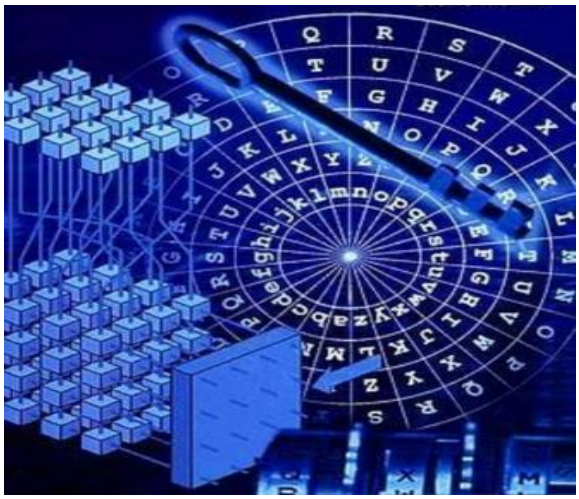
ÉCOLE NATIONALE SUPÉRIEURE D'ARTS ET MÉTIERS
UNIVERSITÉ HASSAN II DE CASABLANCA

Rapport Mini Projet

Méthode de cryptographie en langage VHDL :

Chiffrement de Hill



Réalisé par :

- AZGHOUDI Najwa (MSEI)
- TSOULI Iman (MSEI)
- SOUILI Oumaima (GEM)

Sous la supervision du Professeur :

- Mr. EL MOUMNI

Sommaire :

INTRODUCTION :

La cryptologie est une science très ancienne : les hommes ont toujours eu besoin de dissimuler des informations et de transmettre des messages en toute confidentialité. Le terme cryptologie vient du grec kruptos (κρυπτός) signifiant secret, caché et de logos (λογός) signifiant discours. La cryptologie est donc la science du secret. Elle regroupe la cryptographie et la cryptanalyse : la première a pour but de concevoir des systèmes visant à assurer la sécurité des communications sur un canal de communication public et la seconde vise à trouver des failles dans ces systèmes. La cryptographie est traditionnellement utilisée pour dissimuler des messages aux yeux de certains utilisateurs et le chiffrement des communications militaires a depuis l'Antiquité été une préoccupation majeure des diverses forces armées. Le chiffrement regroupe les techniques mises en œuvre pour brouiller la signification d'un message qui est matériellement visible. Le contenu du message ne doit alors être récupérable que par les personnes auxquelles le message est adressé. Le chiffrement fait appel à deux processus élémentaires de transformation du message pour satisfaire ces propriétés : — la substitution qui consiste à remplacer, sans en modifier l'ordre, les symboles d'un texte clair par d'autres symboles, — et la transposition qui repose sur le bouleversement de l'ordre des symboles (mais pas leur identité).

Il existe des systèmes de chiffrement relativement simples qui ont été utilisés de l'Antiquité (chiffrement de César ou scytale) jusqu'au début du XXe siècle (chiffrement de Vernam, chiffrement de Hill, machine Enigma)

Dans notre projet, on va étudier le chiffrement de Hill qui est une méthode robuste de cryptographie symétrique qui repose sur l'utilisation de matrices et de l'arithmétique modulaire.

En exploitant le langage de programmation VHDL (Hardware Description Language) qui est un langage de description matériel utilisé pour modéliser et concevoir des circuits électroniques, on va découvrir cette méthode de cryptographie, ses points forts et sa capacité à chiffrer des blocs de lettres simultanément.

Le chiffrement de Hill :

Le chiffrement de Hill est une méthode de cryptographie par substitution polygraphique qui utilise des opérations matricielles pour chiffrer et déchiffrer des messages. Il a été inventé par Lester S. Hill en 1929. Voici une explication détaillée du principe de ce chiffrement et des étapes de son application :

Principe de Base

1. Représentation des caractères : Chaque caractère du message est représenté par un nombre. Par exemple, en utilisant un alphabet de 26 lettres (A-Z), on peut attribuer à chaque lettre un nombre entre 0 et 25 (A = 0, B = 1, ..., Z = 25).

