وزارةالتعليـــم العالـــي و البحـــث العلمـــــــي

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

جامعـــة – أكلي محند أولحاج -البويرة

UNIVERSITE Akli Mohand Oulhadj —Bouira

(ALGERIE)

Mémoire de Master

Présenté au département de Génie Electrique
Faculté des Sciences et Sciences Appliquées
Pour obtenir le diplôme

De Master

En:

Systèmes électroniques complexes

Par :

M^elle^.BEN AMMAR Asma

M^elle^.HADDOUCHE Khalissa

Thème

Amélioration de la génération des sous clés de l'algorithme cryptographique DES

Soutenu le 25/09/2017 devant le jury composé de :

Mr. I.Griche	Maître Assistant à l'université de Bouira	*Président*
Mm. K. Dichou	Enseignante à l'université de boumerdes	*Co-Encadreur*
Mr. M.Nourine	Maître de conférence à l'université de Bouira	*Examinateur*
Mr.M.Djibiri	Maître Assistant à l'université de Bouira	*Examinateur*
Mr. M. Rezki	Maître de conférence à l'université de Bouira	*Encadreur*

Remerciements

Nous remercions Dieu le tout puissant de nous avoir accordé la santé, le courage et la volonté d'arriver au terme de ce travail.

Nous remercions chaleureusement notre promoteur Monsieur Dr.REZKI MOHAMED et Mademoiselle Dr. DICHOU KARIMA pour toutes les orientations et les conseils qu'il nous a prodigués tout au long de ce travail.

Nos remerciements s'adressent surtout à l'enseignant (Dr. BENSEGUINI SKANDER) et l'enseignant (Dr.CHALABI IZEDDINE) l'enseignant (Dr. NOURINE MOURAD) et pour toute les orientations et les conseils qu'elle nous a prodigué tout le long de ce travail.

Ainsi que tous nos enseignent qui nous ont enseigné durant nos études au département de Génie Electriques.

À nos familles, notamment nos chers parents qui nous ont imbibés d'amour et d'affection et à nos frères qui nous ont donnés un coup de main et ils ont contribué à ce que ce modeste travail voie le jour.

Nous tenons à remarcier tous nous collégues d'étude, partculiérement notre promotion

Enfin, nous tenons aussi à exprimer notre profonde gratitude à toute personne de près ou de loin, ceux qui nous ont encouragés.

Merci

Dédicaces

Je dédie ce modeste travail à :

A mes très chers parents, ELHACHMI et ZOINA, soucieux de ma réussite, pour leur incitation à avancer dans tous ce que j'entreprends et pour tous les sacrifices qu'ils ont consentis pour mon instruction, qu'ils trouvent ici l'accomplissement de leurs vœux. Que Dieu les garde et les entoure de sa bénédiction.

A ma chère grand-mère : Fatima

A mes chers frères «Mohammed et le petit Radouane»

A mes très chères sœurs : Amal, siham, lobna

A mes oncles et tantes, cousins et cousines, ainsi qu'à toute ma famille

A toutes mes amies : Soumeya, Habiba, Ilhame, Khalissa, Mariam, wassim

Ainsi qu'à toute personne qui m'est chère. Et à tous ceux qui m'ont aidé de prés ou de loin à accomplir ce travail.

BEN AMMAR Asma

Dédicaces

Je dédie ce modeste travail à :

À Mes très chers parents,

qui me sont les plus chers au monde, dont l'amour

et les sacrifices n'ont pas cessé de combler ma vie ;

que Dieu les protège et les garde pour moi

A mes très chers sœurs et frères

qui n'ont cessé d'être pour moi des exemples de persévérance,

de courage et de générosité

À l'esprit du mon frère décédé le grand ***Rabah.***

A mon très cher Mari

Merci pour ton aide, tes conseils et tes encouragements

A ma belle famille

Mes dédicaces spéciales vont particulièrement

pou mes neveux ***AKRAM, WASSIM, AHMED ,***

ABDERRAHIME *et* ***ADAM*** *et ma nièce* ***SIRINE.***

Que le bon dieu les garde pour nous

A Mes professeurs d'AMO qui doivent voir dans ce travail la fierté

d'un savoir bien acquis

A toutes mes amies et mes collègues de la promotion d'infotronique

A ma binôme ***ASMA***

Et à tous ceux qui m'ont aidé de prés ou de loin à accomplir ce travail.

à tous ceux que j'aime,
et à tous ceux qui m'aiment ...

KHALISSA

Table des matières

Chapitre 2 : Principe de chiffrement par DES

Chapitre 3 : Proposition d'un DES amélioré

Chapitre 4 : Implémentation du DES amélioré sur FPGA

Liste des figures

Chapitre 1

Chapitre 2

Chapitre 3

Chapitre 4

Liste des tableaux

Chapitre 1

Chapitre 2

Chapitre 3

Chapitre 4

Liste des abréviations

Abréviation	Signification
AES	Advanced Encryption Standard.
ANSI	American National Standards Institute.
CLB	Configurable logic block.
CPU	Central Processing Unit.
DES	Data Encryption Standard.
ECC	Elliptic Curve Cryptography.
FPGA	Field Programmable Gate Array.
GCLK	Global Clock.
IBM	International Business Machines.
FP	Final Permutation.
IOB	Input Output Bloc
IP	Initial Permutation.
ISE	Integrated Software Environment.
LUT	Look Up Table.
NBS	National Bureau of Standards.
NIST	National Institute of Standards and Technology.
NSA	National Security Agency.
PC1	Permuted Choice 1.
PC2	Permuted Choice 2.
RAM	Random Access Memory.
ROM	Read Only Memory.
RSA	Rivest Shamir Adelman cryptosystem.
RTL	Register Transfer Level.
S-Box	Substitution Box.
TDES	Triple DES.
VHDL	Very height speed integrated circuit Hardware Description Language.
XOR	Exclusive OR.

Introduction Générale

L'information est un élément constitutif et déterminant dans tous les domaines. Depuis l'invention de l'écriture, l'humanité exprime le besoin de transmettre leurs informations de manière sécurisée en les rendant inintelligibles pour toute personne étrangère à l'échange, c'est-à-dire que les messages ne peuvent pas être compris par l'ennemi, même s'ils sont interceptés. Ils se servaient donc d'outils permettant de garder leurs confidences hors d'atteinte des yeux indiscrètes : signes et symboles intelligibles, figures ou couleurs, usage d'expressions ou phrases convenues d'avoir un sens spécifique qui diffère de l'ordinaire, etc. La progression de ces outils primitifs à travers le temps, a permis de concevoir des règles de sécurité plus efficaces et plus logiques qui ont donné naissance à la cryptologie.

La cryptologie est une science mathématique qui étudie les communications secrètes. Elle est composée de deux domaines d'étude complémentaires : la cryptographie et la cryptanalyse [1], le rôle des cryptographes est de construire des systèmes de chiffrement, l'objectif des cryptanalyses est de " casser" ces systèmes [2].

La cryptographie propose un ensemble de techniques permettant d'assurer la confidentialité, l'authentification, l'intégrité des données et la non-répudiation de la source de données. Nous distinguons deux grandes catégories de techniques cryptographiques : celles à chiffrement symétrique ou à clé secrète et celles à chiffrement asymétrique ou à clé publique.

Les algorithmes cryptographiques symétriques sont des fonctions de deux paramètres :l'algorithme et la clé ; la sécurité de tout le crypto-système (cryptographie) réside dans la clé (à cause de) l 'algorithme peut être public. Il existe plusieurs algorithmes symétriques parmi eux on trouve l'algorithme DES .

Le DES possède une clé primaire de 64 bits qui est utilisée pour créer 16 sous-clés de 48 bits [3]; parmi ces 16 sous clés on trouve des sous clés redondantes à cause de l'utilisation des clés faibles ou semi-faibles ce qui réduit la compléxité de chiffrement amplique que diminuer la sécurité d'échange d'informations.

Notre contribution consiste à ajouter une étape au processus de génération des sous-clés dont le but est d'améliorer la génération des sous-clés pour avoire des sous clés non redondant ainsi que pour augmenter la complexité de chiffrement .

Cette mémoire est organisée en quatre chapitres répartis comme suit :

Le Chapitre 1 amène un aperçu des techniques de cryptographie classique et moderne et de ses deux types à savoir cryptographies symétrique et asymétrique tout en définissant les algorithmes de chiffrement les plus connus.

Le chapitre 2 présente une discussion de l'algorithme cryptographique DES avec une explication détaillée de son principe de fonctionnement.

Le Chapitre 3 propose d'un DES amélioré avec l'implémentation de deux versions (classique et améliorée) de DES sur MATLAB dans le but de la validation et la vérification des résultats.

Le Chapitre 4 montre l'implémentation du DES classique et amélioré sur FPGA ainsi que les résultats expérimentaux compilés. Nous terminons ce rapport par une conclusion et la présentation de perspectives.

Chapitre 1

Généralités sur la cryptologie

1.1. Introduction

L'information est un élément constitutif et déterminant dans tous les domaines. Tout au long de l'histoire, l'humanité a essaye d'envoyer des informations d'une façon sécurisée. Alors comment assurer la sécurité d'échange d'information ?

La cryptologie est un moyen approprié pour assurer la sécurité de l'information en utilisant les mathématiques pour atteindre des objectifs tels que : la confidentialité, l'authenticité et l'intégrité des messages échangés à travers des voies de communication. Dans ce chapitre, nous donnons un aperçu des techniques de cryptographie classique et moderne et de ses deux types à savoir cryptographies symétrique et asymétrique tout en définissant les algorithmes de chiffrement les plus connus.

1.2. Domaines de cryptologie

La cryptologie composée de deux domaines d'étude complémentaires : la cryptographie et la cryptanalyse [1]. Généralement on distingue deux types de cryptographie : la cryptographie symétrique à clé secréte et la cryptographie asymétrique à clé publique comme montre la figure suivant.

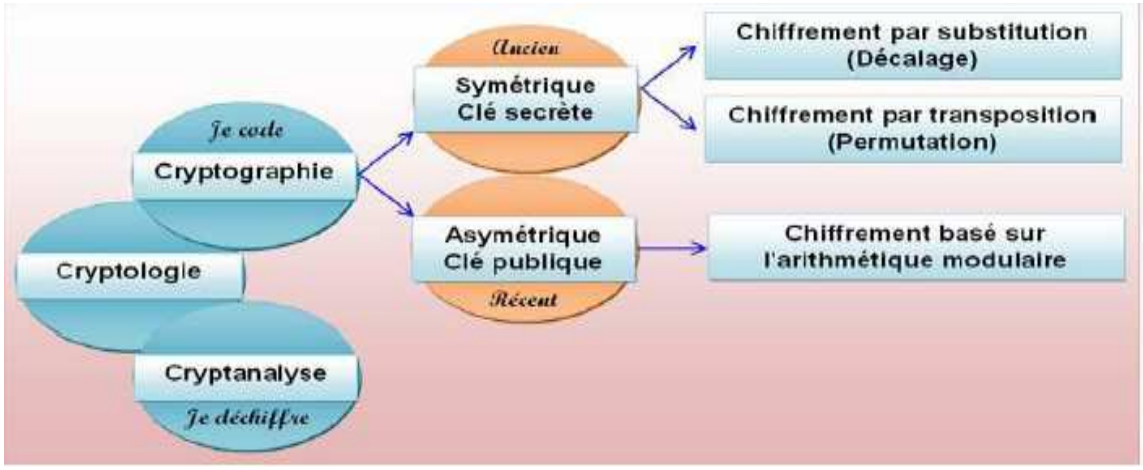


Figure1.1: schéma générale de la cryptologie [1]

1.3. La cryptographie

La cryptographie, partie intégrante de la cryptologie, une science au croisement des mathématiques, de l'informatique, et de la physique, qui étudie l'ensemble de techniques permettant de chiffrer un message et de le rendre inintelligible sauf pour son destinataire : cette opération s'appelle le chiffrement [3] [4]. Elle fournit un message chiffré ou cryptogramme (en anglais *ciphertext*), à partir d'un message en clair (en anglais *plaintext*). Inversement, le déchiffrement est l'action de reconstruire le texte en clair à partir du texte chiffré. Ces fonctions de chiffrement et

déchiffrement sont des fonctions mathématiques appelées algorithmes cryptographiques (cryptosystèmes), qui dépendent d'un paramètre appelé clé [4].

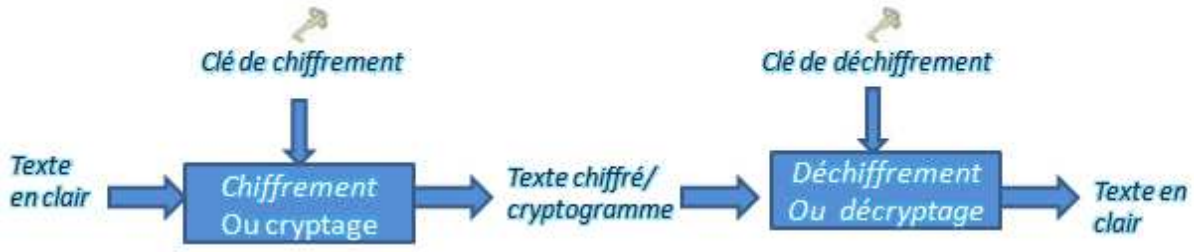


Figure.1.2: Terminologies de la cryptographie : cryptage et décryptage [4].

- **Quelques terminologies [1]:**

Crypter: brouiller l'information, la rendre "incompréhensible"

Décrypter: rendre le message compréhensible.

Texte en clair : c'est le message à protéger.

Texte chiffré : c'est le résultat du chiffrement du texte en clair.

Déchiffrement: c'est la méthode ou l'algorithme utilisé pour transformer un texte chiffré en texte en clair.

Clé: c'est le secret partagé utilisé pour chiffrer le texte en clair en texte chiffré et pour déchiffrer le texte chiffré en texte en clair.

1.3.1. L'objectif de la cryptographie [5] [1] [6] [7]

La confidentialité englobe l'ensemble des dispositions qui assure que l'information reste secrète de toutes les personnes qui ne sont pas autorisées à y accéder.

L'intégrité qui assure que l'information n'a pas été modifiée entre son envoi et sa réception.

L'authenticité qui permet de certifier l'identité de la personne qui envoie le message de façon à limiter l'accès aux données.

La non-répudiation qui désigne l'assurance d'une transmission entre deux personnes, que l'expéditeur peut vérifier qu'un certain destinataire a reçu un message particulier.

1.3.2. La cryptographie classique

La cryptographie classique concerne la période de l'antiquité jusqu'à l'apparition des ordinateurs. Elle traite des systèmes reposant sur les lettres et les caractères d'une langue naturelle (allemand, anglais, français, etc...) [8]. Et inclut tous les mécanismes et algorithmes basés sur des fonctions mathématiques ou logiques (exemple: Cesar, Vigner) [9].

La plupart des méthodes de cryptographie classique reposent sur deux principes essentiels : la substitution et la transposition [10].

1.3.2.1. La cryptographie par substitution

La substitution consiste à remplacer certaines entités (généralement des lettres) par d'autres ou par des symboles dans un message [10]. On distingue généralement plusieurs types de crypto systèmes par substitution [1] [3]:

La substitution mono-alphabétique : est le plus simple à imaginer consiste à remplacer chaque lettre du message par une autre lettre de l'alphabet. Comme le chiffrement par décalage.

La substitution poly alphabétique : Consiste à utiliser une suite de chiffres mono alphabétique réutilisée périodiquement.

La substitution homophonique: Permet de faire correspondre à chaque lettre du message en clair un ensemble possible d'autres caractères.

La substitution de poly grammes : Consiste à substituer un groupe de caractères (poly gramme) dans le message par un autre groupe de caractères.

1.3.2.2. Chiffrement par transposition ou chiffrement par permutation

Les méthodes de chiffrement par transposition consistent à réarranger les données à crypter de telle façon à les rendre incompréhensibles. Il s'agit généralement de réarranger géométriquement les données pour les rendre visuellement inexploitables. Avec le principe de la transposition toutes les lettres du message sont présentées, mais dans un ordre différent [1].

Transposition complexe par colonnes :

Un mot clé secret (avec uniquement des caractères différents) est utilisé pour dériver une séquence de chiffres commençant de 1 et finissant au nombre de lettres composant le mot clé. Cette séquence est obtenue en numérotant les lettres du mot clé en partant de la gauche vers la droite et en donnant l'ordre d'apparition dans l'alphabet. Une fois la séquence de transposition obtenue, on chiffre en écrivant d'abord le message par lignes dans un rectangle, puis on lit le texte par colonnes en suivant l'ordre déterminé par la séquence [8].

Transposition par carré poly bique :

Un mot clé secret est utilisé pour construire un alphabet dans un tableau. Les coordonnées des lignes et des colonnes correspondant aux lettres du texte à chiffrer sont utilisés pour transcrire le message en chiffres. Avec ce procédé chaque lettre du texte en clair est représentée par deux chiffres écrits verticalement. Ces deux coordonnées sont ensuite transposées en les recombinants par deux sur la ligne ainsi obtenue [8].

1.3.3. La cryptographie moderne

Si le but de la cryptographie classique est d'élaborer des méthodes permettant de transmettre des données de manière confidentielle, la cryptographie moderne s'intéresse en fait plus généralement aux problèmes de sécurité des communications [10]. Pour cela, on utilise un certain nombre de mécanismes basés sur des algorithmes cryptographiques [11].

La sécurité des données chiffrées dépend des éléments suivants [11]:

- La robustesse de l'algorithme cryptographique (difficile à casser).
- La confidentialité de la clé.

Les techniques de cryptographie moderne se composent de grandes parties comme le montre la figure.1.3 [9] [12] [13] :

- La cryptographie à clés secrètes ou cryptographie symétrique.
- La cryptographie à clés publiques ou cryptographie asymétrique.

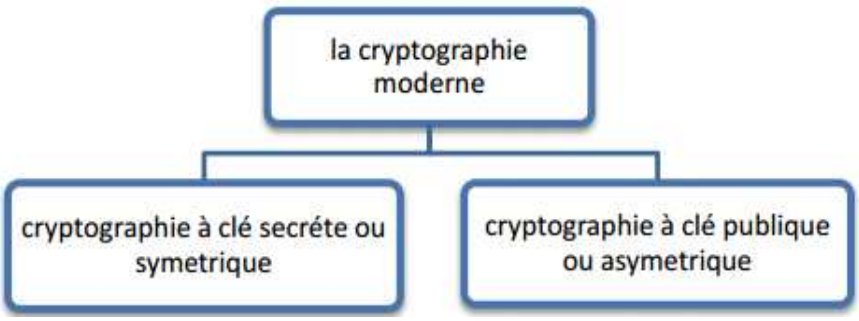


Figure .1.3: les méthodes de la cryptographie moderne [9].

Ils ont tous les deux leurs avantages et leur inconvénients. La différence qui existe entre ces deux types se situe au niveau de la clé [9].

1.3.3.1. La cryptographie symétrique

Le chiffrement symétrique (aussi appelé chiffrement à clé privée ou chiffrement à clé secrète) consiste à utiliser la même clé et le même algorithme pour le chiffrement et le déchiffrement entre deux personnes en communication qui est seulement connue cette clé, comme le représente la figure 1.4. [12] [14].

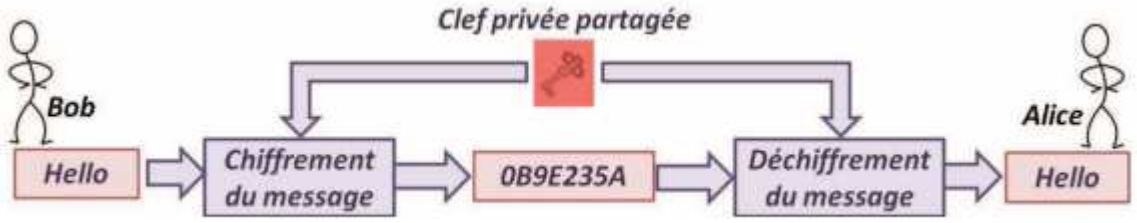


Figure 1.4 : Principe de cryptographie symétrique [4].

-Plus précisément, pour transmettre un message, L'émetteur (Bob) réalise au préalable le chiffrement en utilisant une clé secrète. A la réception, le destinataire (Alice) reçoit le message crypté et applique un algorithme de déchiffrement avec la même clé que Bob utilise pour le message en clair [15] [16].

Le chiffrement symétrique se divise en deux parties : chiffrement par bloc (block ciphers) et chiffrement par flot (stream ciphers) [9].

Chiffrement par flot : dans un crypto système par flots, le cryptage des messages se fait caractère par caractère ou bit par bit, au moyen de substitutions générées aléatoirement, la taille de la clé est donc égale à la taille du message [9] [15].

Chiffrement par bloc : dans un algorithme de chiffrement par bloc, chaque message clair est découpé en blocs de taille fixe de même longueur et chiffré à l'aide d'une clé unique. Ces algorithmes sont en général construits sur un modèle itératif. Il utilise une fonction F qui prend une clé secrète k et un message M de n bits. La fonction F est itérée un certain nombre de fois (nombre de tours). Lors de chaque tour, la clé k est différente et on chiffre le message qui vient d'être obtenu de l'itération précédente. Les différentes clés k(i) qui sont utilisées sont déduites de la clé secrète k. Les algorithmes les plus connus des systèmes cryptographique symétriques sont : le DES et l'AES [2].

1.3.3.1.1. DES (Data Encryption Standard)

La norme DES (*Data Encryption Standard*) est adoptée par NSA en 1967. C'est un algorithme de chiffrement par bloc qui rassemble les deux techniques de base : la confusion (substitution) et la diffusion (permutation) [17]. Et consiste à chiffrer un bloc de texte de 64 bits pour produire un cryptogramme de 64 bits a partir d'une clé de 56 bits [3].

1.3.3.1.2. AES (Advanced Encryption Standard)

L'AES est un algorithme de chiffrement par bloc utilisant des clés de 128, 192 ou 256 bits. Le nombre de rondes dépend de la taille des clés : 10 rondes pour des clés de 128 bits, 12 rondes pour des clés de 192 bits et 14 rondes pour des clés de 256 bits. Enfin, la taille des blocs est de 128 bits. Chaque ronde d'un bloc est constituée d'une succession de XOR avec la sous-clé correspondante, d'une fonction de substitution et de permutation [10]. Les fonctions de substitution et de permutation sont utilisées pour garantir la confusion (aucune propriété statique ne peut être déduite du message chiffré) et la diffusion (toute modification du message en clair se traduit par une modification complète du chiffré). Le choix des clés dépend du niveau de protection que l'on souhaite attribuer aux documents ; c'est ainsi que la NSA recommande l'emploi des clés de 192 ou 256 bits pour des documents top-secrets [18]. Enfin, les algorithmes de chiffrement par bloc sont plus rapides que les algorithmes par flot à cause de leur parallélisme. Sans oublier que tout secret doit être dans la clé et pas dans l'algorithme car les algorithmes de chiffrement sont connus par tout le monde [9].

1.3.3.2. La cryptographie asymétrique

En 1976, Whitfield Diffie et Martin E. Hellman décrit la possibilité de développer un algorithme de chiffrement basé sur deux clés différentes [6] [10] [19].

La cryptographie à clé publique (asymétrique) consiste en l'existence d'une paire de clés de chaque côté (émetteur et récepteur) liées mathématiquement. Chaque paire est composée d'une clé privée (et différente pour chaque utilisateur qui doit être gardée secrète), et d'une clé publique connu par tous les utilisateurs [20] [11] [15].

La cryptographie asymétrique se base sur des fonctions à sens unique. Cela veut dire que les données cryptées avec la clé publique ne peuvent être décryptées que s'il on possède la clé secrète. Ça signifie que même si l'on a obtenu la clé publique, on ne pourra pas déchiffrer les informations [20].

Figure.1.5 : Principe de cryptographie asymétrique [4].

Le protocole le plus connu dans le chiffrement asymétrique est le protocole RSA.

1.3.3.2.1. **L'algorithme RSA**

L'algorithme à clef publique RSA (Rivest, Shamir et Adleman) a été inventé en 1977 par Ronald Rivest, Adi Shamir et Leonard Adleman. Sa sécurité réside dans la difficulté à factoriser le produit de deux grands nombres premiers [19] [21].

Le principe de l'algorithme RSA est le suivant :

Le destinataire choisit deux grands nombres premiers p et q de même ordre et calcule leur produit $N= p\times q$ est (représente le module public). Il choisit également un nombre e tel que e et (p- 1) ×(q-1) soient premiers entre eux. La clef publique est le couple (N, e) La clef privée d est quant à elle calculée comme : e. $d = mod$ ((p-1). (q-1)). N et e sont rendus publics.

