الجمهورية العربية السورية

وزارة التعليم العالي والبحث العلمي

جامعة دمشق

كلية الهندسة الميكانيكية والكهربائية

قسم هندسة الإلكترونيات والاتصالات

تقييم فعالية تنفيذ معيار التشفير المتقدم باستخدام شريحة صفيفة بوابات قابلة للبرمجة حقلياً

رسالة مقدمة لنيل درجة الماجستير في هندسة الاتصالات المتقدمة

إعداد:

المهندسة ميرفت الشريف

إشراف:

الدكتور المهندس عادل خضور علي

2022-2021

صفحة قرار الحكم مع توقيع اللجنة

بموجب قرار مجلس البحث العلمي والدراسات العليا رقم /724 / تاريخ 14 / 11/ 2022 القاضي بتأليف لجنة الحكم من السادة:

الدكتور	نضال زيدان	عضواً
الدكتور	محمد ميهوب	عضواً
الدكتور	عادل خضور علي	عضواً ومشرفًا علمياً

تصريح خطي

أنا الموقع أدناه أصرح بعلمي الكامل وقبولي:

1. أن كل ما ينتج عن البحث والأطروحة هو ملكية فكرية ومالية مع جامعة دمشق، وأنني ألتزم بأخذ موافقة الجامعة في حال رغبتي بنشر البحث أو الأطروحة أو جزء منها نصاً أو مضموناً خارج إطار الجامعة (من دور نشر أو مكتبات أو مواقع الكترونية وغيرها من وسائل النشر).

2. أنه لا يوجد أي جزء من هذه الرسالة تم اقتباسه من عملي آخر أو أنجز للحصول على شهادة أخرى في جامعة دمشق أو أية جامعة أو معهد تعليمي داخل أو خارج الجمهورية العربية السورية.

الاسم والتوقيع

ميرفت الشريف

صفحة شكر وتقدير

لا يسعني إلا أن أتقدم بخالص شكري وامتناني لكل من علمني حرفاً، وأخص بالذكر أساتذتي فرداً فرداً، خاصة أساتذة قسم هندسة الإلكترونيات والاتصالات، وأتقدم لأسرتي بالشكر الجزيل لدعمهم، سائلة الله أن يمنَّ عليكم جميعاً بالصحة والقوة والعمر المديد.

كلمة شكر لأستاذي المشرف الذي ساندني ودعمني وحفزني على إنجاز هذا البحث، سائلة الله أن يمنَّ بالصحة والقوة والعمر المديد.

م. ميرفت الشريف

الإهداء

إلى والدي ووالدتي اللذين رهنا حياتهما لتربية ابنتيهما، وحرصا على أن نكون أشخاصاً ناجحين في حياتنا وفي مجال اختصاصنا، راجية الله أن يمنَّ عليهما بالصحة والقوة والعافية والعمر المديد.

إلى زوجي سندي في هذه الحياة بعد الله، سائلة الله أن يحفظه وأولادي ويوفقنا في تربية أطفالنا ليسيروا على درب العلم والمعرفة.

إلى أختي توءمي، راجية الله أن يسدد خطاها في حياتها ويحمي أطفالها ويجعلهم قرة عين لها.

إلى صديقاتي، وأخص بالذكر صديقتي وزميلة الدراسة الدكتورة هلا أمين، راجية الله أن يحقق أمنياتها في الحياة.

الفهارس

Indexes

فهرس المحتوياتTable of Contents

صفحة قرار الحكم مع توقيع اللجنة.....

تصريح خطي.....

صفحة شكر وتقدير .....

الإهداءDedication.....

فهرس المحتويات.Table of Contents 1

فهرس الأشكال List of Figures 4

فهرس الجداول List of Tables..... 7

قائمة المصطلحات Terminology List..... 8

قائمة المختصرات List of Abbreviations 9

المستخلص 10

المستخلص بالإنكليزية.English Abstract..... 12

الإطار العام للبحث 14

مقدمة..... 15

بعض الدراسات السابقة..... 15

مشكلة البحث..... 21

هدف البحث..... 21

أهمية البحث 21

مسوغات البحث 22

محددات البحث..... 22

حدود البحث 22

منهج البحث وأدواته 23

بنية الرسالة.....24

1. الفصل الأول: الأساسيات النظرية المرتبطة بالبحث.....26

1.1. خوارزميات علم التعمية Algorithms of Cryptography.....27

1.1.1 علم التعمية (التشفير).....27

1.1 .2 أنواع التعمية وأمثلة لبعضها..... 28

2.1. خوارزمية معيار التشفير المتقدمAES algorithm.....30

1.2.1معيار التشفير المتقدم 31

3.1. صفيفة البوابات القابلة للبرمجة حقلياً FPGA 36

1.3.1 العناصر المنطقية القابلة للبرمجة 36

2.3.1 مفهوم شريحة صفيفة البوابات القابلة للبرمجة حقلياً ومكوناتها 37

3.3.1 البنية الشاملة العامة لشرائحFPGA 38

2. الفصل الثاني: التنفيذ العملي والنتائج 50

1.2 المخطط العام لخطوات الدراسة 51

2.2 أدوات البحث ومواصفات الحاسوب المستخدم.....52

3.2 المحاكاة 53

4.2 تحليل النتائج.....54

5.2 تقييم أداء تنفيذ خوارزمية AES على ثماني عائلات لشرائح FPGAs.....66

6.2 تعقيد الخوارزمية.....69

7.2 إقرار صلاحية النتائج Validation 78

8.2. خرج الأداة البرمجية Xilinx planAhead عند تنفيذ خوارزمية AES لكل من

الشرائح المختارة.....80

التوصيات والآفاق المستقبلية .Future Works.....85

المراجع.References.....86

الملاحقAppendices 89

الملحق 1: نسخة من البرنامج لخوارزمية (AES).....89

الملحق 2: نسخة من الدراسات المرجعية المعتمدة في مقارنة نتائج البحث.....108

فهرس الأشكال List of Figures

الشكل الصفحة

الشكل (1-1) مخطط توضيحي لآلية التشفير وفك التشفير.....26

الشكل (1-2) خوارزميةIDE A 29

الشكل (1-3) استبدال عناصر المصفوفة بصندوق S-BOX.....31

الشكل (1-4) صندوق S-BOX.....32

الشكل (1-5) عملية الإزاحة.....33

الشكل (1-6) عملية مزج الأعمدة.....33

الشكل (1-7) المخطط الانسيابي لخوارزميةAES.....34

الشكل (1-8) صندوق Inv s-box 35

الشكل (1-9) العلاقة بين درجة تعقيد التصميم والكلفة.....37

الشكل (1-10) اختلاف البنية العامة لشرائح FPGAs باختلاف الشركة المصنعة.....38

الشكل (1-11) البنية العامة لشرائح FPGA.....39

الشكل (1-12) بنية CLB.....40

الشكل (1-13) طريقة توضع كتل CLBs داخل شريحة FPGA.....40

الشكل (1-14) أنواع الشرائح والمصادر التي تحتويها.....41

الشكل (1-15) البنية الداخلية لوحة IOBs.....42

الشكل (1-16) توضع وحدات Banksعلى أطراف شريحة FPGA.....43

الشكل (1-17) البنية الوظيفية لوحدة DCM.....43

الشكل (1-18) توضع أعمدة كتل ذاكرة RAM المدمجة على FPGA.....44

الشكل (1-19) توضع كتل ذاكرة RAM المدمجة على شريحة Xilinx.....45

الشكل (1-20) الوصلات الداخلية للكتل المنطقية.....45

الشكل (1-21) جوامع وضوارب مدمجة.....46

الشكل (1-22) نوى معالجات مدمجة.....47

الشكل (1-23) وحدة تراسل بيانات عالية السرعة.....47

الشكل (1-24) وحدات معالجة الإشارة الرقمية.....48

الشكل (2-1) مخطط عام لخطوات الدراسة.....51

الشكل (2-2) مواصفات المعالج الخاص بالحاسوب المستخدم.....52

الشكل (2-3) مواصفات الذاكرة الموجودة في الحاسب المستخدم وسعتها.....52

الشكل (2-4) نتيجة المحاكاة باستخدام برنامج Modelsim.....53

الشكل (2-5) مقارنة بين أداء شرائح مختلفة FPGAs عند تنفيذها للخوارزمية

المدروسة من حيث التأخير الزمني وعدد الشرائح والتردد الأعظمي والفعالية.....58

الشكل (2-6) مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقة من

حيث المصادر وعدد الشرائح المتوفرة في الشريحة المختارة.....59

الشكل (2-7) مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقةعند تنفيذ خوارزمية

AES على شرائح FPGA مختلفة من حيث التأخير الزمني (نانو ثانية). 61

الشكل (2-8) مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقة من حيث الإنتاجية

عند تنفيذ خوارزمية AES على شرائح FPGA مختلفة.....63

الشكل (2-9) مقارنة نتائج تنفيذ إجرائية مزج الأعمدة للدراسة الحالية مع دراسات سابقة

من حيث عدد الشرائح.....64

الشكل (2-10) مقارنة نتائج تنفيذ إجرائية مزج الأعمدة للدراسة الحالية مع دراسات

سابقة من حيث التأخير الزمني والتردد الأعظمي.....64

الشكل (2-11) مقارنة نتائج الدراسة الحالية مع نتائج دراسات سابقة من حيث الفعالية.....68

الشكل (2-12) المخطط التدفقي لإجرائية progsub.....69

الشكل (2-13) مخطط تدفقي لإجرائية PROGSH.....71

الشكل (2-14) مخطط تدفقي لإجرائية PROGMIX.....73

الشكل (2-15) المخطط التدفقي لخوارزمية البحث.....75

الشكل (2-16) جودة أداء الخوارزمية وفقاً لدرجة تعقيده.....77

الشكل (2-17) خرج الدراسة [8] 78

الشكل (2-18) خرج الدراسة الحالية.....79

الشكل (2-19) المصادر المستخدمة والتأخير الزمني للشريحة spartan-6.....80

الشكل (2-20) المصادر المستخدمة من الشريحة والتأخير الزمني للشريحة Zynq-7000.....80

الشكل (2-21) المصادر المستخدمة من الشريحة والتأخير الزمني للشريحة Virtex-7.....81

الشكل (2-22) المصادر المستخدمة من الشريحة والتأخير الزمني للشريحة Virtex-5 81

الشكل (2-23) المصادر المستخدمة من الشريحة والتأخير الزمني Kintex-7 82

الشكل (2-24) المصادر المستخدمة من الشريحة والتأخير الزمني للشريحة Virtex-4 82

الشكل (2-25) المصادر المستخدمة والتأخير الزمني للشريحة spartan-3.....83

الشكل (2-26) المصادر المستخدمة من الشريحة والتأخير الزمني للشريحة Virtex-6 83

الشكل (2-27) System On Chip FPGA 85

فهرس الجداول List of Tables

الجدول (2-1) مقارنة بين أداء شرائح FPGA عند تنفيذ خوارزمية AES من حيث التأخير الزمني والإنتاجية وعدد الشرائح.....56

الجدول (2-2) مقارنة نتائج الدراسة الحالية مع دراسات سابقة من حيث المصادر.....58

الجدول (2-3) مقارنة نتائج الدراسة الحالية مع دراسات سابقة من حيث التأخير الزمني.....60

الجدول (2-4) مقارنة نتائج الدراسة الحالية مع دراسات سابقة من حيث Throughput 62

الجدول (2-5) مقارنة نتائج مزج الأعمدة للدراسة الحالية مع دراسات سابقة من حيث المصادر.....64

الجدول (2-6) مقارنة نتائج مزج الأعمدة للدراسة الحالية مع دراسات سابقة.....65

الجدول (2-7) مقارنة نتائج الدراسة الحالية مع دراسات سابقة من حيث الفعالية.....67

قائمة المصطلحات Terminology List

المقابل العربي	المصطلح الإنكليزي
A	
معيار التشفير المتقدم	Advanced Encryption Standard
غير متماثل	Asymmetric
خوارزمية	Algorithm
B	
بنك	Bank
C	
علم التشفير	Cryptography
النص المشفر	Ciphertext
D	
التأخير الزمني	Delay
L	
جدول البحث	Look Up Table
P	
مسارات	Pipelining
النص الصريح	Plaintext
T	
الإنتاجية	Throughput
V	
لغة توصيف العتاد للدارات المتكاملة ذات السرعات العالية جدا	VHDL

List of Abbreviations قائمة المختصرات

A	
AES	Advanced Encryption Standard
E	
ECB	Electronic Code Book
D	
DES	Data Encryption Standard
F	
FPGA	Field Programmable Gate Array
L	
LUT	Look Up Table
N	
NO	Number Of
VHDL	Very High-speed integrated circuit Hardware Description Language
S	
SOC FPGA	System On Chip

المستخلص

خلفية البحث:

عرضت الرسالة اختيار العائلة المناسبة من ثماني عائلات لشرائح صفيفة البوابات القابلة للبرمجة حقلياً

لتحقيق خوارزمية معيار التشفير المتقدم بأفضل أداء من حيث الشرائح المستخدمة من المصادر المتاحة والإنتاجية والفعالية.

هدف البحث:

يهدف البحث إلى تقييم أداء تنفيذ خوارزمية معيار التشفير المتقدم عن طريق تطبيق خوارزمية البحث باستخدام شريحة صفيفة البوابات القابلة للبرمجة حقلياً من حيث الشرائح المستخدمة من المصادر المتاحة والإنتاجية والفعالية.

المواد والطرائق: **المواد** اللازمة للبحث: لزم لإتمام البحث حاسوب بالمواصفات الآتية:

الحاسوب المستخدم من نوع DELL مواصفاته

1-CPU : Intel(R) Core(TM) i3-5005U CPU @ 2.00GHZ

2-Memory 4.0 GB DDR3

الطريقة المتبعة في إنجاز البحث هي بتنفيذ الخوارزمية المدروسة باستخدام أدوات برمجية مناسبة، وهي:

1- برمجية Modelsim 10.5 لإنجاز المحاكاة المطلوبة للخوارزمية المدروسة خوارزمية معيار التشفير المتقدم.
2- برمجية Xilinx planAhead14.7 لإجراء التنفيذ العملي لخوارزمية معيار التشفير المتقدم على شريحة صفيفة البوابات القابلة للبرمجة حقلياً وتقييم فعالية الخوارزمية.

النتائج العلمية:

أثبتت النتائج في هذا البحث بعد تنفيذ خوارزميةAES على شرائح عدة FPGAs:

1- إن أفضل نتيجة من حيث النسبة المئوية للشرائح المستخدمة من المصادر المتاحة على شريحة صفيفه البوابات القابلة للبرمجة حقلياً كانت عائلة Virtex-7 بلغت 1.71% كما في الجدول (2-1).

2- إن أفضل نتيجة من حيث قيمة الإنتاجية التي حصلنا عليها عند تنفيذ الخوارزمية المدروسة كانت الشرائح الأفضل هي Zynq-7000, Virtex-7, Kintex-7 ،حيث حققت جميعها قيمة (Mbps)7203.200 المبينة في الجدول (3-4)

3- حققت شريحةKintex-7 أفضل نتيجة من حيث الفعالية، فحققت قيمة(Mbps/slice)4.06

4-حققت الشريحةSpartan-6 عند تنفيذ خوارزميةAES فعالية أفضل عند مقارنتها مع الدراسات السابقة، حيث بلغت 1.48Mbps للدراسة الحالية. في حين بلغت قيمة فعالية الدراسة[3] عند تنفيذها لخوارزمية معيار التشفير المتقدم0.544Mbps

الاستنتاجات من وجهة نظر الباحث

من خلال هذه الدراسة استخلصنا الآتي:

1- إن أفضل نتيجة من حيث النسبة المئوية للشرائح المستخدمة من المصادر المتاحة على شريحة صفيفه البوابات القابلة للبرمجة حقلياً كانت عائلة Virtex-7 بلغت 1.71% كما في الجدول (2-1).

بينما من حيث قيمة الإنتاجية بلغت (7203.200Mbps) التي حصلنا عليها كما في الجدول (3-4)، كما حققت فعالية بلغت (Mbps/slice)4.06 كما في الجدول (3- 7).

2- من حيث مقارنة نتائج الدراسة الحالية مع الدراسات السابقة: وجدنا عند تنفيذ خوارزميةAES باستخدام الشريحةSpartan-6 حققت فعالية أفضل من المحققة لأجل الدراسة[3] ، حيث بلغت الفعالية لأجل الدراسة الحاليةMbps1.48في حين لأجل الدراسة بينما لأجل الدراسة [3] بلغتMbps0.544 ومن ثم حققت الدراسة تحسينًا بمقدار%172.06 بقيمة الفعالية.

3- من حيث مقارنة نتائج الدراسة الحالية مع الدراسات السابقة: وجدنا عند تنفيذ خوارزميةAES باستخدام الشريحةVirtex-6 حققت فعالية أفضل من المحققة لأجل الدراسة[3] ، حيث بلغت الفعالية لأجل الدراسة الحالية Mbps 3.680في حين لأجل الدراسة بينما لأجل الدراسة [3] بلغتMbps 1.440 ومن ثم حققت الدراسة تحسينًا بمقدار% 155.56 بقيمة الفعالية.

الكلمات المفتاحية:

معيار التشفير المتقدم (AES), شريحة صفيفة البوابات القابلة للبرمجة حقليا (FPGA),

الإنتاجية (Throughput), الفعالية(Efficiency).

English Abstract

Research background

This thesis interduces on evaluation the performance of implementation AES algorithm on eight different families of FPGA chips and determine the number of slices we have used and the throughput we have achieved.

Research Objectives

The aim of this thesis is to evaluate the performance of implementation AES algorithm on FPGA chips.

Materials and methods

This thesis needed - A computer with the following specifications:

-CPU: Intel(R) Core (TM) i3-5005U CPU @ 2.00GHZ

-Memory 4.0 GB DDR3

-Two computer programs are the first program MODELSIM 10.5 used for simulation and the second program Xilinx PlanAhead 14.7 used for implementation.

Methods: this thesis depended on experimental method to achieve thesis objective.

The scientific results

In this thesis we have concluded that the best efficient chip was Virtex-7 by achieving the best result in slice utilization % was1.71%, the best value of throughput was 7203.200(Mbps) and the efficiency was 4.06(Mbps/slice).

When we compare the result with latest studies, we find that spartan-6 achieved (1.48Mbps/slice) efficiency which consider enhancement by 172.06%

Compared with [3] study which achieved (0.544Mbps/slice) in efficiency.

When we compare the result with latest studies, we find that Virtex−6 achieved (3.680Mbps/slice) efficiency which consider enhancement by 155.56%

Compared with [3] study which achieved (1.44Mbps/slice) in efficiency

Conclusion from the researcher point of view

In this thesis we have concluded that the best efficient chip was Kintex−7 by

Achieving the best value of the efficiency was 4.06(Mbps/slice).

Keywords

Advanced Encryption Standard (AES), Field Programmable Gate Array (FPGA), Throughput, Efficiency.

الإطار العام للبحث

الإطار العام للبحث

مقدمة عامة

إن التطور السريع الحاصل في مجال الاتصالات والإلكترونيات دفع الباحثين للبحث عن طرائق عدة لحماية البيانات، خاصة تلك التي لها حساسية خاصة، متل: ملفات مهمة أو حسابات بنكية أو بيانات شخصية فردية، خاصة في أثناء انتقالها في وسائل الاتصال المتنوعة، وهذا أدى إلى ظهور خوارزميات متعددة دعيت خوارزميات تعمية (تشفير).

بعض الدراسات السابقة:

دراسة [1] أنجزت عام 2021 بعنوان

FPGA implementation of AES algorithm for high-speed applications

اقترح الباحثون لزيادة الإنتاجية تقنية جديدة في مرحلة حساب s-box أنشئ فيها وحدةSMC

Squarer and Multiplication with and constant block

ضمت كل وحدات التربيع والضرب بوحدة واحدة من أجل تقليل التأخير الزمني للمسار الحرج كما في الجدول الآتي وهذا نتج عنه تحقيق إنتاجية بلغت قيمتها(Mbps)54133 واستخدام للموارد المتاحة على الشريحة بلغ 7200شريحة، بالإضافة لذلك اقترحت الدراسة طريقة جديدة لحساب مرحلة مزج الأعمدة أطلق عليها اسم Pipelined mix column ، وكانت النتائج وفق الجدول الآتي باستخدام شريحةvirtex-5

	Delay(ns)	Area (in LUT)
Conventional s-box	3.345	209
Non pipelining s-box	2.826	125
Pipelining s-box	2.148	108

نتائج إجرائية s-box

	Delay(ns)	Area in LUT
mix-column using convention method	1.401	384
mix-column using pipelining architecture	0.973	128

نتائج إجرائية مزج الأعمدة

دراسة [2] عام2020 بعنوان

FPGA implementation of hardware architecture with AES encryptor using sub-pipelined s-box techniques for compact applications

اعتمد الباحث في حساب مزج الأعمدة على تقنية Vedic multiplier في تنفيذ عملية الضرب بالمصفوفة الثابتة، وذلك لتحقيق أقل استخدام للموارد المتاحة على الشريحة، فحقق النتائج المبينة في الجدول الآتي

Resources	Conventional mix column	Vedic multiplier
Slices	292	175
LUTs	507	336

نتائج إجرائية مزج الأعمدة للدراسة[2]

كما اقترح الباحث طريقة جديدة لحساب s-box حققت النتائج المبينة في الجدول الآتي عن طريق تنفيذ خوارزميةAES باستخدام شريحة Virtex-5

Parameters	Conventional s-box	Proposed s-box
Delay(ns)	1.74	1.62
Fmax (MHz)	571	617.25
Number of slices	37	35
Number of LUTs	71	69

نتائج إجرائية s-box للدراسة [2]

وحققت الدراسة عند تنفيذها على نوعين من الشرائح النتائج المبينة في الجدول الآتي

Family	Slices	Fmax (MHz)	Throughput (Mbps)
Virtex-4	1120	112.37	14383
Spartan-3xc300	1132	67.75	8672

نتائج تطبيق خوارزميةAES

دراسة [3] عام 2020 بعنوان

An efficient AES implementation using FPGA with enhanced security features

في هذه الدراسة اقترح الباحثون آلية جديدة لتنفيذ الخوارزمية أطلق عليها اسم modified AES

اقترح فيها توليد قيم s-box باستخدام مولد تتابع PN sequence generator ، وأن المفتاح الابتدائي المطلوب لتشفير الدخل أيضا يعتمد على خرج مولد التتابع PN sequence generator

حيث حقق التصميم المقترح عن طريق تنفيذه النتائج المبينة في الجدول الآتي

Throughput (Mbps)	Number of Slices	Device
5930	4095	Xc6vlx240t
3030	5566	Xc6slx150

نتائج تطبيق خوارزميةAES

دراسة[4] عام 2020

Design and implementation of pipelined and parallel AES encryption systems using FPGA

اقترح الباحثون تغيير طرائق المعالجة لخوارزميةAES لجعل الطريقة المقترحة في إنجاز الخوارزمية تستخدم أكبر قدر ممكن من المصادر المتاحة على شريحةFPGA المستخدمة، وذلك بهدف جعل تنفيذ الخوارزمية أسرع من قبل، حيث حققت هذه الدراسة النتائج الآتية في الجدول الآتي.

	Parallel processing	Pipelined processing	Normal Processing
NO. Slices	56845	46745	45368
Throughput(cycle/Byte)	0.125	0.75	------

نتائج تنفيذ خوارزميةAES بالدراسة[4]

دراسة [5] عام 2018 بعنوان

Design and Implementation of AES Using FPGA

نفذ الباحث الخوارزمية باستخدام شريحةSpartan-3an لشركة Xilinx باستخدام برمجية Xilinx ISE14.5

حققت الدراسة قيمة إنتاجيةMbps6400.

دراسة [6] عام 2017 بعنوان:

High throughput FPGA Implementation of Advanced Encryption Standard Algorithm.

أنجز الباحثون الدراسة باستخدام عائلتي VIRTEX-5,VIRTEX-6، حيث حققت النتائج الآتية في الجدول الآتي. اعتمد الباحثون على خواص الحقول المنتهية في الرياضيات في إنجاز مرحلة الاستبدال المتمثلة في صندوق S-BOX بالإضافة لتطبيق تقنيات المسارات متعددة المراحل للوصول إلى أعلى قيم للإنتاجية.

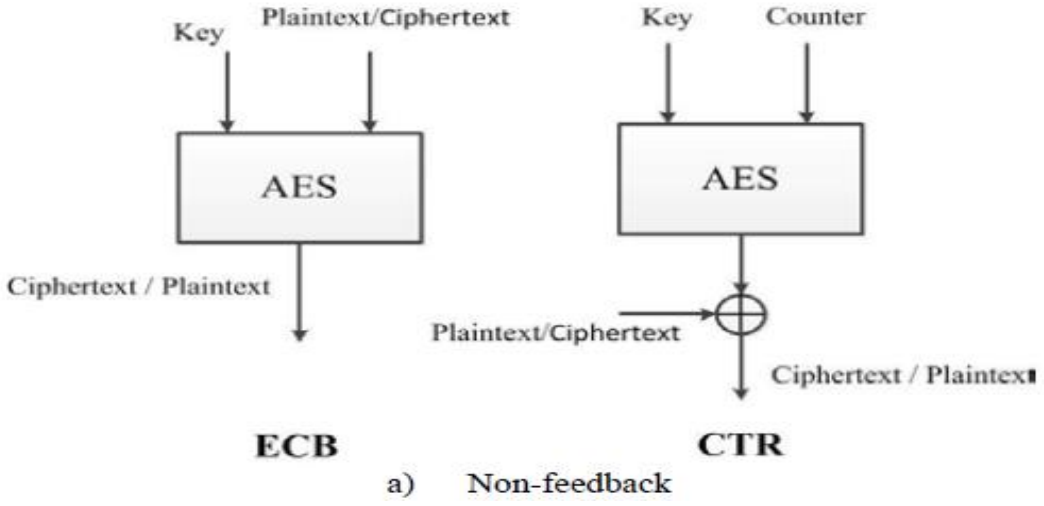
FPGA Family	NO. slices	Delay(ns)	Throughput (Mbps)	Efficiency (Mbps/slice)
Virtex-5	7385	1.56	81680	11.06
Virtex-6	6361	1.17	108690	17.08

نتائج تنفيذ خوارزميةAES وفق الدراسة[6]

دراسة [7] عام 2016 بعنوان

FPGA implementation of the AES−128 algorithm in non−feedback modes of operation

اقترحت الدراسة تنفيذ خوارزمية AES بنمطي عملECB (Electronic Code Book),CTR بدون تغذية خلفية، حيث يسمح هذان النمطان من العمل للبيانات بأن تعالج بشكل متواز، الأمر الذي لا يسمح به نمط التغذية الخلفية في نمط العمل ECB تجري معالجة كل كتلة من النص الصريح المراد تشفيره مباشرة وبشكل مستقل تحول إلى كتلة من النص المشفر الكلي، حيث إن إجرائيات التشفير المتعددة يمكن أن تعمل على التوازي معا كما في الشكل الآتي:

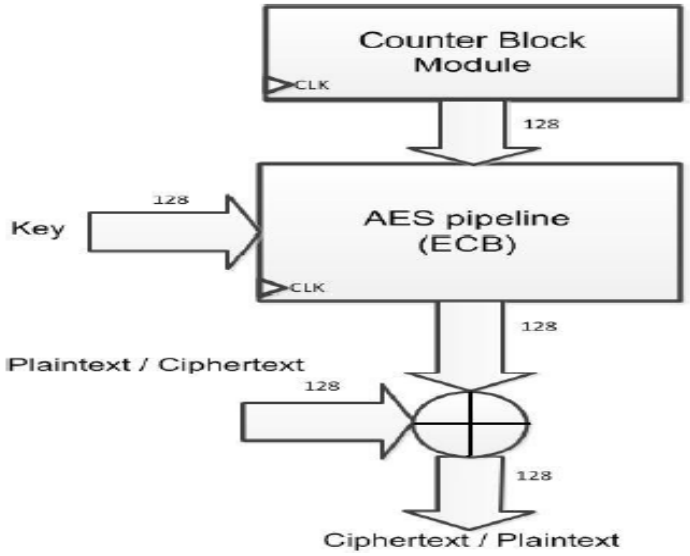


a) Non-feedback

أنماط عمل ECB,CTR بعدم وجود تغذية خلفية

أما نمط العمل(Counter) CTR فيجري فيه تشفير قيمة عداد لتوليد تتابع لكتل الخرج التي تكون خاصة بالنص الصريح المدخل للتشفير، حيث يجب أن تكون كل قيمة للعداد مختلفة لأجل كل كتلة من النص الصريح التي جرى تشفيرها. لذلك اقترح الباحث تصميم وحدة خاصة لكتلة العداد كما في الشكل الآتي، حيث تزود هذه الوحدة بكلمة بطول 128خانة، تنقسم بدورها لقسمين كل منها مكون من 64خانة.