2. Utilisation de matrices : Le chiffrement de Hill utilise une matrice clé carrée K de taille $n \times n$. Cette matrice doit être inversible dans l'arithmétique modulaire, ce qui signifie que son déterminant doit être non nul et avoir un inverse modulaire.

3. Vecteurs de texte clair : Le message en clair est divisé en blocs de n caractères. Chaque bloc est représenté comme un vecteur colonne X.

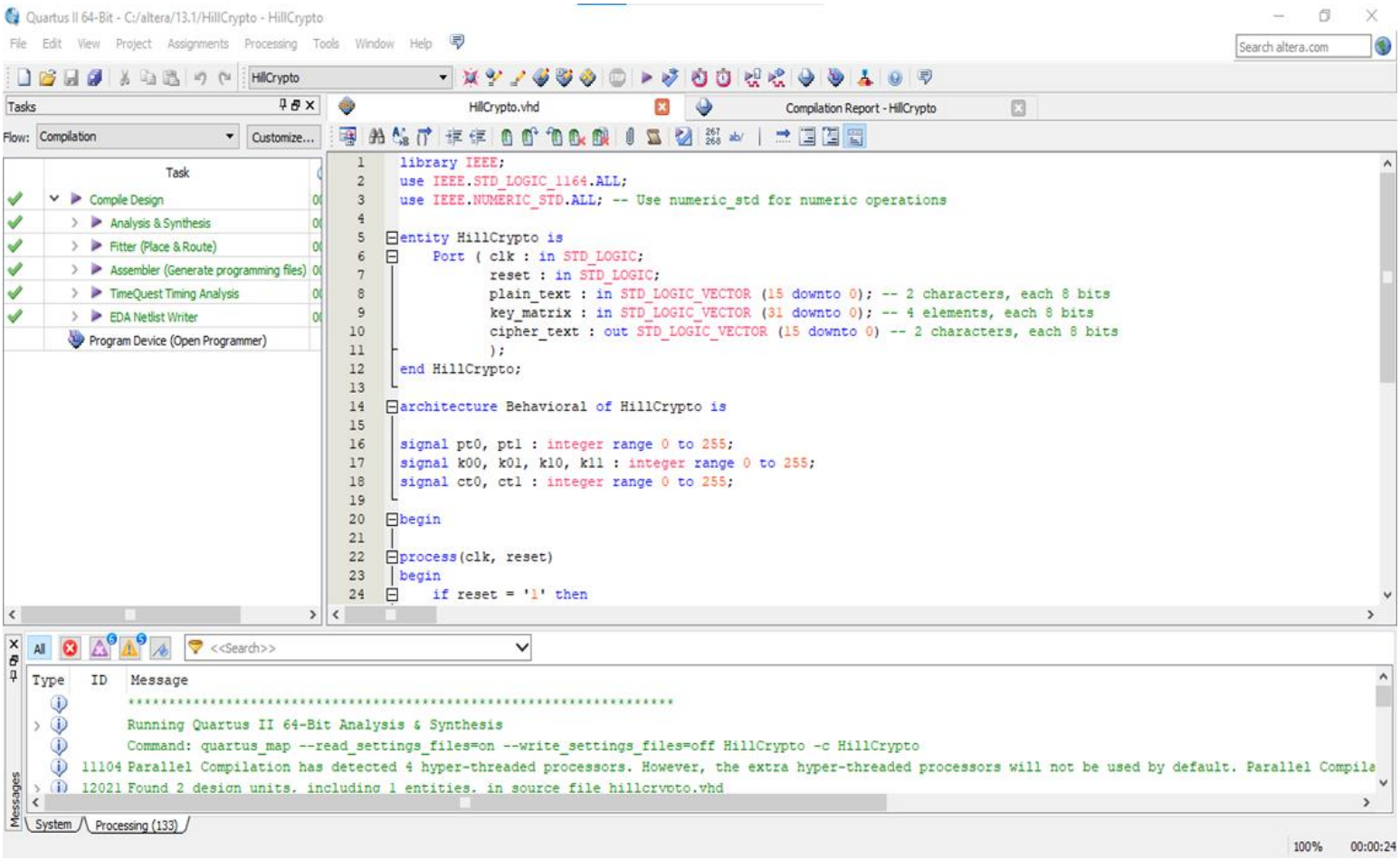
4. Chiffrement : Le vecteur X est multiplié par la matrice clé K pour obtenir le vecteur chiffré Y selon la relation :