Pour chiffrer un message M l'émetteur calcule : $C= M^e$ mod N et envoie C, le récepteur déchiffre le message à l'aide de la clef privée d en calculant $C^d \ mod\ N = (M^e \ mod\ N)^d \ mod\ N = M$ [4] [22]. Ou $M = C^d \ mod$ N [17].

L'expéditeur et le destinataire doivent connaître la valeur de N. L'expéditeur connaît la valeur de e, et seul le récepteur connaît la valeur de d. Ainsi, la clé publique est définie comme $\{e, N\}$ et la clé privée est définie comme $\{d, N\}$ [6].

La sécurité de l'algorithme RSA repose sur la difficulté de factoriser les grands nombres : factoriser un grand nombre N en deux nombres premiers p et q et retrouver la clef privée d à partir de N et e [4] [19].

1.4. Les courbes elliptiques

Les courbes elliptiques sont un sujet très à la mode en mathématiques, très complexes et très riches en même temps. Elles sont à l'origine de nouveaux algorithmes de cryptographie très sûrs, et on entrevoit les prémices de leur utilisation pour la factorisation de grands nombres entiers [23]. En cryptographie, les courbes elliptiques peuvent être utilisées pour des opérations asymétriques comme des échanges de clés sur un canal non-sécurisé ou un chiffrement asymétrique.

Les clés employées pour un chiffrement par courbe elliptique sont plus courtes qu'avec un système fondé sur le problème de la factorisation comme La cryptographie à clé publique du RSA (une clé de 160 bits lorsque RSA utilise une clé de 1024 bits, à un niveau de sécurité équivalent) Cela signifie des calculs plus rapides et une consommation d'énergie plus faible ainsi que des

économies de mémoire et de bande passante. De plus, l'ECC procure un niveau de sécurité équivalent ou supérieur aux autres méthodes [24].

1.5. Les avantages et les inconvénients de cryptographies symétriques et asymétriques

cryptographie	avantages	Inconvénients
Symétrique	-Système rapide de chiffrement /déchiffrement. -Clés relativement courtes (128 ou 256 bits). -Primitive de mécanismes cryptographiques, et Bonne performances et sécurité bien étudié. -Assure la confidentialité des données.	-Gestion des clés difficiles (nombreux clés). -Point faible : l'échange de la clé secrète. -Dans un réseau de N entités susceptibles de communiquer secrètement il faut distribuer N ∗(N-1) /2 clés.
Asymétrique	-Pas de secrèt à transmettre. -Nombre clés à distribuer est réduit par rapport aux clés symétriques. -Très utile pour échanger des messages facilement. -La distribution est simplifiée : La clé privée n'est jamais révélée ou transmise et la clé publique est disponible à tous les utilisateurs.	-Les algorithmes à clé publique nécessitent une capacité de traitement importante, ce qui n'est pas raisonnable pour les systèmes à ressources limitées. -La relation clés publique/clés privée impose : -La taille de clés et relativement longue (généralement entre 512 et 2048 bits). -Gestion de certificats de clés publiques. -Lenteur de calcul. -Pas d'authentification de la source.

Table.1.1 : Les avantages et les inconvénients symétriques/asymétriques [25] [17] [26].

1.6. La cryptanalyse

La cryptanalyse, a l'inverse de la cryptographie, est l'étude des procédés cryptographiques dont le but est de reconstruire les messages en clair correspondant à des messages chiffrés dons on n'est pas destinataire à l'aide de méthodes et techniques mathématiques sans connaître la clef de chiffrement [4] [7] [12] [27] [28].

Une tentative de cryptanalyse d'un système est appelée une attaque, et elle peut conduire à différentes résultats :

Cassage complet : le cryptanalyste retrouve la clef de déchiffrement.

Obtention globale : le cryptanalyste trouve un algorithme équivalent à l'algorithme de déchiffrement, mais qui ne nécessite pas la connaissance de la clef de déchiffrement.

Obtention locale : le cryptanalyste retrouve le texte en clair correspondant à un message chiffré.

Obtention d'information : le cryptanalyste obtient quelques indications sur le texte en clair ou la clef (certains bits de la clef, un renseignement sur la forme du texte en clair,...) [21].

Une attaque cryptanalytique se caractérise selon des données dont elle dispose, ainsi on peut trouver quatre situations de cryptanalyse :

Attaque sur texte chiffré seul : disposition seulement d'un nombre fini de textes chiffrés pour retrouver la clef de déchiffrage.

Attaque à texte clair connu : disposition de couple de message (clairs, chiffrés).

Attaque à texte clair choisi : l'attaquant a accès à l'algorithme de chiffrement et il l'utilise pour générer le couple (clairs, chiffrés).

Attaque à texte chiffré choisi : Le cryptanalyste peut choisir les textes à déchiffrer sans connaître la clef [29].

1.6.1. Cryptanalyse différentielle

L'idée de la cryptanalyse différentielle revient à (Biham & Shamir, 1991) Il s'agit d'une attaque basée sur des données claires choisies [5] [30]. La cryptanalyse différentielle est un type d'attaques qui peut-être utilisé contre les algorithmes de chiffrement par blocs itératifs. Elle utilise une attaque à texte en clair choisi et se base sur l'observation de l'évolution des différences entre deux textes lorsqu'ils sont chiffrés avec la même clef. En analysant ces différences entre paires de textes, il est possible d'attribuer des probabilités à chaque clef possible. A force d'analyser des paires de textes, on finit soit par trouver la clef recherchée, soit par réduire suffisamment le nombre de clefs possibles pour pouvoir mener une attaque exhaustive rapide [21].

1.6.2. Cryptanalyse linéaire

Fût proposée par (H. Gilbert et par M. Matsui 1993) et destinée à casser les chiffrements symétriques [5] [30], La cryptanalyse linéaire utilise une attaque à texte en clair connu et consiste à modéliser l'algorithme de chiffrement par blocs par une approximation linéaire. Avec un nombre suffisant de paires (texte en clair, texte chiffré), on peut deviner certains bits de la clef [21].

Comme la plupart des attaques aux chiffrements par blocs, la cryptanalyse linéaire cible le dernier tour de l'algorithme de chiffrement et tente de déterminer un ensemble de positions des bits i_1 , i_2 ,..., i_m du texte clair et un autre ensemble de positions i_1, i_2 ,..., i_n des bits du texte chiffré (avec *m* pas forcément égal à *n*) tel que la somme de chaque série de bits prend la même valeur pour la plupart des textes clairs utilisés [5] [30].

1.7. Conclusion

Dans ce chapitre, nous avons définit la cryptologie et ses deux domaines d'étude complémentaires : la cryptographie et la cryptanalyse. Après nous avons présenté les deux différents types de chiffrement moderne symétrique tel que AES et DES (clé secrète) et asymétrique tel que le RSA (clé publique) et enfin une comparaison (avantages et inconvénients) entre ces deux types. Dans le chapitre qui suit nous parlerons en détails sur l'un des protocoles de chiffrement symétrique (DES) avec une explication détaillée sur son principe de fonctionnement.

Chapitre 2

Principe de chiffrement par DES

2.1. Introduction

Après avoir présenté des généralités sur la cryptographie, nous avons remarqué que la cryptographie symétrique est rapide et simple à implémenter car elle nécessite des clés de petites tailles. L'algorithme DES figure parmi les algorithmes les plus utilisés pour la sécurité de l'information. Dans ce chapitre, nous allons décrire le principe de fonctionnement de DES tout en définissant ses différentes étapes ainsi que son algorithme et la cryptanalyse réalisée sur lui.

2.2. Historique de DES

DES signifie Data Encryption Standard, selon la dénomination de l'ANSI (American National Standards Institute) [6]. Le développement de ce standard a été lancé en 1972 ; il s'agissait de proposer un algorithme efficace, implantable facilement sur du matériel à bas coût, versatile, et suffisamment sûr pour être proposé comme algorithme « universel » de chiffrement pour l'industrie et l'administration américaines (sauf les communications classées secrètes, dont l'utilisation et le transfert sont beaucoup plus encadrés). Le NBS (National Bureau of Standards, devenu ensuite le NIST, National Institute of Standards and Technology) a lancé deux appels d'offre, en 1973 et 1974 ; IBM a envoyé comme réponse au second l'algorithme Lucifer [3]. Le NBS, IBM et la NSA (National Security Agency) ont ensuite collaboré pour modifier l'algorithme (afin, officiellement, d'augmenter sa sécurité) et régler quelques détails juridiques (IBM avait un brevet sur Lucifer). L'algorithme obtenu a été publié en 1975, et la norme elle-même a été rendue publique en 1977 [31].

2.3. Définition de DES

Le DES est un algorithme de chiffrement symétrique par bloc de 64 bit [3]. C'est notamment celui qui est encore en vigueur pour chiffrer les mots de passe sur les systèmes Unix. Et pour être un système de chiffrement à clé secrète « calculatoirement sûr » [32] [33]

Il a certains domaines d'utilisations très vastes telles que les cartes magnétiques et les cartes à puces [3].

2.4. **La description de DES**

Le DES a été construit pour fonctionner aussi bien de façon logicielle que matérielle. L'algorithme utilise une clé K de 56 bits, pour chiffrer des blocs de 64 bits, et fournit un cryptogramme C de 64 bits en sortie [32].

Le bloc de texte clair subit d'abord à M une permutation initiale IP donnant le message « permuté » M' est en suite divisé en deux mots de 32bits : L_0 (L pour *left*) qui représente la partie gauche de M' et R_0 (R pour *right*) pour la partie droite [32], puis on itère 16 fois une procédure où la moitié droite est recopié telle quelle à gauche et la moitié gauche et transmise à droite en subissant au passage une modification dépendant de la clé [27]. A la fin, on inverse les moitié gauches et droites (ou bien supprime le croisement de la dernière étape comme il est présent dans la figure 2.1), et on applique l'inverse de la permutation initiale pour obtenir le bloc chiffré. Le schéma général de DES est donc le suivant (on a seulement représenté quelques-unes des 16 étapes) [34]:

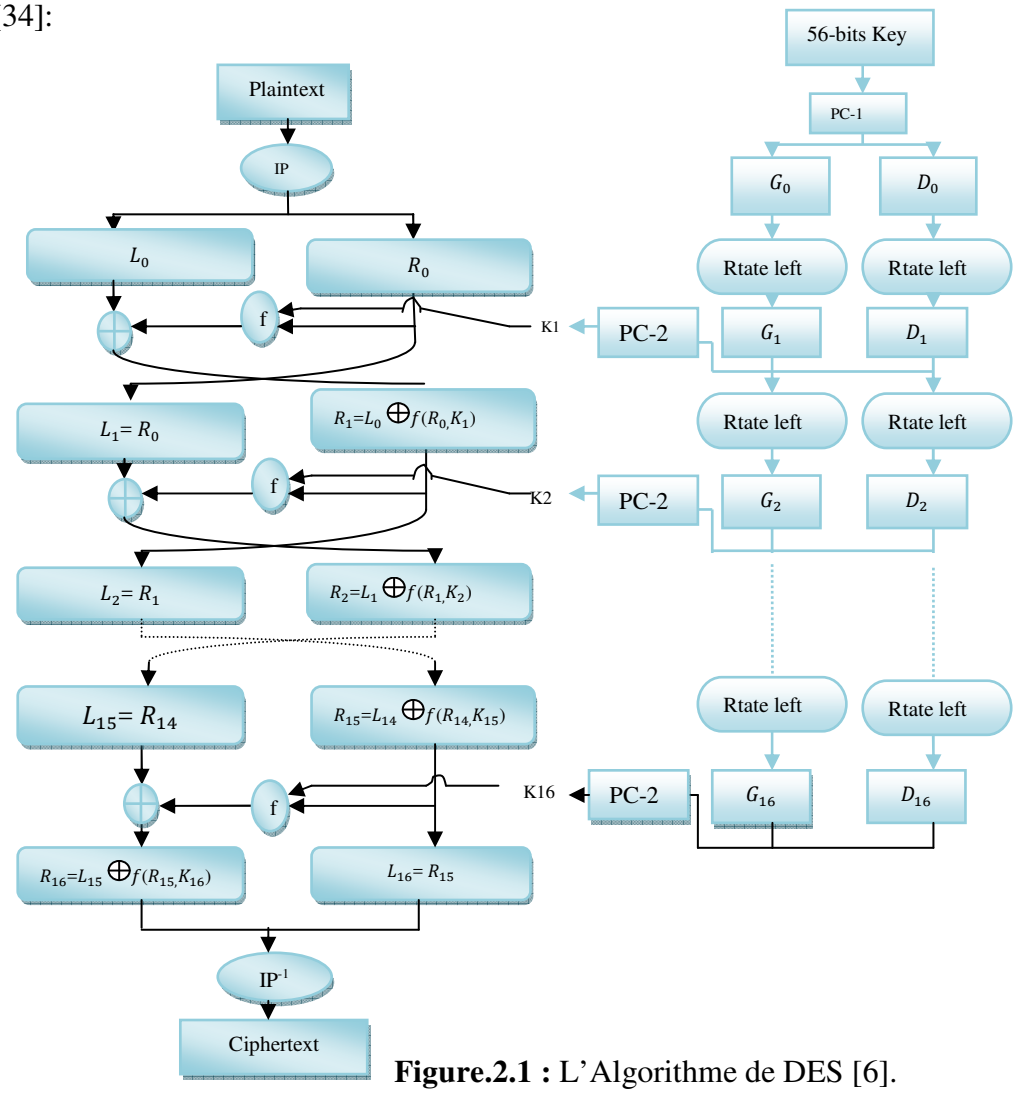


Figure.2.1 : L'Algorithme de DES [6].

La structure de Feistel garanti que le cryptage et le décryptage sont des procédés similaires. La seule différence c'est que les sous-clés sont appliquées dans l'ordre inverse pendant le décryptage. Cela simplifie l'implantation, notamment en matériel, car il n'est point question de séparer les modules de cryptage et de décryptage [34].

2.4.1. DES et son successeur

Dans un premier temps, chaque bit d'un bloc est soumis à la permutation initiale, pouvant être représentée par la matrice de permutation initiale (notée PI).

A la fin des itérations, les deux blocs G_{16} et D_{16} sont obtenus, puis soumis à la permutation initiale inverse [34].

Permutation initiale *IP*							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Permutation initiale inverse IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Tab.2.1 : La permutation initiale et son inverse [5].

La permutation initiale et son inverse sont décrits par (la figure2.1). Les tableaux se lisent de gauche à droite et de haut en bas, le n-iéme nombre est la position avant permutation du bit qui se trouve en n-iéme position après permutation [11].

2.4.2. Scindement en blocs de 32 bits

Une fois la permutation initiale réalisée, le bloc de 64 bits est scindé en deux blocs de 32 bits, notés respectivement G et D (pour gauche et droite. On note Go et Do l'état initial de ces deux blocs. Il est intéressant de remarquer que Go contient tous les bits possédant une position paire dans le message initial, tandis que Do contient les bits de position impaire [34].

G_0	58	50	42	34	26	18	10	2
	60	52	44	36	28	20	12	4
	62	54	46	38	30	22	14	6
	64	56	48	40	32	24	16	8
D0	57	49	41	33	25	17	9	1
	59	51	43	35	27	19	11	3
	61	53	45	37	29	21	13	5
	63	55	47	39	31	23	15	7

Tableau 2.2 : Blocs Go et Do de 32 bits de DES [34].

2.4.3. Processus de chiffrement

Le texte permuté M' de 64 bits, chiffre en 16 tours, suivis par l'inverse de la permutation initiale (IP-1). A chaque tour, les 32 bits du côté droit (R_i) du bloc sont transformées avec la fonction marquée (F) et une sous-clé, puis OU exclusif (XOR) avec le côté gauche (L_i) de 32 bits. Après chaque tour, les deux côtés du bloc de données sont inversés et l'algorithme continue de la même façon pour les autres tours [6] [32].

$L_i = R_{i-1}$ et $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$ [3] [6] [32].

2.4.4. fonction *f*

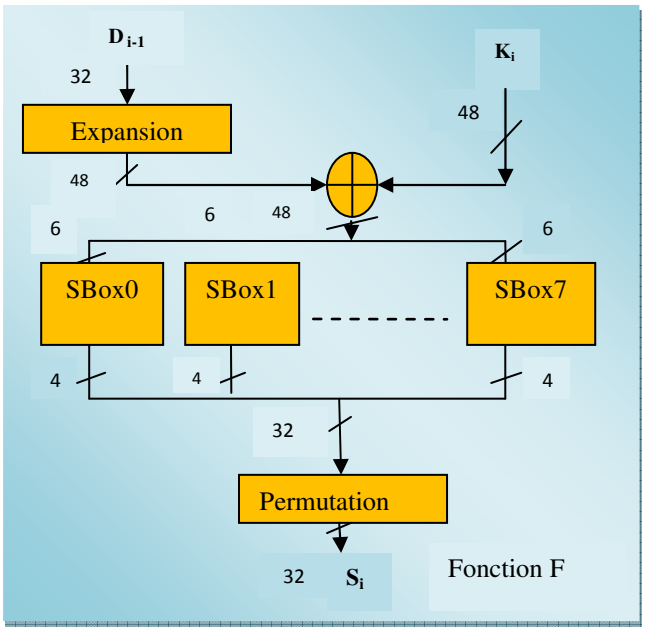


Figure.2.2 : Schéma de la fonction *f* [4].

La fonction *f* est ou cœur de la construction du DES. Tout d'abord l'argument de gauche (L_{i-1}) qui possède de 32 bits est expansé (E) en 48 bits en redoublant certains bits, cette expansion

et précisée par la figure 2.2. Ensuit, on calcule le ou exclusif de cet argument expansé avec le deuxième argument (qui n'est autre que la clé K_i) [3]. La fonction f est défini en utilisant huit permutation appelées s-boxes pour substitution en anglais [10], qui associent à 6 bits d'entrée 4 bit de sortie le résultant possède 48 = 8×6 bits et est transformé en une chaîne de 32 = 8×4 bits, chaque clé de tour K_i contient un sous-ensemble différent des 56 bits de la clé. En fin on applique la permutation d'écrit par le tableau 2.3 à ces 32 bits (pré-cryptogramme $c' = (R_{16},L_{16})$) pour obtenir la valeur de f (le cryptogramme final c) [32].

Fonction *E* d'expansion											
32	1	2	3	4	5	4	5	6	7	8	9
8	9	10	11	12	13	12	13	14	15	16	17
16	17	18	19	20	21	20	21	22	23	24	25
24	25	26	27	28	29	28	29	30	31	32	1

Permutation *P* final							
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Tab.2.3 : L'expansion du premier argument et la permutation finale de f [5].

2.4.5. Boîtes S

Le DES utilise huit boîtes de substitution (les S-Boxes) différentes ($b_1$qui furent l'objet de nombreuses controverses quant à leur contenu [20]. On les représente par des tableaux à 32 ligne et 16 colonnes.les premiers et derniers bits de l'entrée déterminent une ligne du tableau, les autres bits déterminent une colonne [4]. La sélection de la colonne se faisant sur 4 bits, il y a 16 possibilités (0 à 15). Grâce à cette information, la fonction de sélection sélectionne une valeur codée sur 4 bits [34].

S0															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	8	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S1															
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S2															
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	12	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S3															
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S4															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S5															
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	8	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S6															
4	11	2	14	15	0	8	13	3	12	9	7	5	10	3	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S7															
13	2	8	12	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Tab. 2.4 : Les boités S [5].

2.4.6. Génération des clés

La clé initiale K_0 de 64 bits subit, dans un premier temps une première permutation CP_1(une réduction de 8 bits les bits de parité de chaque octet).La chaîne de caractère résultante de 56 bits est alors divisée en 2 blocs de 28 bits (L_0 etR_0). Chacun d'eux subira alors un décalage à gauche. Les deux nouveaux blocs (L_0 etR_0) seront alors concaténé avec une nouvelle permutation CP_2 (une outre réduction de 8 bits). On obtient alors une clé partielle K_1 de 48 bits [6] [10]. On réitérera ces opération 15clés (K_2, K_3,…, K_{16}). La différence résidera dans l'incrémentation des paramètres des deux permutations [11].

Permutation CP_1													
57	49	41	33	25	17	9	1	58	50	42	34	26	18
10	2	59	51	43	35	27	19	11	3	60	52	44	36
63	55	47	39	31	23	15	7	62	54	46	38	30	22
14	6	61	53	45	37	29	21	13	5	28	20	12	4

Permutation CP_2											
10	21	6	15	28	3	5	1	24	11	17	14
2	13	20	27	7	16	8	26	4	12	19	23
48	33	45	51	40	30	55	47	37	31	52	41
32	29	36	50	42	46	53	34	56	39	49	44

Tab.2.5 : Les permutations CP_1et CP_2 [10].

Ensuit, pour chaque itération, les bits de la clé principale subiront des décalages circulaires à gauche d'un ou deux bits selon chaque itération de la manière suivante :

Itération	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Nombre de décalages	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Tab.2.6 : Décalage relatif aux sous-clés [5]

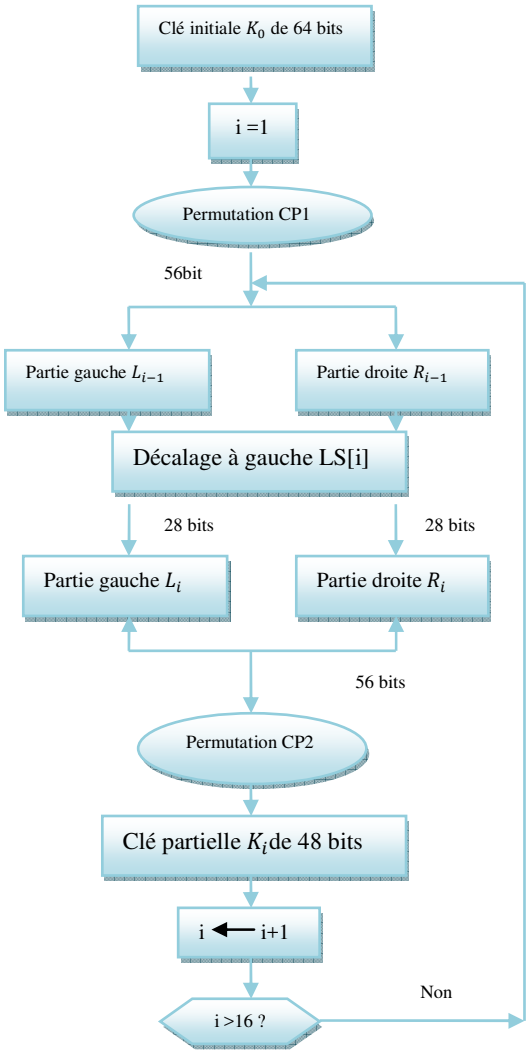


Figure.2.3 : La génération de la clé [10].

2.4.7. Déchiffrement

Le déchiffrement est effectué par le même algorithme, mais cette fois-ci en inversant l'ordre d'application des clés, c'est-à-dire en utilisant K_{16} à la première itération, K_{15}à la seconde…deK_1 à la seizième. Cela est dû ou fait que la permutation final est l'inverse de la permutation initiale et

$$R_{i-1} = L_i \;\; ; \quad L_{i-1} = R_i \oplus f(L_i \, , K_i)$$

Observons que si l'ordre des clés de tour est inversé, l'algorithme, quant à lui, reste identique [10] [11].

2.5. Attaques réalisées sur DES

Plusieurs attaques existent pour casser le DES :

– La recherche exhaustive par force brute : on teste toutes les clés possibles, l'une après l'autre, afin de déchiffrer un bloc de données. On a besoin, en moyenne, de 2 55 essais [35].

– Une machine dédiée au calcul des clés : Deep Crack (Des Cracker, élaboré par l'Electronic Frontier Foundation) a coûté moins de 250'000 dollars. Elle disposait de 1850 processeurs en parallèle pour un temps de recherche de 56 heures (1998) [35].

– Le calcul distribué : un regroupement d'ordinateurs par Internet qui partagent leur puissance de calcul. En 1998, Distributed.net a pu décrypter un message en 39 jours. Le 19 janvier 1999, EFF et Distributed.net ont cassé en travaillant conjointement une clé en 22 heures et 15 minutes [35].

– Depuis 2006, un autre ordinateur dédié, nommé COPACABANA2, permet de casser une clé DES en 9 jours. En 2008, des améliorations logicielles ont même permis de diminuer le temps de recherche à 6,4 jours. Son avantage est son coût : 10.000$. On peut donc facilement en acquérir plusieurs et donc diminuer le temps de recherche en conséquence. Par exemple, pour 4 machines acquises (=40,000$), il faudra compter 1,6 jour de recherche [35].