حقق الباحث باقتراح العداد ميزات حماية إضافية عن نمط ECB، لأن الكتل المشفرة دائما تختلف باختلاف قيم العداد المميزة.



خوارزمية AES CTR mode

حققت الدراسة النتائج المبينة في الجدول الآتي باستخدام شريحةVirtex-5 وبرمجيةXilinx

Efficiency (Mbps/slice)	Throughput (Mbps)	Number of LUTs	Number of Slices	Mode operation
3.07	34890	11359	1289	ECB
2.99	34890	11677	1484	CTR

نتائج تطبيق خوارزميةAES على Virtex-5

دراسة [8] عام 2016 بعنوان

High Throughput AES Algorithm Using Parallel Sub bytes and Mix column

في هذه الدراسة اقترح الباحث لتحقيق سرعة أعلى في تنفيذ الخوارزمية تقنية الوصول المتوازي بثماني مراحل لإجرائيتي sub-byte و mix-colum ،حيث حققت هذه التقنية لأجل إجرائيةsub-byte قيمة(ns)1.013

ولإجرائية mix-column قيمة(ns)0.835.

حيث إن تقنية المعالجة المتوازية التي استخدمها الباحث في الدراسة كانت بهدف زيادة قيمة الإنتاجية

حيث بلغت(Mbps)58764 عند تنفيذ خوارزمية معيار التشفير المتقدم على شريحةvirtex-5، وباستخدام موارد متاحة على الشريحة بلغت 6568slice.

دراسة [9] عام 2015 بعنوان

Performance Analysis of AES Cryptographic Algorithm

جرى فيها تنفيذ خوارزمية AES باستخدام عائلة Virtex-5باستخدام اللوح3-xc5vlx110tff1136 حيث حققت استخدام موارد متاحة على الوحدة المستخدمة 1670شريحة، ومن حيث الإنتاجية حققت قيمة(Mbps)4280، وذلك باستخدام برمجية المحاكاةModelSim6.3 وبرمجية للتنفيذXilinx12.2.

مشكلة البحث:

نظرا لأهمية حماية بيانات المستخدمين من أي هجمات متوقعة تعددت خوارزميات التعمية، منها RSA، DIFFI HELLMAN، DES ،ونظرا للثغرة الأمنية في خوارزمية DES التي تتمثل بكون طول مفتاحها قصير 56بت جرى تطويرها لتصبح خوارزمية DES 3حيث شفرت الرسالة بثلاثة مفاتيح لذلك يكون طول المفتاح 3*56 خانة = 168خانة. وفي عام 2008 كسرت هذه الخوارزمية مما دفع الباحثين لتطوير خوارزمية جديدة سميت: معيار التشفير المتقدم، حيث برزت كثير من الدراسات الحديثة في مجال دراسة فعالية خوارزميات التشفير على اختلاف أنواعها. لذلك نقدم في هذا البحث دراسة لتقييم فعالية خوارزمية معيار التشفير المتقدم باستخدام شريحة صفيفة بوابات قابلة للبرمجة حقليا باستخدام الأدوات البرمجية Xilinx Plan Ahead،Modelsim.

هدف البحث:

تعددت خوارزميات التعمية، لكن خوارزمية معيار التشفير المتقدم أثبتت إلى الآن مناعتها ضد الهجمات، مما دفعنا لدراسة تقييم أداء تنفيذ هذه الخوارزمية باستخدام شريحة صفيفة البوابات القابلة للبرمجة حقليا باستخدام برمجيات مناسبة.

أهمية البحث:

تتمثل أهمية البحث بتوفير حماية للبيانات المهمة والحساسة للمستخدمين في أثناء انتقالها بشبكات الاتصال، نظرا لوجود عدد من المتطفلين بهدف سرقة هذه البيانات والتلصص عليها، حيث تؤمن خوارزمية التشفير المتقدم مستوى من الحماية غير قابل للكسر حتى الآن، لذا في هذا البحث سنتناول تنفيذها باستخدام شريحة صفيفه بوابات منطقية قابلة للبرمجة حقليا، ثم نقيم فعالية تطبيق هذه الخوارزمية بعد إنجاز البحث.

مسوغات البحث

نظرا لأهمية الميزات التي تقدمها شريحة صفيفة البوابات المنطقية القابلة للبرمجة حقلياً ولأهمية التشفير لحماية بيانات المستخدمين كان لابد من دراسة تنفيذ أهم خوارزميات التشفير باستخدام شريحة صفيفة البوابات القابلة للبرمجة حقلياً ودراسة فعالية هذا التنفيذ.

محددات البحث

اقتصرت الدراسة على المحاكاة باستخدام أداة برمجية مختصة بدون التطبيق العملي على الشرائح.

حدود البحث

حد زمني: المدة الزمنية المعطاة لإنجاز البحث، حيث نفذ البحث وفق المخطط الزمني المبين في الجدول الآتي:

المدة الزمنية	الخطوات المنجزة
ثلاثة أشهر	دراسة نظرية لعلم التشفير وأنواع خوارزميات التشفير
ثلاثة أشهر	دراسة نظرية معمقة للخوارزمية المدروسةAES
ستة أشهر	دراسة معمقة للغة VHDL
ستة أشهر	كتابة الكود البرمجي لخوارزميةAES
ثلاثة أشهر	إنجاز المحاكاة باستخدام الأداة البرمجيةMODELSIM10.5
خمسة أشهر	1-إنجاز محاكاة التنفيذ العملي باستخدام الأداة البرمجية Xilinx PlanAhead14.7 باستخدام ثماني عائلات لشرائح FPGAs 2-تحليل النتائج وتقييم فعالية تنفيذ خوارزميةAES باستخدام شرائحFPGAs الثمان.

حد علمي: المنهجية المتبعة والبرمجيات المستخدمة في إنجاز البحث.

منهج البحث وأدواته:

اعتمدنا في هذه الدراسة على المنهج العلمي التجريبي، وذلك عن طريق استخدام أدوات برمجية مناسبة وهي (ModelSim10.5,Xilinx Plan Ahead14.7) ولغة برمجية مناسبة VHDL لتنفيذ الخوارزمية المدروسة وإجراء اختبار الأداء على ثماني شرائح مختلفة من شرائح صفيفة البوابات القابلة للبرمجة حقلياً وتحديد الأفضل.

بنية الرسالة

رتبت موضوعات الرسالة كالآتي:

خصص الإطار العام للبحث لعرض الفقرات الآتية: بعض الدراسات السابقة، مشكلة البحث، هدف البحث وأهميته ثم مسوغات البحث ومحددات وحدود البحث وأخيراً منهج البحث وأدواته.

الفصل الأول خصص للحديث عن الأساسيات النظرية المتعلقة بالبحث، فتناولنا فيه الحديث عن خوارزميات التشفير وأهم أنواعه، ثم تناولنا شرح أساسيات نظرية لخوارزمية معيار التشفير المتقدم.

أخيرا تحدثنا عن شرح أساسيات نظرية لمصفوفة البوابات القابلة للبرمجة حقلياً.

الفصل الثاني خصص للإجراء العملي، فتناولنا فيه الخطوات التي أنجزنا فيها البحث والنتائج العملية التي حصلنا عليها، وأخيرا تقييم الأداء لتنفيذ معيار التشفير المتقدم على شرائح صفيفة البوابات القابلة للبرمجة حقلياً.

الفصل الأول:

الأساسيات النظرية المرتبطة بالبحث

Theoretical Basic Related to the Research

الفصل الأول:

الأساسيات النظرية المرتبطة بالبحث

مقدمة

مفهوم الخوارزمية (Algorithm) قديم، فقد وضع البابليون في عهد حمورابي (حوالي 1800ق. م) أول توصيف لقواعد حل بعض أنواع المعادلات، واقتصر مفهوم الخوارزمية في تلك الحقبة على أنها طريقة حسابية لحل المسائل. وقد عرف الإنسان القديم كثيراً من الخوارزميات لحل مسائل الأعداد، مثل خوارزمية إقليدس لإيجاد القاسم المشترك الأعظم لعددين صحيحين. جاءت كلمة خوارزمية من اسم العالم الإسلامي محمد بن موسى الخوارزمي، وهو عالم رياضي من القرن التاسع الميلادي. تطور مفهوم الخوارزمية مع الزمن واستخدم في الرياضيات ليعني مجموعة من الخطوات الرياضية لحل مسالة معينة[10] . فإذا كانت خطوات الحل محدودة تكون الخوارزمية منتهية، ومن ثمّ هناك حل للمسألة، لكن إن كانت خطوات الحل غير محدودة فالخوارزمية غير منتهية.

نعرف الخوارزمية: بأنها مجموعة من الخطوات المنطقية التي يجرى تنفيذها بحسب ترتيب محدد، تصف بدقة ووضوح طريقة عامة لحل مسألة معينة.

واعتماداً على مفهوم الخوارزمية يمكن أن نحدد بعضاً من الخصائص الأساسية للخوارزمية، منها:

1- يجب أن تكون خطوات الخوارزمية مرتبة ومنتهية.
2- يجب أن تنفذ خطوات الخوارزمية أعمالاً بسيطة.
3- يجب أن تكون الخوارزمية معرفة جيداً، وألاَّ تكون غامضة.
4- يجب أن تكون طريقة الحل عامة، فكثير من المسائل يمكن حلها بطرق خاصة بها ولا يمكن الاستفادة منها عند حل مسائل أخرى، مثل هذه الطرق لا تعد خوارزميات.
5- يجب أن تكون الخوارزمية فعالة وغير معقدة من حيث الخطوات، أي يجب ألاَّ تكون تكلفتها عالية من حيث زمن المعالجة والمساحة المشغولة في الذاكرة الرئيسة للحاسوب.
6- يجب أن تكون الخوارزمية قابلة للتعبير بطرق عدة كتابياً، بالإضافة للمخطط التدفقي.

1.1. خوارزميات علم التشفير:

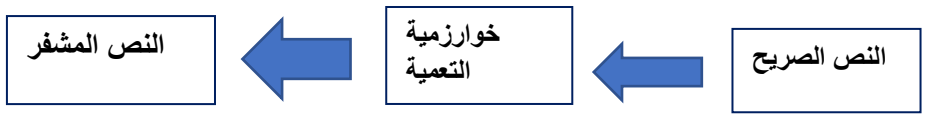
1.1.1. علم التشفير Cryptography:

عرف هذا العلم منذ القدم، حيث استخدمه الإنسان في المجال الحربي والعسكري لحماية رسائله من الوقوع في أيدي الأعداء، فيذكر أن أول من استخدمه الفراعنة للتراسل بين قطاعات الجيش، كما استخدمه يوليوس قيصر في حماية رسائله بين فرق الجيش. ومن الرياضيات جاءت كلمة التعمية، وفي اللغات اللاتينية جاءت بمعنى كلمة سيفر. وفي القرون الماضية اقتصر علم التشفير على أمن المعلومات العسكرية والمراسلات الدبلوماسية وحماية الأمن الوطني، لكن في عصرنا الحديث مع تطور علم الاتصالات توسع مفهوم علم التعمية ليشمل مفهوم عدم الاختراق ومنع التجسس والقرصنة الالكترونية وتوفير سبل التجارة الالكترونية.

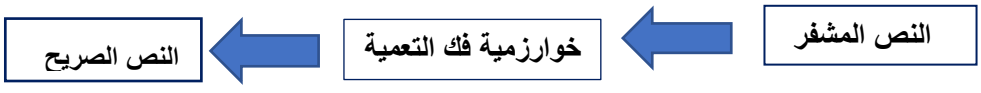
علم التشفير: هو الحقل المهتم بالتقنيات اللغوية والرياضية لتحقيق أمن المعلومات، خاصة في عملية الاتصال، فهو علم يبحث عن تشفير معطيات حساسة وتحليلها.

فالتشفير نعرفه أنه طريقة لتحويل المعلومات من صيغة يفهمها جميع المستخدمين وتسمى النص الصريح (Plaintext) إلى صيغة أخرى تسمى النص المشفر (Ciphertext)لا يفهمها إلاَّ المستخدم الذي لديه معلومات خاصة تسمى مفتاح التشفير(التعمية) تمكنه من فك تشفير تلك المعلومات وفهمها.

من خلال خوارزمية التعمية تجري عملية تحويل النص الصريح لنص معمى وفق الشكل الآتي:



وتجرى عملية الإظهار للنص المعمى وفق المراحل الآتية:



الشكل (1-1) مخطط توضيحي لآلية التشفير وفك التشفير

2.1.1. أنواع التشفير:

تنقسم التعمية إلى نوعين وفق [11],[12] :

2.1.1. 1.التعمية التقليدية (Conventional Cryptography)

تدعى كذلك التعمية المتماثلة (Symmetric Cryptography) ويستخدم فيها مفتاح واحد للتعمية ولإظهار البيانات (Encryption key same decryption key). يعتمد هذا النوع من التشفير على سرية المفتاح المستخدم، فالشخص الذي يملك هذا المفتاح بإمكانه إظهار وقراءتها محتوى الرسائل والملفات، أما الثغرة الكبيرة في هذا النوع تكمن في توفير تبادل مفتاح التعمية بأمان.

من أشهر خوارزميات التعمية:

Caesar Cipher, DES,3DES, IDE, Towfish.RC4, Rijndael (AES)

وتدعى هذه الخوارزميات بخوارزميات التعمية متناظرة المفاتيح، لكونها تستخدم مفتاح تعمية واحد لعمليتي الإظهار والتعمية. في مثل هذا النوع تتطلب قناة نقل آمنة كما يجب توزيع المفاتيح بسرية تامة.

منظومات التعمية المتناظرة تنقسم إلى كلاسيكية:

أ- مشفرات الاستعاضة: ولها أربعة أنواع الاستعاضة البسيطة، الاستعاضة الهومفونية، الاستعاضة كثيرة الأبجدية، الاستعاضة البوليغرامية.

ب- مشفرات إبدال الموقع.

2.2.1.1 التشفير بالمفتاح العام (Public key Cryptography):

يعرف بالتشفير اللامتماثلة Asymmetric، وفيه يختلف مفتاح التعمية عن مفتاح الإظهار، أي هناك مفتاحان أحدهما عام يرسل لجميع المستخدمين يستخدم لتعمية الرسائل، والآخر خاص شخصي يحتفظ فيه صاحبه لا يرسله لأحد، ويستخدم لإظهار الرسائل المشفرة، وهذه الطريقة جاءت حلاً لمشكلة التوزيع الآمن للمفاتيح في التعمية المتناظرة.

هناك حالياً ثلاث منظومات للتعمية بالمفتاح العام:

1- منظومة العوامل الصحيحة، مثالها خوارزمية RSA.

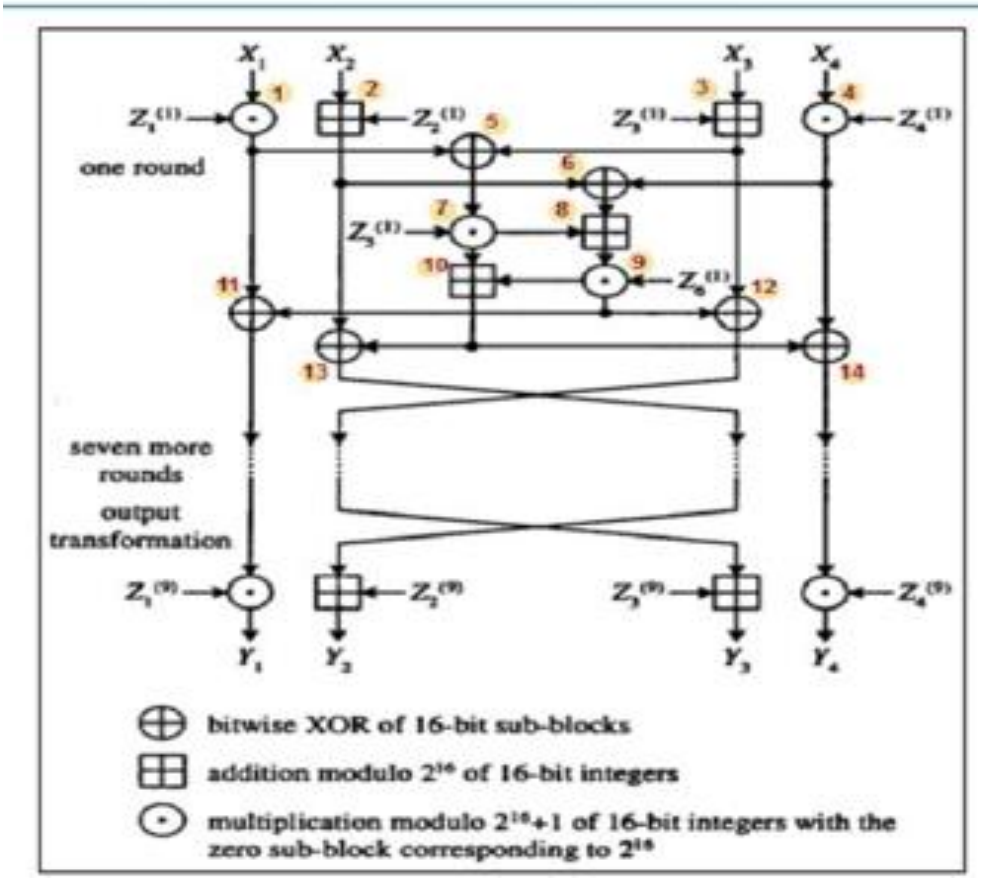
2- منظومة اللوغاريتم المتقطع (المنفصل)، مثالها ElGamal

3- منظومة تعمية المنحني الإهليجي، مثالها Elliptic curve cryptography.

مثال على التشفير المتناظر الخوارزمية العالمية لتشفير المعطيات (IDEA):

وهي مثال للتشفير الكتلي، حيث تعالج كتل النص الصريح المكونة من 64bits، وتحوله إلى كتل نص مشفر (معمى) بطول 64bits وبطول مفتاح 128bits يستخدم نفسه للتعمية وللإظهار. تعتمد IDEA على ثلاث عمليات منطقية: وهي عملية : XOR و الجمع بقياس 2^16 و الضرب قياس 1+(2^16)

تعتمد الخوارزمية على تقسيم النص المراد تعميته (64 bits) إلى 4 كتل كل منها x1,x2,x3,x4 كل منها بطول 16 bits ،و كذلك اشتقاق 52 مفتاحاً فرعياً من مفتاح التعمية الأساسي المؤلف من 128 bits كما أنها تعمل نظام الدورات (ثماني دورات) يتألف دخل كل دورة من نص مؤلف من 64 bits و مقسم إلى أربعة أقسام (كل منها 16 بت) ،و كذلك تستخدم مفاتيح فرعية أيضاً يتألف كل مفتاح من z1,z2,z3,z4,z5,z6 بطول 16 bits.تتألف الخوارزمية من ثماني دورات أساسية وواحدة نهائية تسمى :دورة تشكيل الخرج، حيث تُعد خرج كل دورة دخلاً للدورة الثانية، الشكل (1-2) يلخص الخوارزمية IDEA



الشكل (1-2) خوارزميةIDE A

2.1. خوارزمية معيار التشفير المتقدم

مقدمة:

بعد كسر تشفير خوارزمية التشفير القياسي الأمريكي (DES) بدأت الدعوة من جديد إلى اعتماد خوارزمية جديدة تكون قوية بما يكفي للمستقبل، ففي عام 1997 نشر المركز الوطني للمعايير و التكنولوجيا National Institute of Standards and Technology(NIST) [13] دعوة لتقديم عروض لمعيار تعمية جديد AES يمتلك قوة أمن تساوي أو أفضل من 3DES ،فقد حدد NIST أن AESيجب أن تكون من نوع التعمية المتناظرة وبطول كتل 128 bits ويدعم مفاتيح بأطوال مختلفة من bits (128,192,256) ،قدمت مجموعة من العروض لخوارزميات مقترحة كانت في البداية 21 عرضاً, وخلال تسعة شهور جرى اختيار خمس عشر خوارزمية و ذلك في الجولة الأولى عام 1998 م وفي الجولة

الثانية عام 1999 م رُشحت خمس خوارزميات من أصل خمس عشرة، وبعد كثير من الاختبارات جرى التصويت للخوارزميات الخمس المتأهلة بالترتيب الآتي:

1- خوارزمية رايندول (Rijndael) صوت لها 86 صوتاً إيجابياً و 10أصوات سلبية.
2- خوارزمية سربينت (SERPENT) صوت لها 59صوتاً إيجابياً و 7 أصوات سلبية.
3- خوارزمية تووفيش (TOWFISH) صوت لها 31 صوتاً إيجابياً و 21صوتاً سلبياً.
4- خوارزمية أرسي6 (RC6) صوت لها 23 صوتاً إيجابياً و 37صوتاً سلبياً.
5- خوارزمية مارس (MARS) صوت لها 13 صوتاً إيجابياً و 83 صوتاً سلبياً.

من خلال التصويت جرى ترشيح خوارزمية معروفة باسم (Rijndael) تلفظ رايندول (حيث جاءت التسمية من مزيج أسماء مخترعَيهَا عَالمَي التشفير البلجيكيين:) Dr.Joan Daemen –Dr.Vincent Rijmen وبعد أن حُللت وَاختبرت جرى اعتمادها بوصفها معياراً التشفير من قبل حكومة الولايات المتحدة .وأعلنها المركز الوطني للمعايير و التكنولوجيا (NIST) ونشر المعيار النهائي في نوفمبر 2001 كخوارزمية تشفير قياسية للمعيار (AES)، وغدت الآن واحدة من أكثر خوارزميات المفتاح الواحد قوة وشعبية.

خوارزمية رايندول:

نبدأ بمدخلات هذه الخوارزمية التي تكون عبارة عن النص الصريح المراد تعميته بالإضافة لمفتاح التعمية، حيث يجري تحويل كل من النص والمفتاح إلى التمثيل السداسي عشر، ثم يخزن النص المدخل ضمن مصفوفة مربعة (4*4) تسمى بمصفوفة State، وهي متغيرة بحسب مراحل عملية التشفير، وكذلك يخزن مفتاح التشفير ضمن مصفوفة مربعة (4*4) تسمى مصفوفة المفتاحCipher Key.

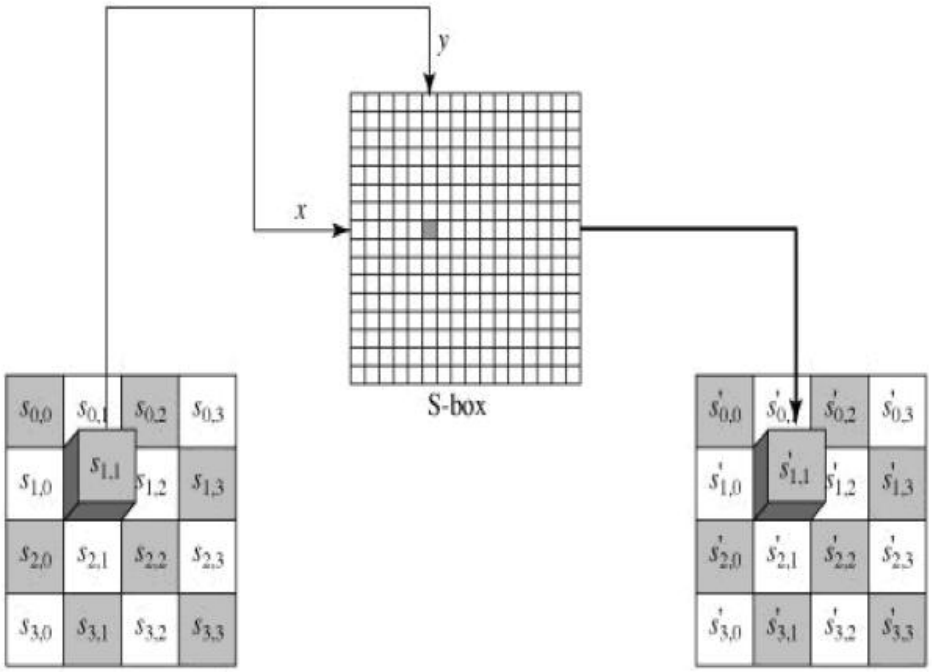
1.2.1. معيار التشفير المتقدم خطوات التشفير وفك التشفير:

1-2-1-1 خطوات التشفير تتألف من ثلاث مراحل كما جاء في[15,14]:

1- المرحلة الأولى: تتضمن الدورة الابتدائية التي يجري من خلالها إضافة مفتاح التشفير إلى النص المدخل باستخدام العملية الثنائية XORعلى عناصر مصفوفتي State و Cipher Key.

2- المرحلة الثانية تتضمن تسع دورات، وكل دورة تحوي أربع تحويلات، هي:

2-1- Bytes_Sub: يجرى من خلالها استبدال عناصر المصفوفة الناتجة من المرحلة السابقة بعناصر من الجدول الجاهز S_box وفق الشكل (1-3).



الشكل (1-3) استبدال عناصر المصفوفة بصندوق S-BOX

وهي عملية استعاضة (استبدال) بسيطة تحول كل بايت إلى قيمة مختلفة، حيث إن هذه المرحلة تعرف 256 قيمة من أجل عملية الاستعاضة الشكل (1-4) حيث يستخدم عناصر مصفوفة الحالة كدليل لكل بايت في جدول الاستعاضة بطول (256بايت)، وتستبدل البايت بالقيم المستخلصة من جدول الاستعاضة، حيث إن محتويات جدول الاستعاضة، ليست قيماً عشوائية بل تُحسب باستخدام صيغة رياضية.

hex	y															
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

x

الشكل (1-4) صندوق S-BOX

2-2- Shift Rows:

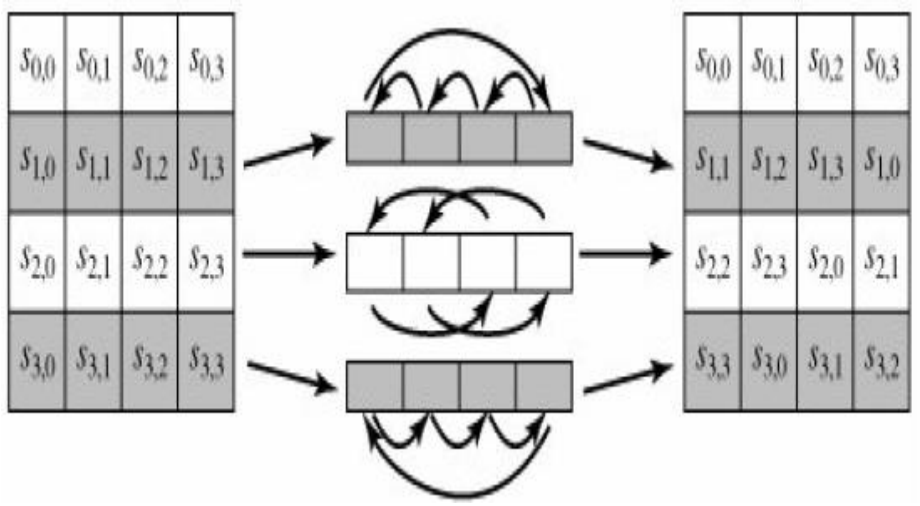
وهي عبارة عن إجراء إزاحة سطرية لعناصر المصفوفة الناتجة عن المرحلة السابقة كما في الشكل (1-5)

السطر الأول: يُدور بمقدار صفر بايت (أي لا يتغير).

السطر الثاني: يُدور بمقدار بايت واحد.

السطر الثالث: يُدور بمقدار بايتين.