$$Y = K \times X \pmod{26}$$

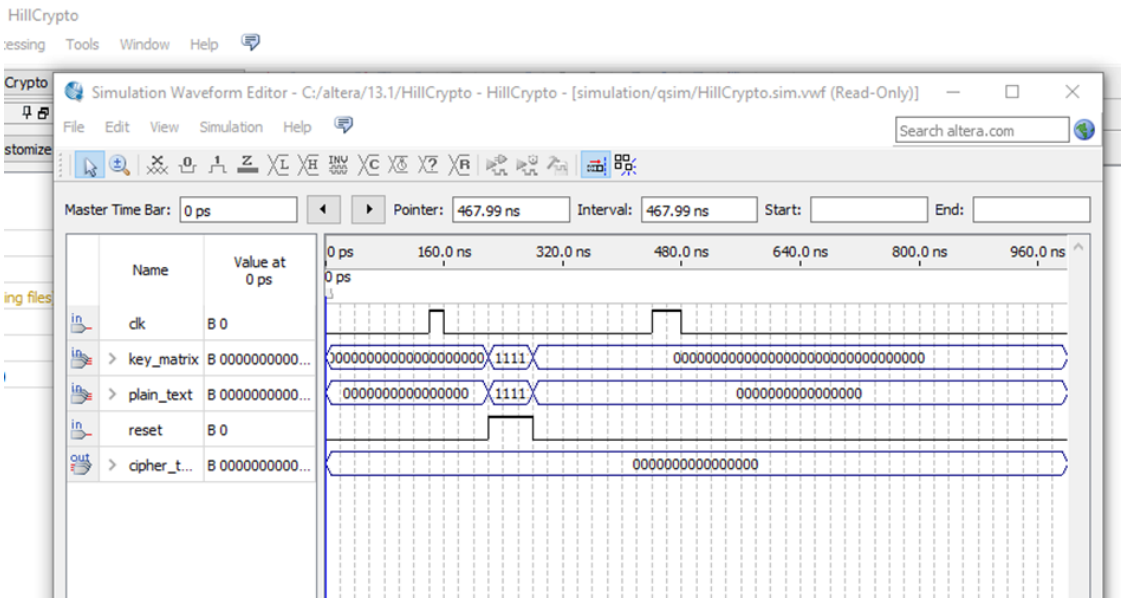
5. Déchiffrement : Pour déchiffrer le message, il faut utiliser l'inverse de la matrice clé K^{-1}. Le vecteur chiffré Y est multiplié par K^{-1} pour récupérer le vecteur en clair X :