– Cryptanalyse différentielle : La cryptanalyse différentielle est conçue par Biham et Shamir en 1993. C'est une attaque à message clairs choisis [10]. Il s'agit de la possibilité de produire au moyen de textes clairs choisis les textes chiffrés correspondants avec une clé inconnue. On analyse les différences résultantes sur les textes chiffrés et on donne des probabilités aux clés testées. En affinant, on trouve la clé la plus probable. La meilleure attaque différentielle connue demande 2 47 textes clairs choisis. Donc elle peut casser l'algorithme DES après 247 couples de texte choisi [17] [35].

– Cryptanalyse linéaire : Pour casser l'algorithme DES, Mitsuri Matsui a conçu en 1993 cette technique de cryptanalyse linéaire. Cet algorithme est une attaque à texte clair connu [10].Son principe tire profit des probabilités élevées des occurrences des expressions linéaires déduites du texte clair et du texte chiffré. Ces expressions linéaires sont construites à partir d'approximation linéaire de l'algorithme à crypter. Pour crypter l'algorithme DES, une bonne implémentation aura

besoin à peu près de 239 couples chiffrés par la même clé. La faille du DES réside à une certaine caractéristique linéaire de ses S-Box (table de substitution) qui doivent être normalement non linéaires. Chaque algorithme de cryptage doit résister à cette attaque [17] [35].

– Il existe d'autres attaques spécifiques au DES (Davie's attack, ou des attaques sur des versions simplifiées du DES) mais qui n'entrent pas dans le cadre de cette mémoire [35].

2.6. Double DES

Suite aux failles du DES, quelques modifications ont été apportées, mais pas toujours avec succès. Ce fut notamment le cas avec le 2DES.

Il faut tout d'abord choisir deux clefs K_1 et K_2. Le principe du 2DES est de chiffrer deux fois le message :

$$\mathrm{E}\,(K_2, \mathrm{E}\,(K_1, m))$$

Il a été prouvé que 2DES était équivalent à un DES avec une clé de 57 bits. Il faut donc seulement deux fois plus de travail pour le briser ($2^{57} = 2 * 2^{56}$).

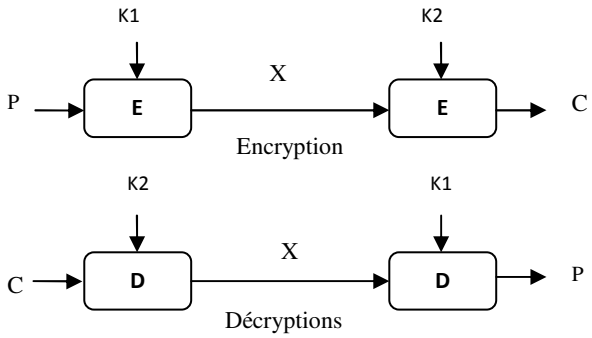


Fig.2.4 : Double DES [35] [36].

Le 2DES est sensible à l'attaque Meet-in-the-middle, autrement dit, un intrus peut s'introduire dans l'échange et retrouver la clé utilisée [35].

2.7. Triple DES (TDES ou 3DES)

Avec la technologie actuelle (des CPU très rapides et moins chers), il est hautement possible de cracker DES. La solution a été au premier temps d'adopter le triple DES (TDES ou 3DES) : 3 applications de DES à la suite avec 2 clés différentes (112 bits) (Figure 2.5) [1].

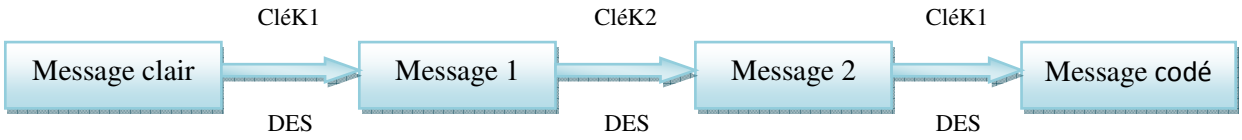


Figure.2.5 : Algorithme de TDES [1].

Le TDES est une variante du DES qui consiste à appliquer l'algorithme trois fois à chaque bloc : chiffrement, déchiffrement puis de nouveau chiffrement. Les clefs utilisées pour chaque application du DES peuvent être toutes les trois distinctes, ou bien on peut utiliser seulement deux clefs distinctes : une pour le chiffrement et une pour le déchiffrement. Dans tous les cas, la longueur efficace de la clef du Triple DES ne dépasse pas 112 bits [21] [36]. Le TDES permet d'augmenter significativement la sécurité du DES, toutefois il a l'inconvénient majeur de demander également plus de ressources pour le chiffrement et le déchiffrement [11].

2.8. Conclusion

Dans ce chapitre, nous avons présenté la sécurité du standard de chiffrement symétrique DES, avec une explication détaillée sur son principe de fonctionnement ; nous allons voir, dans le chapitre suivant, l'approche proposée pour améliorer la génération des sous-clés de l'algorithme cryptographie DES, avec l'implémentation de deux versions classiques et améliorées de DES sur MATLAB pour valider et vérifier les résultats de notre contribution.

Chapitre 3

Proposition d'un DES amélioré

3.1. Introduction

Dans le chapitre précédent nous avons étudié le principe de fonctionnement du DES. Nous avons vu que la sécurité de cette algorithme repose sur la génération de sous clés. Ces dernières sont parfois redondantes à cause de l'utilisation de certaines clés faibles.

Dans ce chapitre, nous allons proposés une approche pour avoir des sous clés non redondantes ainsi que pour augmenter la complexité de génération de sous clés. Par la suite, nous allons implémenter de deux versions du DES avec/ et sans l'approche proposée.

3.2. Problème de génération de sous-clés [36] [37]

DES utilise 16 sous clés de 48 bits générées à partir d'une clé principale de 56 bits (64 bits si l'on considère également les bits de parité).

• Clés faibles: qui génèrent des sous clés redondantes.

• Résultat: réduire la complexité de chiffrement.

• DES a 4 clés faibles :

- 01010101 01010101
- FEFEFEFE FEFEFEFE
- E0E0E0E0 F1F1F1F1
- 1F1F1F1F 0E0E0E0E

• DES a 6 clés demi faibles

-01 FE 01 FE 01 FE 01 FE
-1F E0 1F E0 1F E0 1F E0
-01 E0 01 E1 01 F1 01 F1
-1F FE 1F FE 0E FE 0E FE
-01 1F 01 1F 01 0E 01 0E
-E0 FE E0 FE F1 FE F1FE

3.3. Techniques possibles pour améliorer l'algorithme DES [36]

-Chiffrement multiple avec DES.

-Étendant le DES aux chemins de données de 128 bits et aux clés de 112 bits.

-Étendre le calcul de l'expansion des clés.

3.4. L'approche proposée

Nous avons proposé une approche pour avoir des sous-clés non redondantes, cette approche expose l'utilisation de ou exclusif entre les 64 bits de la clé initiale KK comme il est bien expliqué dans la figure 3.1

Tout d'abord nous avons appliqué une rotation par bit qui implique que : RK = rotation de KK par 1bit.

En suit l'application de (ou exclusif) qui implique : k= KK ⊕ RK.

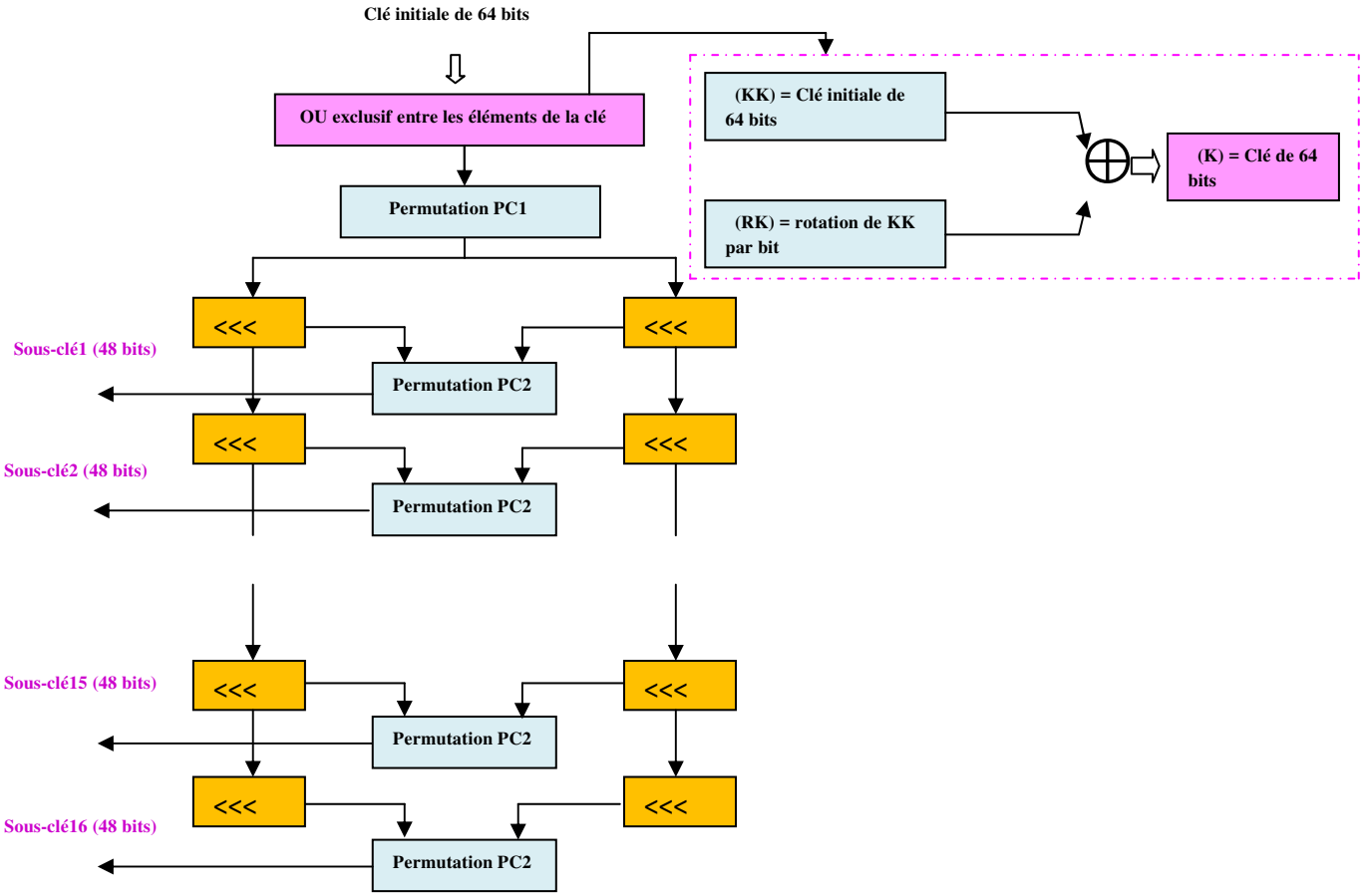


Figure.3.1: Diagramme de la génération des sous clés de l'algorithme DES amélioré.

Programmation:

Nous allons faire une programmation en deux langages différents pour les raisons. Nous utilisons MATLAB pour vérifier la validité de l'algorithme (présentée dans ce chapitre) et VHDL pour les matérielles de réalisation du protocole (présenté dans le chapitre 4).

3.5. Résultats d'implémentation sur MATLAB

Après l'utilisation de (ou exclusif) entre les 64 bits de la clé initiale KK, les 4 clés faibles de DES ne donnent pas des sous-clés redondantes.

Pour avoir les résultats d'implémentation sur MATLAB, nous avons pris la clé K $(E0E0E0E0F1F1F1F1)_{16}$ comme un exemple.

- **Résultat d'implémentation de la génération des sous clés de DES classique sur MATLAB :**

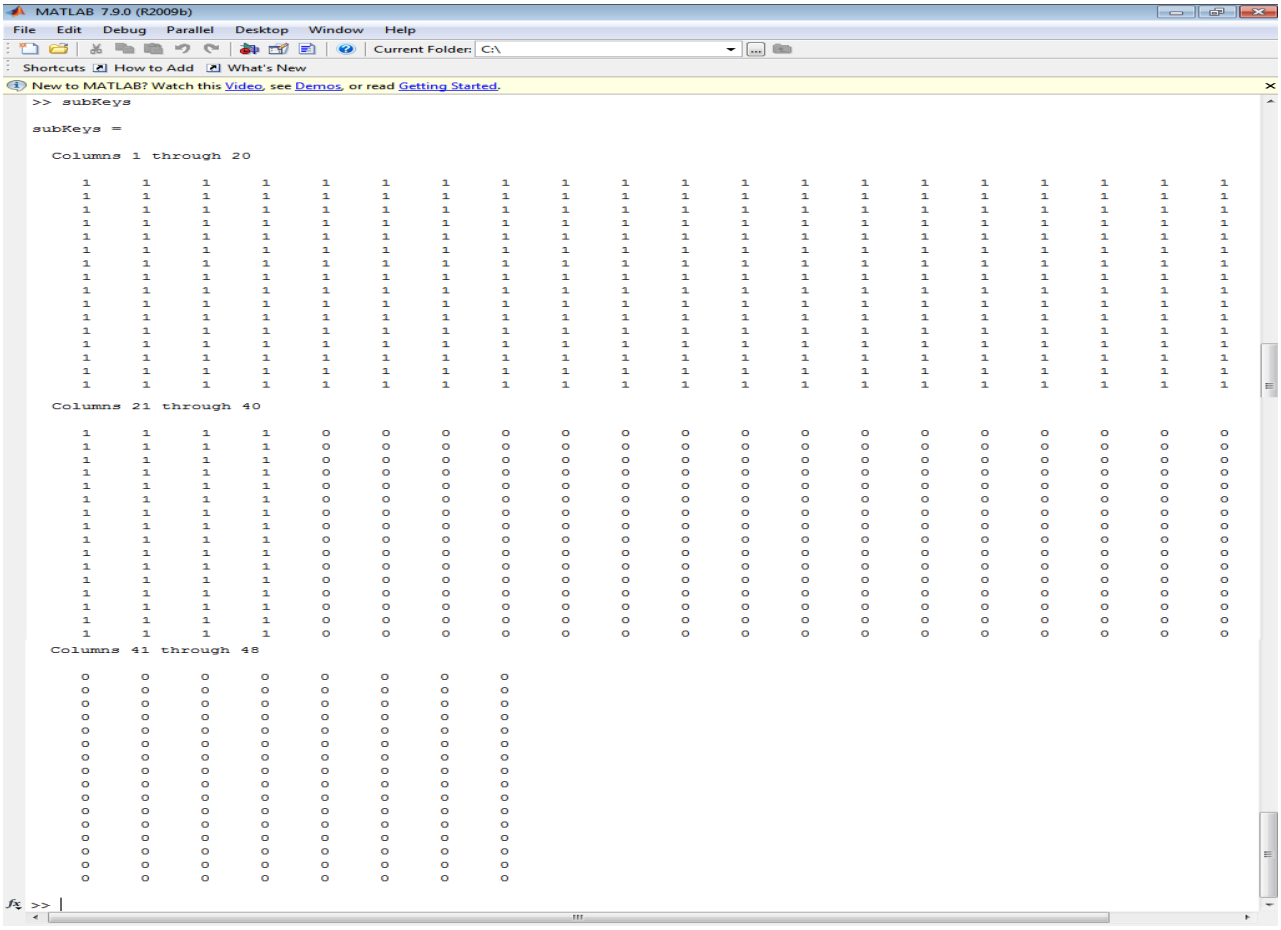


Figure.3.2 : Résultat d'implémentation de DES classique sur MATLAB.

Nous avons remarqué que le résultat de l'implémentation de la clé faible k = $(E0E0E0E0F1F1F1F1)_{16}$ tout les 16 sous clé résultant sont la même (il y a une redondantes).

(Le code MATLAB de la génération des sous clés de DES classique se trouve dans l'annexe A).

- **Résultat d'implémentation de DES amélioré sur MATLAB :**

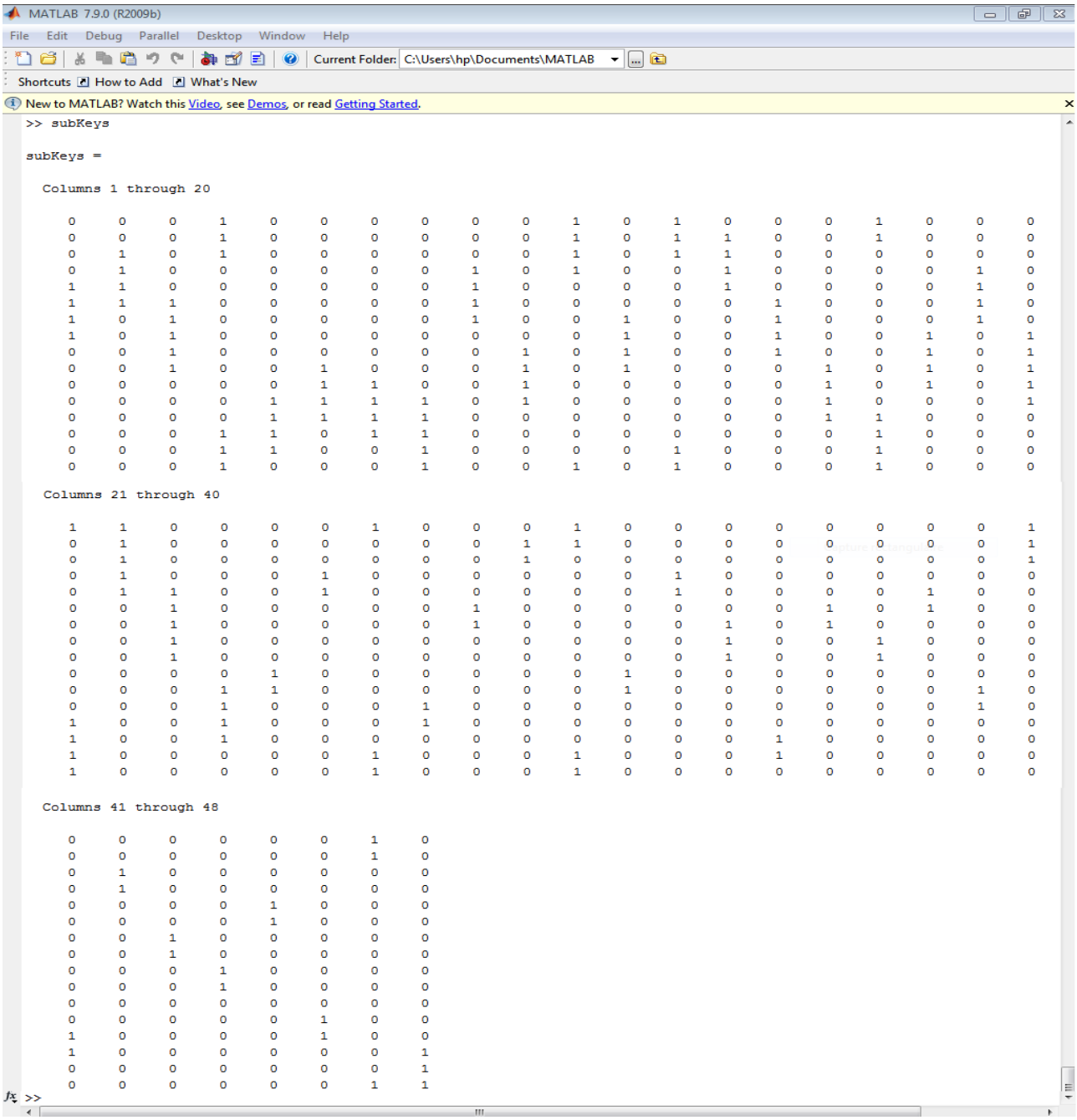


Figure.3.3 : Résultat d'implémentation de la génération dessous clés de DES amélioré sur MATLAB.

La figure 3.3 représente le résultat de la simulation de la clé faible k = $(E0E0E0E0F1F1F1F1)_{16}$ dans le programme amélioré de la génération des sous clés de DES sur MATLAB, nous avons remarqué que tous les 16 sous clés résultant sont déférents (pas de redondance).

(Le code MATLAB de la génération des sous clés de DES amélioré se trouve dans l'annexe B).

3.6. Comparaison des résultats obtenus

- Résultat d'implémentation de la génération des sous clés de DES classique et amélioré des 4 sous clés faibles :

clé	Classique	Amélioré
E0E0E0E0F1F1F1F1	**1)** 000000FFFFFF	10288C 220102
	2) 000000FFFFFF	102C8 4060102
	3) 000000FFFFFF	502C04 040140
	4) 000000FFFFFF	814848 808040
	5) 000000FFFFFF	C08426 408408
	6) 000000FFFFFF	E08222 081408
	7) 000000FFFFFF	504911 042810
	8) 000000FFFFFF	A01252 004820
	9) 000000FFFFFF	205252 004810
	10) 000000FFFFFF	245150 810010
	11) 000000FFFFFF	064151 810200
	12) 000000FFFFFF	0F4111 100204
	13) 000000FFFFFF	0F4111 100204
	14) 000000FFFFFF	1B0089 002081
	15) 000000FFFFFF	190888 222001
	16) 000000FFFFFF	112888 220003
FEFEFEFEFEFEFEFE	1) FFFFFF FFFFFF	000000 222183
	2) FFFFFF FFFFFF	000000262183
	3) FFFFFF FFFFFF	000000 1300A3
	4) FFFFFF FFFFFF	000000 468142
	5) FFFFFF FFFFFF	000000 448548
	6) FFFFFF FFFFFF	000000 489448
	7) FFFFFF FFFFFF	000001 235428
	8) FFFFFF FFFFFF	000000 085C28
	9) FFFFFF FFFFFF	000000 085C30
	10) FFFFFF FFFFFF	000000 894830
	11) FFFFFF FFFFFF	000000 814A10
	12) FFFFFF FFFFFF	000000 910214
	13) FFFFFF FFFFFF	000000 244084
	14) FFFFFF FFFFFF	000000 102285
	15) FFFFFF FFFFFF	000000 322085
	16) FFFFFF FFFFFF	000000 222087
01010101 01010101	**1)** 000000 000000	000000 222183
	2) 000000 000000	000000 1310C3
	3) 000000 000000	000000 260143
	4) 000000 000000	000000 468142
	5) 000000 000000	000000 448548
	6) 000000 000000	000000 489448
	7) 000000 000000	000000 48D428
	8) 000000 000000	000000 085C28

	9) 000000 000000	000000 894830
	10) 000000 000000	000000 894830
	11) 000000 000000	000000 814A10
	12) 000000 000000	000000 910214
	13) 000000 000000	000000 910284
	14) 000000 000000	000000 102285
	15) 000000 000000	000000 322085
	16) 000000 000000	000000 222087
1F1F1F1F 0E0E0E0E	1) 000000 111111	10288C 220102
	2) 000000 111111	102C84 060102
	3) 000000 111111	281404040140
	4) 000000 111111	40A424408040
	5) 000000 111111	C08426408408
	6) 000000 111111	E08222081408
	7) 000000 111111	A09222085020
	8) 000000 111111	500929002820
	9) 000000 111111	205252004810
	10) 000000 111111	245150810010
	11) 000000 111111	64151810200
	12) 000000 111111	F4111100204
	13) 000000 111111	F0189100084
	14) 000000 111111	1B0089002081
	15) 000000 111111	190888222001
	16) 000000 111111	112888220003
00000000 00000000	Toute les sous-clés = 000000 000000	Toute les sous-clés=000000 000000
FFFFFFFF FFFFFFFF	Toute les sous-clés =111111 111111	Toute les sous-clés=000000 000000

Tab.3.1 : Résultats d'implémentation des clés faibles de la génération des sous clés de DES classique et amélioré.

D'après la simulation des clés faibles dans les 2 programmes classique et amélioré .nous avons remarqué que : notre approche proposée réduite la faiblesse dans les 2 clés faibles $(FEFEFEFEFEFEFEFE)_{16}$ et $(01010101\ 01010101)_{16}$. Et pour les autre clé nous avons obtenue des très bons résultats.