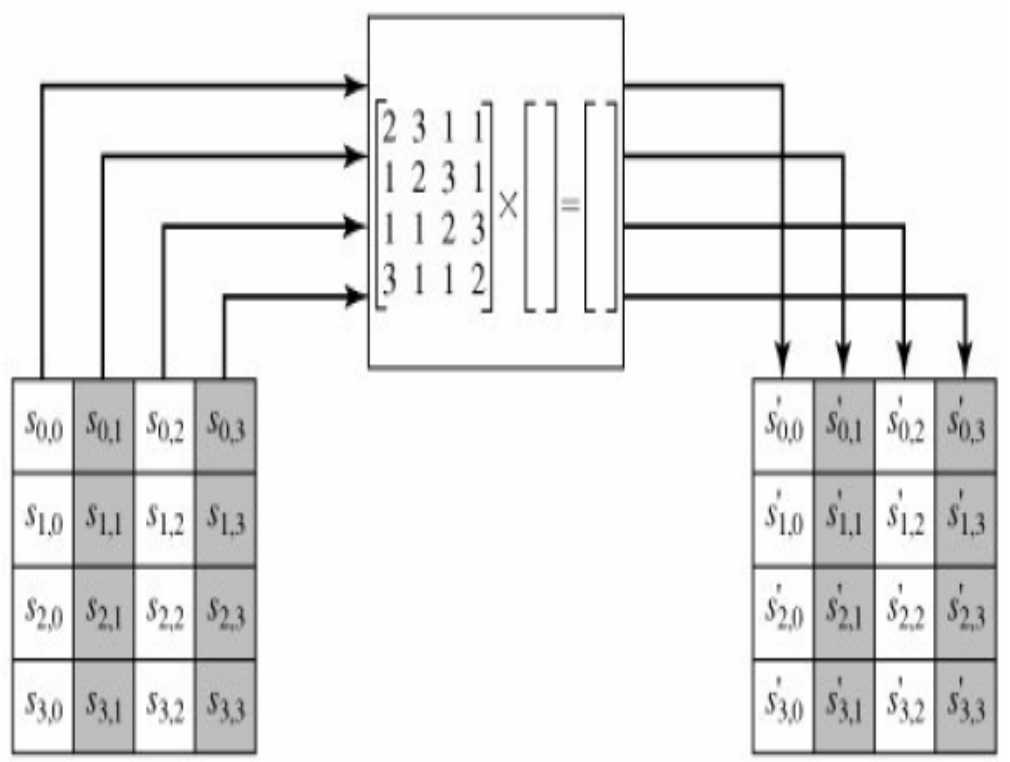
السطر الرابع: يُدور بمقدار ثلاثة بايتات.



الشكل (1–5) عملية الإزاحة

2–3– مزج الأعمدة (Mix columns):

في هذه المرحلة يجري فيها ضرب المصفوفة الناتجة من المرحلة السابقة بمصفوفة مربعة محددة مسبقا كما في الشكل (1–6) الآتي:



الشكل (1–6) عملية مزج الأعمدة

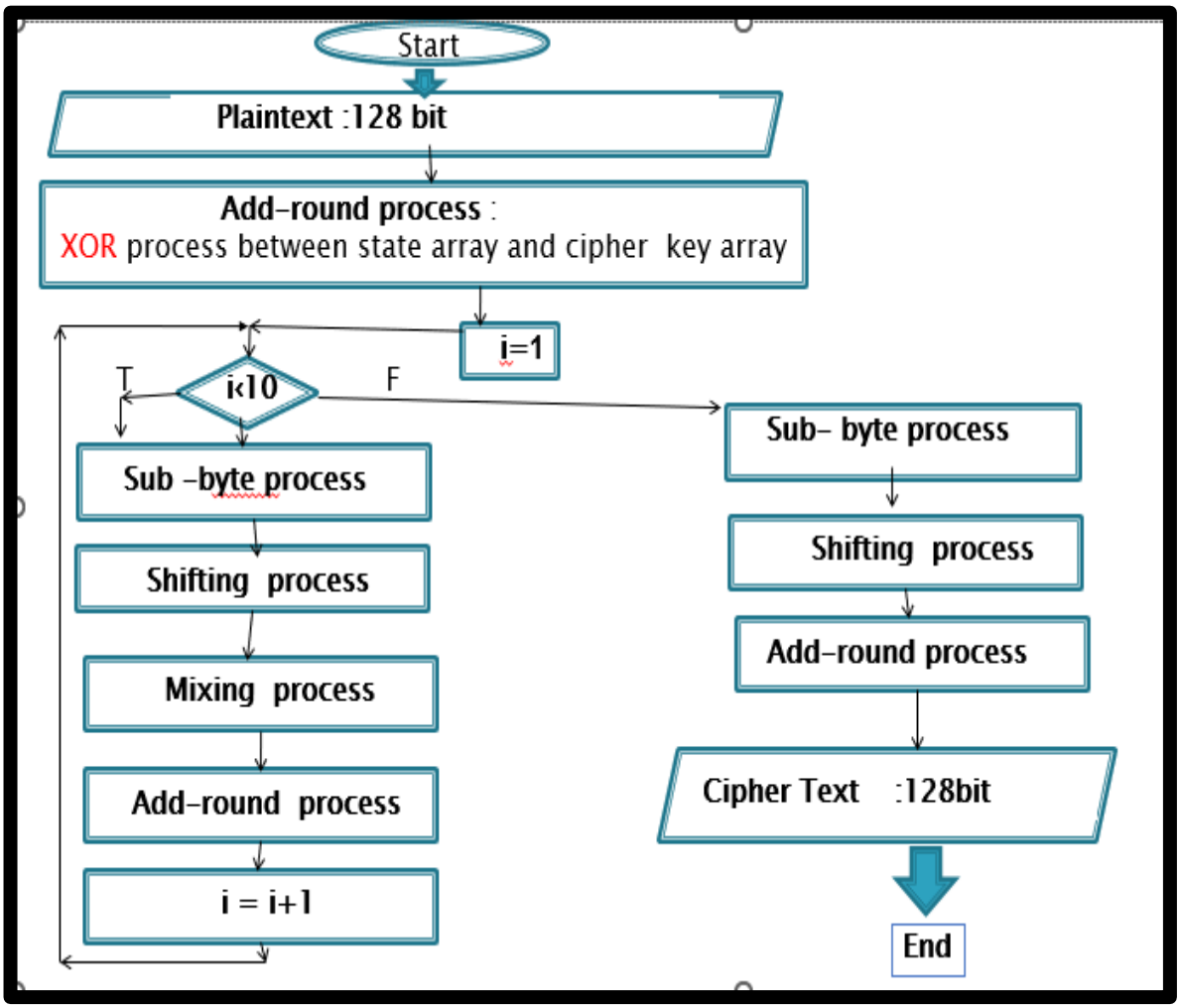
4-2- Add round key:

في هذه المرحلة يضاف مفتاح التعمية للمصفوفة الناتجة من المرحلة السابقة باستخدام عملية xor.

3- الدورة النهائية (الدورة العاشرة): وفيها تنفذ التحويلات نفسها في المرحلة الثانية، عدا عملية مزج الأعمدة، فلا تنفذ في هذه الدورة، ونحصل بنهاية هذه التحويلات على النص المشفر (Cipher text).

المخطط الانسيابي الآتي الشكل (1-7) يبين مراحل تنفيذ الخوارزمية:

حيث i رمز لرقم الدورة في الخوارزمية التي تكون من (1 إلى 10)



الشكل (1-7) المخطط التدفقي لخوارزمية الدراسة AES.

2-1-2-1 – فك التشفير Decryption:

تجري عملية فك التشفير باستخدام معكوس العمليات السابقة وبترتيب معاكس، فهي تتضمن عملية عكس كل العمليات التي تستخدم في التشفير، كما في المراحل الآتية

1-المرحلة الأولى عملية XOR لا تحتاج إلى تابع عكس، لأن إجراء عملية (XOR) مرتين تعطينا القيمة الأصلية.

3- المرحلة الثانية وهي الاستعاضة بصندوق الاستعاضة العكسي Inv s-box تعمل بنفس طريقة Sub bytes لكن باستخدام جدول مختلف يعيد القيم الأصلية أي الجدول المبين بالشكل (1-8).

hex		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

الشكل (1-8) صندوق Inv s-box

2- مرحلة الازاحة العكسية للأسطر نحصل عليها بإجراء التدوير نحو اليمين عوضاً عن اليسار.

3-المرحلة الأخيرة تكون بإجراء عملية مزج الأعمدة العكسية وذلك باستخدام مصفوفة ثابتة هي مقلوب المصفوفة الأصلية المستخدمة في مرحلة مزج الأعمدة وتكون معطاة بالترميز السداسي عشر، حيث إن جداء المصفوفة بمقلوبها يعطينا المصفوفة الواحدية، كما الآتي:

أما بالنسبة إلى عملية مزج الأعمدة العكسية فتستخدم مصفوفة ثابتة هي مقلوب المصفوفة الأصلية المستخدمة في مرحلة مزج الأعمدة وتكون معطاة بالترميز السداسي عشر، حيث إن جداء المصفوفة بمقلوبها يعطينا المصفوفة الواحدية، كما الآتي:

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.1. شريحة صفيفة البوابات القابلة للبرمجة حقلياً

في بداية الثمانينيات لوحظ وجود فجوة هائلة في تسلسل ظهور الدارات المنطقية المتكاملة، فمن جانب نجد العناصر المنطقية القابلة للبرمجة مثل SPLD,CPLD التي كانت قابلة لإعادة التعديل والبرمجة ولا تحتاج إلى زمن كبير للتصميم والتطوير، غير إنها ليست قادرة على دعم وظائف متقدمة ومعقدة، من جانب آخر نجد تقنية ASICالتي تدعم وظائف متقدمة ومعقدة غير أن كلفة التصميم والتصنيع مرتفعة نسبياً، كما إنها تحتاج إلى خبرة كبيرة لتصميمها، بالإضافة إلى عدم إمكانية تعديل التصميم بعد برمجته على الشريحة [16].

في عام 1984أعلنت شركةXilinx عن تطويرها لصنف جديد من العناصر المنطقية القابلة للبرمجة، أطلقت عليه Field Programmable Gate Array اختصاراً (FPGA) حيث كانت شرائح (XC2064) الأولى من شرائح FPGA التي تستخدم خلايا SRAM التي تعتمد تقنية ترانزستورات CMOS وتحوي على بضعة آلاف من العناصر المنطقية.

1.3.1. العناصر المنطقية القابلة للبرمجة (Programmable Logic Devices):

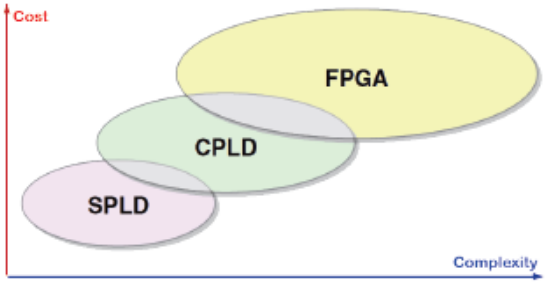
كانت بداية ظهورها أواسط السبعينيات [17] على شكل ذاكرة قابلة للقراءة فقط (PROM) ،والتي كانت بسيطة نوعاً ما، ومع نهاية السبعينيات ظهرت أجهزة أخرى أكثر تعقيداً عرفت بالعناصر المنطقية القابلة للبرمجة PLDs ،حيث قسمت بحسب بنيتها إلى ثلاثة فروع رئيسة:

1- SPLDs (Simple Programmable Logic Devices) :العناصر المنطقية القابلة للبرمجة البسيطة.

2- CPLDs (Complex Programmable Logic Devices) :العناصر المنطقية القابلة للبرمجة معقدة البنية.

3- FPGAs (Field Programmable Gate Arrays) :صفيفة البوابات المنطقية القابلة للبرمجة حقلياً.

تختلف هذه التصنيفات الثلاثة من حيث تعقيد البنية الداخلية، كما تشترك فيما بينها بكثير من الميزات. الشكل (1-9) يبين العلاقة بين درجة تعقيد التصميم والكلفة للتصنيفات الثلاثة:



الشكل (1-9) العلاقة بين درجة تعقيد التصميم والكلفة

2.3.1. مفهوم صفيفه البوابات القابلة للبرمجة حقليا (FPGA):

عبارة عن شريحة متكاملة قابلة للبرمجة حقلياً بعد التصنيع على مستوى الكيان الصلب، لذلك أخذت تسمية قابلة للبرمجة حقلياً، كما تمتاز بكونها ذات مستوى عال من التكامل (VLSI) وتكلفة مقبولة[17] .

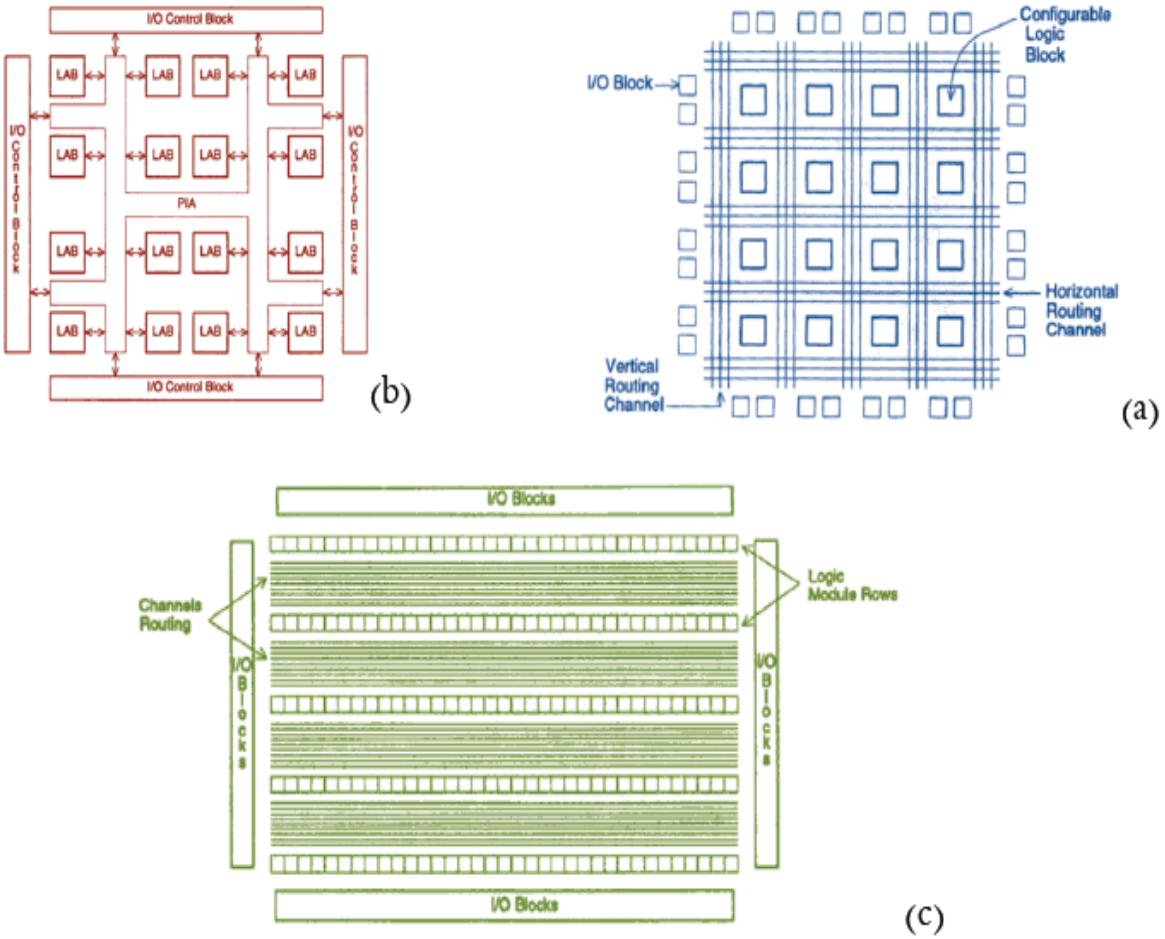
البنية العامة لشرائح FPGA:

تتألف البنية العامة من مجموعة من الكتل المنطقية القابلة للبرمجة والتشكيل تدعى (CLBs or PLBs) [17]،[18]،[19]،بالإضافة إلى مجموعة من الوصلات الداخلية تدعى RMIs التي تصل مجموعة الكتل المنطقية .إن الشكل العام لهذه الكتل مع مجموعة الوصلات يمثل مصفوفة مترابطة تحوي آلاف الكتل المنطقية المترابطة .إضافة إلى ذلك تملك شرائح FPGA وحدات دخل وخرج (IOBs) المسؤولة عن التخاطب مع الإشارات الفيزيائية للوسط الخارجي بالإضافة لمكونات أخرى سنأتي على ذكرها لاحقاً. الشكل (1-10) يبين البنية العامة لشرائح FPGA لشركات عدة مصنعة هي مثلا:

(a) Xilinx

(b) Altera

Actel(c)



الشكل (1–10) اختلاف البنية العامة FPGAs باختلاف الشركة المصنعة

على الرغم من أن تقنية FPGA تشترك في عناصر البنية العامة الجوهرية الموضحة في الشكل (1–10) إلا أن عناصر بنيتها الداخلية تختلف بحسب الشركة المصنعة والتقنية المستخدمة في التصنيع، فهي تمتلك أحجاماً متعددة مختلفة في خصائصها البنيوية والوظيفية.

يبين الشكل (1–11) البنية الشاملة العامة لشرائح FPGA.

3.3.1. البنية الشاملة العامة لشرائحFPGA وهي:

1–CLBs(Configurable Logic Blocks) الكتل المنطقية القابلة للبرمجة

2–IOBs(Input/Output Blocks) وحدات الدخل و الخرج.

3-DCM(Digital Clock Management) وحدة إدارة تردد عمل الشريحة .

4-ERM(Embedded RAM Memory) ذاكرة RAMمدمجة.

5-RMIs(Routing Matrix Interconnects) الوصلات الداخلية للكتل المنطقية.

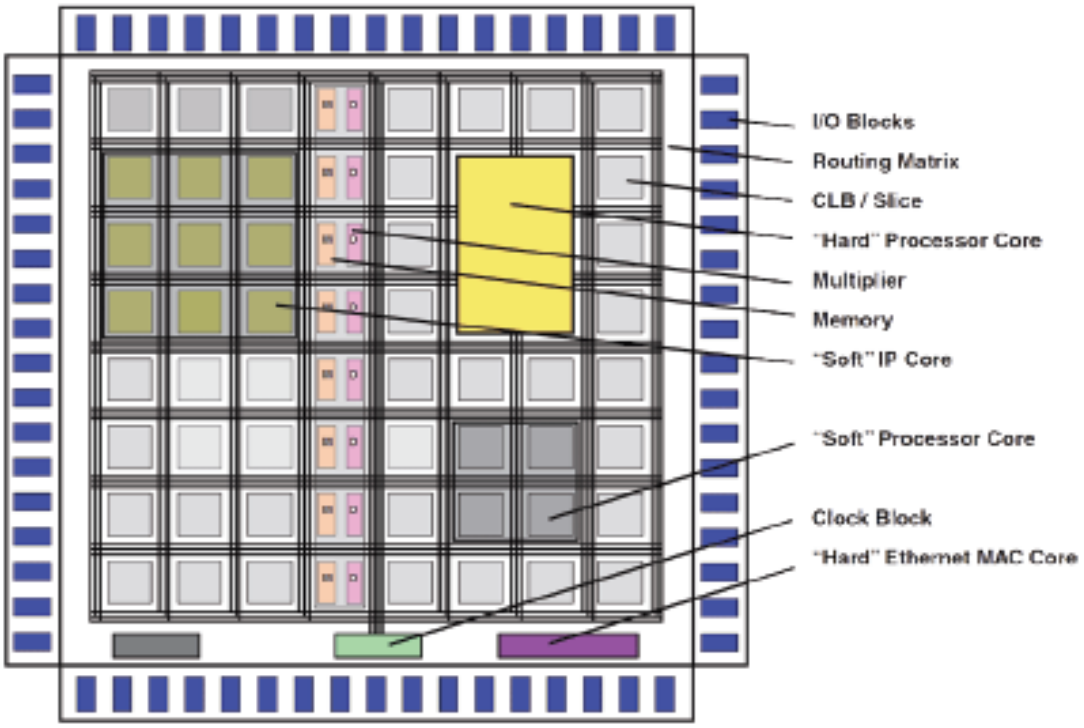
6-EMAs(Embedded Multipliers, Adders) جوامع وضوارب مدمجة .

7-EPCs(Embedded Processor Cores) نوى معالجات مدمجة .

8-GbTs(Gigabit Transceivers) وحدة تراسل بيانات عالية السرعة.

9- IPs (Intellectual Property) وحدات برمجية وظيفية جاهزة.

10- DSPBs (Digital Signal Processing Blocks)وحدات مخصصة لتطبيقات معالجة الإشارة الرقمية.



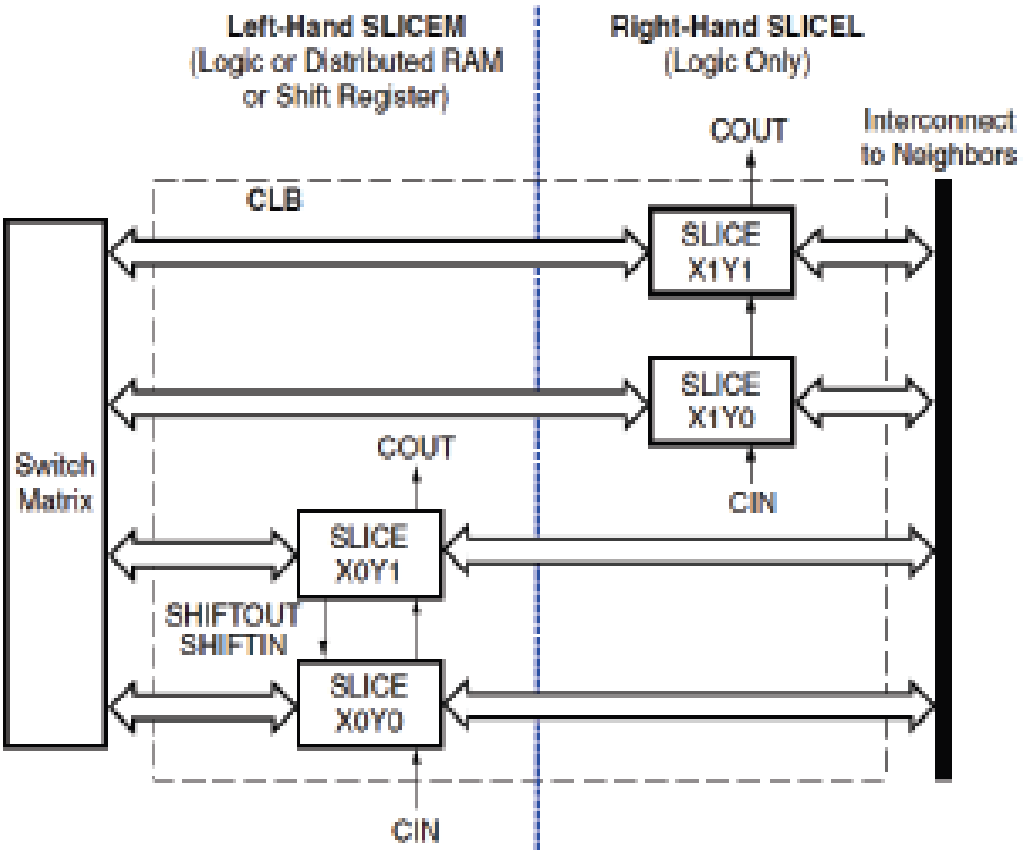
الشكل (1-11) البنية العامة لشرائح FPGA

1-CLBs (Configurable Logic Blocks) الكتل المنطقية القابلة للبرمجة :

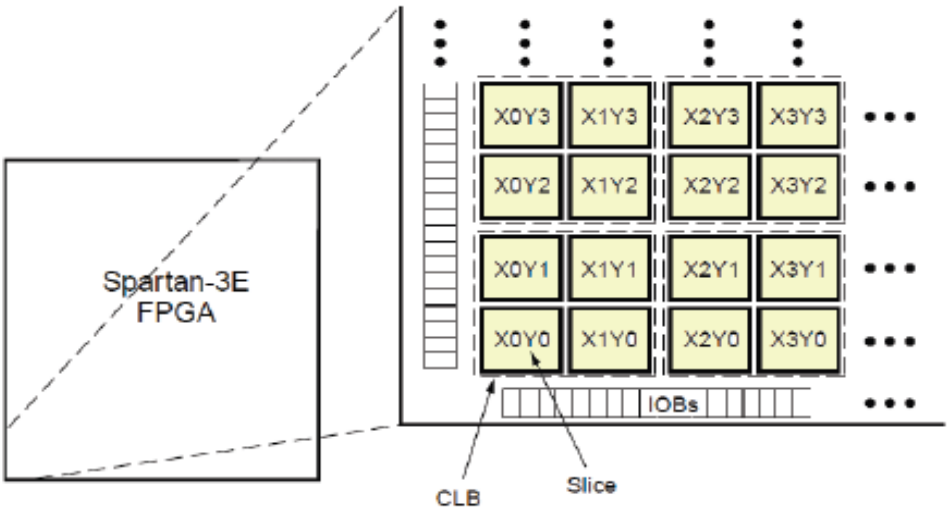
إذ تعتبر الوحدة المنطقية الأساسية في بنية FPGA كما إن عدد هذه الوحدة وميزاتها يختلف من شريحة إلى أخرى، بعض الشركات (Altera) تستخدم تسمية LAB (Logic Array Block) بدلاً من CLB.

تتألف من 4-slice وكل slice يمكن أن تستخدم لتشكيل ذاكرة RAM بعرض1*16(RAM16) أو مسجل إزاحة بعرض (SRL16) 16 bit الذي يمكن أن يعمل المسجل كقلاب أو ماسك.

كما تحوي CLB على دارات منطقية أخرى (Multiplexer, Carry Logic). الشكل (1-12) يبين بنية CLB لشركة Xilinx. يجري توضع الكتل المنطقية القابلة للبرمجة (CLBs) داخل شريحة FPGA على شكل مصفوفة تتألف من أسطر وأعمدة، عدد هذه الكتل ضمن الشريحة يختلف من شريحة إلى أخرى. كما بالشكل (1-13) المبين.



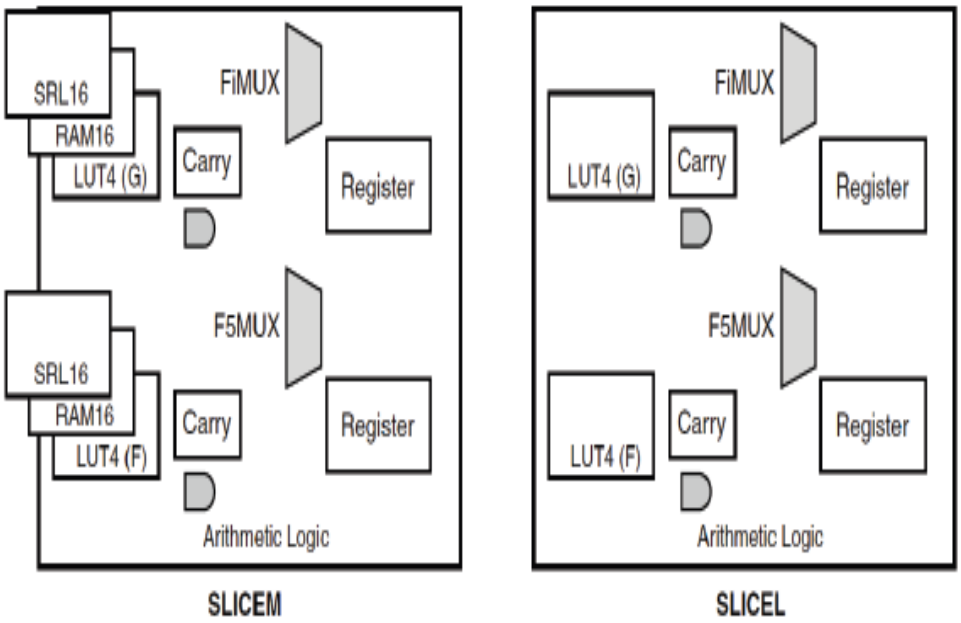
الشكل (1-12) بنية CLB



الشكل (1–13) طريقة توضع كتل CLBs داخل شريحة FPGA

إن كل كتلة منطقية قابلة للبرمجة تحوي أربعة slices موزعة على شكل أزواج، بحيث كل زوج نُظم كعمود مع وحدة منطقية مستقلة للحامل. الزوج الأيسر يسمى SLICEMوهو يدعم الوظائف المنطقية (Logic) والتخزينية (RAM 16,SRL16) كما أن الزوج الأيمن يسمى SLICEL وهو ويدعم فقط الوظائف المنطقية.

الشكل (1–14) يبين المصادر الموجودة على كلا ال SLICES.

