количество академических часов: Лекции: 15 часов Практические занятия: 15 часа	УМКД «Защита информации в телекоммуникациях» для бакалавриата 3-го года обучения содержит: силлабус, содержание

		8. Заключение: Внедрение нового раздела "Аппаратно-программное обеспечение для кодирования сигналов на основе Программируемых логических интегральных схем" в ОП "Радиотехника, электроника и телекоммуникации" является важным шагом для обеспечения подготовки специалистов в области кодирования сигналов и обеспечения надежности передачи данных в радиотехнических и телекоммуникационных системах.

Материалы к настоящему акту рассмотрены на расширенном заседании кафедры физики твердого тела и нелинейной физики физико-технического факультета. «21» августа 2023 г. (протокол № 1)

Председатель комиссии:
Член Правления - проректор по
научно-инновационной деятельности Ж.Н.Айтжанова

Член комиссии:
Директор департамента
по академической работе Д. Нұрмуханқызы

Декан физико-технического
факультета Н.Ә. Бейсен

11.09.2023 г.