- **Résultat d'implémentation de DES classique et amélioré des sous clés demi-faibles :**

clé	Classique	Amélioré
01 FE 01 FE 01 FE 01 FE	**1)** 9153E5 431BBD	**1)** 000000 222183
	2) 6EAC1A BCE642	**2)** 000000 262183
	3) 6EAC1A BCE642	**3)** 000000 260143
	4) 6EAC1A BCE642	**4)** 000000 468142
	5) 6EAC1A BCE642	**5)** 000000 448548
	6) 6EAC1A BCE642	**6)** 000000 489448
	7) 6EAC1A BCE642	**7)** 000000 48D428
	8) 6EAC1A BCE642	**8)** 000000 085C28

	9) 6EAC1A BCE642	**9)** 000000 085C30
	10) 9153E5 431BBD	**10)** 000000 894830
	11) 9153E5 431BBD	**11)** 000000 814A10
	12) 9153E5 431BBD	**12)** 000000 910214
	13) 9153E5 431BBD	**13)** 000000 910284
	14) 9153E5 431BBD	**14)** 000000 102285
	15) 9153E5 431BBD	**15)** 000000 322085
	16) 6EAC1ABCE642	**16)** 000000 222087
1F E0 1F E0 0E F1 0E F1	1) 9153E5 BCE642	**1)** 402C24 000000
	2) 6EAC1A 4319BD	**2)** 40A424 000000
	3) 6EAC1 A4319BD	**3)** C08426 000000
	4) 6EAC1 A4319BD	**4)** E08222 000000
	5) 6EAC1 A4319BD	**5)** A09222 000000
	6) 6EAC1 A4319BD	**6)** A01252 000000
	7) 6EAC1 A4319BD	**7)** 245250 000000
	8) 6EAC1 A4319BD	**8)** 065150 000000
	9) 9153E 5 BCE642	**9)** 064151 000000
	10) 9153E5 BCE642	**10)** 0F4111 000000
	11) 9153E5 BCE642	**11)** 0F0189 000000
	12) 9153E5 BCE642	**12)** 1B0089 000000
	13) 9153E5 BCE642	**13)** 190888 000000
	14) 9153E5 BCE642	**14)** 10288C 000000
	15) 9153E5 BCE642	**15)** 102C04 000000
	16) 6EAC1A 4319BD	**16)** 40A424 000000
01 E0 01 E1 01 F1 01 F1	**1)** 9153E5 000000	**1)** 100084 222103
	2) 6EAC1A 000000	**2)** 002C00 260183
	3) 6EAC1A 000000	**3)** 402C00 060143
	4) 6EAC1A 000000	**4)** 40A400 468140
	5) 6EAC1A 000000	**5)** 408402 408548
	6) 6EAC1A 000000	**6)** 608002 489408
	7) 6EAC1A 000000	**7)** 208002 485428
	8) 6EAC1A 000000	**8)** 200012 085828
	9) 9153E5 000000	**9)** 005240 084C30
	10) 9153E5 000000	**10)** 005140 894810
	11) 9153E5 000000	**11)** 004141 814210
	12) 9153E5 000000	**12)** 014101 910204
	13) 9153E5 000000	**13)** 010181900284
	14) 9153E5 000000	**14)** 110081102285
	15) 9153E5 000000	**15)** 110080 322081
	16) 6EAC1A 000000	**16)** 002808 222087
1F FE 1F FE 0E FE 0E FE	1) 9153E5 FFFFFF	**1)** 002808 220183
	2) 6EAC1A FFFFFF	**2)** 100084 262102
	3) 6EAC1A FFFFFF	**3)** 100004 240142
	4) 6EAC1A FFFFFF	**4)** 000024 448042
	5) 6EAC1A FFFFFF	**5)** 800024 448448
	6) 6EAC1A FFFFFF	**6)** 800220 089448
	7) 6EAC1A FFFFFF	**7)** 801220 08D420

	8) 6EAC1A FFFFFF	**8)** 801240084C20
	9) 9153E5 FFFFFF	**9)** 200012 005830
	10) 9153E5 FFFFFF	**10)** 240010 810830
	11) 9153E5 FFFFFF	**11)** 060010 810A10
	12) 9153E5 FFFFFF	**12)** 0E0010 110214
	13) 9153E5 FFFFFF	**13)** 0E0008 110084
	14) 9153E5 FFFFFF	**14)** 0A0008 002085
	15) 9153E5 FFFFFF	**15)** 080808 222085
	16) 6EAC1A FFFFFF	**16)** 110080 222003
01 1F 01 1F 01 0E 01 0E	**1)** 000000 BCE642	**1)** 100084 222102
	2) 000000 4319BD	**2)** 002C00 060183
	3) 000000 4319BD	**3)** 402C00 060141
	4) 000000 4319BD	**4)** 40A400 428140
	5) 000000 4319BD	**5)** 408402 408508
	6) 000000 4319BD	**6)** 608002481408
	7) 000000 4319BD	**7)** 208002 485028
	8) 000000 4319BD	**8)** 200012 005828
	9) 000000 BCE642	**9)** 005240 084C10
	10) 000000 BCE642	**10)** 005140 894010
	11) 000000 BCE642	**11)** 001050 604200
	12) 000000 BCE642	**12)** 014101 900204
	13) 000000 BCE642	**13)** 010181 900284
	14) 000000 BCE642	**14)** 110081 102281
	15) 000000 BCE642	**15)** 110080 322001
	16) 000000 4319BD	**16)** 002808220087
E0 FE E0 FE F1 FE F 1FE	1) FFFFFF 4319BD	**1)** 002808 220183
	2) FFFFFF BCE642	**2)** 100084 262102
	3) FFFFFF BCE642	**3)** 100004 240142
	4) FFFFFF BCE642	**4)** 000024 448042
	5) FFFFFF BCE642	**5)** 800024 448448
	6) FFFFFF BCE642	**6)** 800220 089448
	7) FFFFFF BCE642	**7)** 801220 08D420
	8) FFFFFF BCE642	**8)** 801240 084C20
	9) FFFFFF 4319BD	**9)** 200012 005830
	10) FFFFFF 4319BD	**10)** 240010 810830
	11) FFFFFF 4319BD	**11)** 060010 810A10
	12) FFFFFF 4319BD	**12)** 0E0010 110214
	13) FFFFFF 4319BD	**13)** 0E0008110084
	14) FFFFFF 4319BD	**14)** 0A0008 002085
	15) FFFFFF 4319BD	**15)** 080808 222085
	16) FFFFFF BCE642	**16)** 110080 222003

Tab.3.2 : Résultats d'implémentation des clés demi-faibles du DES classique et amélioré.

D'après la simulation des clés demi-faibles dans les deux programmes classique et amélioré nous avons remarqué que notre aproche proposé réduit la faiblesse dans certaine clés comme la clé

k = (01FE01FE 01FE01FE) $_{16}$ et la clé (1FE01FE0 0EF10EF1)$_{16}$. Et pour les restes clés nous avons obtenue des trés bonne résultats car notre approche proposée éliminé définitivement le redondantes.

Remarques

- Pour l'implémentation de DES classique (les 4 clés faibles), toutes les sous-clés produites sont identique et redondantes 16 fois. Et aussi les 6 paires clés demi-faibles produites des sous clés redondantes.
- Dans l'implémentation de DES amélioré nous remarquons qu'il n'existe aucune sous-clés redondantes pendant les 16 rondes.
- La clé faible (FFFFFFFF FFFFFFFF) $_{16}$ doit être évitée.
- L'utilisation de la clé (00000000 00000000)$_{16}$ permettre la vérification et la validation du programme.

3.7. Conclusion

Après l'étape de l'amélioration au niveau de la génération des sous clés, le DES amélioré donner des sous clés non redondantes.

L'approche proposée, l'utilisation du "ou exclusif XOR", est une méthode utiliser pour augmenter la complexité de génération de sous clés et l'élimination de la touche faible de DES (réduit dans certain cas et éfficace pour les autres cas).

Nous allons entamer dans le chapitre suivant, l'implémentation des deux versions : classique et amélioré du DES sur FPGA.

.

Chapitre 4

Implémentation du DES amélioré sur FPGA

4.1. Introduction

L'implémentation de deux versions classique et amélioré de l'algorithme DES sur un circuit FPGA (*Field Programmable Gate Array*) consiste en la programmation des circuits à l'aide du langage de description de haut niveau VHDL (*Very height speed integrated circuit Hardware Description Language*). Nous allons donc décrire les circuits sous la forme d'algorithme VHDL, et ce, après avoir choisi les contraintes d'implémentation. La seconde étape consiste à effectuer une simulation fonctionnelle des circuits à l'aide du simulateur MODELSIM. La dernière étape consiste à comparer les résultats obtenus.

4.2. Implémentation VHDL de la génération des sous clés de l'algorithme DES

Pour l'implémentation VHDL de la génération de sous clés de l'algorithme DES nous avons besoin de :

1- Les boites de permutation : réorganisent les bits d'une chaîne de bits [38].

2- les registres : Les registres (tampons de données) peuvent être mis en œuvre soit dans la logique combinatoire, soit en utilisant des éléments de mémoire RAM. La plupart des FPGA modernes ont des éléments RAM / ROM intégrés qui sont plus efficaces que la logique combinatoire à ces fins [38].

3- Registres de décalage : sur laquelle les données doivent être déplacées à gauche ou bien a droite [38].

4- Multiplexeurs : Les multiplexeurs peuvent être facilement implémentés à l'aide d'une logique combinatoire qui permet d'aiguiller une entrée parmi 2^n vers une sortie en fonction d'entrée de sélection [38].

Tout d'abord, la clé initiale de 64 bits passe par la boite de permutation (la permutation PC1) et le multiplexeur pour sélectionner 56 bits. Les 56 bits sont alors divisés en deux parties de 28 bits chacune. A partir de ce moment chaque moitié est traitée séparément. Dans des itérations (rondes) successives les deux parties de la clé sont rotationnels d'un ou deux bits (spécifique à chaque itération) [34]. Dans cette partie nous avons besoin de registre de décalage pour les données déplacées à gauche. Après cette partie nous avons besoin de registre (de 56 bits) pour stocker les résultats de chaque boucle et les transmettre au multiplexeur [38]. La sortie du registre de données

passe par la permutation PC2 et alors une sous clés de 48 bits est choisi par le PC2, 24–bits de la moitie gauche et 24 bit de la moitie droite [34].

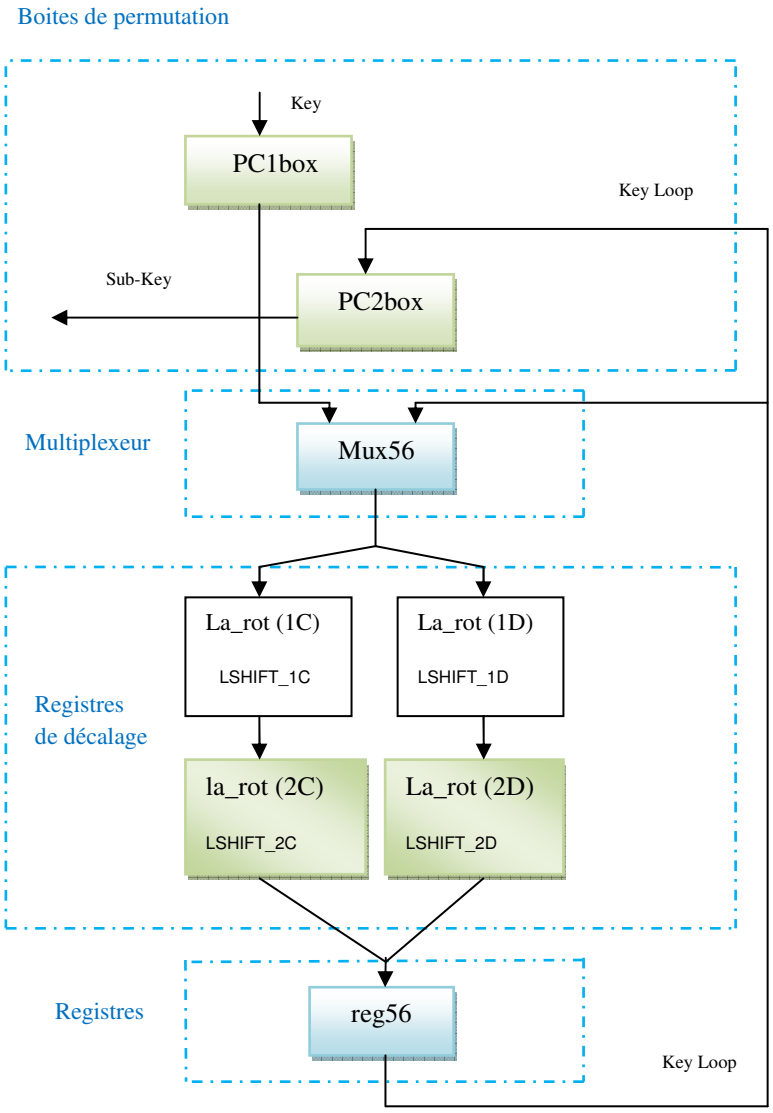


Figure 4.1 : Diagramme de la génération des sous clés de l'algorithme DES classique [38].

4.3. Synthèse et Résultats d'implémentation de la génération de sous clés de DES classique

- La synthèse est effectuée en utilisant ISE9.2 et le périphérique cible est Xilinx FPGA. Nous avons choisi deux types de FPGA de la famille la plus fréquente, Spartan3 (XC3S4000L) et Virtex (XCV1000).

- Le but principal de la synthèse est de déterminer l'efficacité de la mise en œuvre sur la carte FPGA en vérifiant la quantité de ressources logiques nécessaires pour implémenter cette conception sur la carte FPGA.

- La simulation a été effectuée en utilisant le simulateur Modèlsim.

- Le but principal de la simulation est d'assurer que les résultants obtenus par MTLAB et VHDL sont identiques.

- Les résultants de la simulation de la génération de sous clés de l'algorithme DES classique à partir du simulateur Modèlsim sont présentées dans les figures (4.2, 4.3, 4.4, 4.5, 4.6 4.7, 4.8, 4.9, 4.10, 4.11).

- Nous avons choisi les 4 sous-clés faibles et les 6 demi-clés faibles pour la simulation.

- **Résultats de La simulation des 4 sous-clés faibles classique**

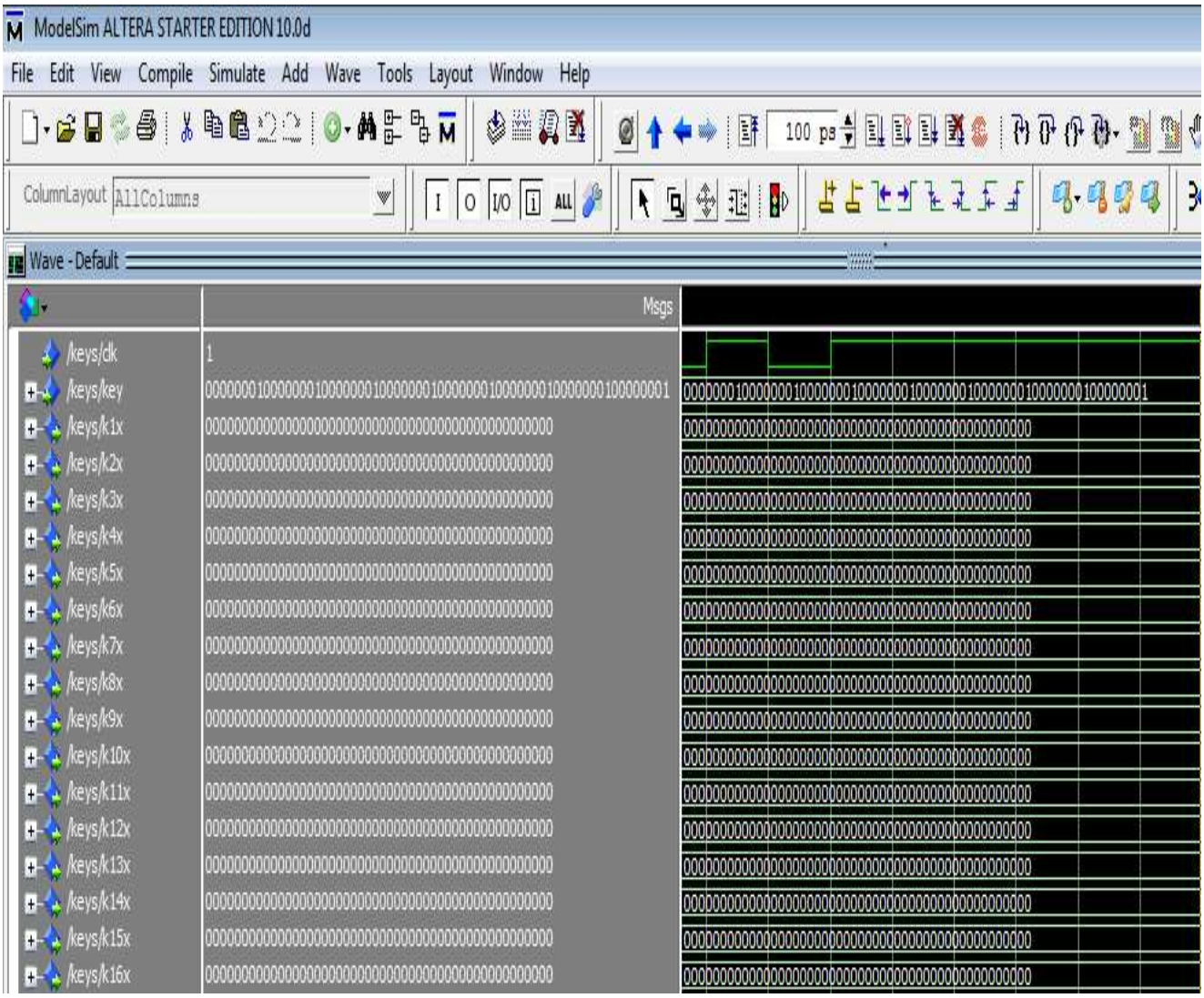


Figure.4.2: Résultat de simulation avec K= $(01010101\ 01010101)_{16.}$

D'après la simulation de la clé faible $(01010101\ 01010101)_{16}$ toutes les16 sous clés résultants sont identiques (tout les sous clés = $(000000\ 000000)_{16}$).

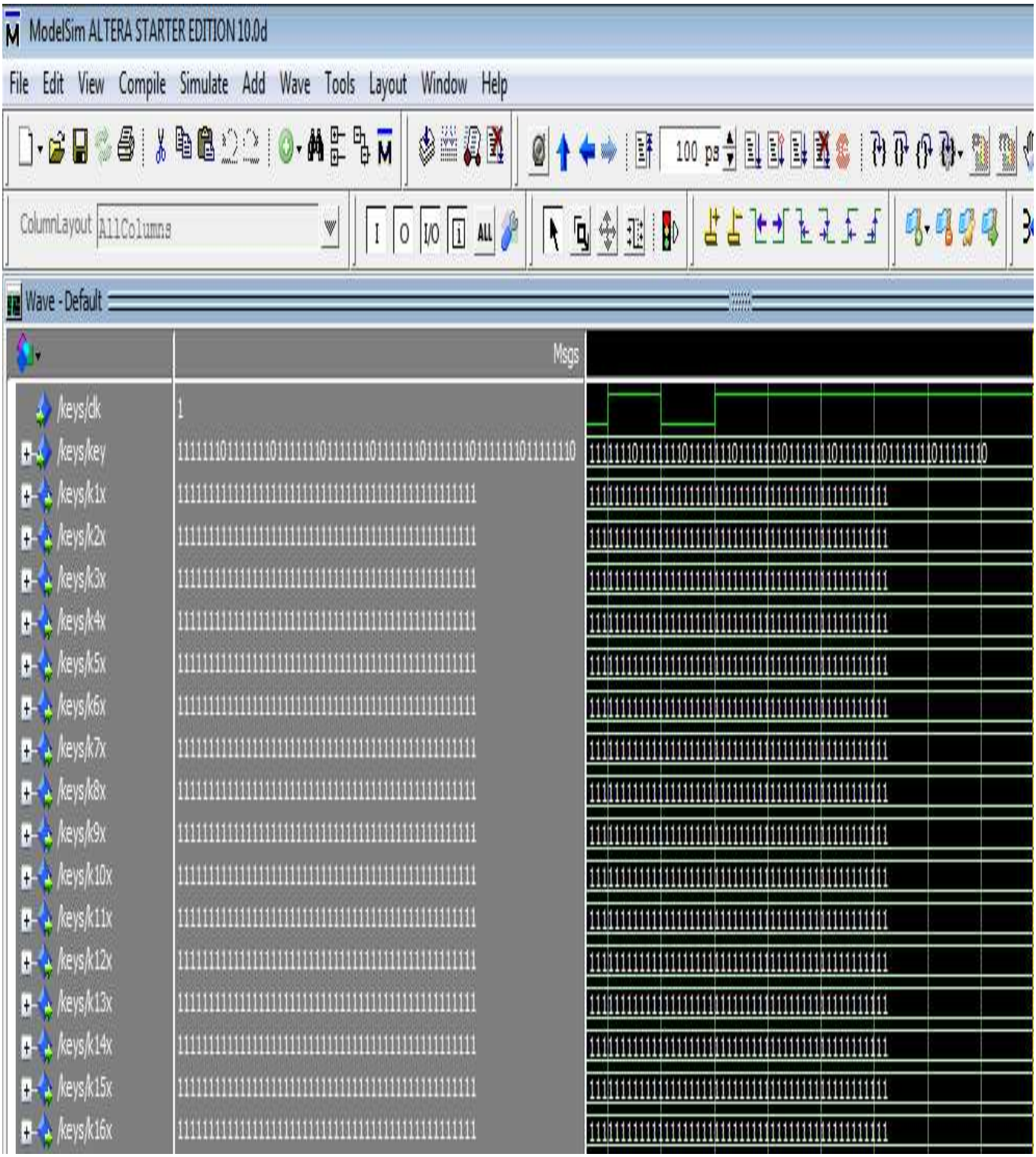


Figure.4.3: Résultat de simulation avec K= (FEFEFEFE FEFEFEFE) $_{16}$.

D'après la simulation de la clé faible (FEFEFEFE FEFEFEFE$_{)\ 16}$ toutes les16 sous clés résultants sont identiques (tout les sous clés = (FFFFFF FFFFFF)$_{16}$).

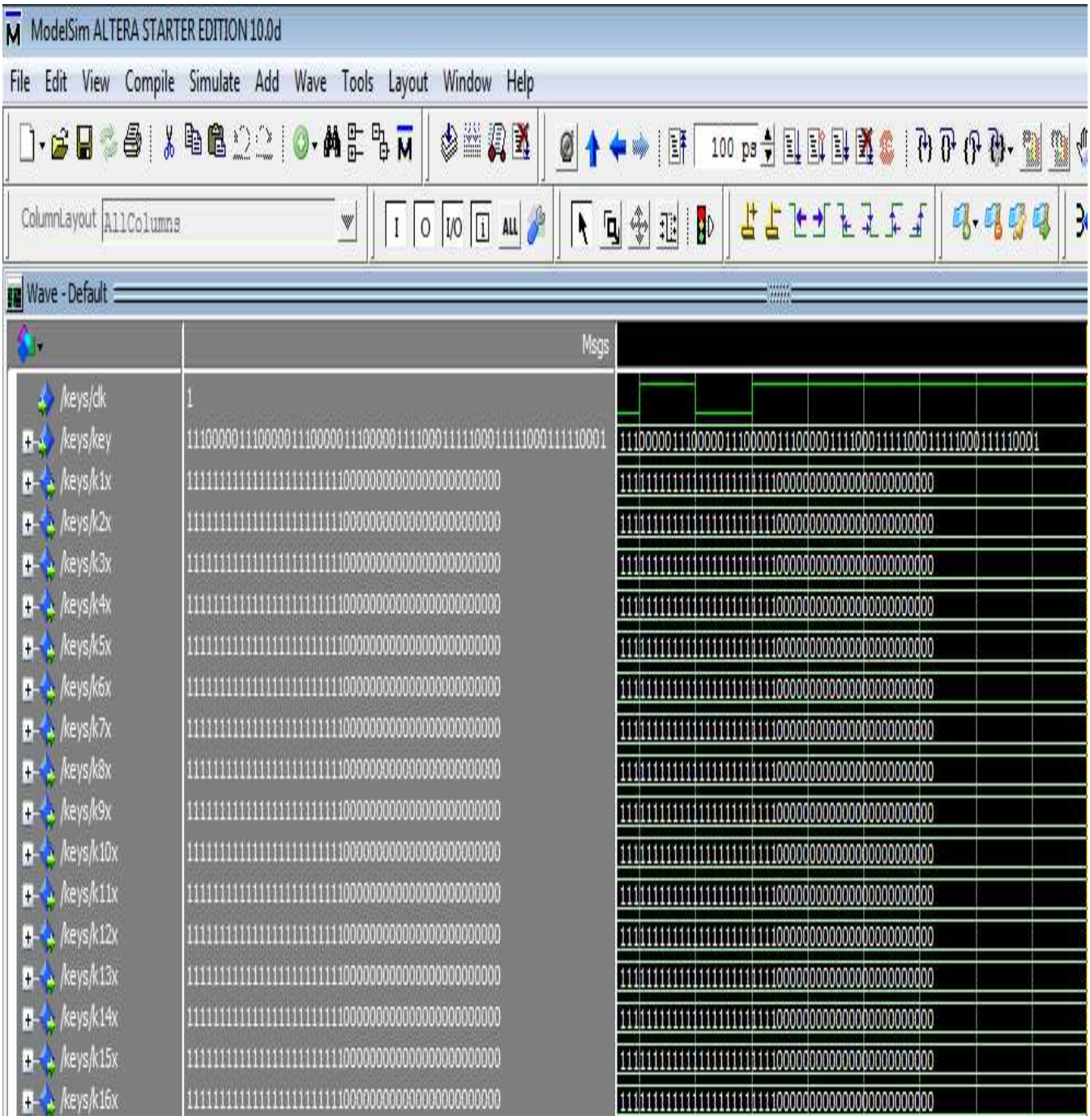


Figure.4.4: Résultat de simulation avec K= $(E0E0E0E0\ F1F1F1F1)_{16}$.