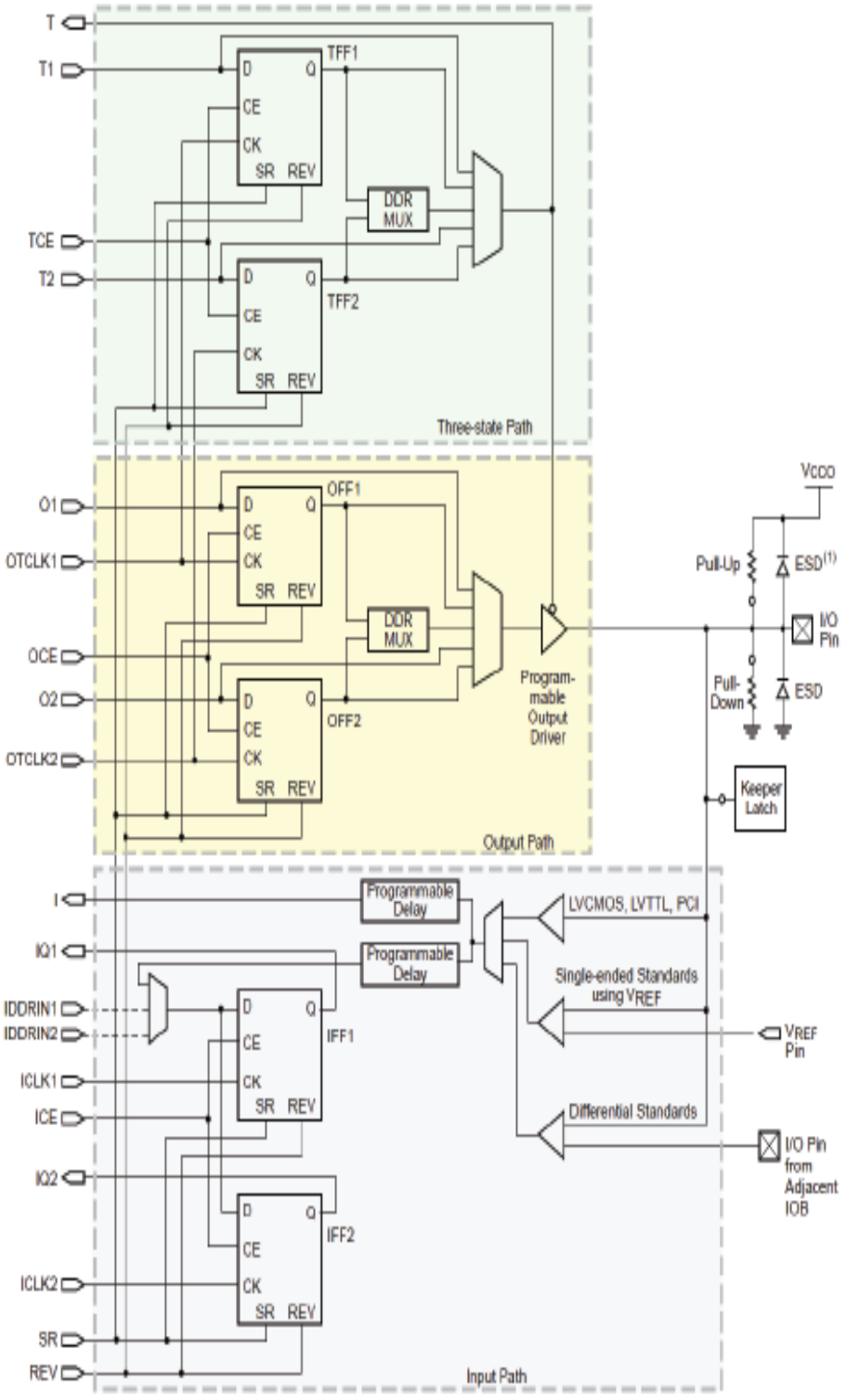


الشكل (1–14) أنواع الشرائح والمصادر التي تحتويها

2-IOBs(Input/Output Blocks) وحدات الدخل و الخرج:

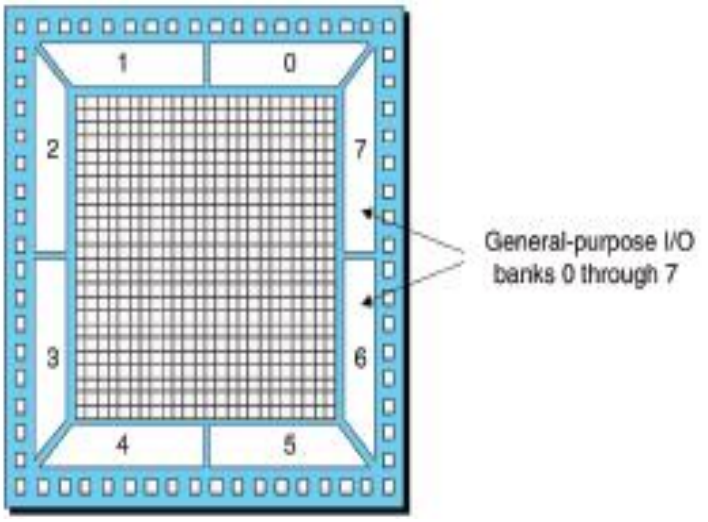
تزود وحدات الدخل والخرج واجهة الاتصال (أحادية أو ثنائية الاتجاه) بين الأقطاب الفيزيائية للشريحة والدارات المنطقية الداخلية. حيث يوجد ثلاث مسارات رئيسة للإشارات:

مسار الدخل (Input path) مسار الخرج (Output path) ومسار حالة ممانعة عالية (3-state path). الشكل (1-15) يبين البنية الداخلية لوحدات الدخل / الخرج. إن الكتل المنطقية لبوابات (IOBs) يجري تنظيمها وفق ما يسمى البوابات على أطراف الشريحة المعنى المماثل لما هو عليه في المتحكمات ويختلف عددها من شريحة إلى أخرى.



الشكل (1–15) البنية الداخلية لوحة IOBs

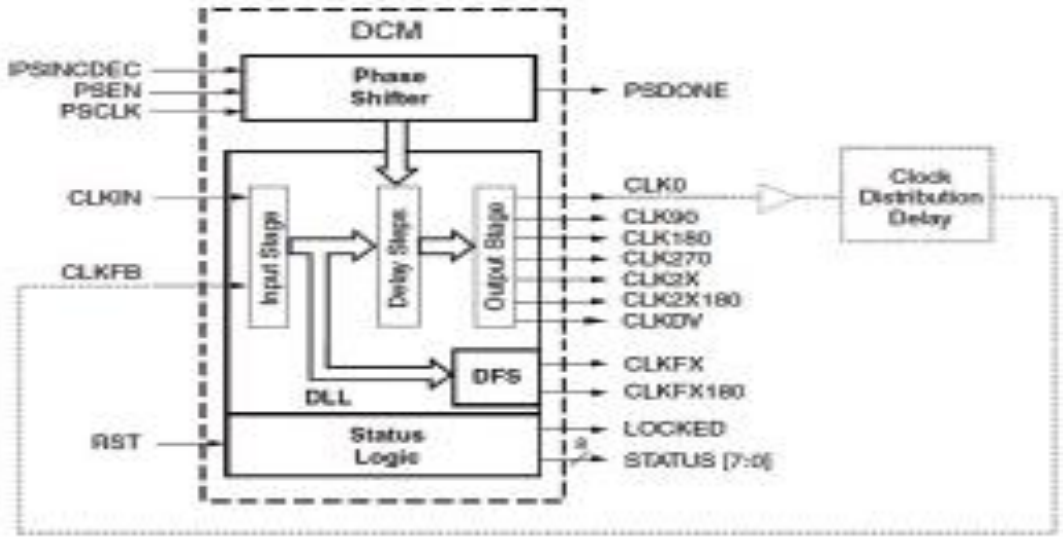
الشكل (1–16) يبين تمثيلاً لتوضع Banksعلى أطراف شريحة FPGA تحوي من صفر إلى سبعة Banks.



الشكل (1-16) توضع وحدات Banksعلى أطراف شريحة FPGA

3-DCM(Digital Clock Management) وحدة إدارة تردد عمل الشريحة :

تتوضع هذه الوحدة غالباً على الحواف العليا و السفلى لشريحة السيليكون، وتتألف من أربع وحدات وظيفية متكاملة مبينة على المخطط الوظيفي المبين في الشكل (1-17),



الشكل (1-17) البنية الوظيفية لوحدة DCM

وهي كالآتي:

- DLL (Delay- Locked-Loop)تقوم على تقليل الإشارات العشوائية والتشوه لنبضات الساعة

-DFS(Digital Frequency Synthesizer) : تستخدم لتوليد تردد إشارة الساعة الداخلية التي هي الجداء بين إشارة الساعة على المدخل CLKIN ومعامل الجداء الذي يحدده المستخدم .

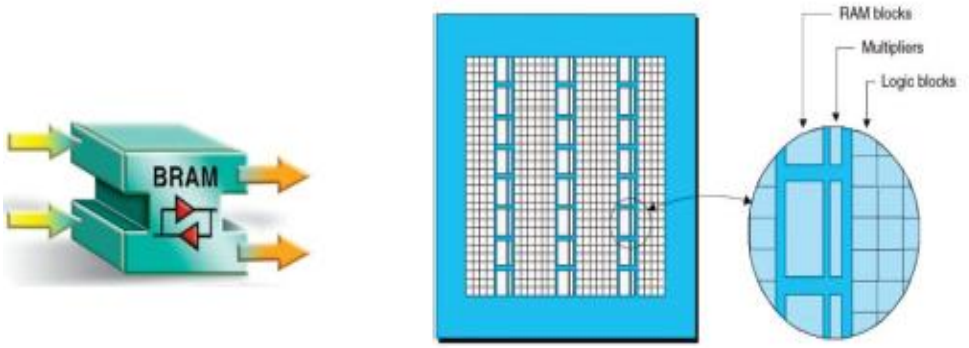
- SL(Status Logic) تستخدم لتشفير وحدة DCM والإشارة إلى الحالة الحالية للوحدة المنطقية .

- PS (Phase Shifter) تستخدم لإزاحة إشارة وفقا لمعاملات إزاحة يحددها المستخدم وهي:

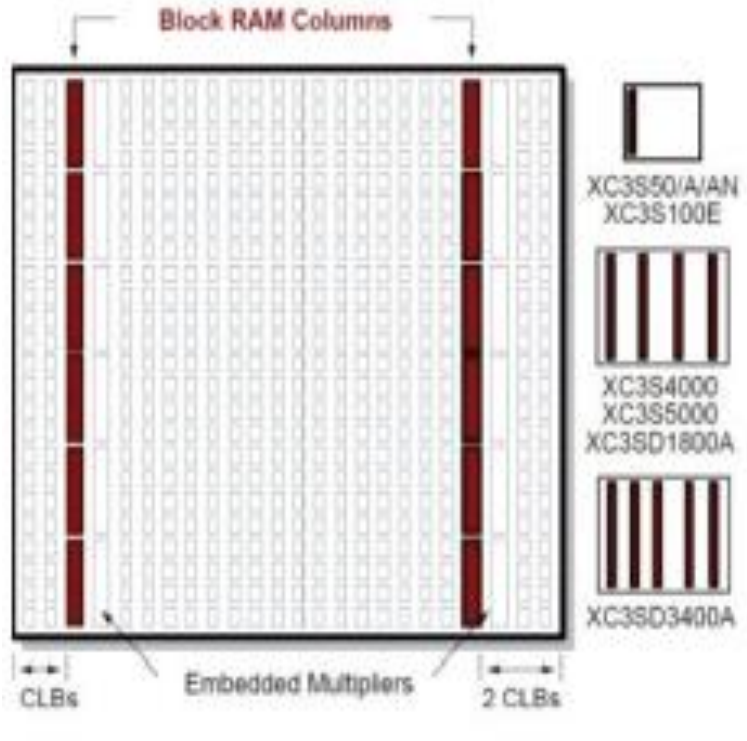
CLK90, CLK0, CLK180, CLK270, CLK2X, CLK2X180, CLKFX, CLKFX180

ERM-4(Embedded RAM Memory) ذاكرة RAM مدمجة:

إن معظم التطبيقات تحتاج بشكل أساسي إلى استخدام الذواكر لذلك تزود شرائح FPGA بوحدات مدمجة من النوع ERAMتتوضع على الشريحة إما على أطراف الشريحة الداخلية وإما أن تنظم في أعمدة بين الكتل المنطقية. كما هو موضح في الشكلين (1-18) و(1-19).



الشكل (1-18) توضع على شكل أعمدة لكتل ذاكرة RAM المدمجة على شريحة FPGA

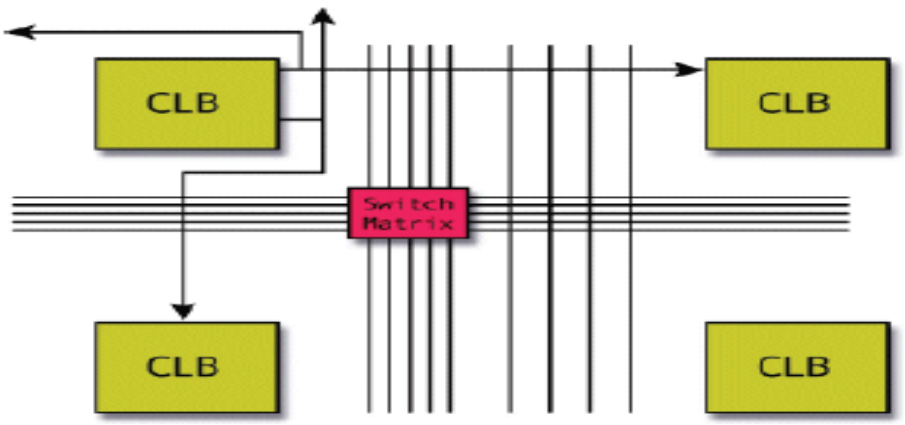


الشكل (1-19) توضع كتل ذاكرة RAM المدمجة على شريحة Xilinx

5-RMIs(Routing Matrix Interconnects) الوصلات الداخلية للكتل المنطقية:

وهي شبكة من مسارات الإشارات (الوصلات) بين مداخل ومخارج الوحدات الوظيفية داخل شريحة FPGA مثل: IOBs,CLB,DCM,BRAMs كما أن هذه الشبكة قابلة للبرمجة، وترتبط فيها الوصلات إلى مصفوفة تبديل تدعى Switch Matrix و تقوم على وصل أنواع مختلفة من الوصلات.

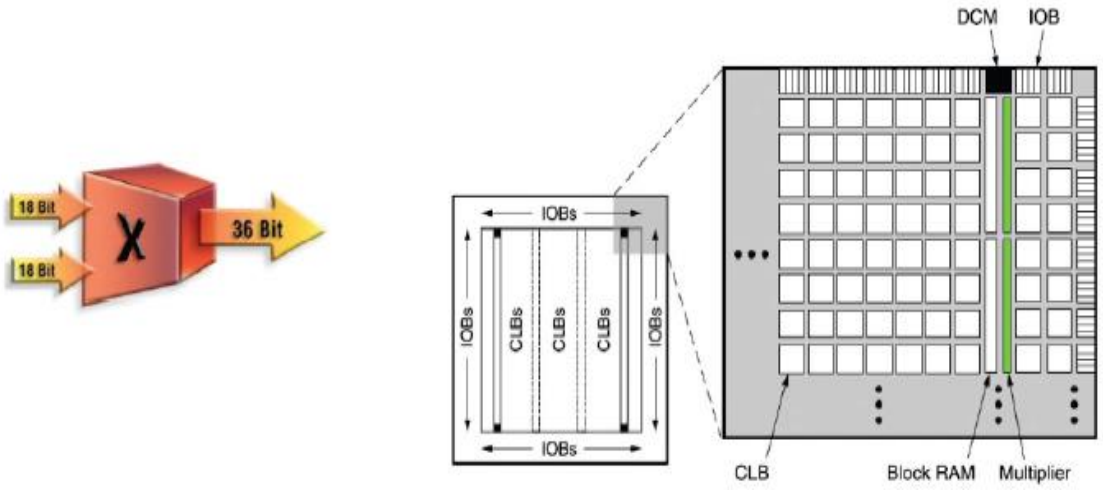
كما في الشكل (1-20)



الشكل (1-20) الوصلات الداخلية للكتل المنطقية

6-EMAs(Embedded Multipliers, Adders) جوامع وضوارب مدمجة :

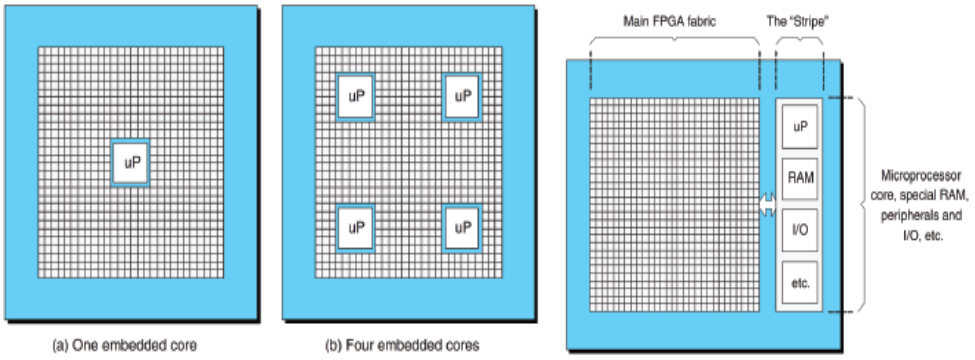
إن بعض التوابع كالضوارب مثلا تكون بطيئة جدا عندما تشكل من وحدات CLBs، ولما كانت عمليات الضرب والجمع تُعد جوهرية في الكثير من التطبيقات لاسيما في تطبيقات معالجة الإشارة فإن كثيراً من شرائح FPGA تتضمن كتلاً منطقية لوظائف الضرب والجمع مدمجة على الكيان الصلب للشريحة تتوضع هذه الكتل بالقرب من كتل الذاكرة RAM المدمجة كما في الشكل (1-21) وذلك لأن هناك ارتباطاً وظيفياً بين الضوارب وكتل الذاكرة.



الشكل (1-21) جوامع وضوارب مدمجة

7-EPCs(Embedded Processor Cores) نوى معالجات مدمجة :

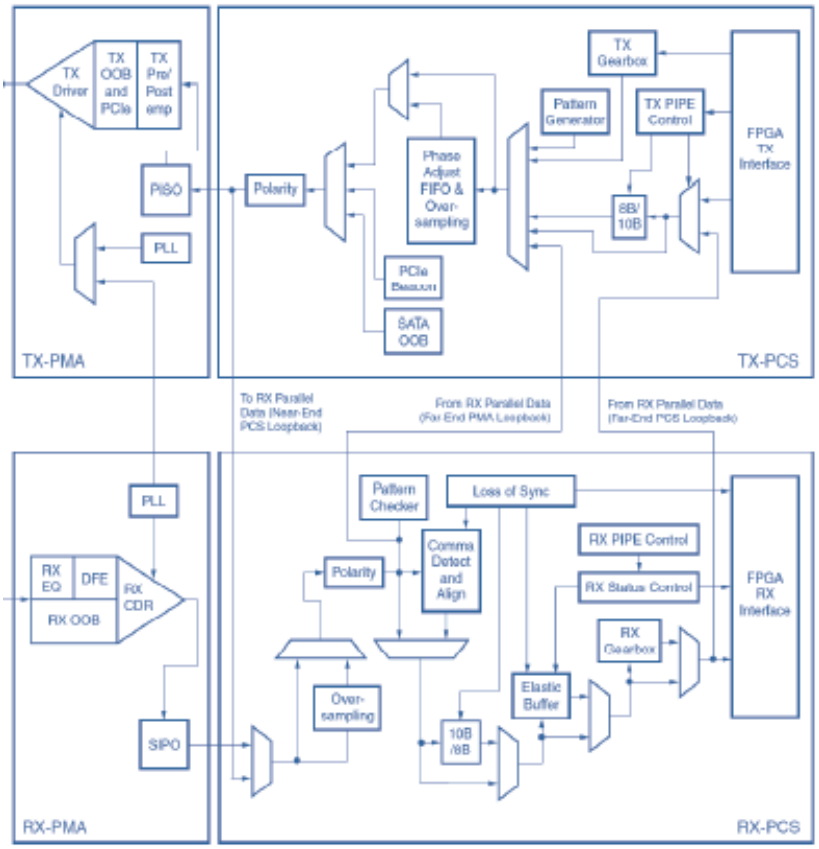
في معظم التطبيقات تظهر الحاجة بشكل أو بآخر إلى وجود معالج مصغر، وحتى وقت متأخر كان يجب وضع المعالج بوصفه عنصراً منفصل على الدارة المطبوعة، أما في شرائح FPGA الحديثة فقد أصبح من الممكن أن تحوي الشريحة على واحد أو أكثر من المعالجات المدمجة، ومن ثُم يستغنى كلياً عن المهام التي تنجز بوساطة معالجات خارجية، وهذا بدوره يحقق كثيراً من المزايا منها: التخلص من تكلفة وجود شريحتين منفصلتين وتقليل نقاط الاتصال والأقطاب على الدارة المطبوعة، ومن ثم تصبح الدارة المطبوعة أصغر حجما وأخف وزنا. كما في الشكل (1-22) حيث a,b تمثل عناصر مدمجة ضمن الكيان الرئيسي لشريحة FPGA في حين أن الشكل الأيمن عناصر مدمجة خارج الكيان الرئيس للشريحة.



الشكل (1–22) نوى معالجات مدمجة

8–GbTs(Gigabit Transceivers) وحدة تراسل بيانات عالية السرعة:

تتضمن شرائح FPGA المتقدمة اليوم كياناً صلباً خاصاً يسمى Transceiver Block مخصصاً لنقل البيانات بسرعات عالية تصل إلى عشرات غيغابت بشكل عام تتضمن هذه الوحدة كتلتين منطقيتين: كتلة مخصصة للإرسال وأخرى مخصصة للاستقبال. كما في الشكل (1–23)



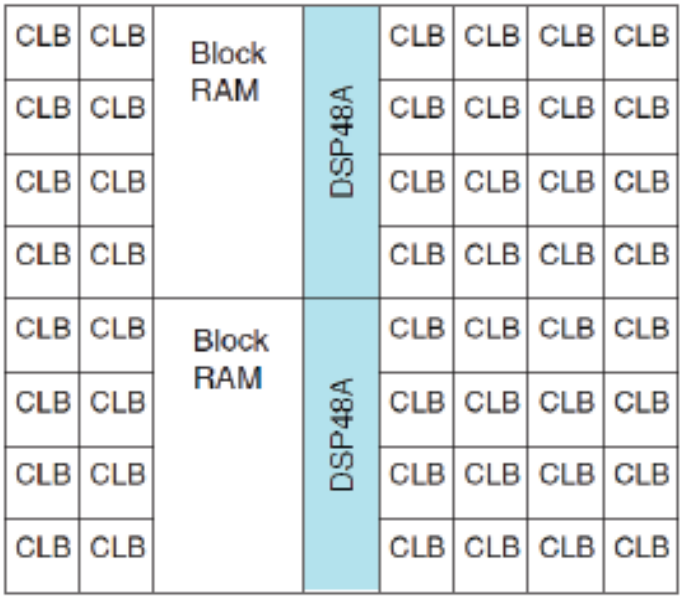
الشكل (1–23) وحدة تراسل بيانات عالية السرعة

9- IPs (Intellectual Property) وحدات برمجية وظيفية جاهزة:

وهي عبارة عن وظائف مستقلة توصف برمجياً باستخدام لغات الكيان الصلب (HDL) بدلاً من بنائها فيزيائياً على الكيان الصلب، إن الهدف من هذه الوحدات البرمجية هو إضافة وظائف منطقية إضافية من دون الحاجة إلى كتابتها برمجياً، و من ثم يمكن بناء نظام معقد بزمن قصير نسبيا من خلال تضمين هذه الوحدات البرمجية في النظام وربط وظائفها. حيث توفر الشركات المصنعة بيئات برمجية لتوليد IPs.

10- DSPBs (Digital Signal Processing Blocks)وحدات مخصصة لتطبيقات معالجة الإشارة الرقمية:
إضافة إلى وحدات الكيان الصلب السابقة فإن بعض شرائح FPGA غالباً ما تكون من العائلات المخصصة لتطبيقات معالجة الإشارة الرقمية تحوي على وحدات كيان صلب DSP الهدف من هذه الوحدات هو دعم الخوارزميات الرياضية المستخدمة في تحليل الإشارة الرقمية، بحيث تكن أسرع ما يمكن بأعلى أداء فعال و أقل استهلاك للطاقة و أقل عدد ممكن من المساحات السيليكونية على شريحة FPGA.حيث تتوضع هذه الكتل بين الكتل المنطقية القابلة للبرمجة و كتل الذواكر المدمجة RAM

كما في الشكل (1-24).



الشكل (1-24) وحدات معالجة الإشارة الرقمية

11- العناصر المنطقية المخصصة (Dedicated Logic)

إضافة لما ذكرنا من العناصر الأساسية المكونة لبنية FPGA فإنها تحوي على عناصر منطقية مبنية في الكيان الصلب للشريحة السيليكونية ومكرسة لعمليات منطقية خاصة بالعناصر الأساسية، هذه العناصر هي:

- Multiplexer Logic: تقوم على الوصل بين LUTs , Slices .
- Carry Chains: تساعد في تسريع العمليات الرياضية .
- Multiplier AND gate: تساعد في تسريع عمليات الضرب للضوارب المبنية بالاعتماد على وحدات LUTs.
- Shift Register LUT: مسجلات إضافية تبنى بالاعتماد على وحدات LUTs.

الفصل الثاني

التنفيذ العملي والنتائج

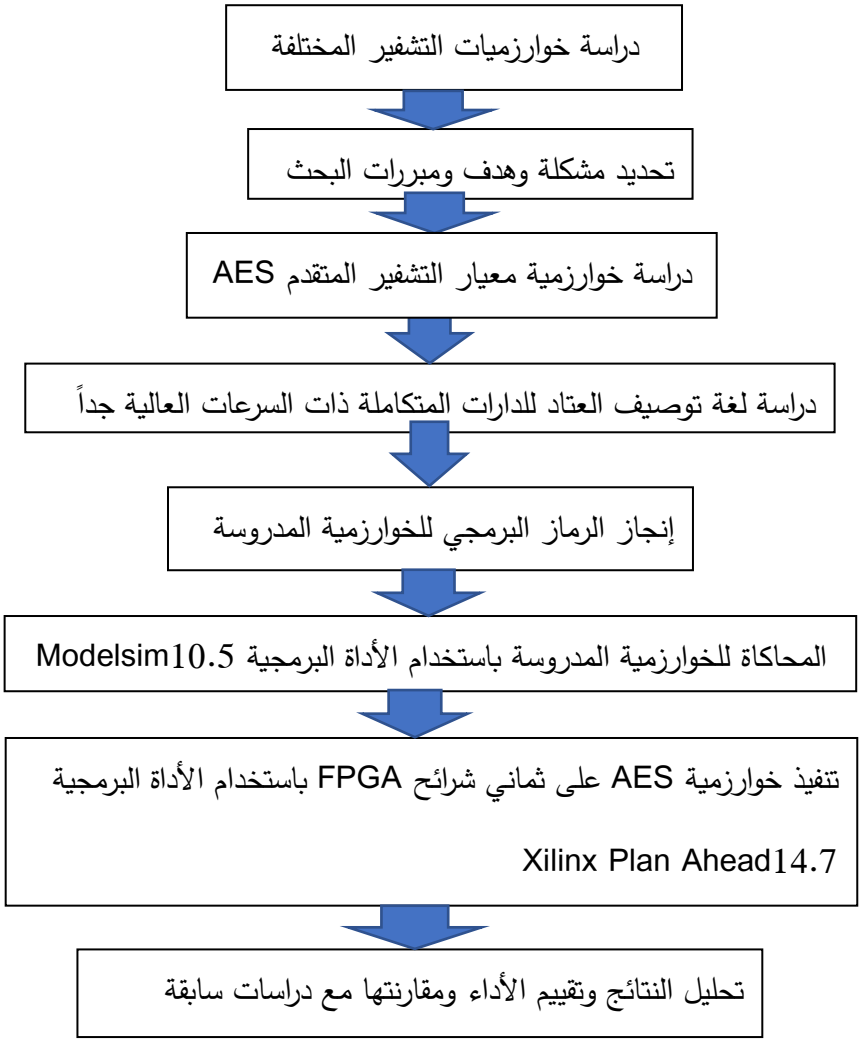
Implementation and Results

الفصل الثاني

التنفيذ العملي والنتائج

1.2 مخطط عام لخطوات الدراسة

يبين الشكل الآتي خطوات الدراسة:



الشكل (2-1) مخطط عام لخطوات الدراسة

2.2 أدوات البحث ومواصفات الحاسوب المستخدم:

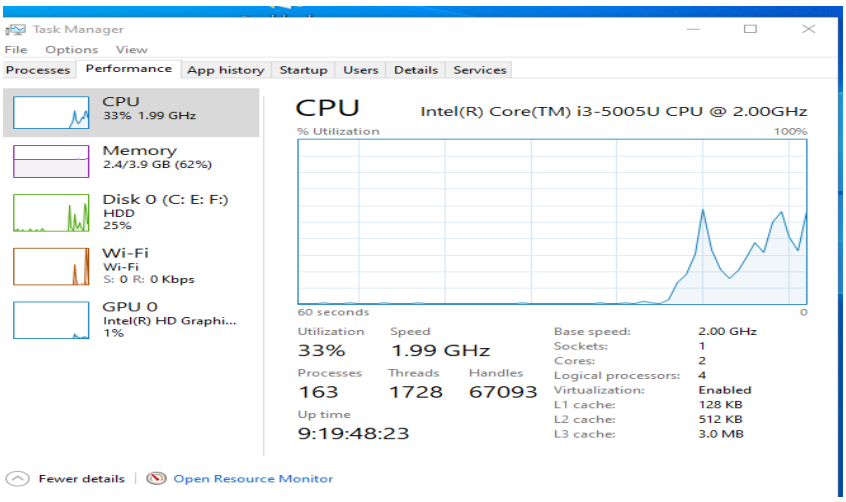
1.2.2. لزم لإتمام البحث أدوات برمجية، وهي:

1- برنامج Modelsim(version10.5) وذلك لإجراء المحاكاة للرماز البرمجي المنجز لخوارزمية الدراسة AES.