$$X = K^{-1} \times Y \pmod{26}$$

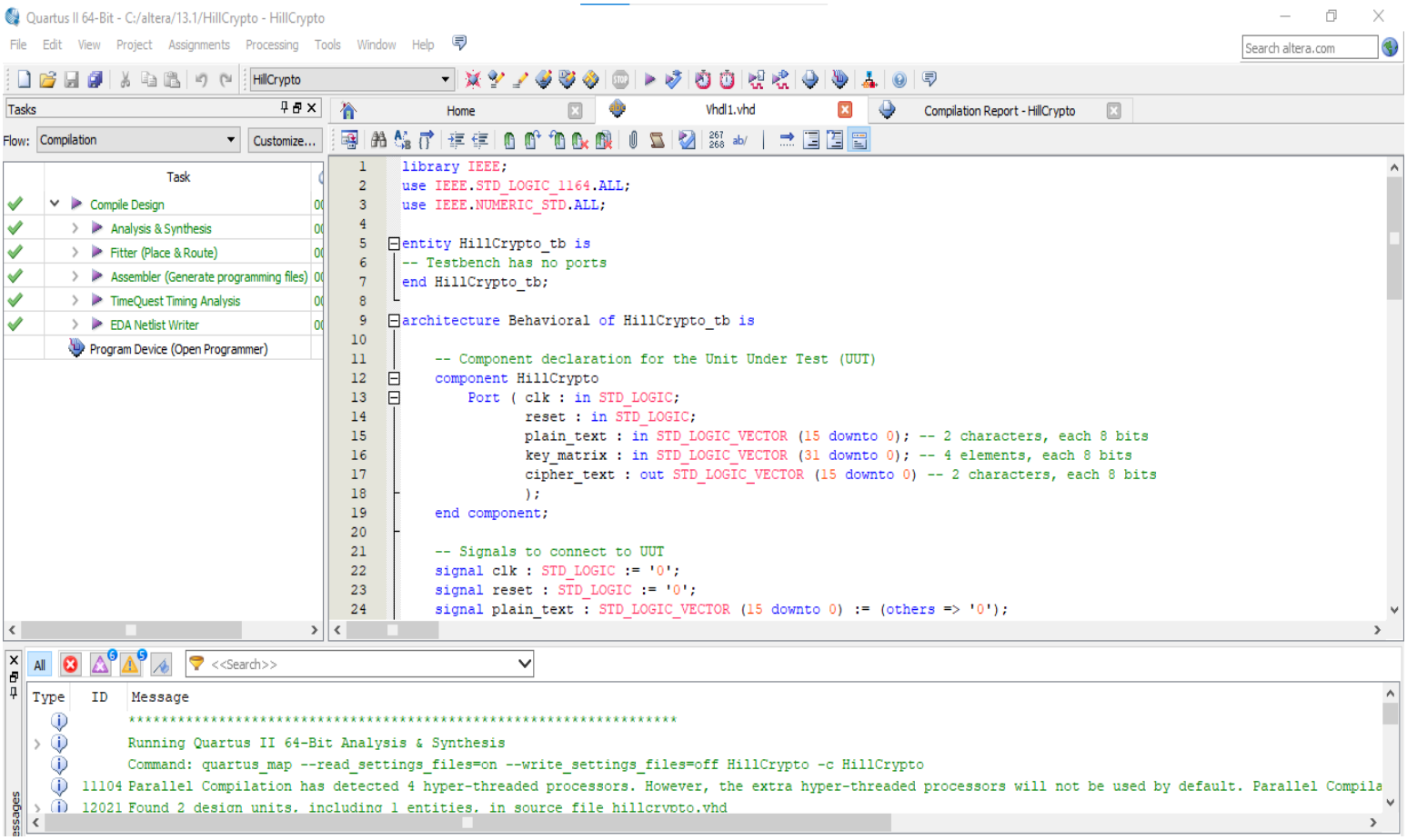
Compilation du code de chiffrement de Hill sur Quartus 13.1 en langage VHDL :



