TP 1 : INITIATION A LA PROGRAMMATION VHDL

PREMIERE APPROCHE DU LOGICIEL SIMULATEUR DE VHDL Altera Max+Plus II

Objectif : L'objectif primordial de ce premier TP est de se familiariser avec un outil de programmation des circuits logiques (numériques) programmables. Nous nous intéressons à un outil de la famille **Altera** : logiciel **Max+plus II**.

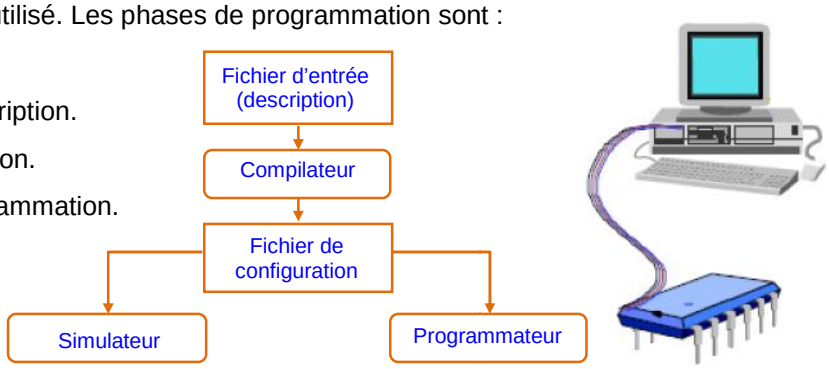
I- Synthèse : Circuits logiques programmables et outils de conception

Un circuit logique programmable se définit comme un composant électronique standard contenant des modules de logique combinatoires et séquentiels, dont les interconnexions internes sont désignées par programmation (interconnexions entre différentes cellules logiques ; voir cours). Il peut être configuré et reconfiguré par l'utilisateur, pour la réalisation de diverses fonctions logiques (numériques).

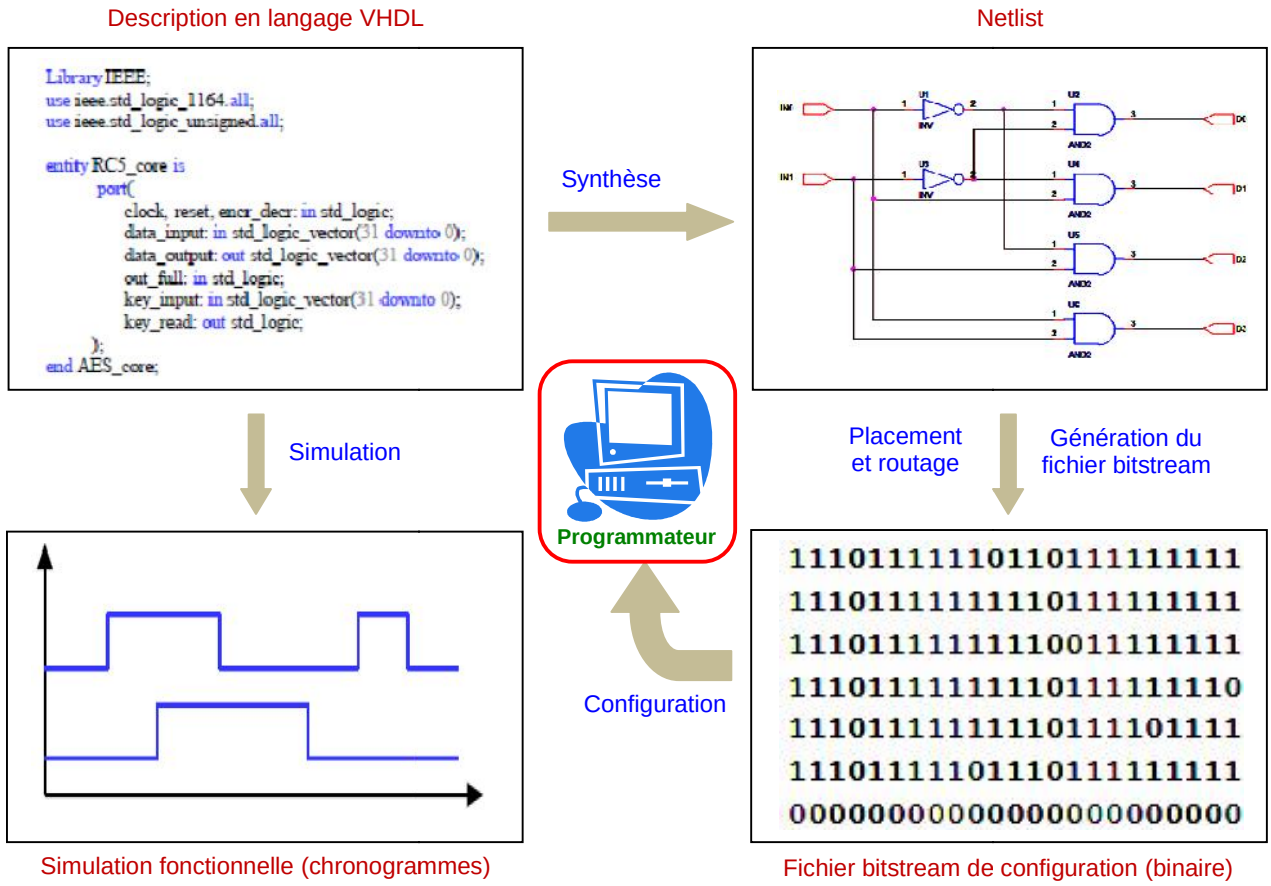
Plusieurs solutions sont possibles pour programmer un circuit logique programmable. Elles dépendent essentiellement de l'outil de développement (outil de conception) utilisé. Les phases de programmation sont :