2-برنامج Xilinx Plan Ahead (version 14.7) وذلك لتنفيذ الرماز البرمجي المنجز لخوارزمية الدراسة على شرائح مختلفة من FPGAs.

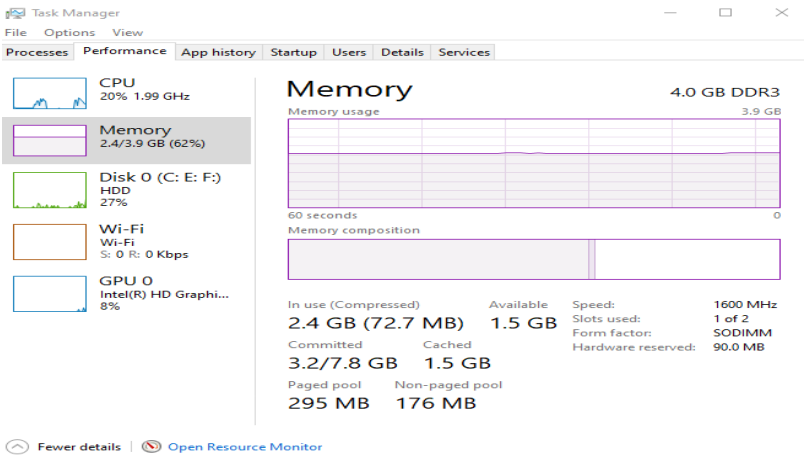
2.2.2. مواصفات الحاسوب المستخدم من نوع DELL

1-CPU: Intel(R) Core (TM) i3-5005U CPU @ 2.00GHZ



الشكل (2-2) مواصفات المعالج الخاص بالحاسوب المستخدم.

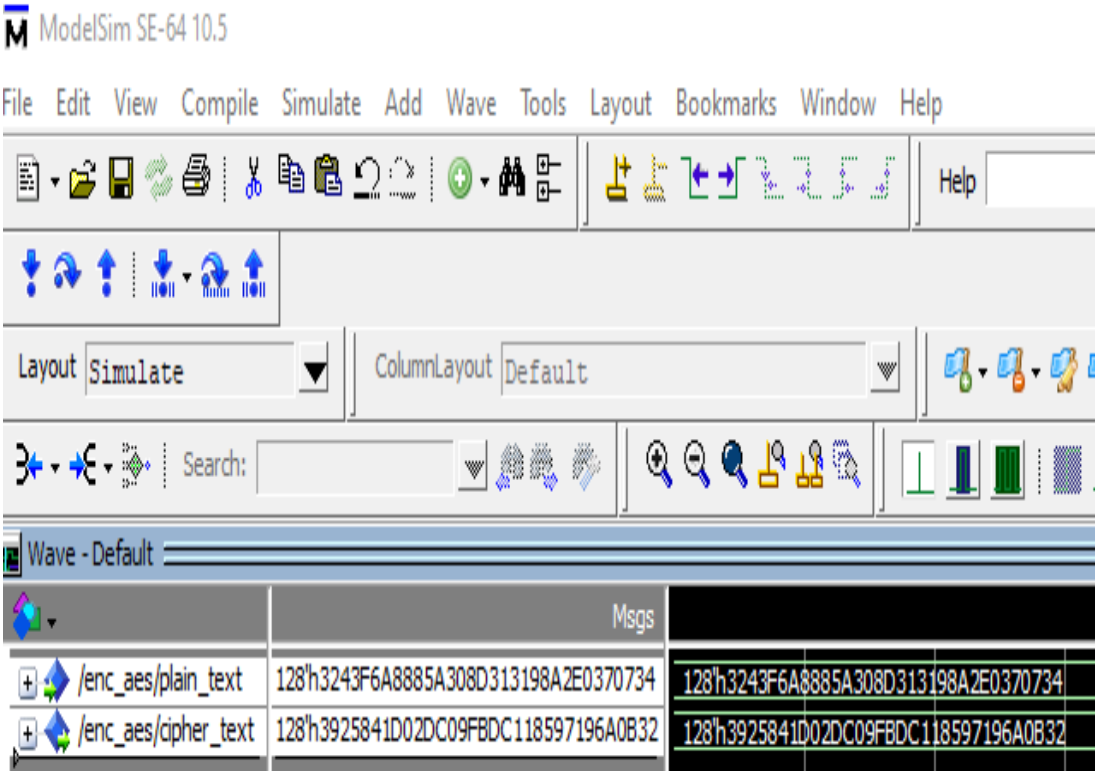
2-Memory 4.0 GB DDR3



الشكل (2-3) مواصفات الذاكرة وسعتها الموجودة في الحاسب المستخدم.

3.2 المحاكاة:

لإجراء المحاكاة للرماز البرمجي المنجز لخوارزمية معيار التشفير المتقدم أُستخدمت الأداة البرمجية Modelsim(version10.5) وكانت النتائج للمحاكاة المنجزة مبينة في الشكل (2-4).



الشكل (2-4) نتيجة المحاكاة باستخدام برنامج Modelsim(version10.5)

2-4- تحليل النتائج:

2-4-1- الإجراء التنفيذي للخوارزمية المدروسة AES:

أُستخدمت الأداة البرمجية(Xilinx Plan Ahead (version 14.7، حيث حصلنا على النتائج المبينة في الجدول (2-1) وذلك بتطبيق الرماز البرمجي المنجز للخوارزمية المدروسة على ثماني شرائح FPGA لشركة Xilinxوذلك لثماني عائلات هي:

1- (Kintex-7) لوح Xc7k480tffg1156-3.
2- (Zynq-7000) لوح Xc7z045ffg900-3
3- (Virtex-7) لوح Xc7vx690tffg1930-3
4- (Spartan_6)لوح Xc6slx150tffg900-3
5- (Virtex-5)لوح Xc5vlx155tffg1760-3
6- (Virtex-4) لوح Xc4vsx55ff1148-12
7- (Virtex-6) لوح Xc6vlx365tf1759-3
8- (Spartan-3) لوح xc3s4000lfg900-4

حققت الدراسة الحالية النتائج الآتية في الجدول (3-1) كما في كل من الأشكال (2-19)، (2-20)، (2-21)، (2-22),(2-24),(2-25),(2-26).

تحليل النتائج وتفسيرها:

1-إن أفضل نتيجة من حيث النسبة المئوية للشرائح المستخدمة من المصادر المتاحة على شريحة صفيفه البوابات القابلة للبرمجة حقلياً كانت عائلة Virtex-7 بلغت 1.71% كما في الجدول (2-1).

2- إن أفضل نتيجة من حيث قيمة الإنتاجية التي حصلنا عليها عند تنفيذ الخوارزمية المدروسة كانت الشرائح الأفضل هي Zynq-7000, Kintex-7,Virtex-7 ، حيث حققت جميعها قيمة (Mbps)7203.200المبينة في الجدول (2-4).

3- حققت شريحةKintex-7 أفضل نتيجة من حيث الفعالية فحققت قيمة(Mbps/slice)4.06.

4-حققت الشريحةSpartan-6 عند تنفيذ خوارزميةAES فعالية أفضل عند مقارنتها مع الدراسات السابقة

حيث بلغت 1.48 Mbps/slice للدراسة الحالية في حين بلغت قيمة فعالية الدراسة[3] عند تنفيذها لخوارزمية معيار التشفير المتقدم 0.544 Mbps/slice من ثم حققت الدراسة الحالية تحسيناً بمقدار % 172.06.

5-حققت الشريحة Virtex-6عند تنفيذ خوارزميةAES فعالية أفضل عند مقارنتها مع الدراسات السابقة حيث بلغت 3.680 Mbps/slice للدراسة الحالية في حين بلغت قيمة فعالية الدراسة[3] عند تنفيذها لخوارزمية معيار التشفير المتقدم 1.44 Mbps/slice من ثم حققت الدراسة الحالية تحسيناً بمقدار % 155.56.

الاستنتاج

من خلال نتائج هذه الدراسة استخلصنا الآتي:

1-عند مقارنة نتائج البحث ضمن العائلات الثمان المنفذ باستخدامها خوارزمية البحثAES :

استخلصنا أن عائلةVirtex-7 كانت الأفضل من حيث النسبة المئوية لاستخدام الشرائح المتوفرة فبلغت 1.71% كما في الجدول (2-1)، أما من حيث قيمة الإنتاجية فبلغت (7203.200Mbps) كما في الجدول (2-4) لأجل العائلات Virtex-7,Kintex-7,Zynq-7000، بينما بلغت الفعالية (Mbps/slice)4.06 كما في الجدول (2- 7).

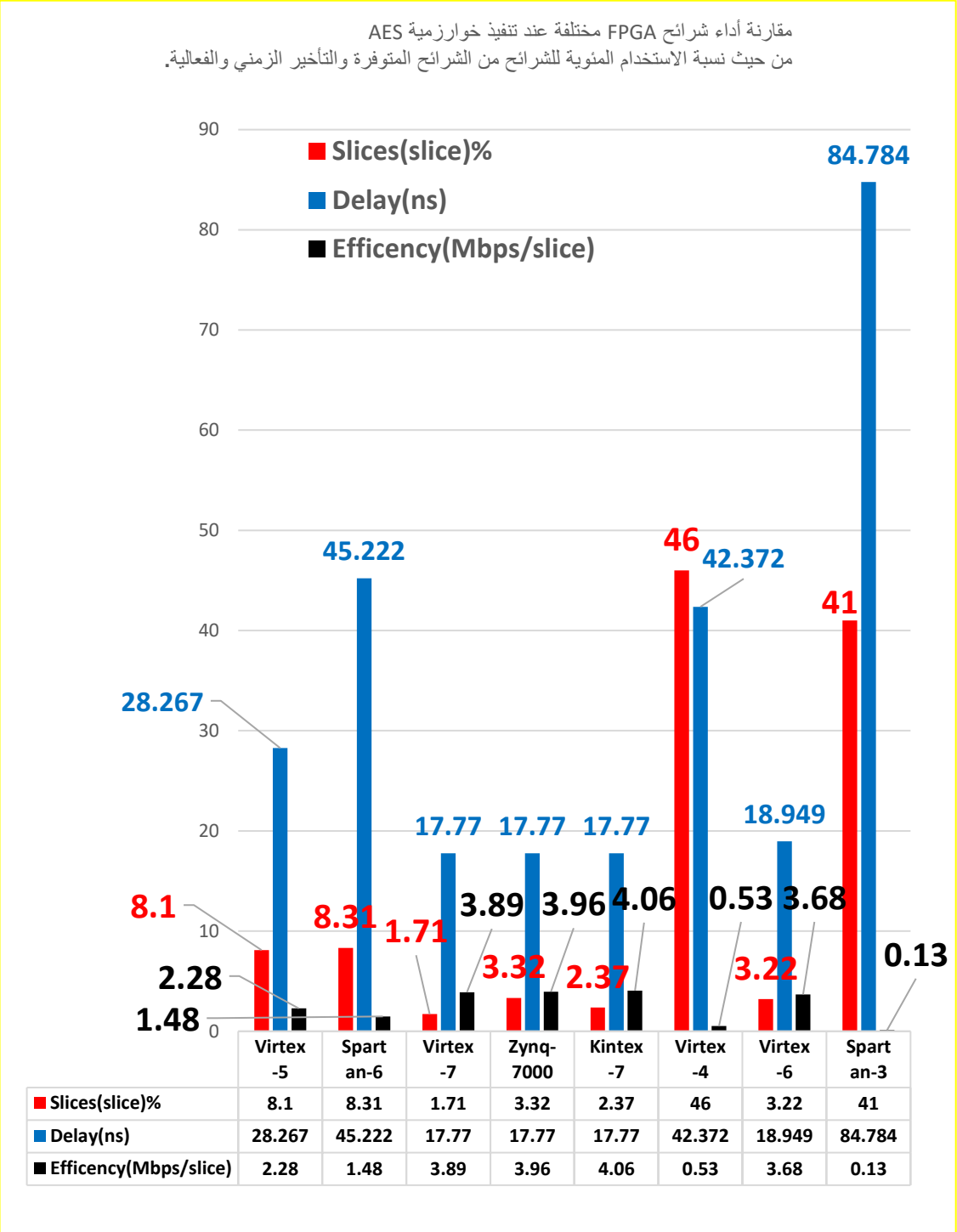
2- من حيث مقارنة نتائج الدراسة الحالية مع الدراسات السابقة:

وجدنا عند تنفيذ خوارزمية AES باستخدام الشريحة Spartan-6 أنها حققت فعالية أفضل من المحققة لأجل الدراسة[3] ، حيث بلغت الفعالية لأجل الدراسة الحالية 1.48Mbps/slice في حين أنها لأجل الدراسة[3] بلغت 0.544Mbps/slice بالنتيجة حققت الدراسة تحسيناً بمقدار % 172.06 بقيمة الفعالية.

بينما حققت الشريحة Virtex-6عند تنفيذ خوارزميةAES فعالية أفضل عند مقارنتها مع الدراسات السابقة حيث بلغت 3.680 Mbps/slice للدراسة الحالية في حين بلغت قيمة فعالية الدراسة[3] عند تنفيذها لخوارزمية معيار التشفير المتقدم 1.44 Mbps/slice من ثم حققت الدراسة الحالية تحسيناً بمقدار % 155.56.

Family	Slice utilization %	Delay path (ns)	Number of Available slices	Number Of Slices
Zynq-7000	3.32	17.770	54650	1818
Virtex-7	1.71	17.770	108300	1855
Spartan-6	8.31	45.222	23038	1915
Kintex-7	2.37	17.770	74650	1776
Virtex-5	8.1	28.267	24320	1986
Virtex-4	46	42.372	12288	5658
Spartan-3	41.1	84.784	27648	11355
Virtex-6	3.22	18.949	56880	1836

الجدول (2-1) مقارنة بين أداء شرائح مختلفة FPGAsعند تنفيذها للخوارزمية المدروسة من حيث التأخير الزمني ونسبة الاستخدام المئوية للشرائح من الشرائح المتوفرة والتردد الأعظمي.



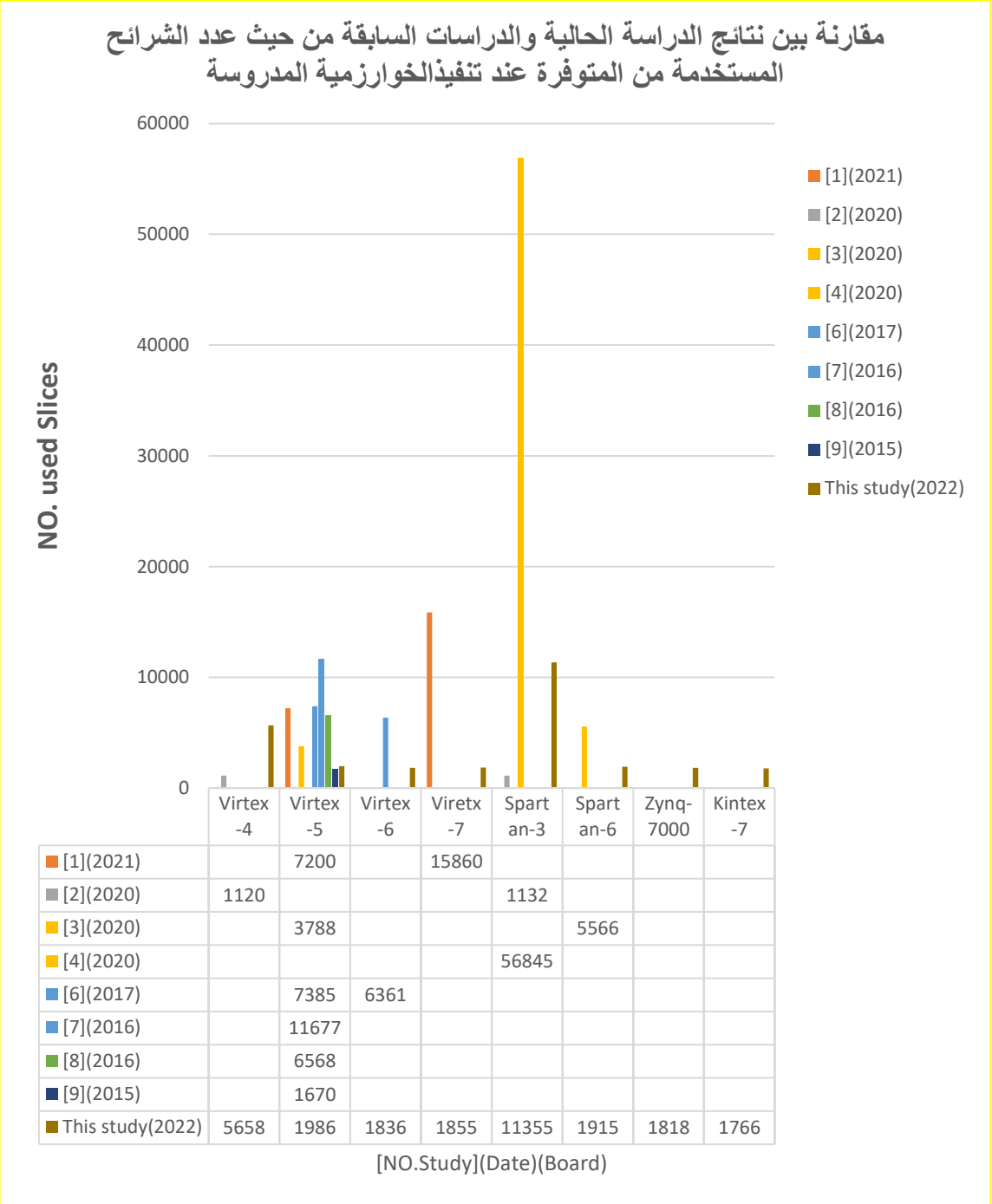
الشكل (2–5) مقارنة بين أداء شرائح مختلفة FPGAs عند تنفيذها للخوارزمية المدروسة من حيث التأخير الزمني ونسبة الاستخدام المئوية للشرائح من الشرائح المتوفرة والفعالية.

2-4-2- مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقة ذات الصلة بموضوع البحث:

2-4-2-1- من حيث المصادر عدد الشرائح المستخدمة من المتوفر في الشريحة المختارة:

Percentage of Slice usage	Number of all Available Slices	Number of slices(slice)	Device family	Date	No. study
55.07 10.85	28800 126800	7200 15860 13760	Virtex- 5 Virtex- 7 Virtex- 7 (Conventional AES)	2021	[1]
10.42 58.96	10752 1920	1120 1132	Virtex-4 Spartan-3	2020	[2]
---------- ----------	---- ----	4095 5566	Virtex-5 Spartan-6	2020	[3]
----------	-----	56845	Spartan-3	2020	[4]
----------	----	7385 6361	Virtex-5 Virtex-6	2017	[6]
---------- ----------	----- -----	11359ECB(Mode) 11677CTR(Mode)	Virtex-5	2016	[7]
----------	----	6568	Virtex-5	2016	[8]
2.42	69120	1670	Virtex-5	2015	[9]
3.32	54650	1818	Zynq-7000	2022	This study
1.71	108300	1855	Virtex-7	2022	This study
8.31	23038	1915	Spartan-6	2022	This study
2.37	74650	1776	Kintex-7	2022	This study
8.1	24320	1986	Virtex-5	2022	This study
46	12288	5658	Virtex-4	2022	This study
3.22	56880	1836	Virtex-6	2022	This study
41	27648	11355	Spartan-3	2022	This study

الجدول (2-2) مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقة ذات الصلة بموضوع البحث من حيث عدد الشرائح.



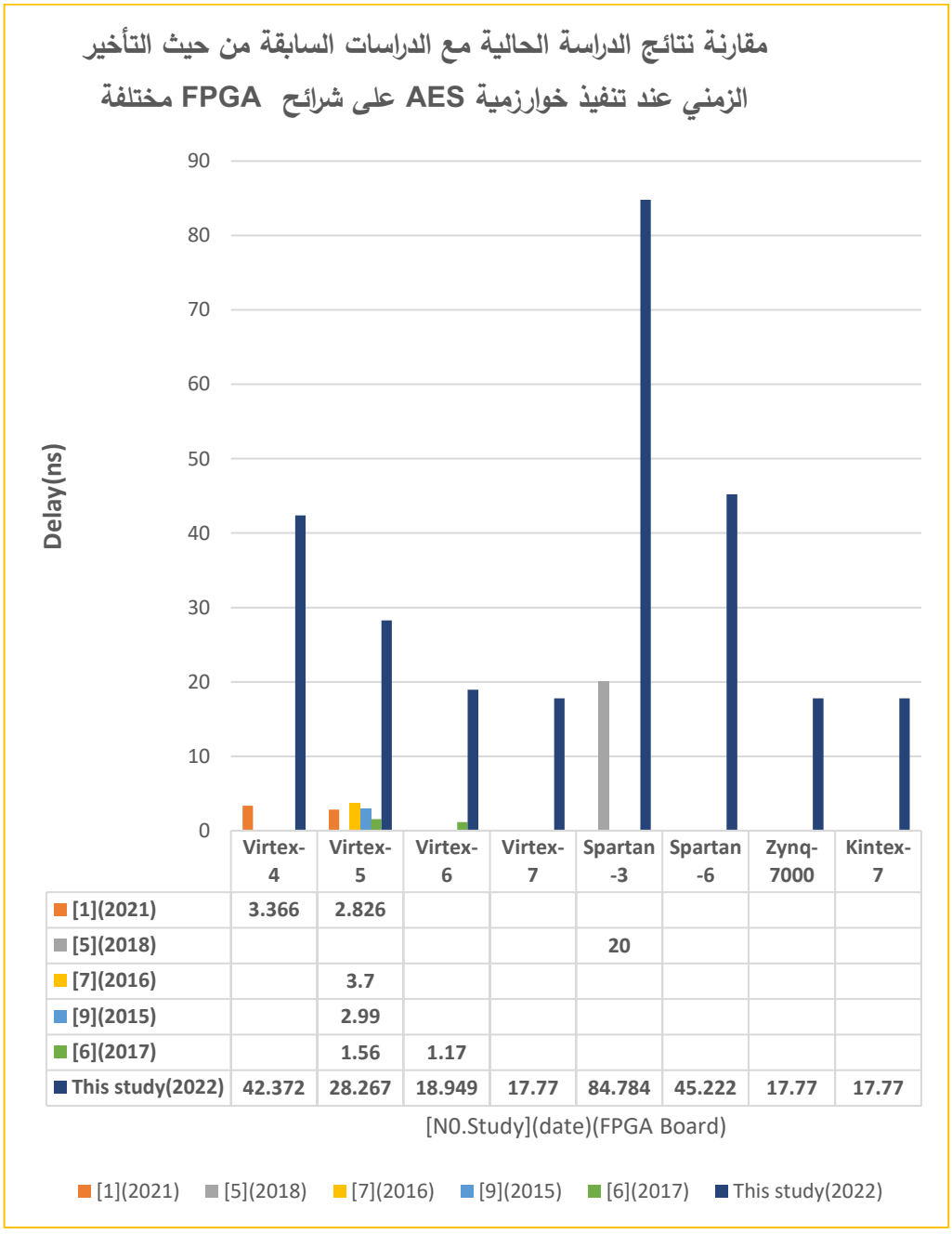
	Virtex -4	Virtex -5	Virtex -6	Viretx -7	Spartan-3	Spartan-6	Zynq-7000	Kintex -7
[1](2021)		7200		15860				
[2](2020)	1120				1132			
[3](2020)		3788				5566		
[4](2020)					56845			
[6](2017)		7385	6361					
[7](2016)		11677						
[8](2016)		6568						
[9](2015)		1670						
This study(2022)	5658	1986	1836	1855	11355	1915	1818	1766

الشكل (2–6) مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقة من حيث المصادر عدد الشرائح المستخدمة من المتوفرة في الشريحة المختارة. استخلصنا من الشكل (2–6) حققت العائلات الآتية Virtex–6, Spartan–6, Virtex–7 استخداماً للشرائح أقل من باقي الدراسات السابقة.

2-4-2-2- مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقة ذات الصلة بموضوع البحث من حيث التأخير الزمني (نانو ثانية):

Delay path (ns)	Device family	Date	Number of study
17.770	Zynq-7000	2022	This study
17.770	Virtex-7	2022	This study
28.267	Virtex-5	2022	This study
45.222	Spartan-6	2022	This study
84.784	Spartan-3	2022	This study
18.949	Virtex-6	2022	This study
42.372	Virtex-4	2022	This study
17.770	Kintex-7	2022	This study
2.826 3.366	Virtex-5 Virtex-4	2021	[1]
20	Spartan-3	2018	[5]
3.7	Virtex-5	2016	[7]
3	Virtex-5	2015	[9]
1.56 1.17	Virtex-5 Virtex-6	2017	[6]

الجدول (2-3) مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقة ذات الصلة بموضوع البحث من حيث التأخير الزمني. من الجدول (2-3) استخلصنا أن التأخير الزمني للدراسة الحالية كان أعلى من الدراسات المرجعية الأخرى ذلك بسبب استخدام نسبة مئوية للشرائح المستخدمة على الرقاقة أعلى من الدراسات الأخرى كما في الشكل (2-6) وهذا سبب تأخيراً زمنياً أكبر في إنجاز عملية التشفير المطلوبة.



الشكل (2–7) مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقةعند تنفيذ خوارزمية AES على شرائح FPGA مختلفة حيث التأخير الزمني (نانو ثانية).