D'après la simulation de la clé faible $(E0E0E0E0\ F1F1F1F1)_{16}$ toutes les16 sous clés résultants sont identiques (tout les sous clés = $(FFFFFF\ 000000)_{16}$).

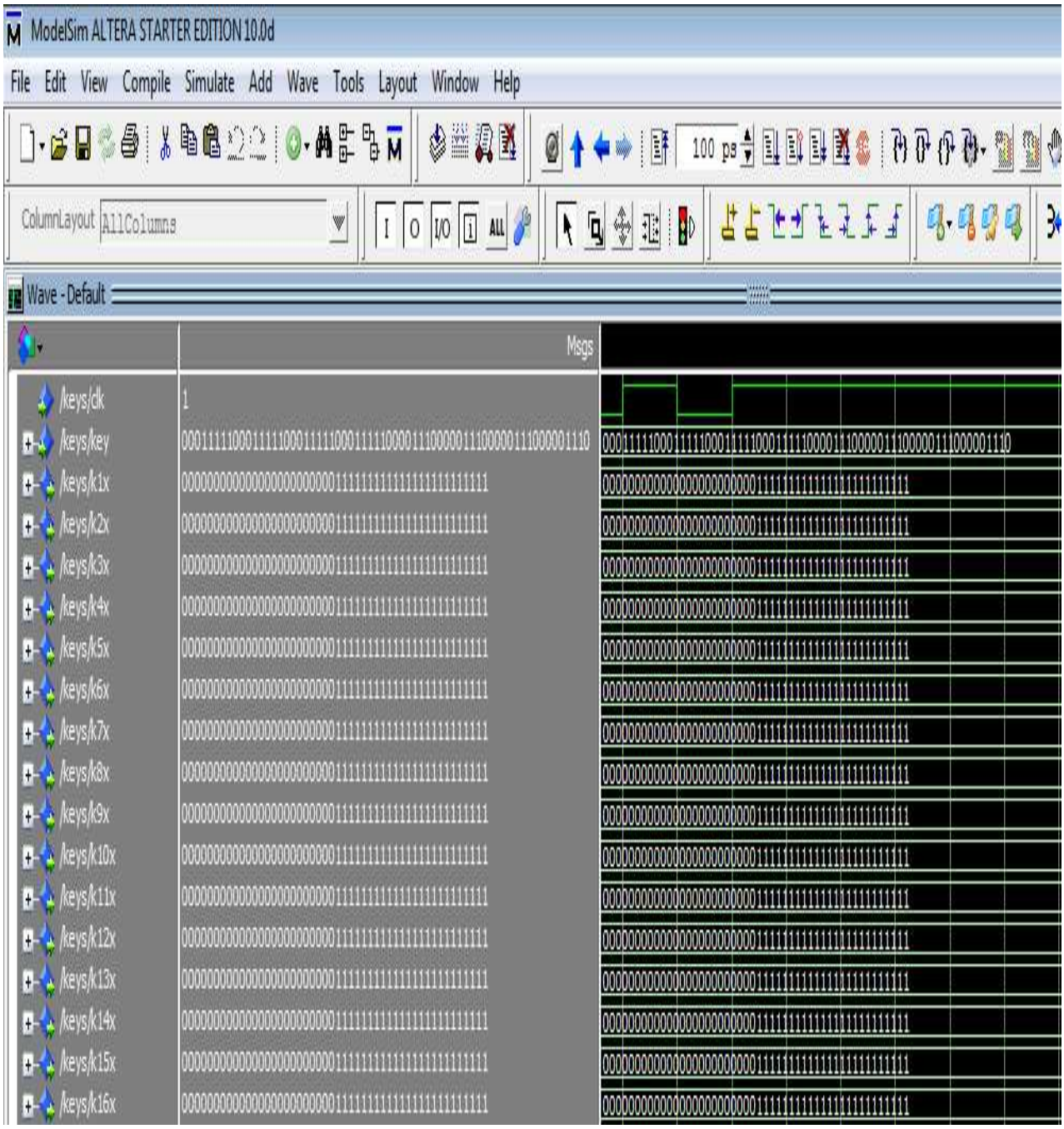


Figure.4.5: Résultat de simulation avec K= (1F1F1F1F 0E0E0E0E) $_{16}$.

D'après la simulation de la clé faible (1F1F1F1F 0E0E0E0E)$_{16}$ toutes les16 sous clés résultants sont identiques (tout les sous clé= (000000 111111)$_{16}$).

4.3.1. Résultats de La simulation des 6 clés demi-faibles classique :

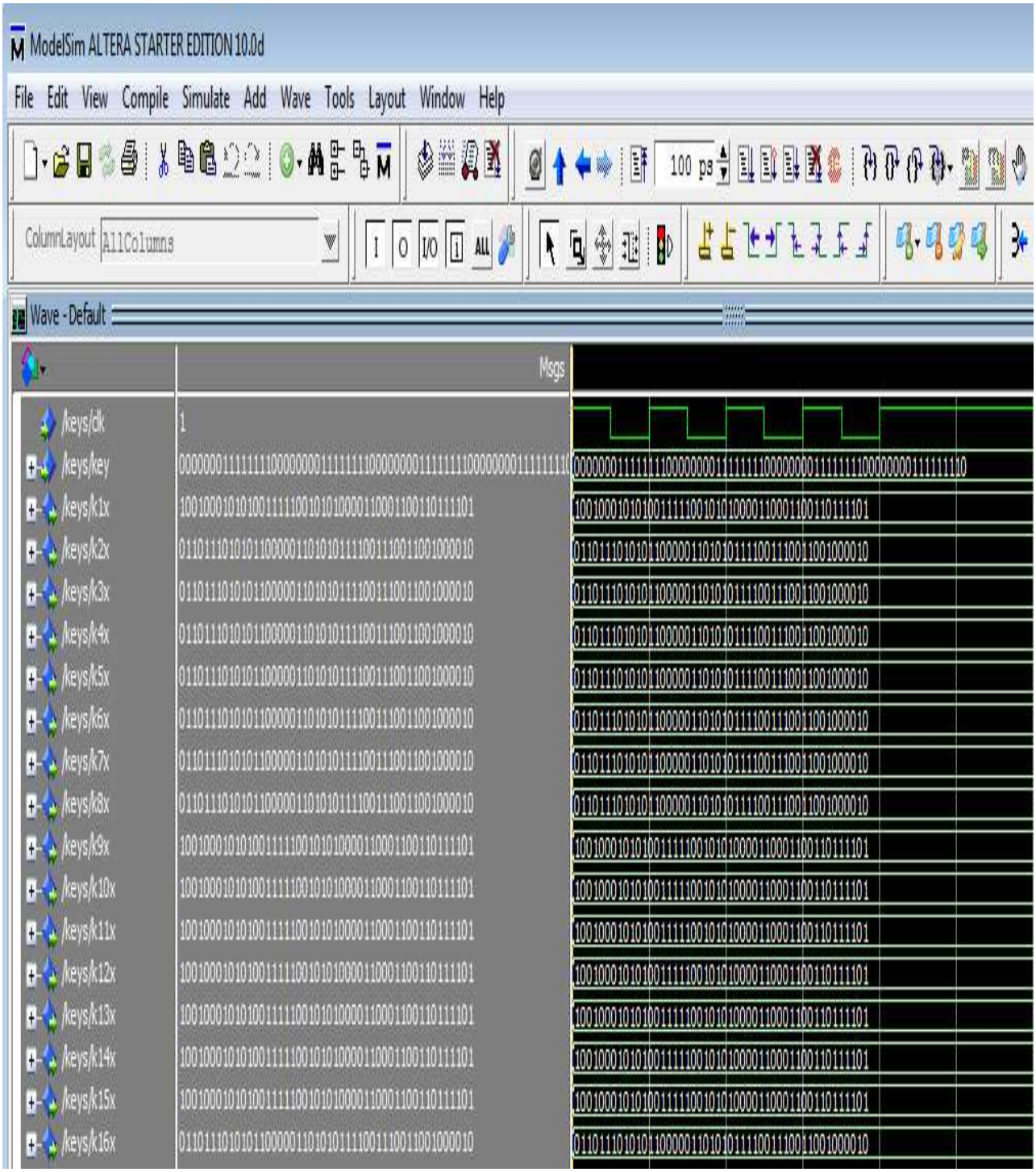


Figure.4.6: Résultat de simulation avec K= (01 FE 01 FE 01 FE 01 FE) $_{16}$.

La simulation de la clé demi-faible (01 FE 01 FE 01 FE 01 FE) $_{16}$ crée deux sous-clés différent (9153E5 4319BD) $_{16}$ et (6EAC1A BCE642) $_{16}$ chaque un est répété 8 fois.

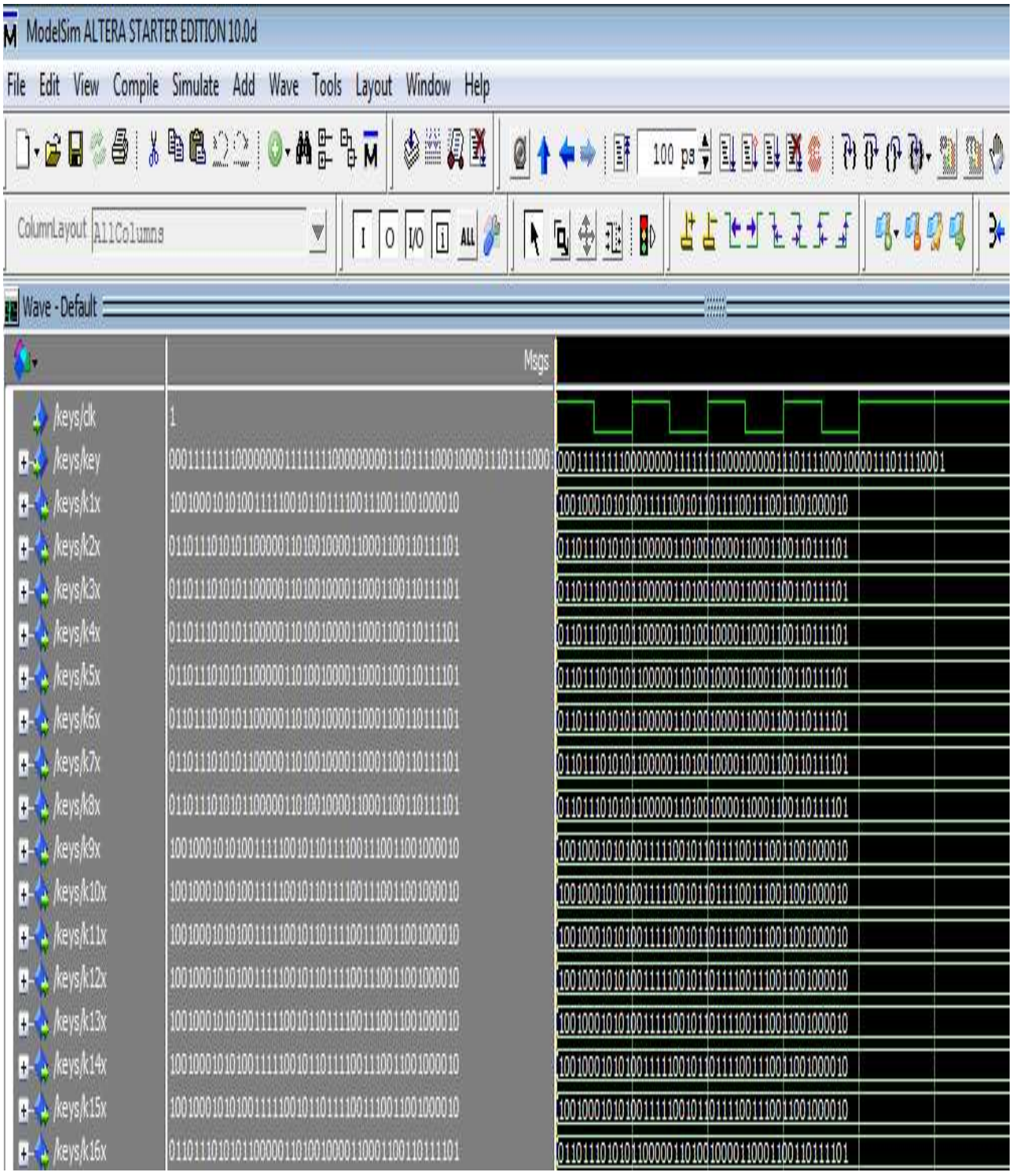


Figure.4.7: Résultat de simulation avec K= $(1F\ E0\ 1F\ E0\ 0E\ F1\ 0E\ F1)_{16}$.

La simulation de la clé demi-faible $(1F\ E0\ 1F\ E0\ 0E\ F1\ 0E\ F1)_{16}$ crée deux sous-clés différentes $(9153E5\ BCE642)_{16}$ et $(6EAC1A\ 4319BD)_{16}$ chaque un est répété 8 fois.

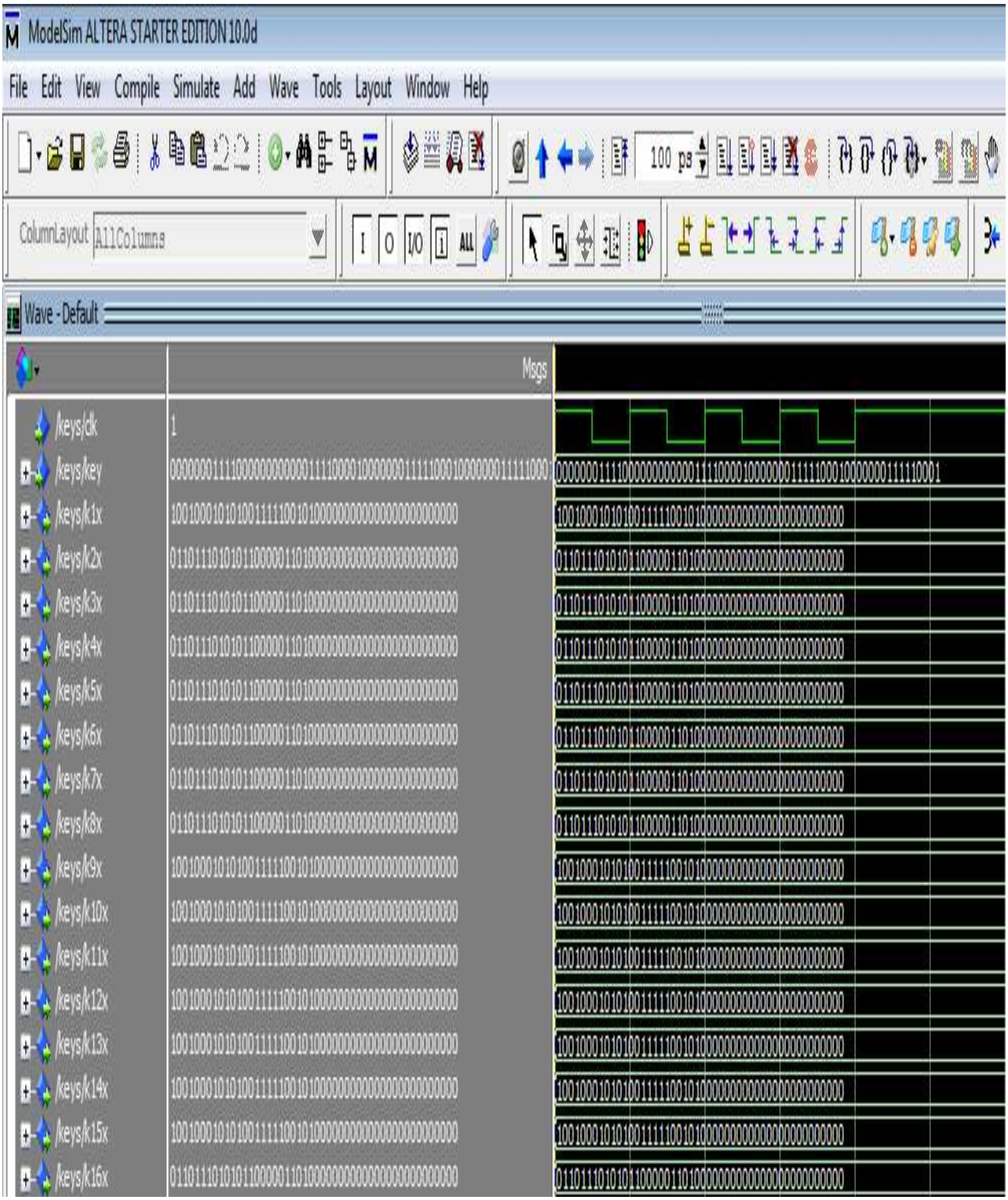


Figure.4.8: Résultat de simulation avec K= (01 E0 01 E1 01 F1 01 F1) $_{16}$.

La simulation de la clé demi-faible (01 E0 01 E1 01 F1 01 F1) $_{16}$ crée deux sous-clés différentes (9153E5 000000) $_{16}$ et (6EAC1A 000000) $_{16}$ chaque un est répété 8 fois

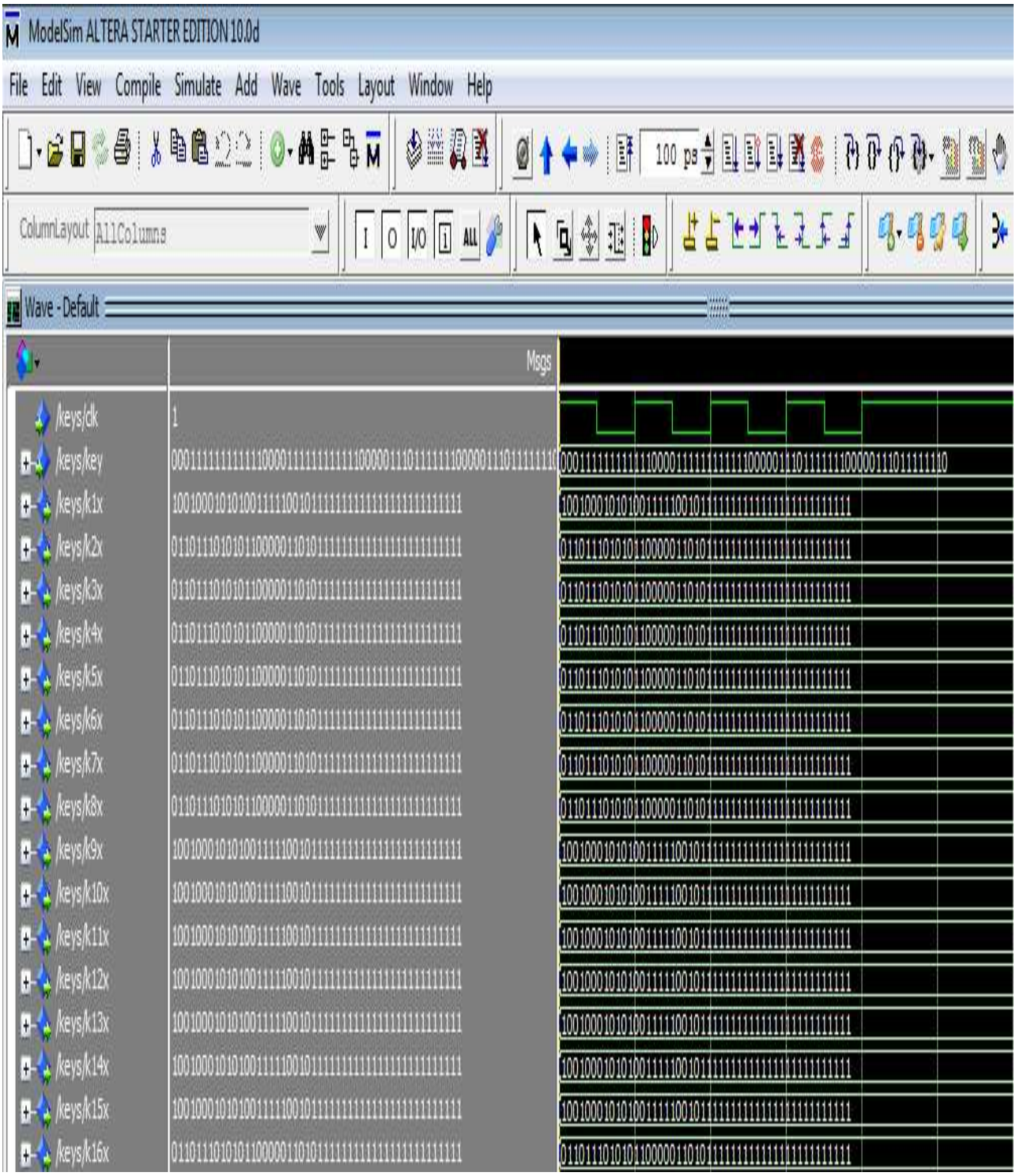


Figure.4.9: Résultat de simulation avec K= (1F FE 1F FE 0E FE 0E FE) $_{16}$.

La simulation de la clé demi-faible (1F FE 1F FE 0E FE 0E FE) $_{16}$ crée deux sous-clés différentes (9153E5 FFFFFF) $_{16}$ et (6EAC1A FFFFFF) $_{16}$ chaque un est répété 8 fois.

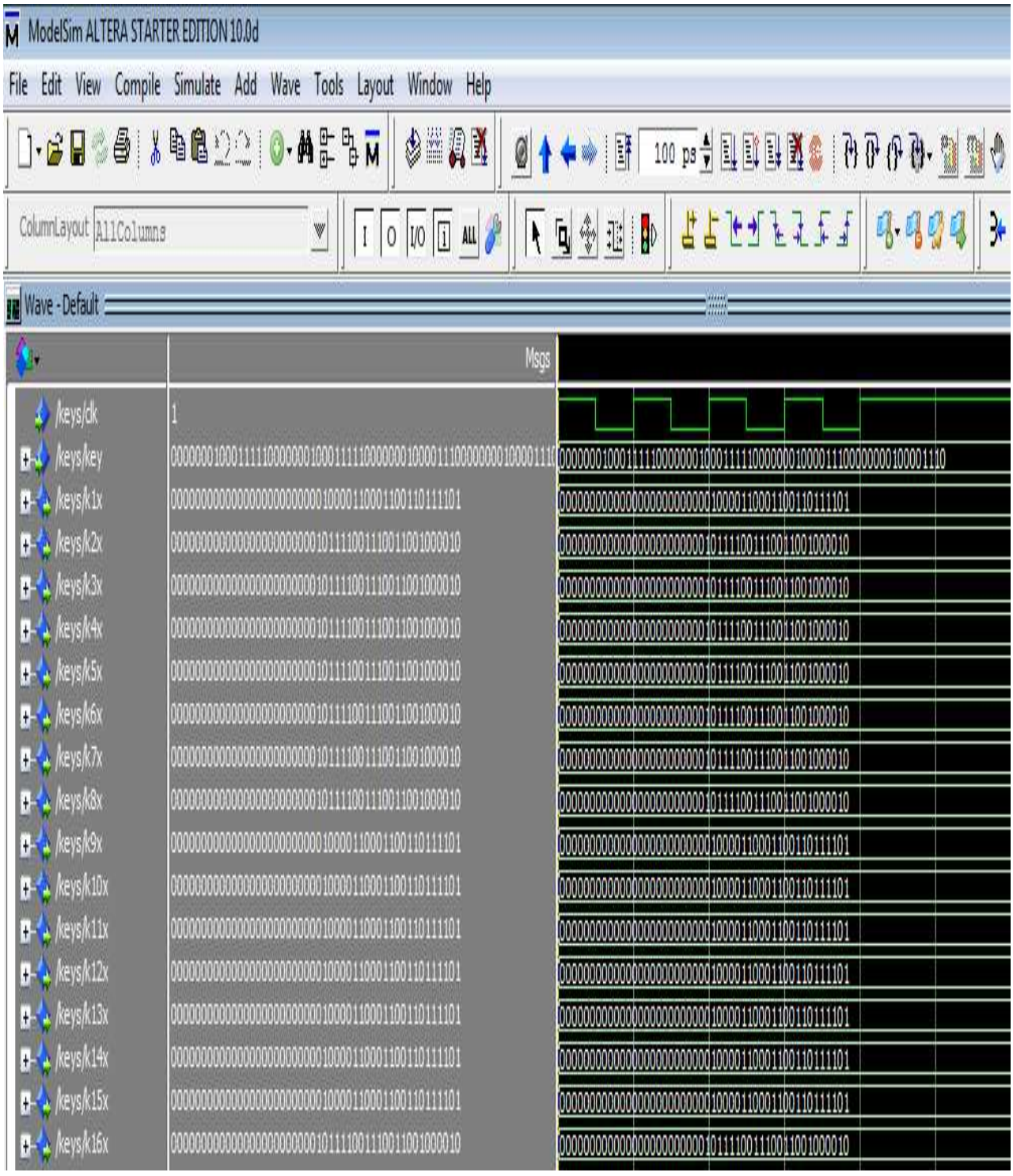


Figure.4.10: Résultat de simulation avec K= (01 1F 01 1F 01 0E 01 0E) $_{16}$.