- Saisie de la description du circuit logique.
- Compilation pour la vérification de la description.
- Simulation, synthèse (Netliste) et vérification.
- Insertion du circuit à programmer et programmation.

La figure ci-contre résume les différentes étapes de programmation d'un tel circuit.



Le synthétiseur de l'outil de conception assistée par ordinateur (CAO) génère dans un premier temps une **Netlist** qui décrit la connectivité de l'architecture. Puis l'outil de placement-routage place de façon optimale tous les composants et effectue le routage entre les différentes cellules logiques.



VHDL (**V**ery high-speed integrated circuits **H**ardware **D**escription **L**anguage) est un langage de description de matériel, destiné à représenter le comportement ainsi que l'architecture d'un système électronique logique ; en particulier des circuits numériques programmables (ASIC, FPGA, CPLD,…etc.).

Actuellement, le marché des circuits programmables est toujours en pleine croissance. Deux fournisseurs particuliers se disputent principalement ce marché : **Xilinx** et **Altera** (environ 80% du marché entre eux deux).

Par conséquence, les principaux fabricants de circuits logiques programmables proposent une version gratuite mais limitée de leurs outils de simulation et de synthèse. Pour débuter en VHDL, nous citons quelques environnements de programmation permettant la saisie d'un fichier VHDL.

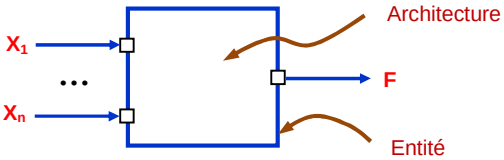
- Intel / Altera : **Max+plus II**, Quartus II, Modelsm,…etc.
- Xilinx : ISE,...etc.

II- Structure générale d'une description VHDL

D'une manière générale, un système numérique est vu comme une "boîte noire". En VHDL, la "boîte noire" est nommée **entité** (entity) et une entité doit toujours être associée avec au moins une description de son contenu, de son implémentation : c'est l'**architecture**. Une description VHDL est donc composée de deux parties indissociables :

- L'**entité** (entity) : elle décrit l'extérieur l'interface du composant et permet ainsi de définir les entrées/sorties. Pour déclarer l'entité on donne un nom et on précise la liste des différents signaux d'entrés/sorties (ports d'E/S).
- L'**architecture** (architecture) : décrit l'intérieur du composant. Elle contient les instructions VHDL permettant de réaliser le fonctionnement attendu. Elle comporte une partie de déclaration et un corps de programme.

La structure générale d'un programme en VHDL est la suivante :



déclaration des bibliothèques IEEE

```
library ieee;
use ieee.std_logic_1164.all;
        .......
```

```
-- Pour écrire des commentaires
```

déclaration des entrées/sorties

```
entity NOM_ENTITE is
 port(
        .......
    );
end NOM_ENTITE ;
```

Nom de l'entité

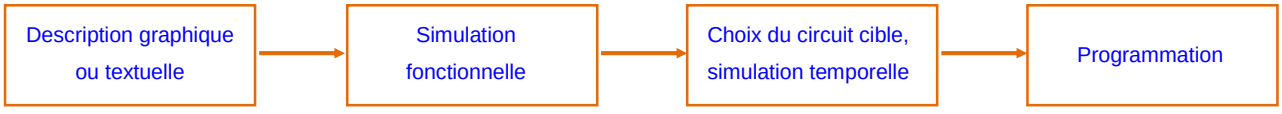
déclaration de l'architecture

```
architecture NOM_ARCHITECTURE of NOM_ENTITE is
 begin
        .......
 end NOM_ARCHITECTURE ;
```