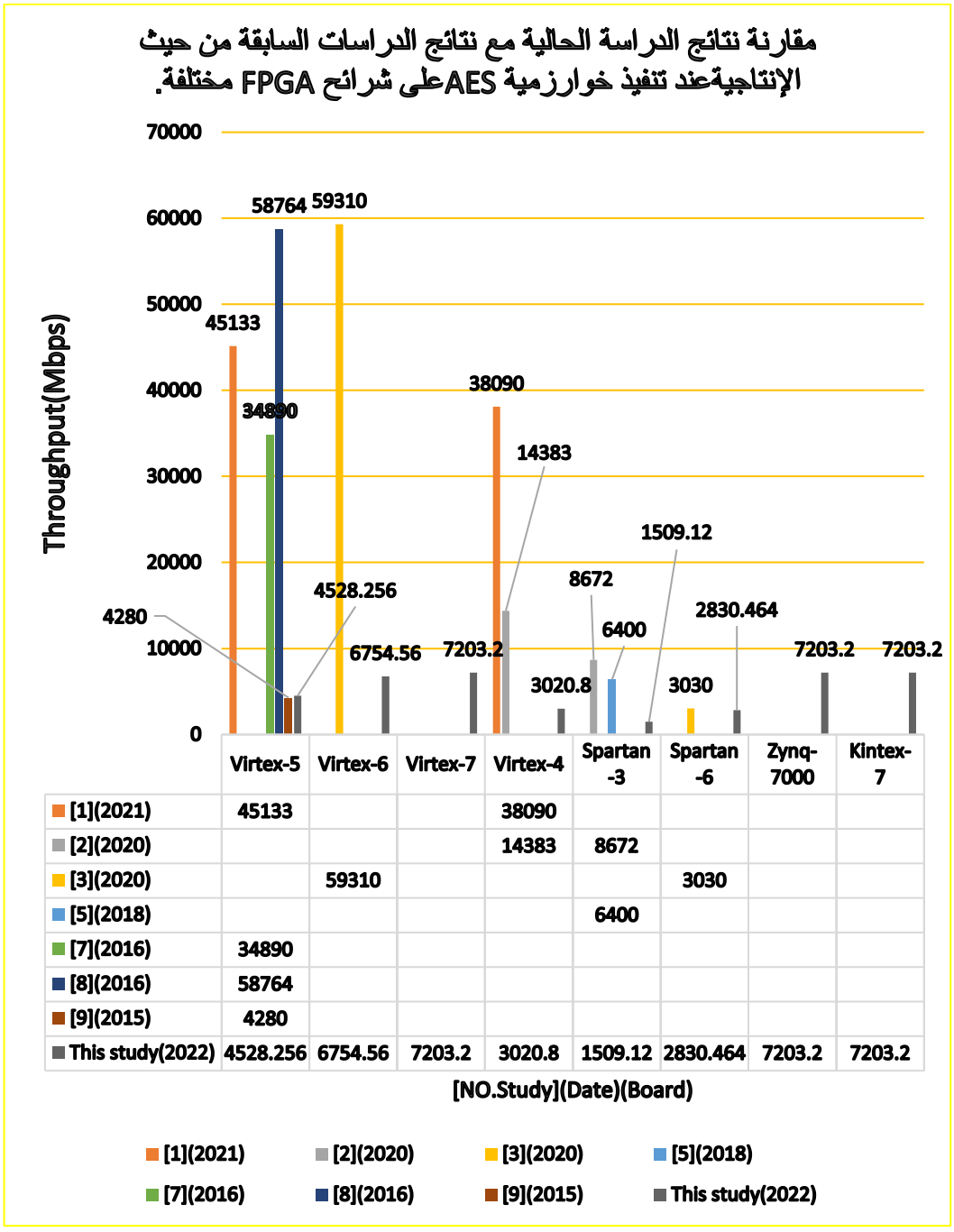
2-4-2-3- مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقة ذات الصلة من حيث الإنتاجية:

Throughput (Mbps)	Device family	Board	Date	No. study
45133 38090	Virtex- 5 Virtex-4		2021	[1]
14383 8672	Virtex-4 Spartan-3		2020	[2]
5930 3030	Virtex-6 Spartan-6	Xc6vlx240t Xc6slx150	2020	[3]
6400	Spartan-3		2018	[5]
34890	Virtex-5	Xc5vlx110t	2016	[7]
58764	Virtex-5		2016	[8]
42809	Virtex-5	Xc5vlx110tff1136-3	2015	[9]
7385 6361	Virtex-5 Virtex-6	Xc5vlx85 Xc6vlx240t	2017	[6]
7203.200	Zynq-7000	Xc7z045ffg900-3	2022	This study
7203.200	Virtex-7	Xc7vx690tffg1930-3	2022	This study
4528.256	Virtex-5	Xc5vlx155tffg1760-3	2022	This study
2830.464	Spartan-6	Xc6slx150tffg900-3	2022	This study
7203.200	Kintex-7	Xc7k480tffg1156-3	2022	This study
3020.8	Virtex-4	Xc4vsx55ff1148-12	2022	This study
1509.12	Spartan-3	Xc3s4000lfg900-4	2022	This study
4528.26	Virtex-5	Xc5vlx155tffg1760-3	2022	This study
6754.56	Virtex-6	Xc6vlx365tff1759-3	2022	This study

الجدول (2-4) مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقة ذات الصلة من حيث الإنتاجية:

حققت ثلاثة عائلات7-Virtex,Zynq-7000,7-Kintex نفس قيمة الإنتاجية ألا وهي 7203.200 Mbps

وعند مقارنة النتائج مع الدراسات السابقة فكانت العائلة Virtex-6 حققت إنتاجية أعلى من الدراسة[6]



الشكل (2-8) مقارنة نتائج الدراسة الحالية مع نتائج الدراسات السابقة من حيث الإنتاجية

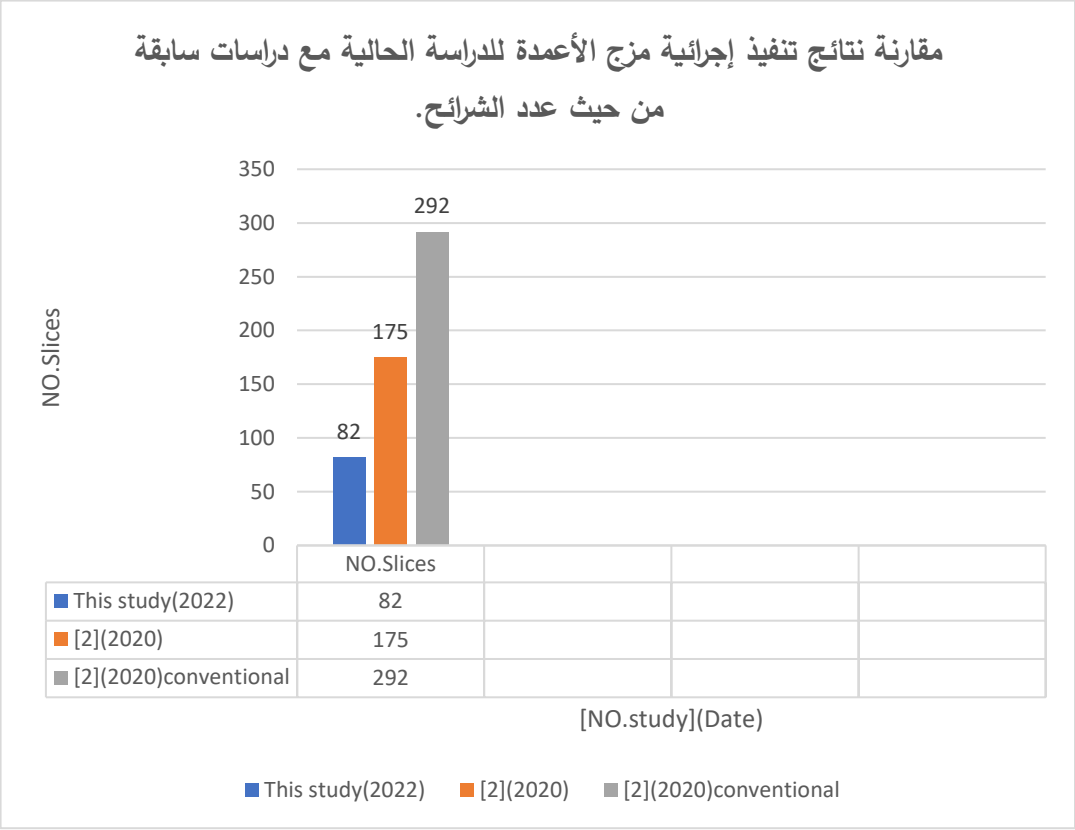
عند تنفيذ خوارزمية AES على شرائح FPGA مختلفة. من الشكل (2-8) استنتجنا أن العائلات الثلاث Virtex-7,Zynq-7000,Kintex-7حققت أعلى قيمة للإنتاجية بلغت 7203.2Mbps

2-4-3-مقارنة نتائج تنفيذ إجرائية مزج الأعمدة مع دراسات سابقة من حيث:

2-4-3-1-استخدام للمصادر المتاحة على شريحة FPGA كما في الجدول (2-5)

Parameters No of LUTs	Parameters No of slices	Date	Number of studies
134	82	2022	This study mix column
336	175	2020	[2] study Vedic multiplier
507	292	2020	[2] Conventional multiplier

الجدول (2-5) مقارنة نتائج مزج الأعمدة للدراسة الحالية مع دراسات سابقة من حيث المصادر

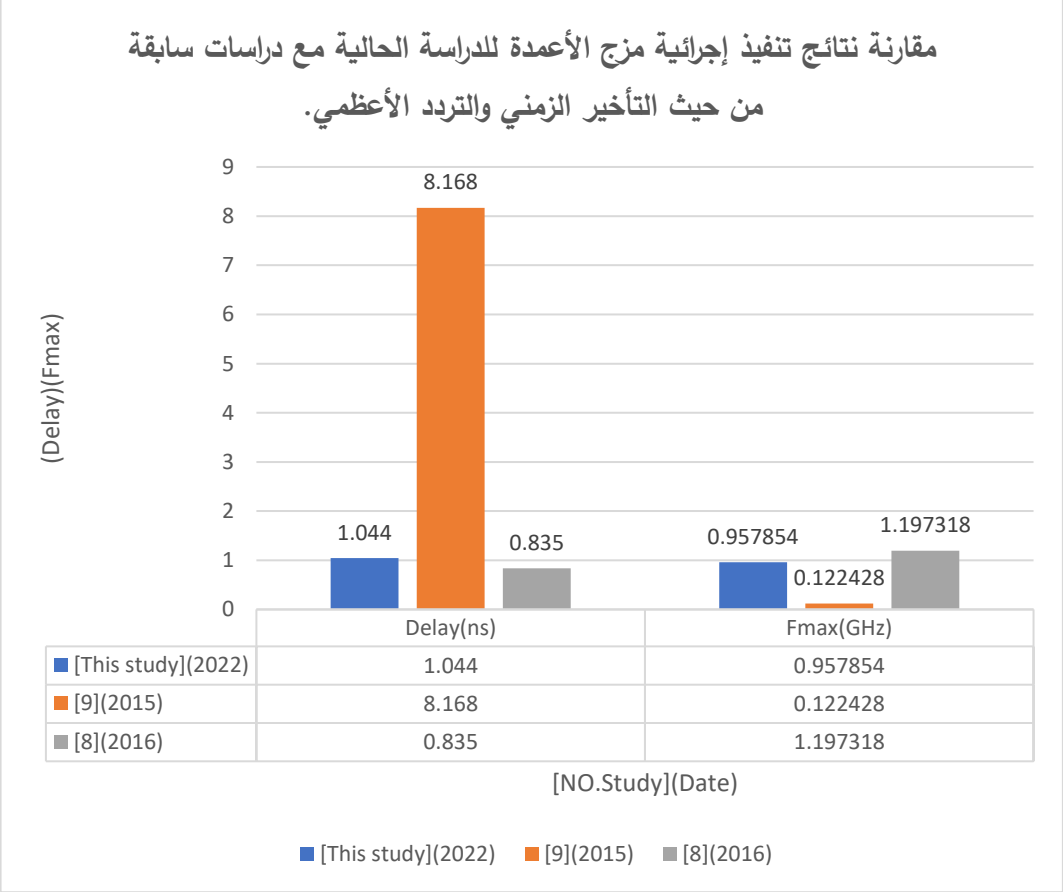


الشكل (2-9) مقارنة نتائج تنفيذ إجرائية مزج الأعمدة للدراسة الحالية مع دراسات سابقة من حيث عدد الشرائح. نستنتج من الشكل السابق أن الدراسة الحالية استخدمت أقل عدد من الشرائح المتوفرة على الشريحة المستخدمة مقارنة مع الدراسات السابقة.

2-4-3-2- مقارنة نتائج تنفيذ إجرائية مزج الأعمدة للدراسة الحالية مع دراسات سابقة من حيث التأخير الزمني والتردد الأعظمي وعددLUT كما في الجدول الآتي (2-6):

Number Of LUT	Fmax (MHz)	Delay (ns)	Date	No. study
134	957.854	1.044	2022	This study mix column
------	122.428	8.168	2015	[9]
300	1197.318	0.835	2016	[8]

الجدول (2-6) مقارنة نتائج تنفيذ إجرائية مزج الأعمدة للدراسة الحالية مع دراسات سابقة



الشكل (2-10) مقارنة نتائج تنفيذ إجرائية مزج الأعمدة للدراسة الحالية مع دراسات سابقة من حيث التأخير الزمني والتردد الأعظمي.

نستنتج من الشكل السابق أن الدراسة الحالية حققت تأخير زمني أقل من المحقق في الدراسة[9] وأعلى من المحقق لأجل الدراسة[8] وذلك بسبب طريقة كتابة الرماز البرمجي للإجرائية مما يساهم في تنفيذ الإجرائية بسرعة أعلى من الدراسة[9] .

2-5-تقييم أداء تنفيذ خوارزمية AES على ثماني عائلات لشرائح FPGAs:

2-5-1- من حيث نسبة الاستخدام المئوية للشرائح من المصادر المتاحة على الرقاقة المختارة:

أثبتت النتائج أن عائلةVirtex-7 كانت الأفضل من حيث النسبة المئوية لاستخدام الشرائح المتوفرة فبلغت 1.71% وفق الجدول (2-1) وهو أقل نسبة استخدام مئوية للشرائح من العائلات الثماني المستخدمة.

2-5-2- من حيث التأخير الزمني:

حققت كل من العائلات الثلاث :Zynq-7000, Kintex-7,Virtex-7 النتائج نفسها المبينة في الجدول (2-3) وهي (ns)17.770.

2-5-3-من حيث الإنتاجية:

حققت كل من العائلات الثلاث :Zynq-7000, Kintex-7,Virtex-7 النتائج نفسها المبينة في الجدول (2-4) وهي (Mbps)7203.200.

2-5-4-من حيث الفعالية:

يعرف مفهوم الفعالية: بأنه حاصل قسمة الإنتاجية على المساحة المستخدمة (أي عدد الشرائح المستخدمة من شريحة FPGA) بحسب المرجع [6].

$$Throughput = \frac{\text{Number of outputted bits}}{\text{Delay of the critical path}}$$

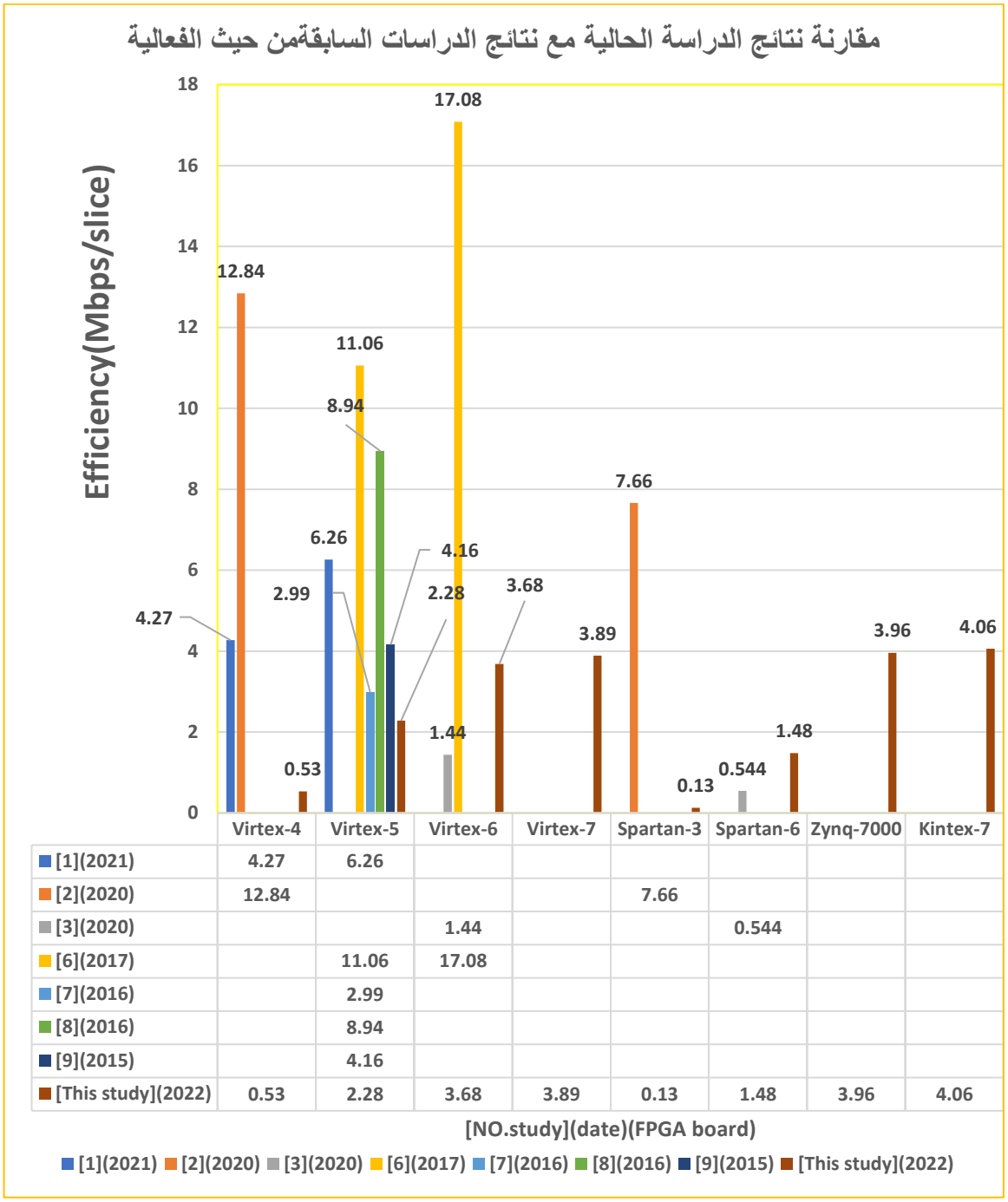
$$Efficiency = \frac{\text{Throughput}}{\text{Area (Slices)}}$$

أثبتت النتائج أن الشريحة Kintex-7 نوع Xc7k480tffg1156-3 هي الأفضل من حيث الفعالية التي بلغت وفق الشكل (2-5) 4.06Mbps/slice.

نبين في الجدول الآتي (2-7) مقارنة نتائج الدراسة الحالية من حيث الفعالية مع دراسات سابقة

Efficiency (Mbps/slice)	Throughput (Mbps)	Device family	Number of Slices	Date	No. study
6.26 4.27	45133 38090	Virtex- 5 Virtex-4	7200 8912	2021	[1]
12.84 7.66	14383 8672	Virtex-4 Spartan-3	1120 1132	2020	[2]
1.44 0.544	5930 3030	Virtex-6 Spartan-6	4095 5566	2020	[3]
-----	6400	Spartan-3	----	2018	[5]
3.07 2.99	34890 34890	Virtex-5 Virtex-5	11359 ECB mode 11677 CTR mode	2016	[7]
8.94	58764	Virtex-5	6568	2016	[8]
2.56	4280	Virtex-5	1670	2015	[9]
11.06 17.08	81680 108690	Virtex-5 Virtex-6	7385 6361	2017	[6]
3.96	7203.200	Zynq-7000	1818	2022	This study
3.89	7203.200	Virtex-7	1855	2022	This study
2.28	4528.256	Virtex-5	1986	2022	This study
1.48	2830.464	Spartan-6	1915	2022	This study
4.06	7203.200	Kintex-7	1776	2022	This study
3.68	6754.56	Virtex-6	1836	2022	This study
0.53	3020.8	Virtex-4	5658	2022	This study
0.13	1509.12	Spartan-3	11355	2022	This study

جدول (2-7) مقارنة نتائج الدراسة الحالية من حيث الفعالية مع الدراسات السابقة.



الشكل (2-11) مقارنة نتائج الدراسة الحالية مع نتائج دراسات سابقة من حيث الفعالية.

من الشكل (2-11) استخلصنا أن العائلةKintex-7 حققت أعلى قيمة من حيث الفعالية عند مقارنة فعالية العائلات الثماني بلغت 4.06Mbps/slice في حين أنه عند مقارنة قيم الفعالية للدراسة الحالية مع الدراسات السابقة وجدنا أن العائلة Spartan-6 حققت فعالية أعلى من الدراسة المرجعية[3] بتحسين بلغ 172.06%.

والعائلة Virtex-6حققت فعالية 3.680 Mbps/slice أعلى من الدراسة المرجعية [3] التي بلغت 1.44 Mbps/slice بتحسين بلغ % 155.56 .

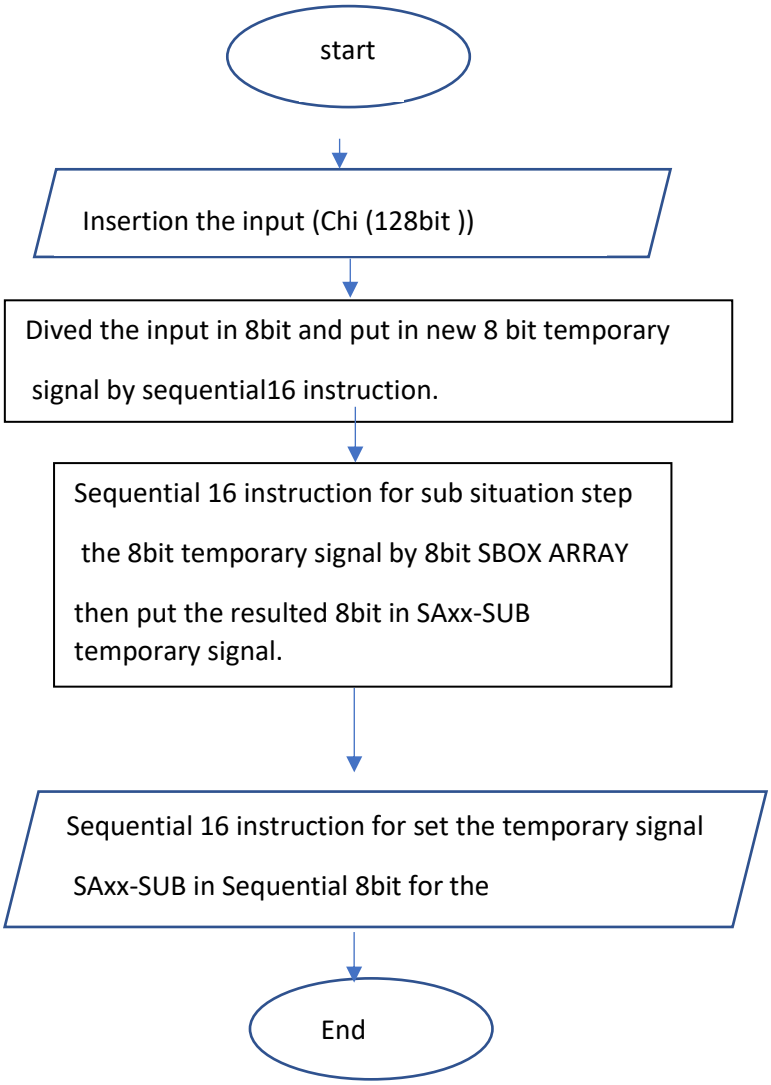
2-6- تعقيد خوارزميةAES المدروسة:

لإيجاد تعقيد هذه الخوارزمية ندرس تعقيد كل من الإجرائيات الثلاث، وهي:

(1-PROGSUB 2-PROGSH 3-PROGMIX) بداية ثم نحدد تعقيد الخوارزمية الكلي

ولإنجاز ذلك نرسم مخطط التدفقي لكل إجرائية ثم نحسب تعقيدها.

1-تعقيد إجرائية PROGSUB



الشكل (2-12) المخطط التدفقي لإجرائية progsub

Steps Cost

1-Insertion instruction (128bits)---------- 1

2-sequential16 instruction put 8bit for input in new 8 bit
temporary signal ---------- n*16=16 n

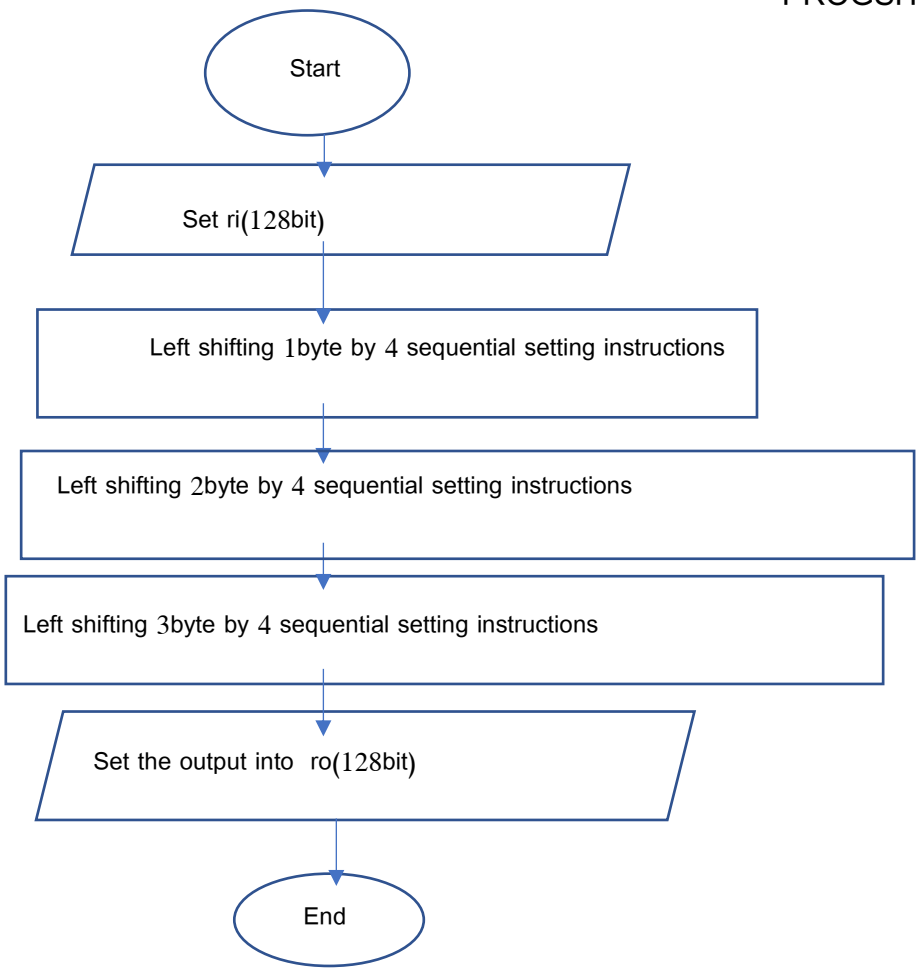
3- Sequential 16 instruction for sub situation step the 8bit
temporary signal by 8bit SBOX ARRAY then put the resulted
8bit in SAxx-SUB temporary signal. ----------16*(n)=16 n

4- Sequential 16 instruction for set the temporary signal
SAxx-SUB in Sequential 8bit for the 128-bit output(chip)---- 16*1=16

Complexity for this procedure O(N)⬅=====17+32 (n)

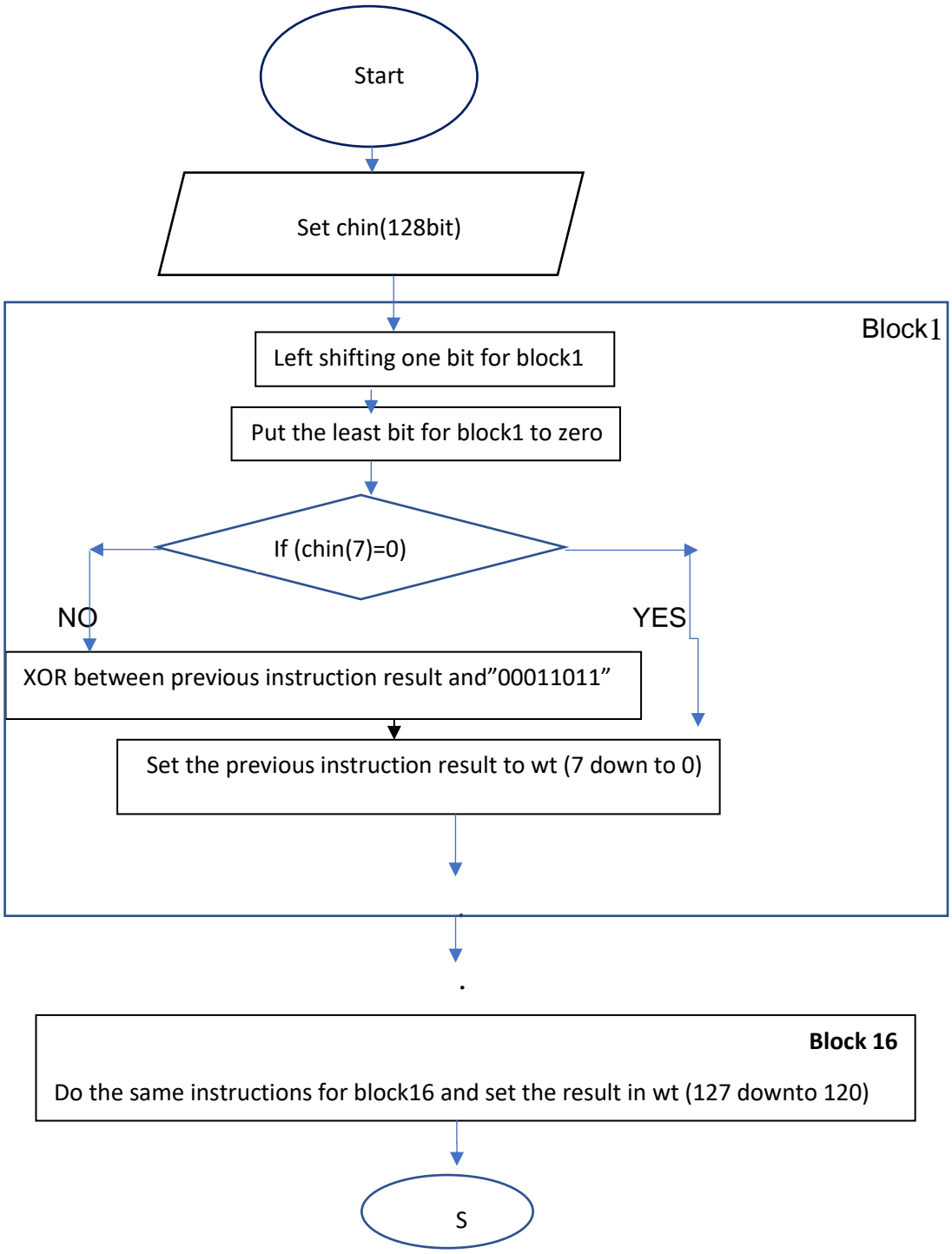
حيث n تمثل عدد عناصر المصفوفةSBOX التي عدد عناصرها 256عنصراً.