La simulation de la clé demi-faible (01 1F 01 1F 01 0E 01 0E) $_{16}$ crée deux sous-clé différentes (000000 BCE642) $_{16}$ et (000000 4319BD) $_{16}$ chaque un est répété 8 fois.

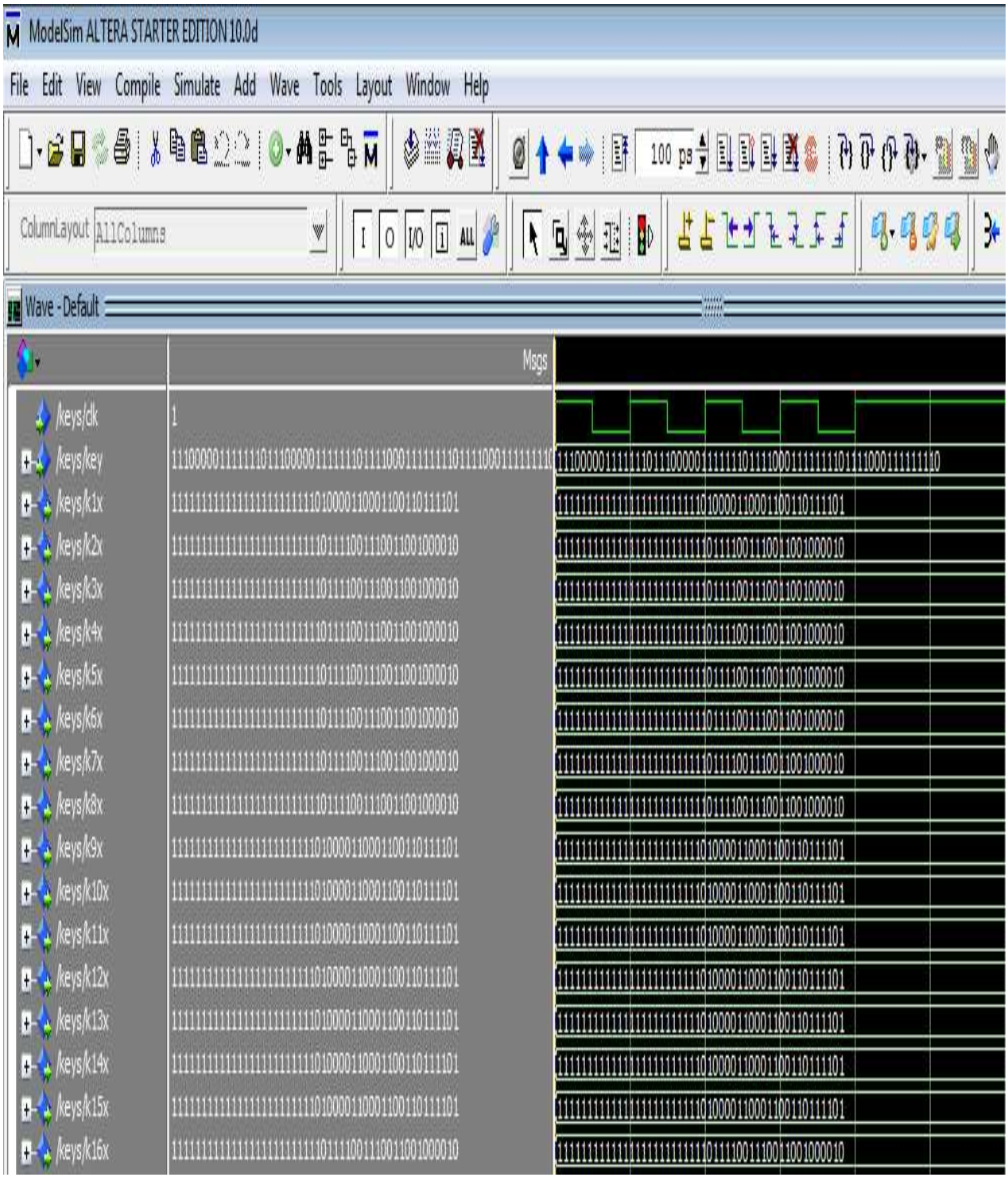


Figure.4.11: Résultat de simulation avec K= (E0 FE E0 FE F1 FE F 1FE) $_{16}$.

La simulation de la clé demi-faible (E0 FE E0 FE F1 FE F1 FE) $_{16}$; crée deux sous-clés différentes (FFFFFF 4319BD) $_{16}$ et (FFFFFF BCE642) $_{16}$ chaque un est répété 8 fois.

(Le code VHDL de la génération des sous clés de DES classique se trouve dans l'annexe C).

4.4. Le schéma RTL :

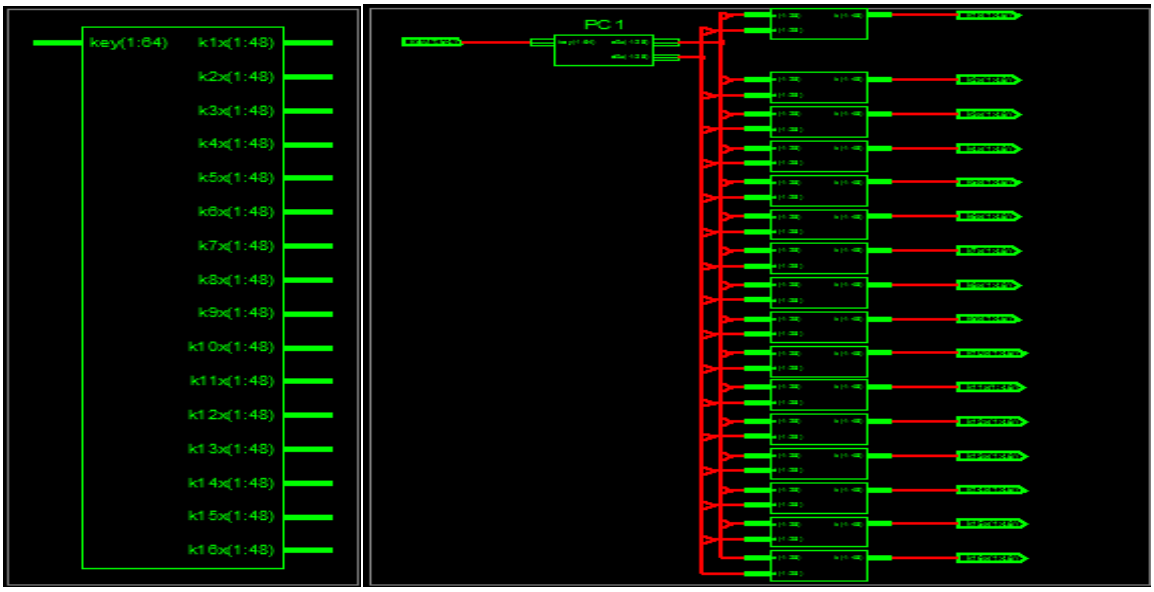


Figure. 4.12 : Le schéma RTL le la génération de sous clés de l'algorithme DES classique.

A travers le schéma RTL, nous avons remarqué que : la boite noire de l'implémentation de notre algorithme représente juste les nombres des entrés/sorties (1 entré "Key" et 16 sorties "K1…K16"). A l'interne de cette boite noire on trouve 17 boites de permutation (boite de permutation PC1 et 16 poites de permutation PC2). Et le chemin d'interconnexion.

Le tableau 1 fournit les données d'exploitation dans le programme classique de la génération des sous clés de DES:

Nous avons choisi deux types de FPGA de la famille la plus fréquente, Spartan3 (XC3S4000L) et Virtex (XCV1000).

Type FPGA	Période (ns)	Fréquence Maximale (MHZ)	Nbr IOBs	Nbr SLICES	Nbr LUT	Nbr Slice Flip Flops	Nbr GCLKs
Spartan3 (XC3S4000L)	3.398	294.291	825 (633)	32 (27648)	56 (55296)	56 (55296)	1 (8)
Virtex (XCV1000)	5.284	189.251	825 (404)	32 (12288)	56 (24576	56 (24576)	1 (4)

Tab.4.1 : Les données d'exploitation de l'algorithme DES classique.

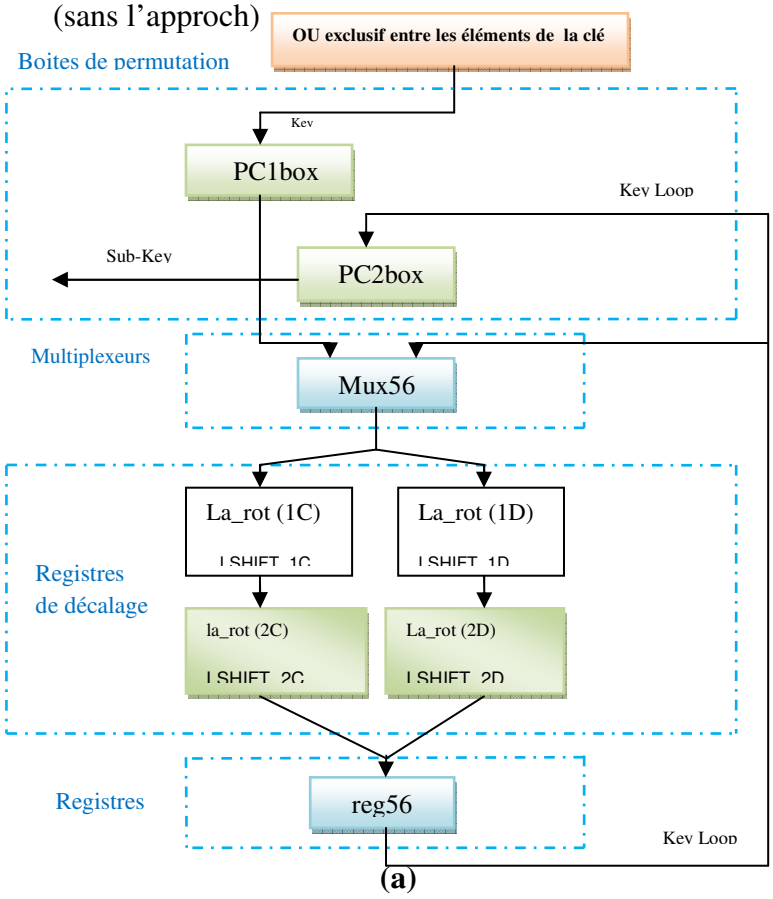
Le tableau précédent montre les ressources matérielles occupées par les FPGA (Spartan3(XC3S4000L), et virtex(XCV1000)) et la fréquence de simulation pour l'implémentation de programme classique.

- Pour les ressources matérielles nous avons utilisé 825 IBOS ,32 slices, 56 luts, 56 flip flops et 1 GCLKs.

- La simulation de programme dans FPGA de type Spartan3, utilisé une fréquence maximale égale 294.291 MHZ dans un période de 3.398 (ns).et pour FPGA de type virtex utilisé moine de fréquence qui égale 189.251(MHZ) dans une période de 5.284.

4.5. Implémentation VHDL de l'approche proposée

Tout d'abord la clé initiale KK de 64 bits passe par la proche proposée (l'ajoute de ou exclusif entre les 64 bits de la clé). Cette partie a besoin de registre pour stocker les résultats de rotation comme il est présenté dans la figure 4.13. Puis, toutes les étapes suivantes sont les mêmes étapes utilisées dans l'implémentation VHDL de la génération de sous clés de l'algorithme DES (sans l'approch)



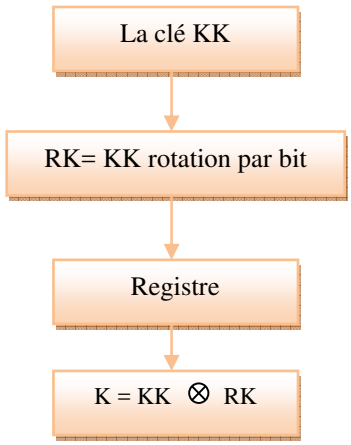


Figure.4.13 : a) Diagramme de la génération des sous clés de l'algorithme DES amélioré. B) Diagramme de OU exclusif entre les éléments de la clé.

4.6. Résultats d'implémentation de la génération des sous clés de l'algorithme DES amélioré

Les résultats de la simulation de la génération des sous clés de l'algorithme DES amélioré a partir du simulateur Modèlsim sont présentées dans les figures (4.14, 4.15, 4.16, 4.17, 4.18 4.19, 4.20, 4.21, 4.22, 4.23)

- **Résultats de La simulation des 4 sous-clés faibles améliorées :**

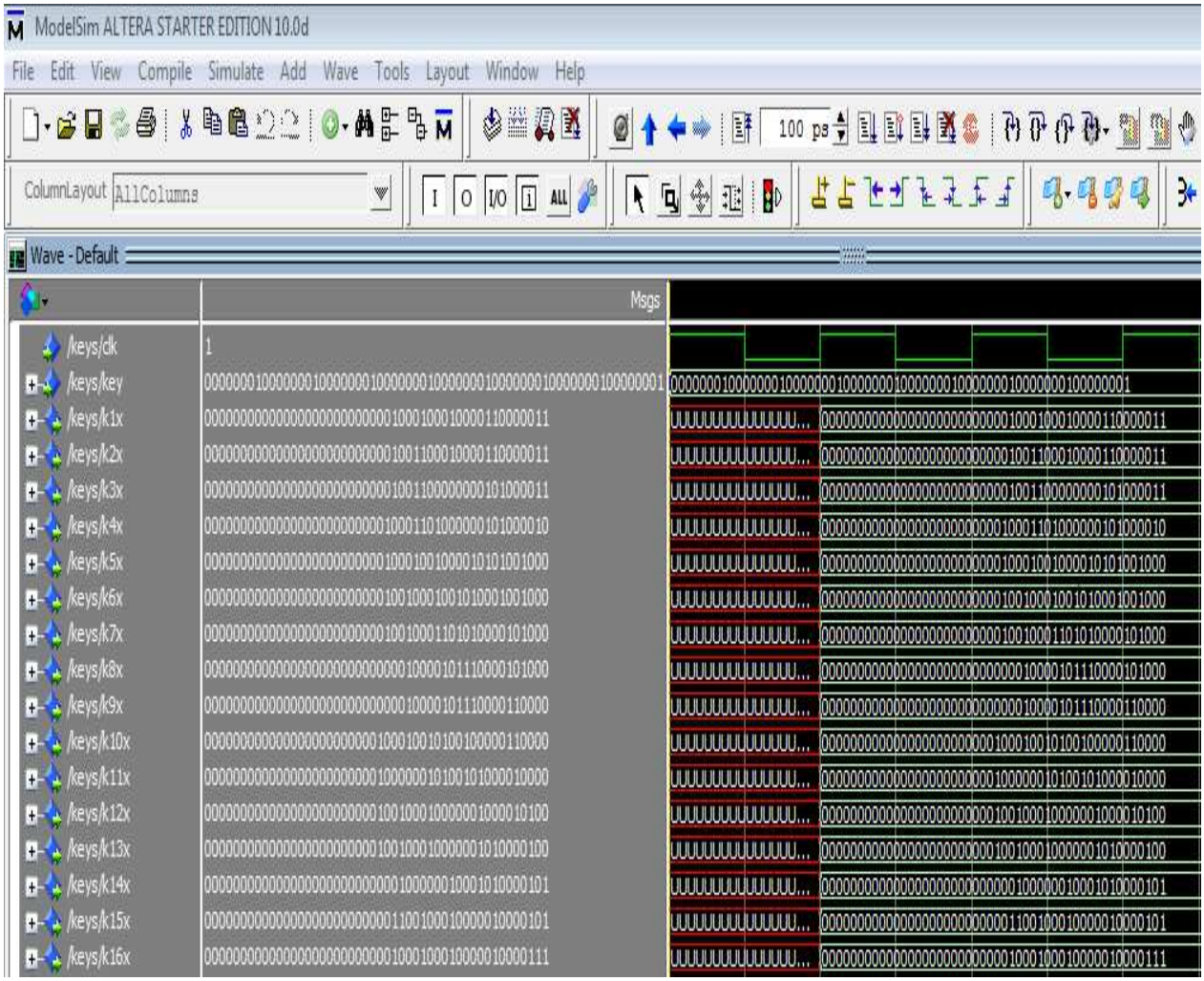


Figure.4.14: Résultat de simulation avec K= $(01010101\ 01010101)_{16.}$

Aprés la simulation de la clé faible $(01010101\ 01010101)_{16}$ dans le programme de la génération des sous clés de DES amélioré nous avons remarqué que tout les sous clés résultant sont défférent.

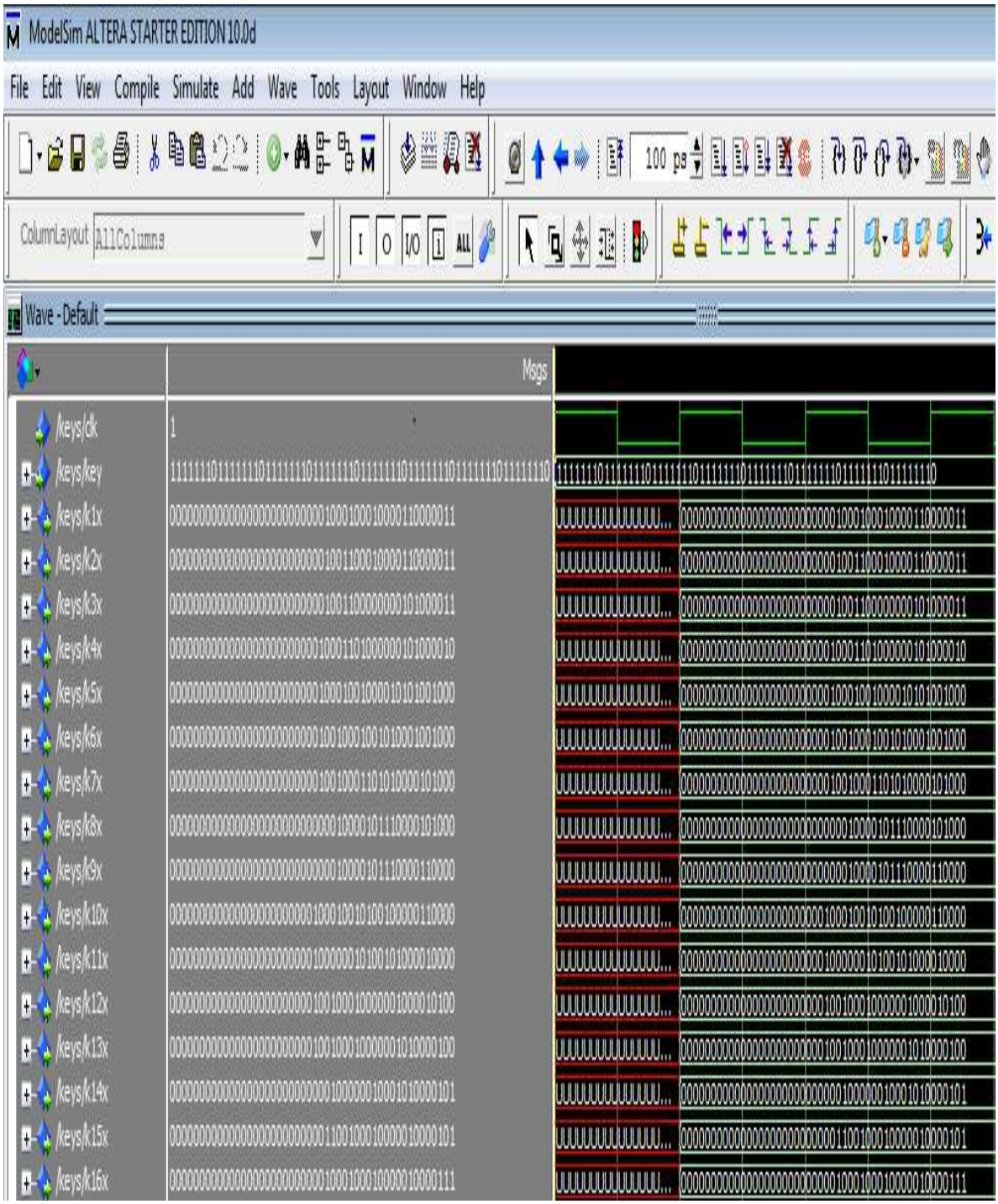


Figure.4.15: Résultat de simulation avec K= (FEFEFEFE FEFEFEFE) $_{16}$.

Aprés la simulation de la clé faible (FEFEFEFE FEFEFEFE) $_{16}$ dans le programme de la génération des sous clés de DES amélioré nous avons remarqué que tout les sous clés résultant sont défférent.

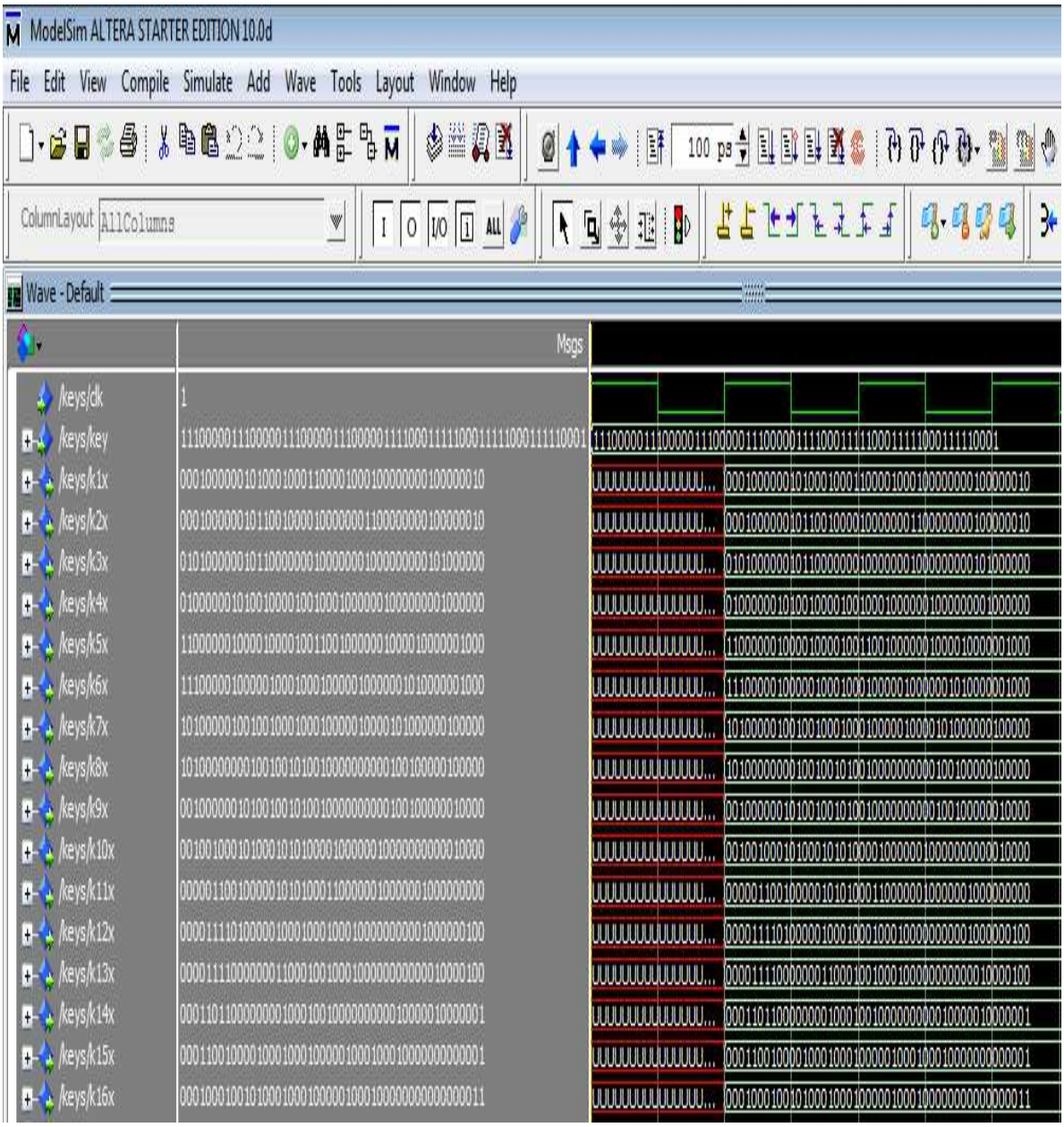


Figure.4.16: Résultat de simulation avec K= (E0E0E0E0 F1F1F1F1)$_{16}$.