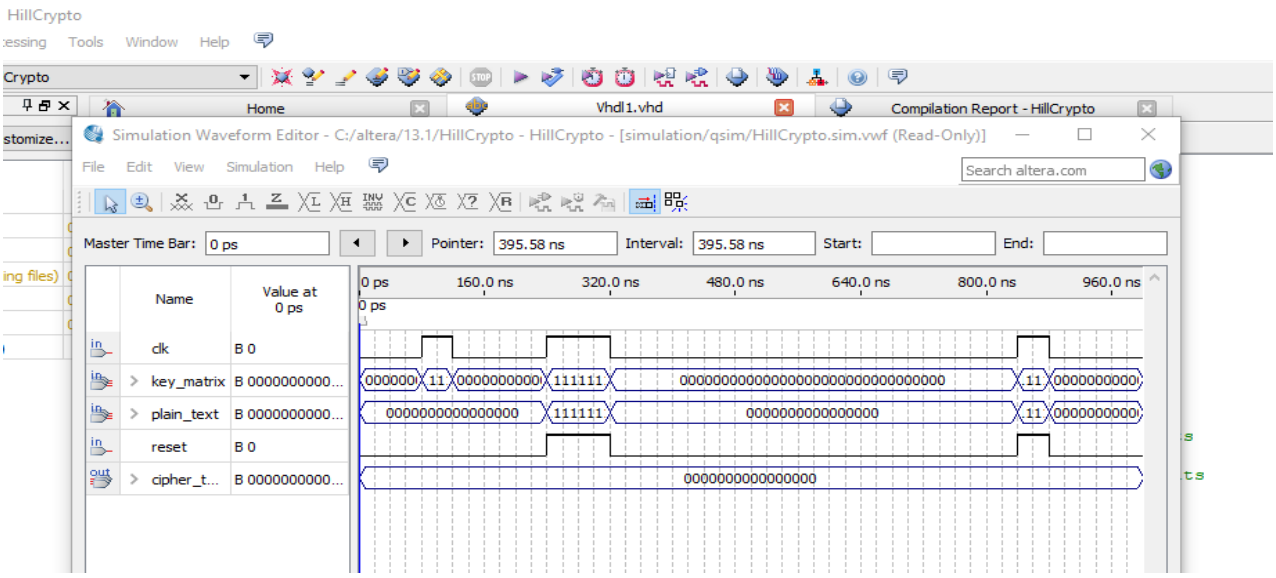
Simulation sur Quartus 13.1 :



Exemple d'application :



Simulation :



Explication :

1. Bibliothèques Utilisées :

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
```

   - Ce bloc de code inclut les bibliothèques IEEE nécessaires pour le design. STD_LOGIC_1164 est utilisé pour les types logiques standard et NUMERIC_STD pour les opérations arithmétiques sur les vecteurs binaires.

2. Déclaration de l'Entité :

```
entity HillCrypto_tb is
-- Testbench has no ports
end HillCrypto_tb;
```

   - Cette partie déclare l'entité HillCrypto_tb, qui est notre testbench. Un testbench n'a pas de ports externes car il sert uniquement à tester une autre entité.

3. Architecture de l'Entité :

```
architecture Behavioral of HillCrypto_tb is
```

   - L'architecture "Behavioral" de l'entité HillCrypto_tb est définie ici. Tout le code de test se trouve dans cette section.

4. Déclaration du Composant (UUT - Unit Under Test)

```
-- Component declaration for the Unit Under Test (UUT)
component HillCrypto
    Port ( clk : in STD_LOGIC;
           reset : in STD_LOGIC;
           plain_text : in STD_LOGIC_VECTOR (15 downto 0); -- 2 characters, each 8 bits
           key_matrix : in STD_LOGIC_VECTOR (31 downto 0); -- 4 elements, each 8 bits
           cipher_text : out STD_LOGIC_VECTOR (15 downto 0) -- 2 characters, each 8 bits
           );
end component;
```

- Cette partie décrit le module de chiffrement Hill (HillCrypto) que nous testons. Il a cinq ports :
  - Clk : l'horloge
  - Reset : signal de réinitialisation
  - Plain_text : texte clair d'entrée (16 bits)
  - Key_matrix : matrice de clé (32 bits)
  - Cipher_text : texte chiffré de sortie (16 bits)

5. Déclaration des Signaux :

```
-- Signals to connect to UUT
signal clk : STD_LOGIC := '0';
signal reset : STD_LOGIC := '0';
signal plain_text : STD_LOGIC_VECTOR (15 downto 0) := (others => '0');
signal key_matrix : STD_LOGIC_VECTOR (31 downto 0) := (others => '0');
signal cipher_text : STD_LOGIC_VECTOR (15 downto 0);
```

6. Définition de la Période de l'Horloge :

```
-- Clock period definition
constant clk_period : time := 10 ns;
```

- Cette constante définit la période de l'horloge à 10 nanosecondes.

7. Instanciation du UUT :