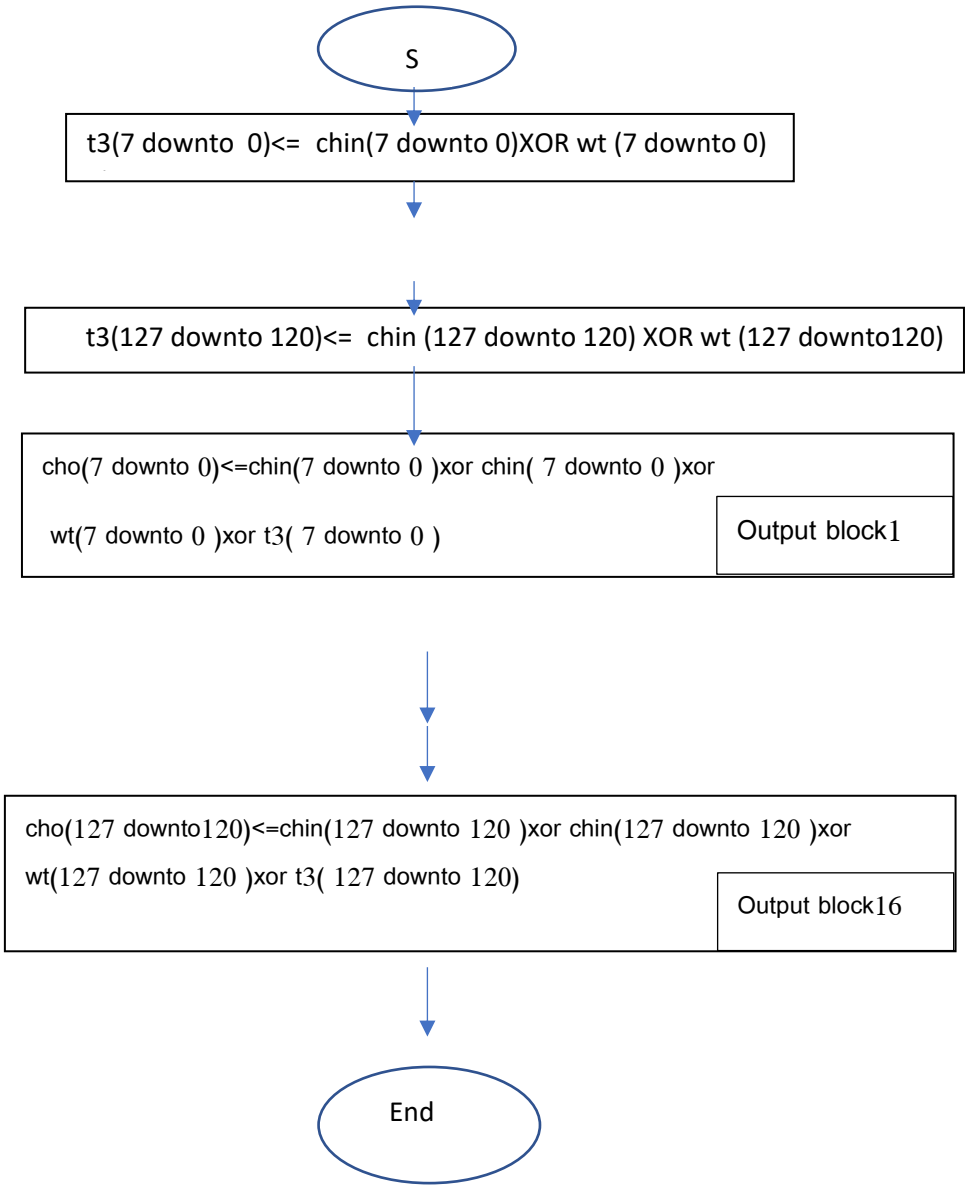
2-تعقيد إجرائية PROGSH



الشكل (2-13) مخطط تدفقي لإجرائية PROGSH

Steps		Cost
1-Insertion (128bits)		1
2-Shiftrow: 4 instructions for shifting	1*4=	4
3-Shiftrow: 4 instructions for shifting	1*4=	4
4-Shiftrow: 4 instructions for shifting	1*4=	4
5-Set the 128bit to ro output		1
Complexity for this procedure	<=O(1) ⬅==========	14

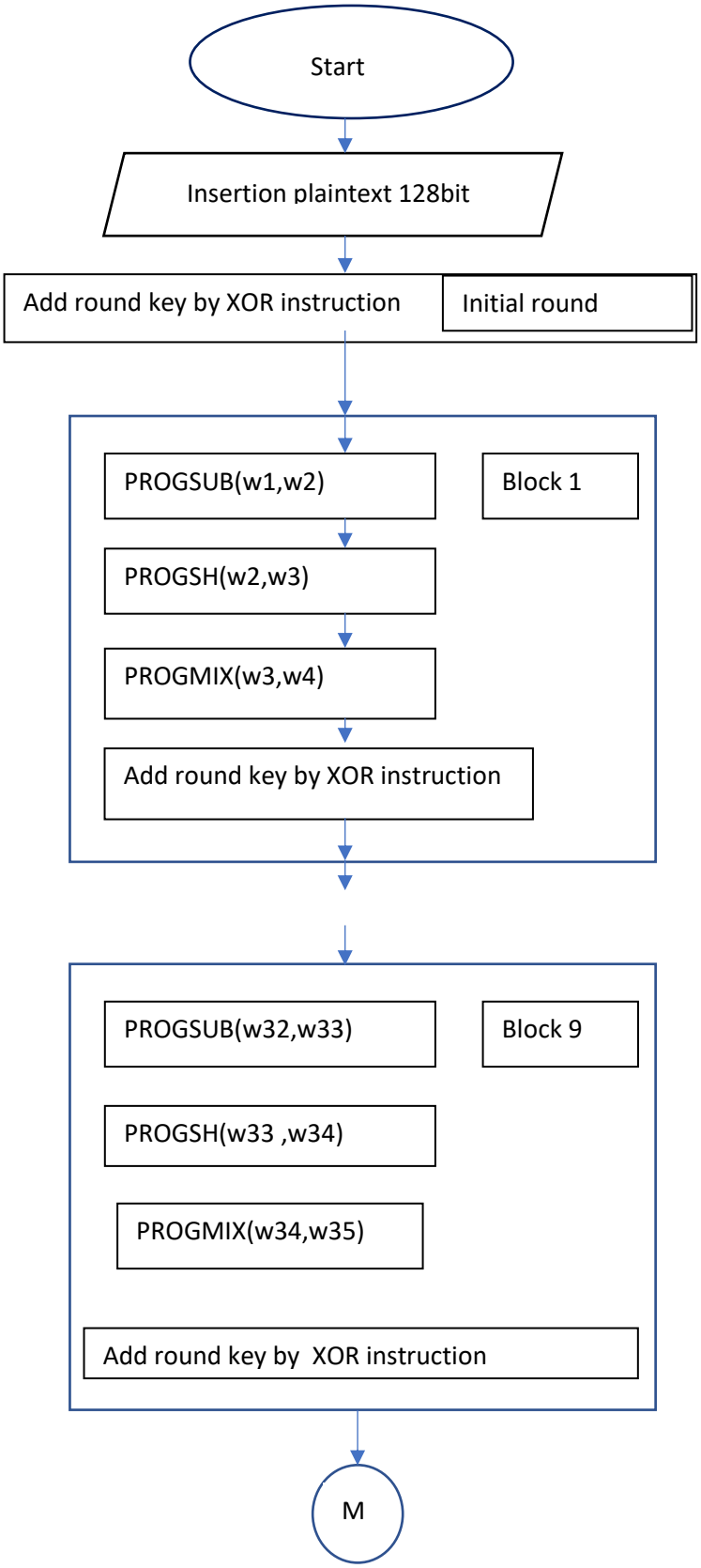

Start
Set chin(128bit)
Block1
Left shifting one bit for block1
Put the least bit for block1 to zero
If (chin(7)=0)
NO
YES
XOR between previous instruction result and"00011011"
Set the previous instruction result to wt (7 down to 0)
Block 16
Do the same instructions for block16 and set the result in wt (127 downto 120)
S

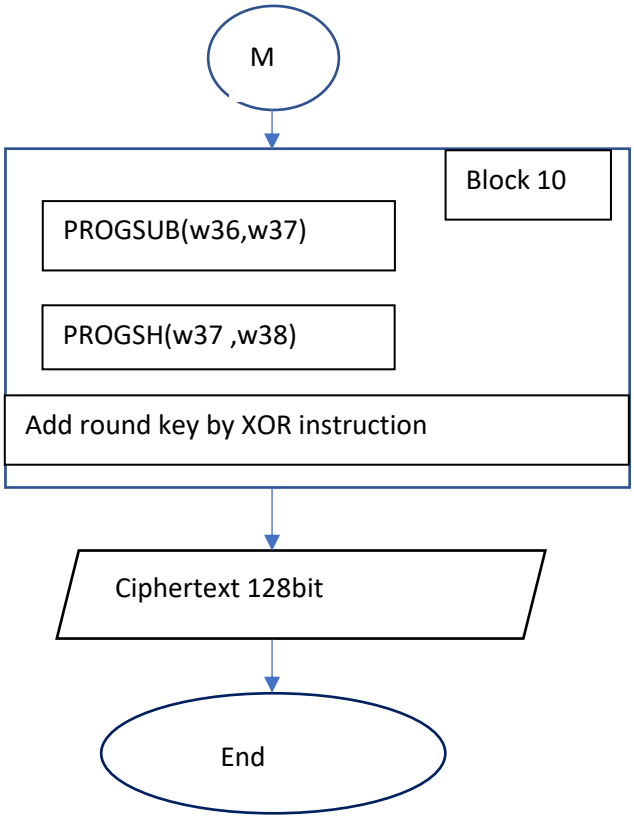


الشكل (2-14) مخطط تدفقي لإجرائية PROGMIX

Steps	Cost
1-Insertion (128bits) ----------	1
2- Left shifting one bit for block1----------	1
3- Set the least weight bit in block NO.1 to zero ----------1	
4-If statement do two instructions 5+6 worst case:	
5-Do XOR operation between the resulted block NO.1 and "00011011" ----------	1
6- Set the resulted block from step 5 to wt(7 downto 0) --------	1
7-Sequential instructions repeated from step 2,3,4,5,6 for The other fifteen blocks ----------	5*15= 75
8- Set the resulted XOR operation between chi(7 downto 0) and wt(7 downto 0) to t3(7 downto 0) ---------- ----1	
9- Sequential instructions repeated for step8 for The other fifteen blocks ----------	-----1*15=15
Complexity for this procedure	<=O(1) ←========== =====96

نحسب الآن تعقيد خوارزميةAES





الشكل (2-15) المخطط التدفقي لخوارزمية AES

تعقيد الخوارزمية المدروسة:

Steps	Cost	Complexity
1-Insertion (128bits) ----------→	1	
2- Add round key XOR instruction ----------→	1	
3-PROGSUB recalling ----------→	17+32n	O(n)
4-PROSH recalling ----------→	14	O(1)
5-PROGMIX recalling ----------→	97	O(1)
2- Add round key XOR instruction ----------→	1	

7-Sequential recalling for procedures PROSUB, PROGSH, PROGMIX,

and add round key XOR instruction for nine sequential blocks

$$9*(32\ n+17+14+96+1) = 288*n+1152 \text{ ------} \rightarrow O(n)$$

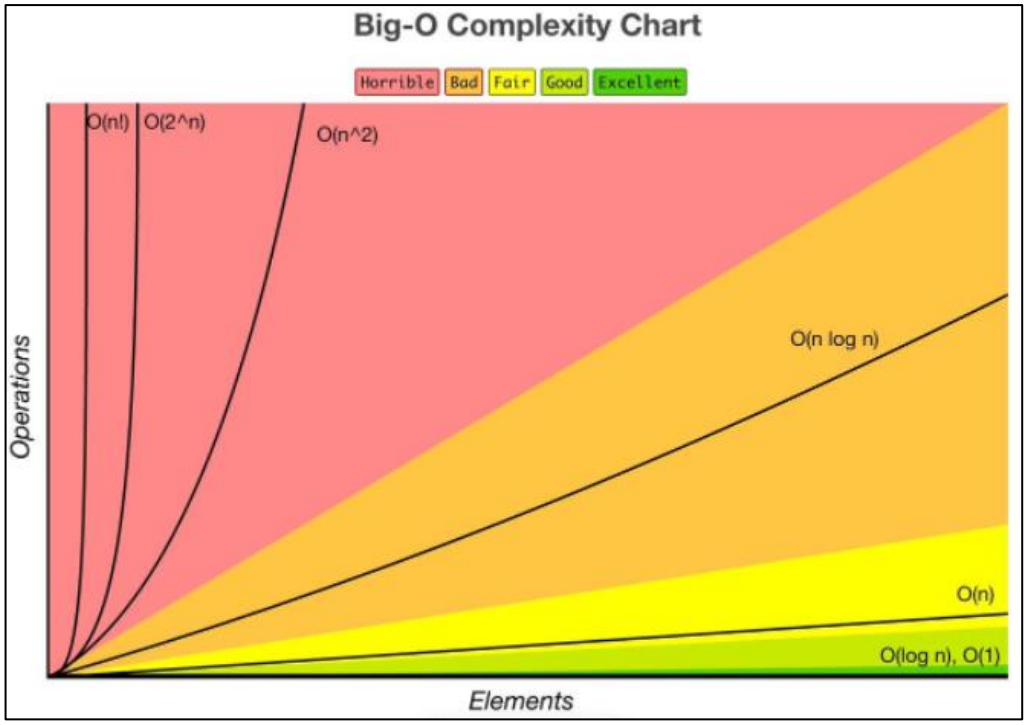
8--Sequential recalling for procedures ROGSUB,

, PROSH, and Add round key XOR instruction for the tenth block

$$(32\ n+17+14+1) = 32n+32\text{----------} \rightarrow O(n)$$

9-Set the output to ciphertext 128 bit 1------ --→ 1

Complexity for this procedure ----------→ O(n)



الشكل (2-16) جودة أداء الخوارزمية وفقا لدرجة تعقيدها[20][Big-O Complexity]

مما سبق نجد أن تعقيد الخوارزمية المدروسة هو O(n) ، وهذا يعد مقبولا (fair) كماهو موضح بالشكل (2-16) الذي يحدد مدى جودة الخوارزمية تبعاً لدرجة تعقيدها.

7.2 – إقرار صلاحية النتائج Validation

2–7–1–إقرار صلاحية نتائج خوارزمية البحثAES:

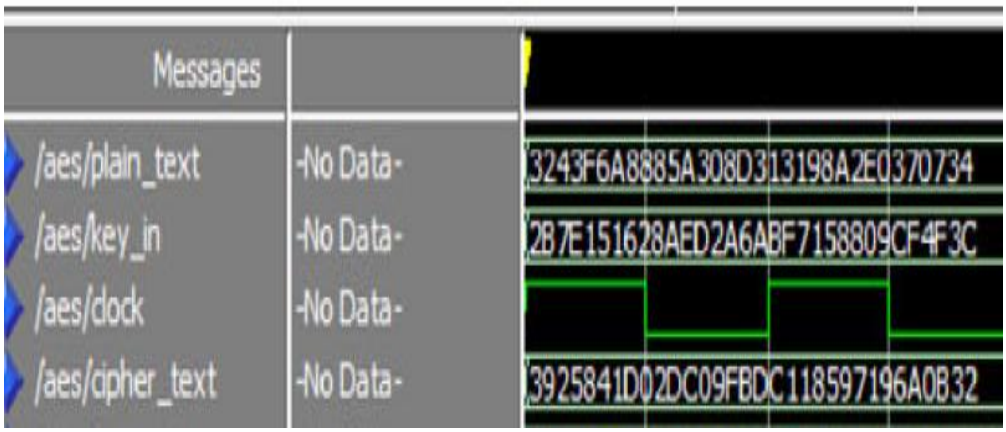
عند إدخال النص الصريح ومفتاح التشفير المطابقين للدراستين [4], [8] المبينة في الشكل (2–17) حصلنا على النص المشفر نفسه كما في الشكل (2–18)لخرج الدراسة.

المدخلات:1–النص الصريح 3243F6A8885A305D313198A2E03070734

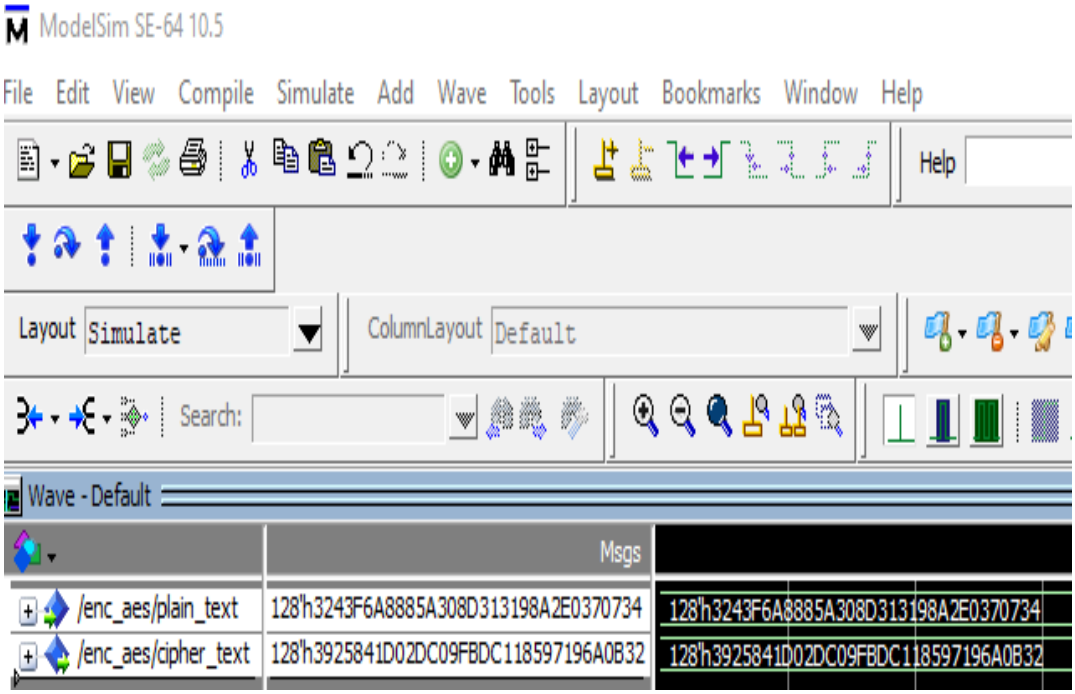
2– المفتاح 2B7E151628AED2A6ABF7158809CF4F3C

حصلنا على الخرج: النص المشفر 3925841D02DC09FBDC118597196A0B32

$$\text{State} = \begin{bmatrix} 32\ 88\ 31\ E0 \\ 43\ 5A\ 31\ 37 \\ F6\ 30\ 98\ 07 \\ A8\ 8D\ A2\ 34 \end{bmatrix} \quad \text{Key} = \begin{bmatrix} 2B\ 28\ AB\ 09 \\ 7E\ AE\ F7\ CF \\ 15\ D2\ 15\ 4F \\ 16\ A6\ 88\ 3C \end{bmatrix} \quad \text{Encrypted output} = \begin{bmatrix} 39\ 02\ DC\ 19 \\ 25\ DC\ 11\ 6A \\ 84\ 09\ 85\ 0B \\ 1D\ FB\ 97\ 32 \end{bmatrix}$$



الشكل (2–17) خرج الدراسة [8]



الشكل (2–18) خرج الدراسة الحالية.

2-8-خرج الأداة البرمجية Xilinx planAhead عند تنفيذ الخوارزمية المدروسة لكل من الشرائح المختارة:

2-8-1-خرج الأداة البرمجية :

```
Maximum combinational path delay: 45.222ns
```

Part: xc6slx150fgg900-3

Resource	Utilization	Available	Utilization
LUT	6540	92152	7%
Slice	1915	23038	8%
IO	256	576	44%

الشكل (2-19) المصادر المستخدمة والتأخير الزمني من الشريحة spartan-6

2-8-2-خرج الأداة البرمجية:

```
Maximum combinational path delay: 17.770ns
```

Part: xc7z045ffg900-3

Resource	Utilization	Available	Utilization
LUT	6494	218600	2%
Slice	1818	54650	3%
IO	256	362	70%

الشكل (2- 20) المصادر المستخدمة والتأخير الزمني من الشريحةZynq-7000

-2-8-3-خرج الأداة البرمجية:

```
Maximum combinational path delay: 17.770ns
```

Part: xc7vx690tffg1930-3

Resource	Utilization	Available	Utilization
LUT	6494	433200	1%
Slice	1855	108300	1%
IO	256	1000	25%

الشكل (2- 21) المصادر المستخدمة والتأخير الزمني من الشريحةVirtex-7

-2-8-4-خرج الأداة البرمجية:

```
Maximum combinational path delay: 28.267ns
```

Part: xc5vlx155tff1760-3

Resource	Utilization	Available	Utilization
LUT	6659	97280	6%
Slice	1986	24320	8%
IO	256	800	32%

الشكل (2-22) المصادر المستخدمة والتأخير الزمني من الشريحةVirtex-5

2-8-5-خرج الأداة البرمجية:

```
Maximum combinational path delay: 17.770ns
```

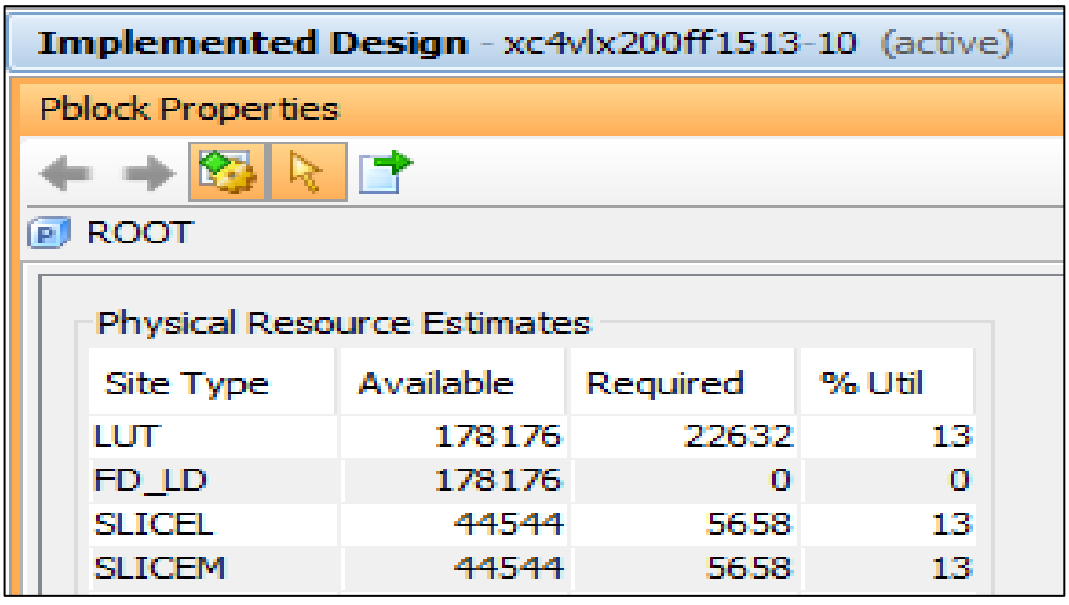
Part: xc7k480tffg1156-3

Resource	Utilization	Available	Utilization
LUT	6494	298600	2%
Slice	1776	74650	2%
IO	256	400	64%

الشكل (2- 23) المصادر المستخدمة والتأخير الزمني من الشريحة Kintex-7

2-8-6-خرج الأداة البرمجية :

```
Maximum combinational path delay: 42.372ns
```



الشكل (2-24) المصادر المستخدمة والتأخير الزمني من الشريحة Virtex-4

2-8-7-خرج الأداة البرمجية :

```
Maximum combinational path delay: 84.784ns
```

Part: xc3s4000ifg900-4

Resource	Utilization	Available	Utilization
4 input LUT	22647	55296	40%
Slice	11355	27648	41%
IO	256	633	40%

الشكل (2-25) المصادر المستخدمة من الشريحة والتأخير الزمني Spartan-3

2-8-8- خرج الأداة البرمجية:

```
Maximum combinational path delay: 18.949ns
```

Part: xc6vlx365tff1759-3

Resource	Utilization	Available	Utilization
LUT	6494	227520	2%
Slice	1836	56880	3%
IO	256	720	35%
OLOGICE1	60	960	6%

الشكل (2-26) المصادر المستخدمة والتأخير الزمني من الشريحةVirtex-6

الخلاصة:

من خلال نتائج هذه الدراسة استخلصنا الآتي:

1-عند مقارنة نتائج البحث ضمن العائلات الثمان المنفذ باستخدامها خوارزمية البحثAES :

استخلصنا أن إن أفضل نتيجة من حيث النسبة المئوية للشرائح المستخدمة من المصادر المتاحة على شريحة صفيفه البوابات القابلة للبرمجة حقلياً كانت عائلة Virtex-7 بلغت 1.71% كما في الجدول (2-1).

أما من حيث قيمة الإنتاجية فبلغت (7203.200Mbps) كما في الجدول (2-4)، بالنتيجة الفعالية بلغت 4.06(Mbps/slice) كما في الجدول (2- 7).

2- من حيث مقارنة نتائج الدراسة الحالية مع الدراسات السابقة:

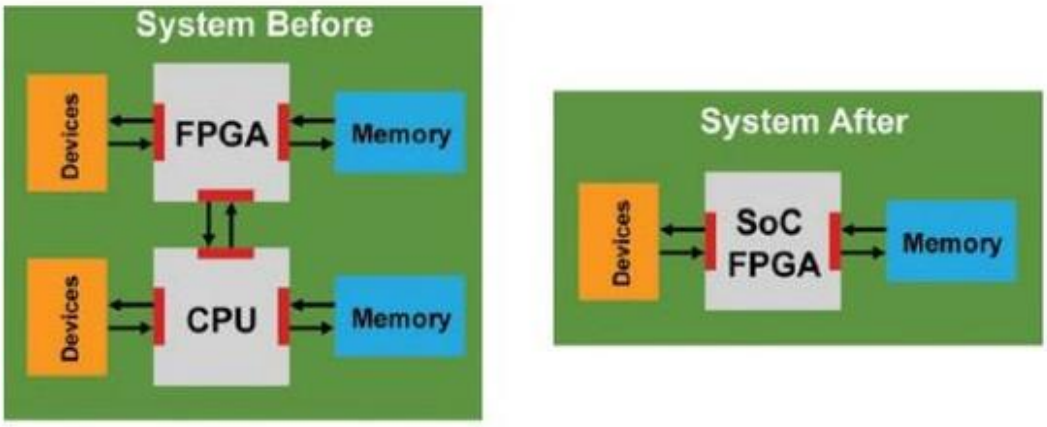
وجدنا عند تنفيذ خوارزمية AES باستخدام الشريحة Spartan-6 أنها حققت فعالية أفضل من المحققة لأجل الدراسة[3] ، حيث بلغت الفعالية لأجل الدراسة الحالية 1.48Mbps/slice في حين أنها لأجل الدراسة[3] بلغت 0.544Mbps/slice بالنتيجة حققت الدراسة تحسيناً بمقدار % 172.06 بقيمة الفعالية.