Aprés la simulation de la clé faible (E0E0E0E0 F1F1F1F1)$_{16}$ dans le programme de la génération des sous clés de DES amélioré nous avons remarqué que tout les sous clés résultant sont défférent.

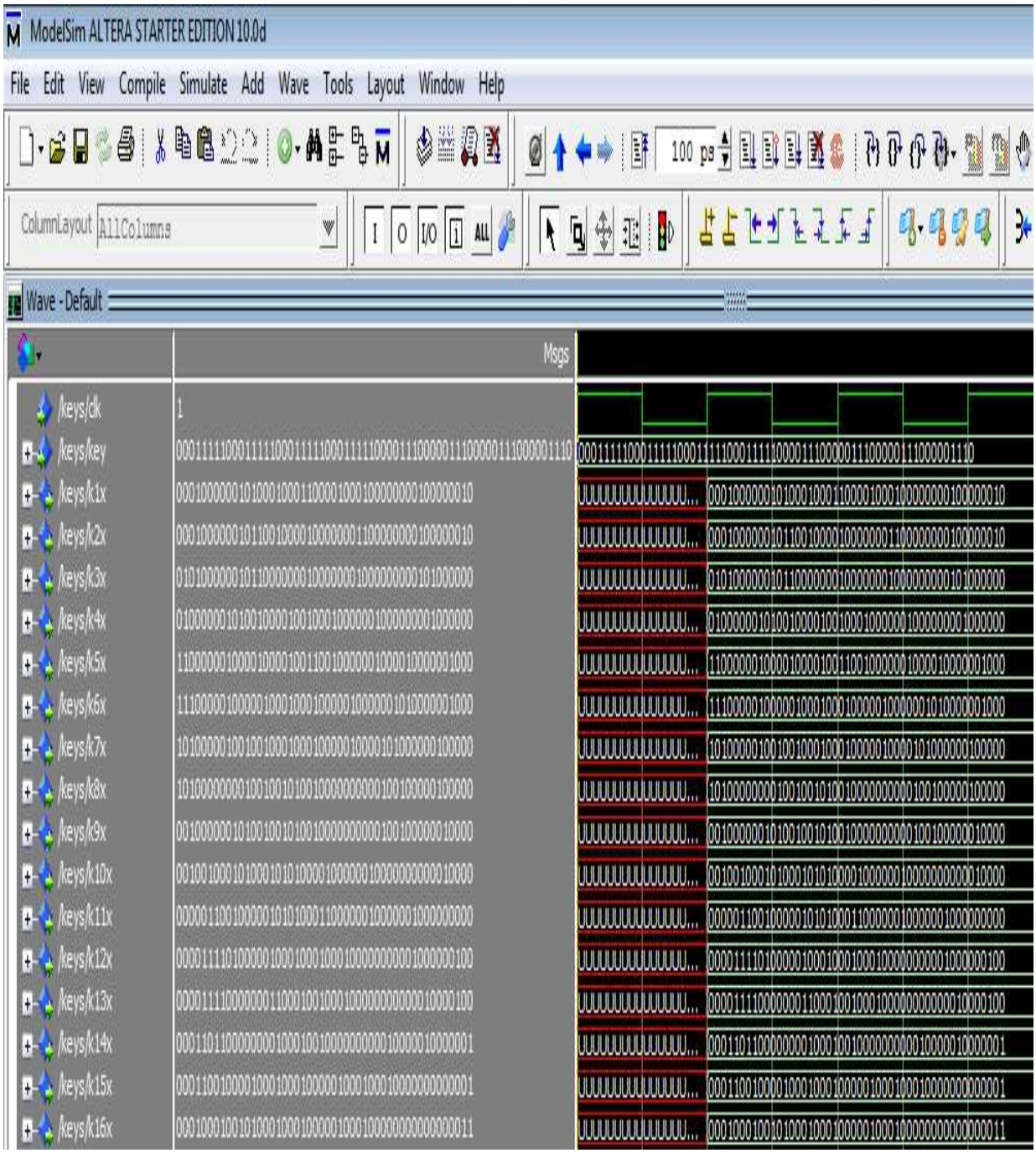


Figure.4.17: Résultat de simulation avec K= (1F1F1F1F 0E0E0E0E) $_{16}$.

Aprés la simulation de la clé faible (1F1F1F1F 0E0E0E0E) $_{16}$dans le programme de la génération des sous clés de DES amélioré nous avons remarqué que tout les sous clés résultant sont défférent.

- **Résultats de La simulation des 6 clés demi-faibles améliorées :**

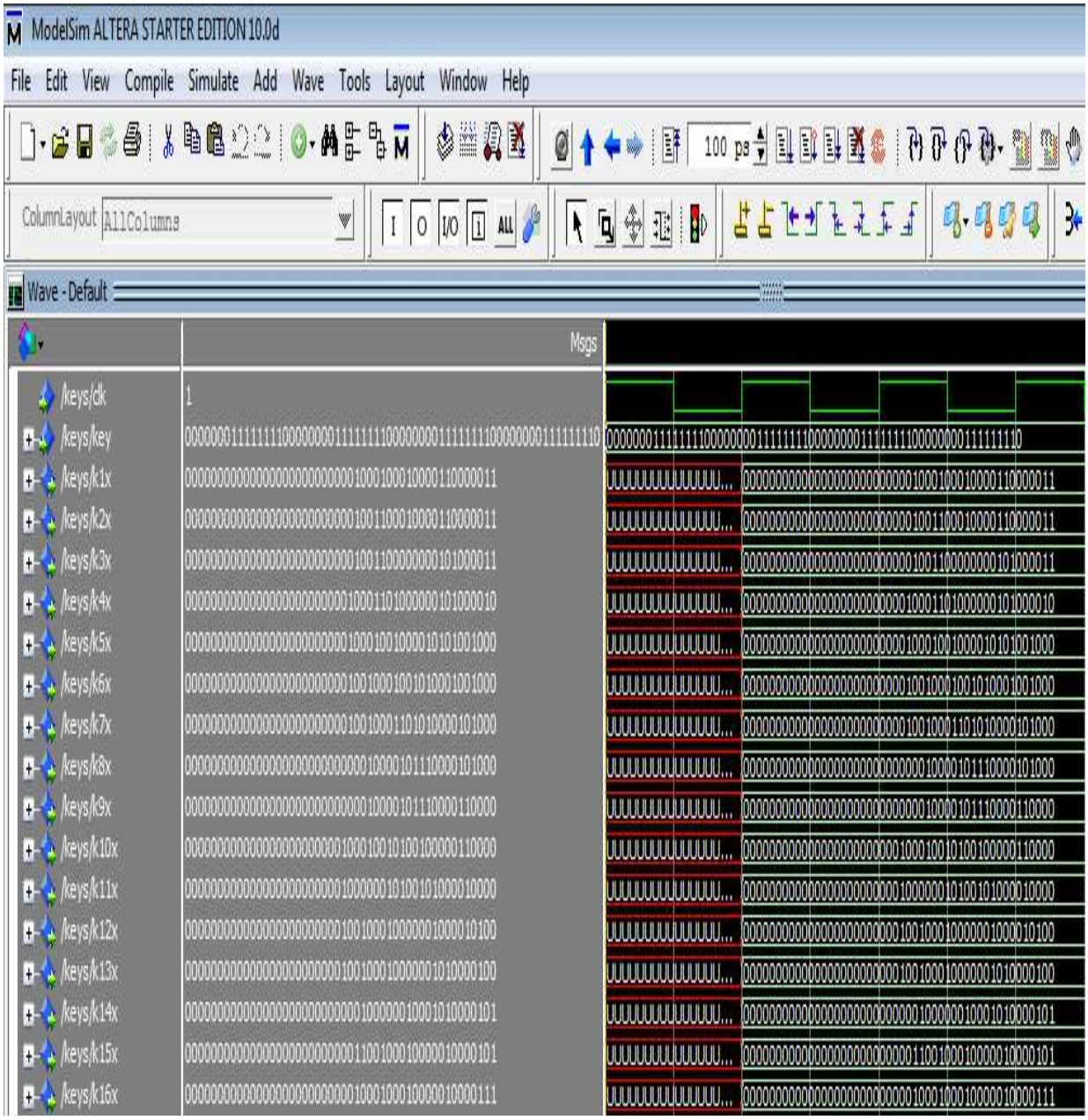


Figure.4.18: Résultat de simulation avec K= (01 FE 01 FE 01 FE 01 FE) $_{16}$.

Apré la simulation de la clé demi faible (01 FE 01 FE 01 FE 01 FE) $_{16}$ dans le programme de DES amélioré tout les sous clés résultants sont déférent.

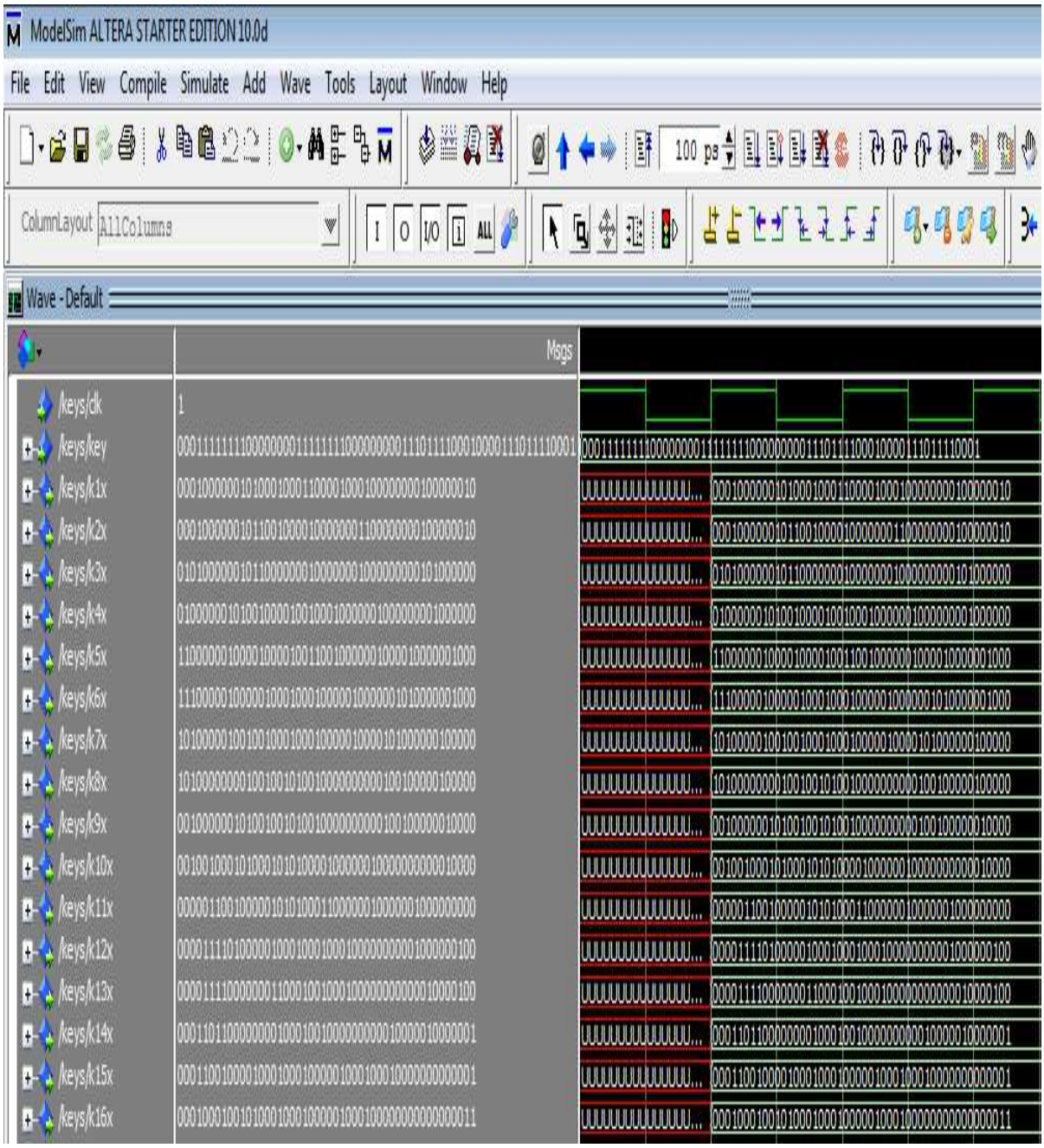


Figure.4.19: Résultat de simulation avec K= (1F E0 1F E0 0E F1 0E F1) $_{16}$.

Apré la simulation de la clé demi faible (1F E0 1F E0 0E F1 0E F1) $_{16}$ dans le programme de la génération des sous clés de DES amélioré tout les sous clés produits sont déférent.

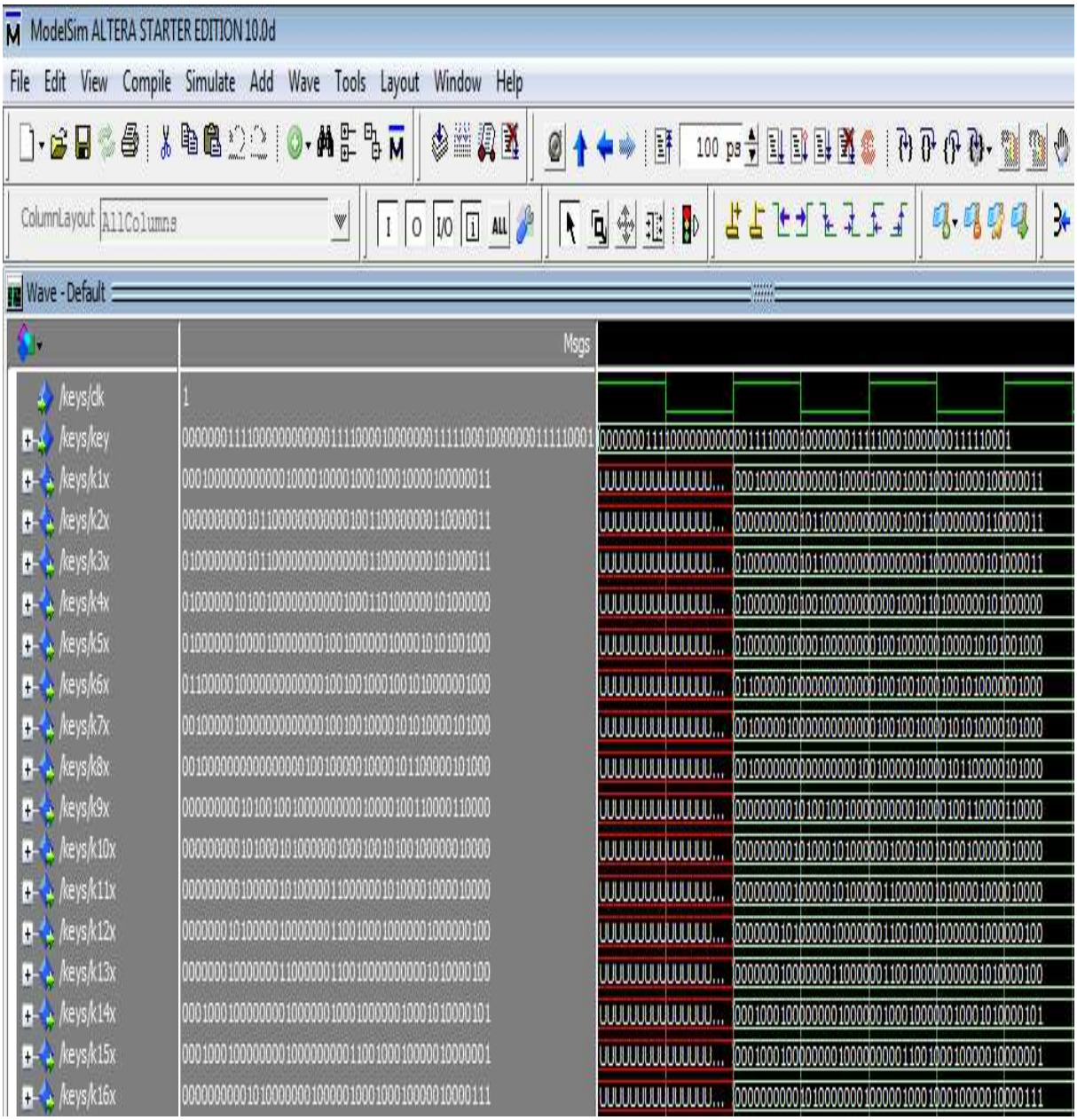


Figure.4.20: Résultat de simulation avec K= (01 E0 01 E1 01 F1 01 F1) $_{16}$.

Apré la simulation de la clé demi faible (01 E0 01 E1 01 F1 01 F1) $_{16}$ dans le programme de la génération des sous clés de DES amélioré tout les sous clés produits sont déférents.

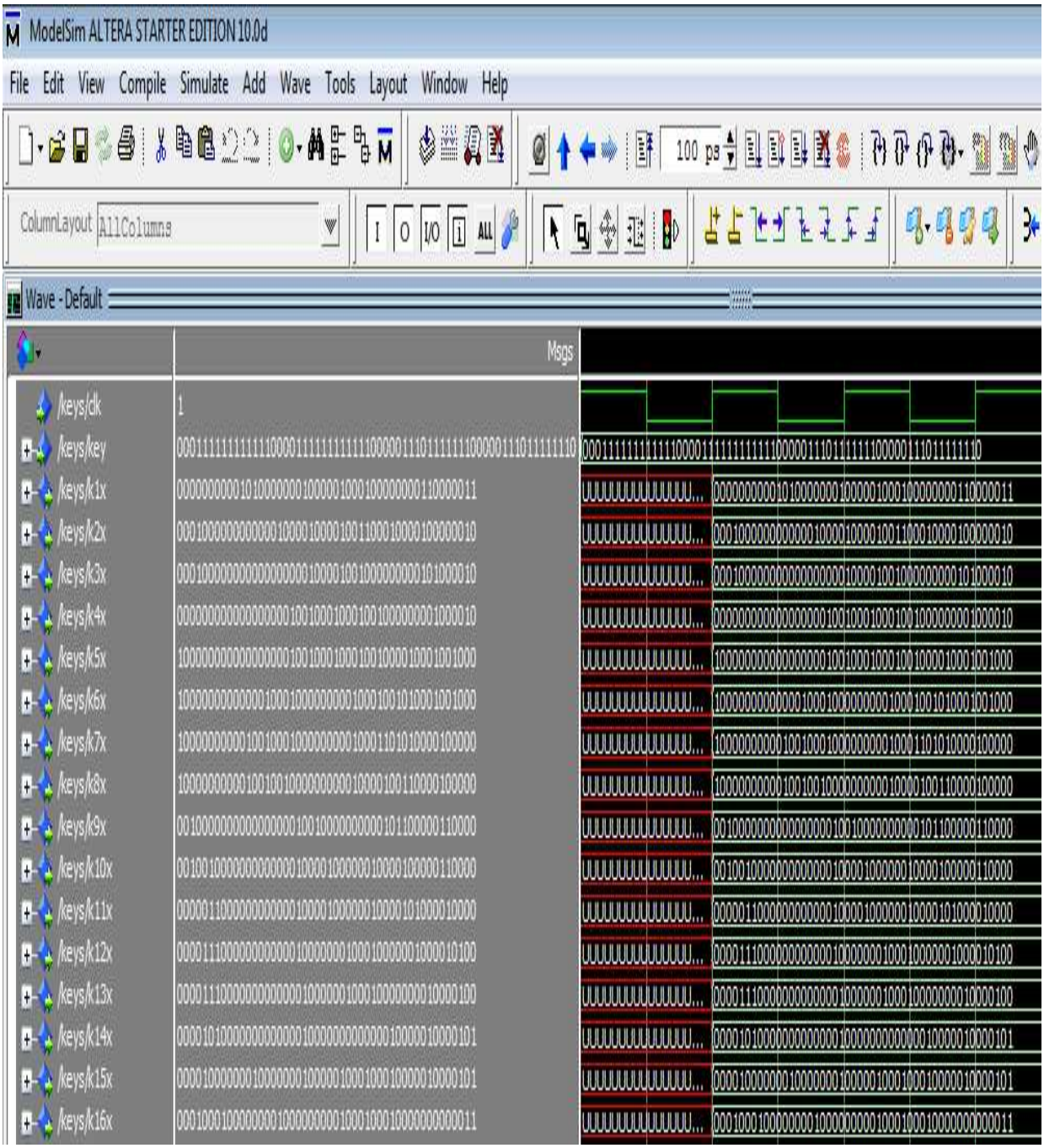


Figure.4.21: Résultat de simulation avec K= (1F FE 1F FE 0E FE 0E FE) $_{16}$.

Apré la simulation de la clé demi faible (1F FE 1F FE 0E FE 0E FE) $_{16}$ dans le programme de la génération des sous clés de DES amélioré tout les sous clés produits sont déférents.

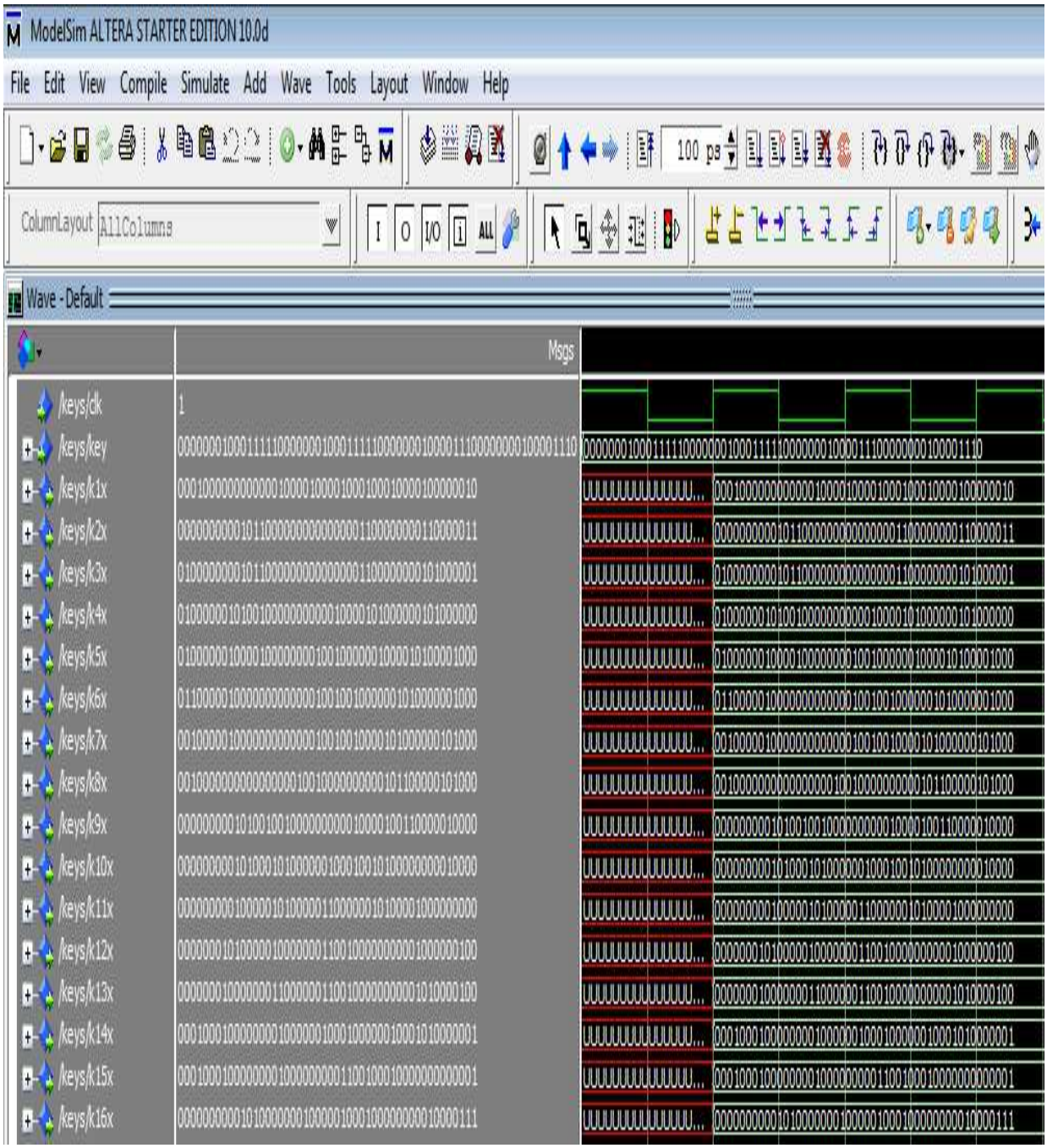


Figure.4.22: Résultat de simulation avec K= (01 1F 01 1F 01 0E 01 0E) $_{16}$.