```
-- Instantiate the Unit Under Test (UUT)
uut: HillCrypto
    Port map (
        clk => clk,
        reset => reset,
        plain_text => plain_text,
        key_matrix => key_matrix,
        cipher_text => cipher_text
    );
```

- Le module HillCrypto est instancié et ses ports sont connectés aux signaux correspondants.

8. Processus d'Horloge :

```
-- Clock process definitions
clk_process :process
begin
    clk <= '0';
    wait for clk_period/2;
    clk <= '1';
    wait for clk_period/2;
end process;
```

- Ce processus génère un signal d'horloge avec une période définie (10 ns).

9. Processus de Stimulation (Stimulus Process) :

```
-- Stimulus process
stim_proc: process
begin
    -- Hold reset state for 20 ns.
    reset <= '1';
    wait for 20 ns;
    reset <= '0';

    -- Test case 1
    -- Example plaintext "HI" (ASCII: H = 72, I = 73) and key matrix
    -- Key matrix elements: k00 = 3, k01 = 3, k10 = 2, k11 = 5
    plain_text <= "0100100001001001";  -- "HI" -> 72, 73
    key_matrix <= "00000011000000110000001000000101";  -- 3, 3, 2, 5
    wait for clk_period;

    -- Wait for 10 ns to check the result
    wait for 10 ns;
```

```
        -- Check the output
        assert (cipher_text = "0111100101001111") -- Expected ciphertext for "HI" with given key matrix
        report "Test Case 1 Failed" severity error;

        -- Test case 2
        -- Add additional test cases as needed
        -- ...

        wait;
    end process;