بينما حققت الشريحة Virtex-6عند تنفيذ خوارزمية AES فعالية أفضل عند مقارنتها مع الدراسات السابقة حيث بلغت 3.680 Mbps/slice للدراسة الحالية في حين بلغت قيمة فعالية الدراسة[3] عند تنفيذها لخوارزمية معيار التشفير المتقدم 1.44 Mbps/slice من ثم حققت الدراسة الحالية تحسيناً بمقدار% 155.56.

التوصيات والآفاق المستقبلية Future Works:

من خلال التطور السريع لتقنيات الاتصال المتنوعة أصبح مجال التشفير ضرورة ملحة لحماية بيانات المستخدمين المختلفة أتت هذه الدراسة لتكون دليلاً في مجال تطبيق خوارزمية التشفير AES في تشفير البيانات باستخدام صفيفة البوابات القابلة للبرمجة حقلياً. من المفضل أيضاً إيجاد آليات تطبيق جديدة لإنقاص التأخير الزمني الذي حصلنا عليه في هذه الدراسة، وذلك للحصول على نتائج أفضل من حيث الفعالية عن طريق إمكانية إعادة كتابة الرماز البرمجي مع اقتراح آليات جديدة لتنفيذ الإجرائيات بتأخير زمني أقل كاستخدام تقنية ضارب فيديك في إنجاز إجرائية مزج الأعمدة. كذلك يمكن البحث في تعديل آلية الخوارزمية للحصول على إجرائية مبسطة وتحقيق نتائج أفضل.

أخيراً يمكن البحث في إمكانية تطبيق خوارزميةAES باستخدام System On Chip FPGA.



الشكل (2-27) System On Chip FPGA

References

1. Priya,s. KarthigaiKumar,P.Teja,N.(2021,October).FPGA implementation of AES algorithm for high-speed applications. Analog Integrated Circuits and Signal Processing. https://doi.org/10.1007/s10470-021-01959-z(retrieved on February 2022).

2. Arul Murugan, C. Karthigaikumar,P.,& Priya,S (2020,September). FPGA implementation of hardware architecture with AES encryptor using sub-pipelined S-box techniques for compact applications. Journal for Control, Measurement, Electronics, Computing and Communications,61(4),682-693.

3.Zodpe,H.Sapkal,A.An efficient AES implementation using FPGA with enhanced security features (2020,July).In Journal of King Saud University-Engineering Sciences ,32,115-122.

4. Nabil, M., M. Khalaf, A., &M.Hassan,s. Design and implementation of pipelined and parallel AES encryption systems using FPGA. (2020), Indonesian Journal of Electrical Engineering and Computer Science,20(1),287-299.

5.Shujaa, M.Hussein,Z. (2018).Design and Implementation of AES Using FPGA. https://www.researchgate.net/publication/324876713.(retrieved on March 2021).

6. Oukili, S., & Bri, S., High throughput FPGA Implementation of Advanced Encryption Standard Algorithm. TELKOMNIKA (Telecommunication Computing Electronics and Control),2017,15(1),494-503.

7.Guzman, I. Nieto, R., &Bernal, A. (2016, April). FPGA implementation of the AES-128algorithm in non-feedback modes of peration. DYANA,83(193),

37-43.

8. Priya,s. KarthigaiKumar,P. (2016,November).High Throughput AES Using Parallel Subbytes and MixColumn. Wireless Personal Communications. ,95(1433),1449.

[10]https://www.google.com/url?sa=AOvVaw33gDu13ylTYZF7WmSZTMU- , (retrieved on 2021 January).

[11] https://www.google.com/url?sa=t&source=AOvVaw0K2tL2lRtiEownsl4uEL, (retrieved on 2021 March).

[12]https://www.google.com/url?sa=AOvVaw0pr4s0XT1AdxBLvmXWJ7O9 , (retrieved on 2021 March).

[13] National Institute of Standards and Technology (NIST).(2001).Advanced Encryption Standard (AES). https://www.google.com/url?sa=AOvVaw3ZlWC6ljetK9SNoOlhUVlm , (retrieved on 2021 March).

[14]https://www.google.com/url?sa=AOvVaw11ENMsryGVunEhilbSWtA , (retrieved on 2021 April).

[15]https://www.google.com/url?sa=AOvVaw0vSVf3JQJ2TJgarTSUFT-T , (retrieved on 2021 June).

[16]https://www.google.com/url?sa=AAOvVawcfHbWtjKl31lD6las4EJJ , (retrieved on 2021 March).

[17]https://www.google.com/url?sa=AOvVaw3skUZ3ZDXH1e9MnTpyqHit , (retrieved on 2021 April).

[18]https://www.google.com/url?sa=AOvVaw0uWk5tofbamuWiPbTnKJ3

, (retrieved on 2021 October).

[19]https://www.google.com/url?sa=AOvVaw2KtTgB8R2DAyo0KHhDki8W

, (retrieved on 2021 March).

[20] https://dev.to/trekhleb/algorithms-and-data-structures-in-javascript-49i3,(retrieved on March 2021).

الملاحق

الملحق الأول:

الملحق –1– نسخة من البرنامج للخوارزمية المدروسة AES:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
Entity progsub is port(-------entitysubbyte ----
--chi:in std_logic_vector(127 downto
0):="00011001001111011110001110111110101000001111010011100010001010
01110011010110001101000110100101010111010011111100001001000000010
000";
chi:in std_logic_vector(127 downto 0);
chip:out std_logic_vector(127  downto 0));
End progsub;
Architecture behavioral of progsub is  -----architecture---
signal sa00_sub, sa01_sub, sa02_sub, sa03_sub: std_logic_vector(7 downto 0);
signal sa10_sub, sa11_sub, sa12_sub, sa13_sub: std_logic_vector(7 downto 0);
signal sa20_sub, sa21_sub, sa22_sub, sa23_sub: std_logic_vector(7 downto 0);
signal sa30_sub, sa31_sub, sa32_sub, sa33_sub: std_logic_vector(7 downto 0);
signal sa00, sa01, sa02, sa03: std_logic_vector(7 downto 0);
signal sa10, sa11, sa12, sa13: std_logic_vector(7 downto 0);
```

```
signal sa20, sa21, sa22, sa23: std_logic_vector(7 downto 0);
signal sa30, sa31, sa32, sa33: std_logic_vector(7 downto 0);
type ARRAY_256 is ARRAY (0 to 255) of STD_LOGIC_VECTOR(7 downto 0);
        type ARRAY_16  is ARRAY (0 to 15) of  STD_LOGIC_VECTOR(7 downto
0);
        constant SBOX : ARRAY_256 :=
        (X"63",X"7c",X"77",X"7b",X"f2",X"6b",X"6f",X"c5",X"30",X"01",X"67",X"2b",X"fe",X"d
7",X"ab",X"76",
        X"ca",X"82",X"c9",X"7d",X"fa",X"59",X"47",X"f0",X"ad",X"d4",X"a2",X"af",X"9c",X"a4"
,X"72",X"c0",
        X"b7",X"fd",X"93",X"26",X"36",X"3f",X"f7",X"cc",X"34",X"a5",X"e5",X"f1",X"71",X"d8"
,X"31",X"15",
        X"04",X"c7",X"23",X"c3",X"18",X"96",X"05",X"9a",X"07",X"12",X"80",X"e2",X"eb",X"
27",X"b2",X"75",
        X"09",X"83",X"2c",X"1a",X"1b",X"6e",X"5a",X"a0",X"52",X"3b",X"d6",X"b3",X"29",X"
e3",X"2f",X"84",
        X"53",X"d1",X"00",X"ed",X"20",X"fc",X"b1",X"5b",X"6a",X"cb",X"be",X"39",X"4a",X"4
c",X"58",X"cf",
        X"d0",X"ef",X"aa",X"fb",X"43",X"4d",X"33",X"85",X"45",X"f9",X"02",X"7f",X"50",X"3c"
,X"9f",X"a8",
        X"51",X"a3",X"40",X"8f",X"92",X"9d",X"38",X"f5",X"bc",X"b6",X"da",X"21",X"10",X"ff"
,X"f3",X"d2",
        X"cd",X"0c",X"13",X"ec",X"5f",X"97",X"44",X"17",X"c4",X"a7",X"7e",X"3d",X"64",X"5
d",X"19",X"73",
        X"60",X"81",X"4f",X"dc",X"22",X"2a",X"90",X"88",X"46",X"ee",X"b8",X"14",X"de",X"5
e",X"0b",X"db",
        X"e0",X"32",X"3a",X"0a",X"49",X"06",X"24",X"5c",X"c2",X"d3",X"ac",X"62",X"91",X"
95",X"e4",X"79",
```

```
        X"e7",X"c8",X"37",X"6d",X"8d",X"d5",X"4e",X"a9",X"6c",X"56",X"f4",X"ea",X"65",X"7
a",X"ae",X"08",
        X"ba",X"78",X"25",X"2e",X"1c",X"a6",X"b4",X"c6",X"e8",X"dd",X"74",X"1f",X"4b",X"b
d",X"8b",X"8a",
        X"70",X"3e",X"b5",X"66",X"48",X"03",X"f6",X"0e",X"61",X"35",X"57",X"b9",X"86",X"
c1",X"1d",X"9e",
        X"e1",X"f8",X"98",X"11",X"69",X"d9",X"8e",X"94",X"9b",X"1e",X"87",X"e9",X"ce",X"5
5",X"28",X"df",
        X"8c",X"a1",X"89",X"0d",X"bf",X"e6",X"42",X"68",X"41",X"99",X"2d",X"0f",X"b0",X"5
4",X"bb",X"16");
begin
sa00<=chi(7 downto 0);
sa01<=chi(15 downto 8);
sa02<=chi(23 downto 16);
sa03<=chi(31 downto 24);
sa10<=chi(39 downto 32);
sa11<=chi(47 downto 40);
sa12<=chi(55 downto 48);
sa13<=chi(63 downto 56);
sa20<=chi(71 downto 64);
sa21<=chi(79 downto 72);
sa22<=chi(87 downto 80);
sa23<=chi(95 downto 88);
sa30<=chi(103 downto 96);
```

```
sa31<=chi(111 downto 104);
sa32<=chi(119 downto 112);
sa33<=chi(127 downto 120);
sa00_sub <= SBOX(conv_integer(sa00));
sa01_sub <= SBOX(conv_integer(sa01));
sa02_sub <= SBOX(conv_integer(sa02));
sa03_sub <= SBOX(conv_integer(sa03));
sa10_sub <= SBOX(conv_integer(sa10));
sa11_sub <= SBOX(conv_integer(sa11));
sa12_sub <= SBOX(conv_integer(sa12));
sa13_sub <= SBOX(conv_integer(sa13));
sa20_sub <= SBOX(conv_integer(sa20));
sa21_sub <= SBOX(conv_integer(sa21));
sa22_sub <= SBOX(conv_integer(sa22));
sa23_sub <= SBOX(conv_integer(sa23));
sa30_sub <= SBOX(conv_integer(sa30));
sa31_sub <= SBOX(conv_integer(sa31));
sa32_sub <= SBOX(conv_integer(sa32));
sa33_sub <= SBOX(conv_integer(sa33));
chip(7 downto 0)<=sa00_sub;
chip(15 downto 8)<=sa01_sub;
chip(23 downto 16)<=sa02_sub;
```

```
chip(31 downto 24)<=sa03_sub ;
chip(39 downto 32)<=sa10_sub;
chip(47 downto 40)<=sa11_sub;
chip(55 downto 48)<=sa12_sub;
chip(63 downto 56)<=sa13_sub;
chip(71 downto 64)<=sa20_sub;
chip(79 downto 72)<=sa21_sub;
chip(87 downto 80)<=sa22_sub;
chip(95 downto 88)<=sa23_sub;
chip(103 downto 96)<=sa30_sub;
chip(111 downto 104)<=sa31_sub;
chip(119 downto 112)<=sa32_sub;
chip(127 downto 120)<=sa33_sub;
end behavioral;
library ieee;--------------library-----
Use ieee.std_logic_1164.all;
use ieee.numeric_bit.all;
use ieee.numeric_std.all;
Use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
Entity progsh is port(-------entityshifing ----
ri:in std_logic_vector(127 downto 0);
```

```
ro:out std_logic_vector(127 downto 0));
End progsh;---end
Architecture behavioral of progsh is  -----architecture---
signal roo:std_logic_vector(127 downto 0);
Begin
roo(31 downto 24)<=ri(31 downto 24);--line1 dont change
roo(63 downto 56)<=ri(63 downto 56);
roo(95 downto 88)<=ri(95 downto 88);
roo(127 downto 120)<=ri(127 downto 120);
roo(23 downto 16)<=ri(119 downto 112);
roo(55 downto 48)<=ri(23 downto 16);
roo(87 downto 80)<=ri(55 downto 48);
roo(119 downto 112)<=ri(87 downto 80);---shift2 end
roo(15 downto 8)<=ri(79 downto 72);
roo(47 downto 40)<=ri(111 downto 104);
roo(79 downto 72)<=ri(15 downto 8);
roo(111 downto 104)<=ri(47 downto 40);----shift3 end
roo(7 downto 0)<=ri(39 downto 32);
roo(39 downto 32)<=ri(71 downto 64);
roo(71 downto 64)<=ri(103 downto 96);
roo(103 downto 96)<=ri(7 downto 0);
ro(127 downto 0)<=roo(127 downto 0);
```

```
end behavioral;library ieee;  ---------------library-----
Use ieee.std_logic_1164.all;
use ieee.numeric_bit.all;
use ieee.numeric_std.all;
Use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
Entity progmix is port(--------entity --mixing ----
chin:in std_logic_vector(127 downto 0);
cho:out std_logic_vector(127 downto 0));
End progmix;---end
Architecture behavioral of progmix is  -----architecture---
signal wt: std_logic_vector(127 downto 0);
signal chr,te,t3: std_logic_vector(127 downto 0);
Begin
c1:process(chin,te,t3)
begin
if chin(7)='0'
then te(7 downto 1)<=chin(6 downto 0);
te(0)<='0';
wt(7 downto 0)<=te(7 downto 0);
end if;
if chin(7)='1'
```

```
then te(7 downto 1)<=chin(6 downto 0);
te(0)<='0';
wt(7 downto 0)<=te(7 downto 0)xor"00011011";
end if;
if chin(15)='0'then te(15 downto 9)<=chin(14 downto 8);
te(8)<='0';
wt(15 downto 8)<=te(15 downto 8);
end if;
if chin(15)='1'then te(15 downto 9)<=chin(14 downto 8);
te(8)<='0';
wt(15 downto 8)<=te(15 downto 8)xor"00011011";
end if;
if chin(23)='0'then te(23 downto 17)<=chin(22 downto 16);
te(16)<='0';
wt(23 downto 16)<=te(23 downto 16);
end if;
if chin(23)='1'then te(23 downto 17)<=chin(22 downto 16);
te(16)<='0';
wt(23 downto 16)<=te(23 downto 16)xor"00011011";
end if;
if chin(31)='0'then te(31 downto 25)<=chin(30 downto 24);
te(24)<='0';
```

```
wt(31 downto 24)<=te(31 downto 24);end if;
if chin(31)='1'then te(31 downto 25)<=chin(30 downto 24);
te(24)<='0';
wt(31 downto 24)<=te(31 downto 24)xor"00011011";
end if;
if chin(39)='0'then te(39 downto 33)<=chin(38 downto 32);
te(32)<='0';
wt(39 downto 32)<=te(39 downto 32);end if;
if chin(39)='1'then te(39 downto 33)<=chin(38 downto 32);
te(32)<='0';
wt(39 downto 32)<=te(39 downto 32)xor"00011011";
end if;
if chin(47)='0'then te(47 downto 41)<=chin(46 downto 40);
te(40)<='0';
wt(47 downto 40)<=te(47 downto 40);end if;
if chin(47)='1'then te(47 downto 41)<=chin(46 downto 40);
te(40)<='0';
wt(47 downto 40)<=te(47 downto 40)xor"00011011";
end if;
if chin(55)='0'then te(55 downto 49)<=chin(54 downto 48);
te(48)<='0';
wt(55 downto 48)<=te(55 downto 48);end if;
```

```
if chin(55)='1'then te(55 downto 49)<=chin(54 downto 48);
te(48)<='0';
wt(55 downto 48)<=te(55 downto 48)xor"00011011";end if;
if chin(63)='0'then te(63 downto 57)<=chin(62 downto 56);
te(56)<='0';
wt(63 downto 56)<=te(63 downto 56);end if;
if chin(63)='1'then te(63 downto 57)<=chin(62 downto 56);
te(56)<='0';
wt(63 downto 56)<=te(63 downto 56)xor"00011011";end if;
if chin(71)='0' then te(71 downto 65)<=chin(70 downto 64);
te(64)<='0';end if;
wt(71 downto 64)<=te(71 downto 64);
if chin(71)='1' then te(71 downto 65)<=chin(70 downto 64);
te(64)<='0';
wt(71 downto 64)<=te(71 downto 64)xor"00011011";end if;
if chin(79)='0'then te(79 downto 73)<=chin(78 downto 72);
te(72)<='0';
wt(79 downto 72)<=te(79 downto 72);end if;
if chin(79)='1'then te(79 downto 73)<=chin(78 downto 72);
te(72)<='0';
wt(79 downto 72)<=te(79 downto 72)xor"00011011";
end if;
```

```
if chin(87)='0'then te(87 downto 81)<=chin(86 downto 80);
te(80)<='0';
wt(87 downto 80)<=te(87 downto 80);end if;
if chin(87)='1'then te(87 downto 81)<=chin(86 downto 80);
te(80)<='0';
wt(87 downto 80)<=te(87 downto 80)xor"00011011";
end if;
if chin(95)='0'then te(95 downto 89)<=chin(94 downto 88);
te(88)<='0';
wt(95 downto 88)<=te(95 downto 88);end if;
if chin(95)='1'then te(95 downto 89)<=chin(94 downto 88);
te(88)<='0';
wt(95 downto 88)<=te(95 downto 88)xor"00011011";
end if;
if chin(103)='0'then te(103 downto 97)<=chin(102 downto 96);
te(96)<='0';
wt(103 downto 96)<=te(103 downto 96);end if;
if chin(103)='1'then te(103 downto 97)<=chin(102 downto 96);
te(96)<='0';
wt(103 downto 96)<=te(103 downto 96)xor"00011011";
end if;
if chin(111)='0'then te(111 downto 105)<=chin(110 downto 104);
```

```
te(104)<='0';
wt(111 downto 104)<=te(111 downto 104);end if;
if chin(111)='1'then te(111 downto 105)<=chin(110 downto 104);
te(104)<='0';
wt(111 downto 104)<=te(111 downto 104)xor"00011011";
end if;
if chin(119)='0'then te(119 downto 113)<=chin(118 downto 112);
te(112)<='0';
wt(119 downto 112)<=te(119 downto 112);end if;
if chin(119)='1'then te(119 downto 113)<=chin(118 downto 112);
te(112)<='0';
wt(119 downto 112)<=te(119 downto 112)xor"00011011";
end if;
if chin(127)='0'then te(127 downto 121)<=chin(126 downto 120);
te(120)<='0';
wt(127 downto 120)<=te(127 downto 120);end if;
if chin(127)='1'then te(127 downto 121)<=chin(126 downto 120);
te(120)<='0';
wt(127 downto 120)<=te(127 downto 120)xor"00011011";
end if;
end  process c1;
t3(7 downto 0)<=wt(7 downto 0)xor chin(7 downto 0);---**3
```

```
t3(15 downto 8)<=wt(15 downto 8)xor chin(15 downto 8);

t3(23 downto 16)<=wt(23 downto 16)xor chin(23 downto 16);

t3(31 downto 24)<=wt(31 downto 24)xor chin(31 downto 24);

t3(39 downto 32)<=wt(39 downto 32)xor chin(39 downto 32);

t3(47 downto 40)<=wt(47 downto 40)xor chin(47 downto 40);

t3(55 downto 48)<=wt(55 downto 48)xor chin(55 downto 48);

t3(63 downto 56)<=wt(63 downto 56)xor chin(63 downto 56);

t3(71 downto 64)<=wt(71 downto 64)xor chin(71 downto 64);

t3(79 downto 72)<=wt(79 downto 72)xor chin(79 downto 72);

t3(87 downto 80)<=wt(87 downto 80)xor chin(87 downto 80);

t3(95 downto 88)<=wt(95 downto 88)xor chin(95 downto 88);

t3(103 downto 96)<=wt(103 downto 96)xor chin(103 downto 96);

t3(111 downto 104)<=wt(111 downto 104)xor chin(111 downto 104);

t3(119 downto 112)<=wt(119 downto 112)xor chin(119 downto 112);

t3(127 downto 120)<=wt(127 downto 120)xor chin(127 downto 120);

chr(7 downto 0)<=t3(31 downto 24)xor wt(7 downto 0)XOR chin(23 downto 16)xor
chin(15 downto 8);---out

chr(15 downto 8)<=chin(31 downto 24)xor chin(23 downto 16)xor wt(15 downto
8)xor t3(7 downto 0);

chr(23 downto 16)<=chin(31 downto 24)xor chin(7 downto 0)xor wt(23 downto
16)xor t3(15 downto 8);

chr(31 downto 24)<=chin(15 downto 8)xor chin(7 downto 0)xor wt(31 downto
24)xor t3(23 downto 16);
```

chr(39 downto 32)<=chin(47 downto 40)xor chin(55 downto 48)XOR t3(63 downto 56)xor wt(39 downto 32);

chr(47 downto 40)<=chin(55 downto 48)xor chin(63 downto 56)XOR t3(39 downto 32)xor wt(47 downto 40);

chr(55 downto 48)<=chin(39 downto 32)xor chin(63 downto 56)XOR t3(47 downto 40)xor wt(55 downto 48);

chr(63 downto 56)<=chin(47 downto 40)xor chin(39 downto 32)XOR t3(55 downto 48)xor wt(63 downto 56);

chr(71 downto 64)<=chin(79 downto 72)xor chin(87 downto 80)xor wt(71 downto 64)xor t3(95 downto 88);

chr(79 downto 72)<=chin(87 downto 80)xor chin(95 downto 88)XOR t3(71 downto 64)xor wt(79 downto 72);

chr(87 downto 80)<=chin(95 downto 88)xor chin(71 downto 64)XOR t3(79 downto 72)xor wt(87 downto 80);

chr(95 downto 88)<=chin(71 downto 64)xor chin(79 downto 72)XOR t3(87 downto 80)xor wt(95 downto 88);

chr(103 downto 96)<=chin(111 downto 104)xor chin(119 downto 112)XOR t3(127 downto 120)xor wt(103 downto 96);

chr(111 downto 104)<=chin(119 downto 112)xor chin(127 downto 120)XOR t3(103 downto 96)xor wt(111 downto 104);

chr(119 downto 112)<=chin(127 downto 120)xor chin(103 downto 96)XOR t3(111 downto 104)xor wt(119 downto 112);

chr(127 downto 120)<=chin(103 downto 96)xor chin(111 downto 104)XOR t3(119 downto 112)xor wt(127 downto 120);

cho(127 downto 0)<=chr(127 downto 0);

```
end behavioral;

library ieee;----------------library- Project PROGRAM---

Use ieee.std_logic_1164.all;

use ieee.numeric_bit.all;

use ieee.numeric_std.all;

Use ieee.std_logic_arith.all;

use ieee.std_logic_unsigned.all;

entity enc_aes is

    Port ( plain_text : in STD_LOGIC_VECTOR (127 downto
0):="00110010010000111111011010101000100010000101101000110000100011
01001100010011000110011000101000101110000000110111000001110011010
100";

--plaintext : in STD_LOGIC_VECTOR (127 downto
0):="00000000000100010010001000110011010001000101010101100110011101
11110001000100110011010101010111011110011001101110111011101110111111
111";

--plain_text : in std_logic_vector (127 downto
0):="01010100011101110110111100100000010011110110111001100101001000
00001001110011010010110111001100101001000000101010001110111011011
111";

--clk2:out std_logic;

          cipher_text : out STD_LOGIC_VECTOR (127 downto 0));

end enc_aes;

architecture Behavioral of enc_aes is
```

```
component progsub is port(
chi:in std_logic_vector(127 downto 0);
chip:out std_logic_vector(127 downto 0));
end component;
component progsh is port(ri:in std_logic_vector(127 downto 0);
ro:out std_logic_vector(127 downto 0));
end component;
component progmix is port(
chin:in std_logic_vector(127 downto 0);
cho:out std_logic_vector(127 downto 0));
end component;
signal w1,w2,w3,w4,w5,w6,w7,w8,w9,w10,w11,w12,w13,w14,w15,w16
:std_logic_vector(127 downto 0);
signal
w17,w18,w19,w20,w21,w22,w23,w24,w25,w26,w27,w28,w29,w30,w31,w32,w33,w
34,w35,w36,w37,w38,w39 :std_logic_vector(127 downto 0);
signal key0,k1,k2,k3,k4,k5,k6,k7,k8,k9,k10:std_logic_vector(127 downto 0);
signal clk1: std_logic:='1';
--constant cfreq:integer:=1e6;
--constant cperiod : time:= 1000 ms /cfreq;
begin
  --u0:progr port map(plaintext,w1);
```

key0<="00101011011111100001010100010110001010001010111011010010101001101010101111110111000101011000100000001001110011110100111100111100";

k1<="10100000111110101111111000010111100010000101010000101100101100010010001110100011001110010011100100101010011011000111011000000101";

k2<="11110010110000101001010111110010011110101001011010111001010001101010110010011010110000000011110100111001101011001111101100111111111";

k3<="00111101100000000100011101111101010001110001011011111110001111100001111000100011011111100100010001101101011110101000100000111011";

k4<="11101111010001001010010101000001101010000101001001011011011111101101100111000100100101001110111101101100001011101011010000000000";

k5<="11010100110100011100011011111000011111001000001110011101100001111100101011100010111100000100011111100100010101101111100";

k6<="01101101100010001010001101111010000100010000101100111110111101101111111001100001100100000111001010000000001001001111111101";

k7<="01001110010101001111011100001110010111110101111111001001111100011100001001010011001001111101100100100111010100110110111000100111111";

```
k8<="11101010110100100111001100100001101101011000110110111010110100100011000100101011111101010110000001111111100011010010100100101111";

k9<="10101100011101110110011011110011000110011111101011011100001000010010100011010001001010010100000101010111010111000000000001101110";

k10<="11010000000101001111100110101000110010011110111000100101100010011110000100111111000011001100100010110110011000110000110010100110";

w1<=key0 xor plain_text;

 u1: progsub port map(w1,w2);

u2: progsh port map(w2,w3);

u3: progmix port map(w3,w4);

w5<=w4 xor k1;

---round1

u5: progsub port map(w5,w6);

u6: progsh port map(w6,w7);

u7: progmix port map(w7,w8);

---round2

w9<=w8 xor k2;

u9: progsub port map(w9,w10);

u10:progsh port map(w10,w11);

u11: progmix port map(w11,w12);
```

```
 --round3
w13<=w12 xor k3;
u13:progsub port map(w13,w14);
u14:progsh port map(w14,w15);
u15:progmix port map(w15,w16);
 --round 4
w17<=w16 xor k4;
u17:progsub port map(w17,w18);
u18:progsh port map (w18,w19);
u19:progmix port map (w19,w20);
  ---round 5
w21<=w20 xor k5;
u21:progsub port map (w21,w22);
u22:progsh port map(w22,w23);
u23:progmix port map (w23,w24);
--u24:progr port map (w24,w25); ---round 6
w25<=w24 xor k6;
u25:progsub port map (w25,w26);
u226:progsh port map (w26,w27);
u27:progmix port map (w27,w28);
--u28:progr port map (w28,w29); ---round  7
w29<=w28 xor k7;
```

```
u29:progsub port map (w29,w30);

u30:progsh port map (w30,w31);

u31:progmix port map(w31,w32);

 --round 8

w33<=w32 xor k8;

u33:progsub port map (w33,w34);

u34:progsh port map (w34,w35);

u35:progmix port map (w35,w36);

 ---round 9

w37<=w36 xor k9;

u37:progsub port map(w37,w38);

u38:progsh port map (w38,w39);

--round 10

cipher_text<=w39 xor k10;

end behavioral;
```

الملحق 2– الدراسات المرجعية المعتمدة في المقارنة نسخة منها

Syrian Arab Republic

Ministry of Higher Studies and Scientific Research

Damascus University

Faculty of Mechanical & Electrical Engineering

Electronics And Communication Engineering

Department

Evaluation of the Performance of Implementing Advanced Encryption Standard (AES)using Field Programmable Gate Array(FPGA) Chip

A Thesis Submitted in Partial Fulfillment of

the requirement for the Degree of Advanced Communication Engineering

By

Eng.Mervat Alsharef

Supervised by

Dr.Eng.Adel Khadour Ali

2021-2022