Apré la simulation de la clé demi faible (01 1F 01 1F 01 0E 01 0E) $_{16}$ dans le programme de la génération des sous clés de DES amélioré tout les sous clés produits sont déférents.

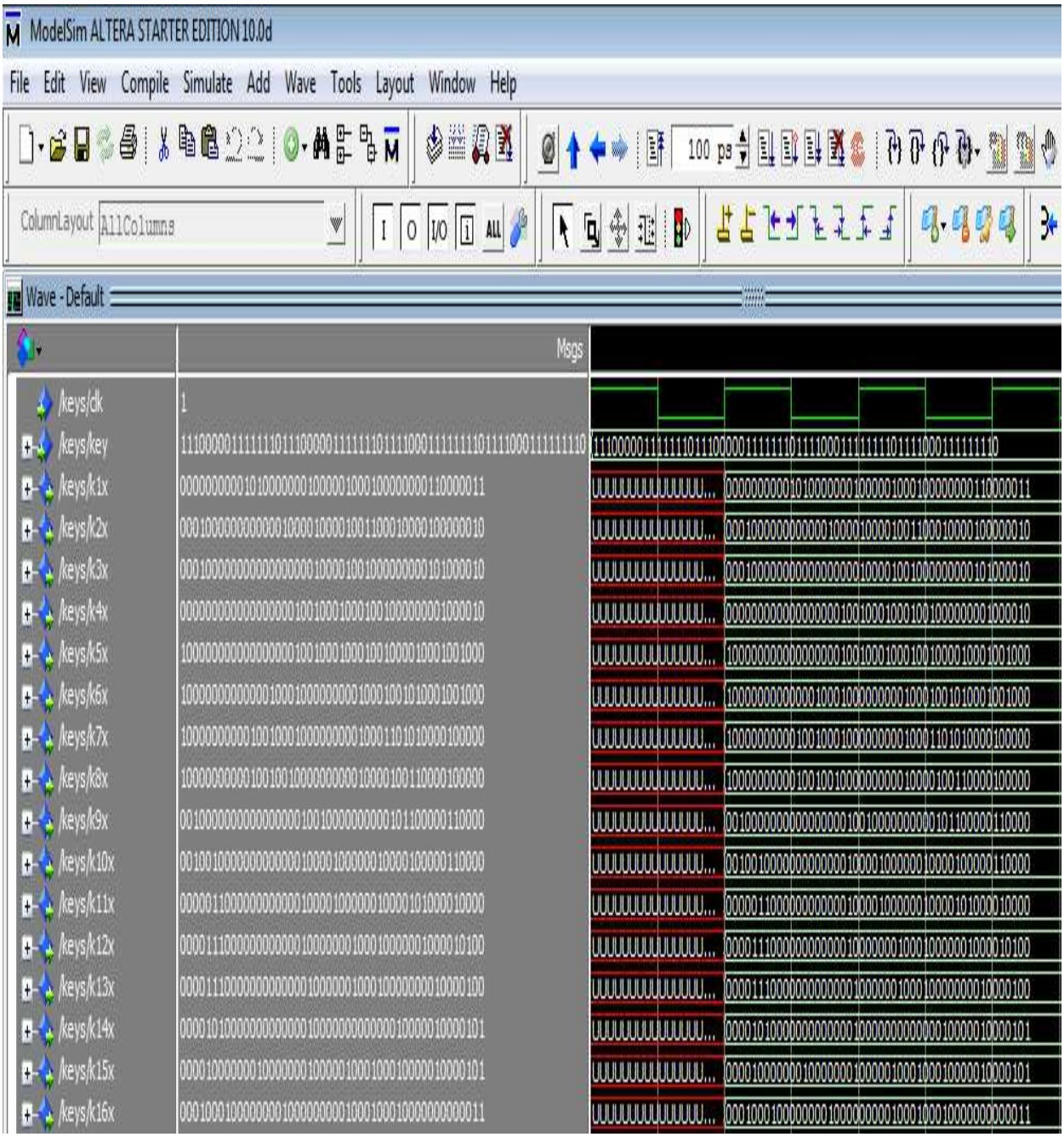


Figure.4.23: Résultat de simulation avec K= (E0 FE E0 FE F1 FE F1 FE) $_{16.}$

Apré la simulation de la clé demi faible (E0 FE E0 FE F1 FE F1 FE) $_{16}$ dans le programme de la génération des sous clés de DES amélioré tout les sous clés produits sont déférents.

(Le code VHDL de la génération des sous clés de DES amélioré se trouve dans l'annexe D).

Le tableau 2 fournit les données d'exploitation de la génération des sous clés de l'algorithme DES amélioré (les ressources utilisées) pour les deux type de FPGA Spartan3 et Virtex.

Type FPGA	Période (ns)	Fréquence Max (MHZ)	Nbr IOB_S	Nbr SLICES	Nbr LUT	Nbr Slice Flip Flops	Nbr GCLKs
Spartan3 (XC3S4000L)	3.398	294.291	833 (633)	97 (27648)	112 (55296)	168 (55296)	1 (8)
Virtex (XCV1000)	5.284	189.251	833 (404)	97 (12288)	112 (24576)	168 (24576)	1 (4)

Tab.4.2 : Les données d'exploitation de l'algorithme DES amélioré.

D'après l'analyse de ce tableau nous remarquons qu'il y a une augmentation dans les ressources matériales les IOBs augmenté par 8 nombre, les slices par 65 nombre, les luts par 56nombre, les flips flops par 112 nombre et le nombre de GCLKs reste 1.

Nous avons enregistré la même fréquence pour l'implémentation, même si nous avons consommé plus des ressources

4.7. Comparaison des résultats

- Les résultats sont complètement corrects et compatibles avec l'algorithme appliqué.

- après la vérification des résultats, nous avons remarqué que les sous clés de l'algorithme DES amélioré dans MATLAB sont les mêmes avec les sous clés de l'algorithme DES amélioré dans VHDL.

- Pour comparer les résultats nous avons choisi deux types de FPGA de la famille la plus fréquente, Spartan3 (XC3S4000L) et Virtex (XCV1000).

- A partir de l'analyse des résultats de tableaux précédant (Tab4.1 et Tab4.2), nous avons observé que :

- Dans la simulation du programme amélioré nous avons utilisé des ressources matérielles (IOBs, slices, lut) plus que dans le programme classique.
- La même fréquence utilisé dans les deux programmes classique et amélioré pour les deux type de FPGA, même si Il ya une occupation des ressources matérielles plus para port a

l'implémentation de programme amélioré qui implique une augmentation dans la chemine critique.

- l'implémentation sur FPGA Spartan3 réalise un meilleur résultat que FPGA virtex parvenant à une fréquence plus élevée.

4.8. Conclusion

Dans ce chapitre nous avons présenté les résultats de l'implantation FPGA de deux versions classique et amélioré de la génération de sous clés de DES sur Xilinx. Les résultats obtenus sont satisfaisantes du point de la fréquence de fonctionnement et les ressources utilisées. Les résultats de simulation de l'approche proposée produisent des sous clés non redondantes ce qui augmente sa résistance aux attaques.

Conclusion générale

La nécessité de sécuriser la transmission de données augmente chaque jour et nécessite des dispositifs de chiffrement de données sécurisés pour préserver la confidentialité et l'authentification des échanges des applications critiques.

Ce travail se concentre sur l'algorithme de cryptage symétrique DES qui a une clé de 64 bits. L'intérêt de l'algorithme DES dans le domaine de la cryptographie n'est plus à démontrer, et son intégration sur circuit numérique (FPGA) est la cible d'un grand nombre de travaux,

Nous avons étudié le DES dans le but d'amélioration de génération de sous-clés pour avoir des sous-clés non redondance.

Notre étude nous a conduits à considérer les points suivants :

Nous avons d'abord fais un tour d'horizon rapide dans le domaine du cryptage et les différentes algorithmes existant. Ensuite, nous avons abordé l'algorithme DES par son aspect théorique et la problématique qui se pose au niveau de la génération de sous clés qui est la redondance des sous-clés.

Par la suite, nous avons proposé l'ajout d'une étape à la génération de sous clés qui réduit le problème de redondance de sous clés. Nous avons simulé le fonctionnement de notre approche sous le logicielle (MATLAB) que nous avons comparé avec le DES classique.

Les architectures réalisant la génération de sous clés du DES (améliore et classique) ont été conçue puis décrite en utilisant le langage VHDL implémentée sur un circuit FPGA de la famille Spartan3 (XC3S4000L) et Virtex (XCV1000) de xilinx sous l'environnement ISE 9.2.

Les résultats de l'implémentation sur FPGA obtenus sont relativement satisfaisantes puisque les fréquences dans les deux programmes (classique et amélioré) sont les mêmes; L'utilisation de l'approche proposée a réduit la redondance dans la génération des sous-clés et elle a réduit la vulnérabilité à l'attaque, et elle a amélioré l'efficacité du cryptage. Les résultats présentés de la mis en œuvre fournissent un haut niveau d'assurance et d'exactitude.

Perspectives :

Comme travaux future :

- l'implémentation de notre nouvel algorithme pour des applications basées sur l'echange d'informations (images, texte ...etc).

-Utiliser d'autres versions plus performantes et actualisées que les versions utilisées : Spartan3 (XC3S4000L) et Virtex (XCV1000).

- L'amélioration de notre algorithme DES dans le but d'obtenir une meilleure sécurité, réduire le temps de cryptage total...etc.

- Réduire et modifier le séquenceur (machine d'état) pour avoir une fréquence élevée.

Voila quelques idées qui peuvent être prise en compte si l'on souhaite continue dans le même axe de recherche. Ce domaine est loin d'être fini et ne constitue que le premier jalon.

Références

[1] B. DEBBAGH, N. BOUNEGEB, *“ Etude de comparaison de principaux systèmes crypto fournis par le package de Bouncy Castel plat forme Java SDK* ", Mémoire de Master académique, Université KASDI MERBAH OUARGLA (Algérie) ,05.Juin.2016.

[2] M. BOUSALHA, Y. TIFOUR, "*contribution à la conception d'un crypto système symétrique flexible sur circuit FPXGA* ". Mémoire de Master, Université M'HAMED BOUGARA DE BOUMERDES, (Algérie), 2016.

[3] C. THUILLET, "*Implantations cryptographiques sécurisées et outils d'aide à la validation des contremesures contre les attaques par canaux cachés* ", Thèse de Doctorat, Université BORDEAUX I, (France) ,30.Mars.2012.

[4] K. BOUSSELAM, "*Résistance des circuits cryptographique aux attaques en faute* ", Thèse de Doctorat, Université MONTPELLIER II, (France), 25. Septembre. 2012.

[5] T.MEKHAZNIA, "*Analyse cryptographique par les méthodes heuristiques* ", Thèse de Doctorat, Université de BATNA 2(Algérie), 25. Février. 2017.

[6] K. DICHOU, " *contribution à l'étude des cartes à puce avancées* ", Thèse de Doctorat-LMD, Université M'HAMED BOGARA-BOUMERDES, 2016.

[7] M. BOUCHEMA, "*Exploitation des transformées paramétriques dans le cryptage des images fixes* ", Mémoire de Magistère, Université FERHAT ABBAS –SETIF 1, (Algérie), 28. Décembre. 2012.

[8] S. BOUALLAGUI, "*Techniques d'optimisation déterministe et stochastique pour la résolution de problèmes difficiles en cryptologie* ", Thèse de Doctorat, Institut national des sciences appliquées de ROUEN, 05. Juillet. 2010.

[9] M. VIDEAU, "*Critère de sécurité des algorithmes de chiffrement à clé secrète* ", Thèse de Doctorat, Université de Paris 6, (France), 10. Novembre. 2005.

[10] F. OMARY, "*Applications des algorithmes évolutionnistes à la cryptographie* ", Thèse de Doctorat d'état, Université MOHAMMED V – AGDAL RABAT (MAROC), 26. Juillet. 2006.

[11] A. K. BENHAOUA, "*Approche cryptographie base sur les algorithmes génétique pour la sécurité des réseaux Ad hoc* ", Mémoire de Magistère, Université D'ORAN ES-SENIA, (Algérie), 2005.

[12] O. AZZOUZI, "*Système embarqué flexible pour un chiffrement hybride symétrique/asymétrique* ", Mémoire de Magister, Ecole nationale supérieure d'informatique, (Algérie) ,2014.

[13] A. SALHI, K.BAKIRI, "*FPGA-Based cryptosystem* ", Mémoire de master, Université M'Hamed BOUGARA-BOUMERDES, (Algérie) ,2015.

[14] T. FUHR, "*Conception, preuves et analyse de fonctions de hachage cryptographiques* ", Thèse de Doctorat, Agence national de la sécurité des systèmes d'information de Paris, (France), 03. Octobre. 2011.

[15] M.A.FILALI, "*Etude et Implémentation Pipeline sur FPGA de L'algorithme de Chiffrement AES* ", Mémoire de Magister, Université MOHAMED BOUDIAF d'ORAN, (Algérie), 29. Juin. 2015.

[16] M. ABID, "*des mécanismes d'authentification basé sur l'identité de l'utilisateur pour renforcer la sécurité des réseaux* ", Thèse de doctorat, Université de Pierre et Marie curie ,30 Mars 2012.

[17] G. ZAIDI, "*Sécurisation par dynamiques chaotiques des réseaux locaux sans fil au niveau de la couche MAC* ", Thèse de Doctorat, L'École Nationale d'Ingénieurs de Sfax, (Tunisie), 06. Décembre. 2012.

[18] L. TING, " *Optimisation par synthèse architecturale des méthodes de partitionnement temporel pour les circuits reconfigurables* ", Thèse de doctorat, Université Henri Poincaré, Nancy 1, Mai 2008.

[19] A. BERZATI "*Analyse cryptographique des altérations d'algorithmes* ", Thèse de Doctorat, Université de Versailles Saint-Quentin, 29. Septembre. 2010.

[20] Y. M.AIT AMEUR, "*sécurisation de communication dans les réseaux d'ordinateurs (couche SSL)* ", Mémoire de Master, Université Mohamed Khider Biskra, (Algérie), Juin. 2014.

[21] S. BEKHOUCH, " *Fondements mathématiques et fonctionnement du standard de chiffrement avancé Rijndael (AES)* ", Mémoire de Magister, Université des Sciences et de la Technologie Houari Boumediene, (Algérie), 2006.

[22] A. ALI PACHA, N.HADJ-SAID, "*La cryptographie et ses principaux systèmes de références* ", Vol. 12, n°01, pp.173-192, 2002.

[23] M. RAMDHANI, " *problème de sécurité dans les réseaux capteurs avec prise en charge de l'énergie* ", Mémoire de magistère, Université SAAD DAHLAB DE BLIDA (Algerie), Novembre 2013.

[24] H. BOUMERZOUG, "*gestion de sécurité dans les réseaux de capteurs sans fil* ", Mémoire de magistère, Université de Québec a trois rivières, Octobre 2011.

[25] B. KEBIR, S. RAHMOUNI, "*Développement d'une application pour l'échange des messages sécurisé* ", Mémoire de fin d'études, Université de Abou Bakr Belkaid, Telemcén, (Algérie), Mai 2015.

[26] A. KARRAY, "*Conception, mise en œuvre et validation d'un environnement logiciel pour le calcul sécurisé sur une grille de cartes à puce de type Java* ", Thèse de Doctorat, Université BORDEAUX I, (France) ,10. Décembre. 2008.

[27] S. LAABIDI. "*Méthodologie de conception de composants intégrés protégés contre les attaques par corrélation soutenue* ". Thèse de Doctorat, l'École Nationale Supérieure des Mines de Saint-Étienne, (France), 19. Janvier. 2010.

[28] B. SAAD, "*intégration des problèmes de satisfaction distribués et sécurisés dans les systèmes d'aide à la décision à base de connaissance* ", Thèse de Doctorat, Université de PAUL VERLAINE – METZ, 10.Décembre.2012.

[29] A. R. KIHEL, "*système cathodique pour la transmission sécurisée de données* ", Mémoire de Magister, Université Mohamed Khider – Biskra, (Algérie), 26.novembre. 2016.

[30] M. BOUKHATEM, "*Application des techniques de cryptage pour le transmission sécurisé d'images MSG* ", Mémoire de magistère en électronique, Université Moloud Mammeri, Tezi-Oizou, (Algérie),11.Mars.2015.

[31] T. POURNIN, "*Implantation et optimisation des primitives cryptographiques*", Thèse de Doctorat, Université de Paris 7, (France), 25. Octobre. 2001.

[32] M. BRUNO, " *codage cryptologie et application* ", Presse polytechniques et universitaires romandes, Disponible sur <http//fribok.blogspot.com/>, 17.Mars.2017.

[33] L. BLOCH, C. WOLFHUGEL, " *sécurité informatique : principe et méthodes* ", Paris, 2007, ISBN : 2-212-12021-4 • ISBN 13 : 978-2-212-12021-9.

[34] M. AMOUD, "*Design et implémentation sur FPGA d'un algorithme DES",* Mémoire présenté à la Faculté des études supérieures, Université de Montréal, Décembre.2008.

[35] R. DUMONT, "*cryptographie de sécurité informatique* ", Mémoire de master, Université de Liége, 2010.

[36] V.S. BAGAD, I.A. DHOTRE, "*Networks and information Security* ", Edition 2009, Strictly According to the Revisted Syllabus of University of Pune (UOP)-2003 cours, ISBN 978-81-8431-570-7.

[37] K. LOUDON, *" Algorithms With C ": useful Techniques forme sorting to encryption,* 1er Edition, published by o'reilly Media,in the united states of America, 1999, 537 p, ISBN 978-1-56592-453-6.

[38] J.PETER KAPS, "*High speed FPGA architectures for the Data Encription Standard* ", Thèse de master on science, Worcester polytechnic institute, Mai 1998.

Conclusion générale

La nécessité de sécuriser la transmission de données augmente chaque jour et nécessite des dispositifs de chiffrement de données sécurisés pour préserver la confidentialité et l'authentification des échanges des applications critiques.

Ce travail se concentre sur l'algorithme de cryptage symétrique DES qui a une clé de 64 bits. L'intérêt de l'algorithme DES dans le domaine de la cryptographie n'est plus à démontrer, et son intégration sur circuit numérique (FPGA) est la cible d'un grand nombre de travaux,

Nous avons étudié le DES dans le but d'amélioration de génération de sous-clés pour avoir des sous-clés non redondants.

Notre étude nous a conduit à considérer les points suivants :

Nous avons d'abord fais un tour d'horizon rapide dans le domaine du cryptage et les différentes algorithmes existant. Ensuite, nous avons abordé l'algorithme DES par son aspect théorique et la problématique qui se pose au niveau de la génération de sous clés qui est la redondants des sous-clés.

Par la suite, nous avons proposé l'ajout d'une étape à la génération de sous clés qui réduit le problème de redondance de sous clés. Nous avons simulé le fonctionnement de notre approche sous le logiciel (MATLAB) que nous avons comparé avec le DES classique.

Les architectures réalisant la génération de sous clés du DES (amélioré et classique) ont été conçue puis décrite en utilisant le langage VHDL implémentée sur un circuit FPGA de la famille Spartan3 (XC3S4000L) et Virtex (XCV1000) de xilinx sous l'environnement ISE 9.2.

Les résultats de l'implémentation sur FPGA obtenus sont relativement satisfaisantes puisque les fréquences dans les deux programmes (classique et amélioré) sont les mêmes; L'utilisation de l'approche proposée a réduit la redondance dans la génération des sous-clés et elle a réduit la vulnérabilité à l'attaque, et elle a amélioré l'efficacité du cryptage. Les résultats présentés de la mis en œuvre fournissent un haut niveau d'assurance et d'exactitude.

Perspectives :

Comme travaux future :

- l'implémentation de notre nouvel algorithme pour des applications basées sur l'echange d'informations (images, texte ...etc).

-Utiliser d'autres versions plus performantes et actualisées que les versions utilisées : Spartan3 (XC3S4000L) et Virtex (XCV1000).

- L'amélioration de notre algorithme DES dans le but d'obtenir une meilleure sécurité, réduire le temps de cryptage total...etc.

- Réduire et modifier le séquenceur (machine d'état) pour avoir une fréquence élevée.

Voila quelques idées qui peuvent être prise en compte si l'on souhaite continuet dans le même axe de recherche. Ce domaine est loin d'être fini et ne constitue que le premier jalon.

Résumé

Depuis la création de l'algorithme cryptographique DES (Data Encryption Standard). Plusieurs contributions ont été faites pour améliorer sa sécurité et sa résistance aux différentes attaques. C'est un algorithme à clé symétrique dont la même clé est utilisée pour le cryptage ainsi que pour le décryptage. A base de la clé principale, un processus permet de générer différentes sous-clés pour chaque tour de cet algorithme. La sécurité du DES repose sur la génération des sous clés. Certaines sous clés sont faibles ou semi-faibles, en conséquence il est déconseillé de les utiliser pour le cryptage. Notre contribution consiste à ajouter une étape au processus de génération de sous-clés. Il s'agit de la conversion de bit paire / Impaire, permettant ainsi d'avoir des sous-clés non redondantes ce qui n'est pas le cas des contributions conventionnelles.

Mots clés: Cryptographie, DES, VHDL, FPGA, MATLAB.

Abstract

Since the creation of the DES (Data Encryption Standard) as a cryptographic algorithm; several contributions have been made to improve its security and resistance to different attacks. It is a symmetric key algorithm whose main key is used for encryption as well as for decryption. Based on the main key, a process can generate different sub-keys for each round of this algorithm. The security of the DES is based on the generation of sub keys. Some sub-keys are weak or semi-weak. Therefore it is advised against using them for encryption. Our contribution is to add a step to the sub-keys generation process. It is the odd bit / event bit conversion, thus allowing non-redundant subkeys, which is not the case for conventional contributions.

Keywords: Cryptography, DES, VHDL, FPGA, MATLAB.

ملخص

منذ إنشاء خوارزمية التشفير*DES* (تشفير البيانات الموحدة).اجريت العديد من المساهمات لتعزيز الأمن ومقاومة الهجمات. هذه الخوارزمية تستخدم نفس المفتاح للتشفير وفك التشفير. و استنادا إلى المفتاح الرئيسي، تجرى عملية توليد مختلف المفاتيح الفرعية لكل دورة من الخوارزمية. يستند أمن *DES* على توليد المفتاح الفرعي. بعض المفاتيح الفرعية ضعيفة أو شبه ضعيفة، نتيجة لذلك ينصح بعدم استخدامها للتشفير. مساهمتنا تتمثل في إضافة خطوة إلى عملية توليد المفاتيح الفرعية. تتعلق بتحويل بت زوجي/ فردي، و التي تسمح بتوليد مفاتيح فرعية غير متكررة، وهذا ليس هو الحال بالنسبة للمساهمات التقليدية.

الكلمات المفتاحية: الترميز ,MATLAB, FPGA, DES .