end Behavioral;
```

- Ce processus applique des stimuli au UUT pour vérifier son comportement :

- Le signal de réinitialisation (reset) est activé pendant 20 ns, puis désactivé.
- Le premier cas de test applique un texte clair ("HI" codé en ASCII) et une matrice de clé au UUT.
- Après un cycle d'horloge, le texte chiffré produit est vérifié par une instruction assert. Si la sortie ne correspond pas à la valeur attendue, un message d'erreur est généré.

CONCLUSION :

Ce code VHDL teste un exemple de chiffrement Hill en appliquant des signaux d'entrée spécifiques et en vérifiant que les sorties sont correctes. Le processus d'horloge génère un signal d'horloge continu, tandis que le processus de stimulation applique les signaux d'entrée et vérifie les résultats.

La démarche mathématique :

La cryptographie de Hill utilise des opérations matricielles pour chiffrer des blocs de texte. La démarche mathématique de base pour chiffrer un texte en utilisant la méthode de chiffrement de Hill peut être décrite comme suit :

1. Représentation des données :
   - Texte en clair (Plaintext) : Le texte en clair est divisé en blocs de longueur n. Chaque bloc est représenté comme un vecteur P de taille n.
   - Clé (Key) : La clé est représentée par une matrice K de taille $n \times n$. La clé doit être une matrice carrée inversible (c'est-à-dire que son déterminant ne doit pas être nul et qu'elle doit avoir un inverse modulo 26).
   - Texte chiffré (Ciphertext) : Le texte chiffré est obtenu en multipliant la matrice de clé K par le vecteur de texte en clair P.

2. Procédure de chiffrement :

Soit PP le vecteur de texte en clair de taille n, et K la matrice de clé de taille $n \times n$.

La formule mathématique pour obtenir le vecteur de texte chiffré C est :

$C = K \cdot P \pmod{26}$

3. Exemple avec des valeurs spécifiques :

Supposons que $n=2$ et que nous voulons chiffrer le texte "HI" avec la clé donnée.

Texte en clair :

Le texte "HI" est représenté en ASCII :

- H correspond à 72
- I correspond à 73

Cependant, dans le chiffrement de Hill, nous utilisons une base de 26 (A-Z). Donc :

- $H=7$ (car 'A' est 0, 'B' est 1, ..., 'H' est 7)
- $I=8$

Le vecteur de texte en clair P est donc : $P = \begin{pmatrix} 7 \\ 8 \end{pmatrix}$

Matrice de clé :

La matrice de clé K

$$K = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix}$$

Calcul du texte chiffré :

Multiplions la matrice de clé par le vecteur de texte en clair : $C=K \cdot P$

$$C = \begin{pmatrix} 3 & 3 \\ 2 & 5 \end{pmatrix} \cdot \begin{pmatrix} 7 \\ 8 \end{pmatrix} = \begin{pmatrix} 3.7 + 3.8 \\ 2.7 + 5.8 \end{pmatrix} = \begin{pmatrix} 45 \\ 54 \end{pmatrix}$$

Pour obtenir les valeurs dans la plage de 0 à 25 (modulo 26) :
45 (mod 26) =19
54 (mod 26) =2

Le vecteur de texte chiffré *C* est donc : $C = \begin{pmatrix} 19 \\ 2 \end{pmatrix}$

Conversion en caractères :

- 19 correspond à ‘T’(puisque 'A' est 0, 'B' est 1, ..., ‘T’est 19)
- 2 correspond à 'C'

Donc, le texte chiffré pour "HI" avec la matrice de clé donnée est "TC".

1. Vérification dans le testbench :

Dans le code VHDL, cette opération est représentée en binaire. Le texte clair "HI" en binaire est "0100100001001001" et la matrice de clé en binaire est "00000011000000110000001000000101". Le code vérifie que le texte chiffré obtenu correspond à la valeur attendue en binaire, qui serait "0111100101001111" pour "TC".
En conclusion, le processus de chiffrement de Hill implique la conversion des caractères en nombres, l'utilisation d'une multiplication matricielle pour transformer le texte clair en texte chiffré, puis la conversion de ce texte chiffré en caractères.

CONCLUSION :

Le chiffrement de Hill présente une méthode cryptographique intéressante, offrant une combinaison de simplicité et d'efficacité pour sécuriser les données. Son importance réside dans sa capacité à fournir une couche de sécurité raisonnable pour le transfert de données sensibles, tout en étant relativement simple à mettre en œuvre. En effet, son processus de chiffrement repose sur des opérations matricielles basiques, ce qui le rend accessible même aux personnes ayant des connaissances limitées en cryptographie. La faisabilité de son implémentation en VHDL est élevée, car le langage offre les outils nécessaires pour réaliser les opérations matricielles requises et permet une simulation et une vérification efficaces avant le déploiement dans des systèmes réels. Ainsi, le chiffrement de Hill combiné à la flexibilité et à l'efficacité de VHDL constitue une solution viable pour sécuriser les données dans diverses applications. Enfin, VHDL facilite l'intégration du chiffrement de Hill dans des systèmes plus vastes, offrant ainsi une solution sécurisée pour diverses applications. En somme, l'efficacité, la flexibilité et la capacité de modélisation des opérations mathématiques complexes font de VHDL un outil inestimable pour la conception et l'implémentation du chiffrement de Hill, garantissant la sécurité des données dans un large éventail de contextes.

REMERCIEMENT :

A la fin de notre cours, nous venons d’achever une formation extrêmement enrichissante sur les composants programmables en tenant compte à l’immense importance du langage VHDL. Nous en profitons pour exprimer notre profonde gratitude et nos sincères remerciements à notre professeur Mr. EL MOUMNI pour l'accompagnement et le soutien précieux qu’il nous a apporté tout au long de notre formation en programmation VHDL, afin d’apprendre les concepts et les importants outils du langage VHDL sur le logiciel Quartus. Grace à sa patience et sa passion pour l'enseignement qui ont créé un environnement d'apprentissage stimulant, enrichissant et motivant, les travaux pratiques que nous avons réalisés sous sa supervision ont été particulièrement bénéfiques, nous permettant de mettre en pratique nos connaissances. Effectivement, nous avons réussi de finaliser ce mini projet qui reflète notre compréhension de la logique d’exploitation de ce langage de programmation robuste à l’aide de ses conseils avisés et à son encadrement. Enfin, Nous sommes honorés d'avoir pu bénéficier de son précieuse expertise, qui nous a permis de renforcer nos connaissances théoriques et développer des compétences pratiques essentielles pour notre avenir professionnel dans le domaine de l'ingénierie électronique.