Nom de l'architecture

III- Présentation du logiciel

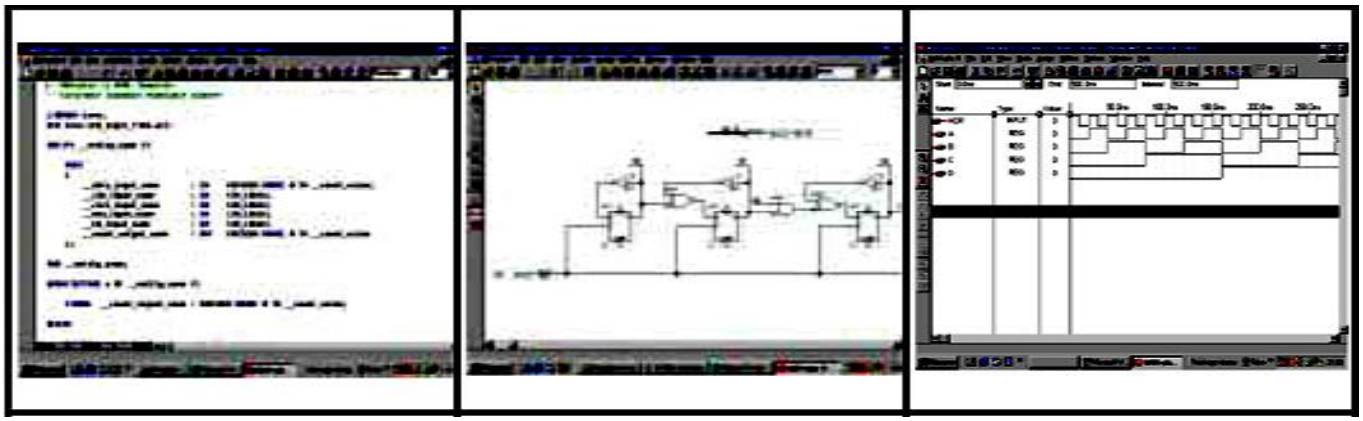
Le logiciel **Max+plus II** permet entre autres, la description d'un projet (système numérique), sa compilation, sa simulation logique et temporelle, son analyse temporelle et la programmation d'un circuit cible (CPLD ou FPGA).



La description du système numérique (logique) peut être faite à l'aide d'une des entrées suivantes :

- Editeur de texte : pour l'utilisation du langage VHDL ou VERILOG.
- Editeur graphique : permet d'utiliser les composants prédéfinis des bibliothèques fournies par le logiciel.
- Editeur de chronogrammes : avec lequel on représentera l'évolution temporelle et celle attendue des sorties.

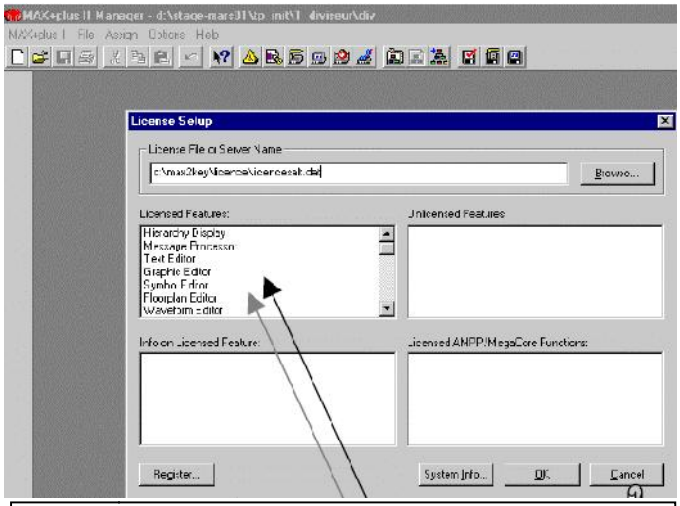
Editeur de texte | Editeur graphique | Editeur de chronogrammes



A chaque description est associé un symbole graphique du composant ainsi réalisé. L'éditeur graphique permettra alors de relier éventuellement ces composants les uns aux autres. Chaque sous-ensemble puis le système global est ensuite compilé, puis simulé par le simulateur logique, puis analysé et envoyé vers le circuit cible via le programmateur.

III-1- Installation du logiciel Max+plus II

Pour installer Max+plus II, télécharger depuis le serveur Altéra et suivre les instructions pour une installation complète (Full installation). Installer le fichier licence. Ce dernier est un fichier texte récupéré par e-mail, auprès du service Altera moyennant la fourniture d'un numéro d'identification de disc dure. Il faut donc changer l'extension du fichier (**license.dat**). Copier ce fichier dans le **C:** de votre PC et déclarer ensuite le chemin dans le logiciel (**Option/License setup/Browse**).



Toutes les possibilités du logiciel sont utilisables

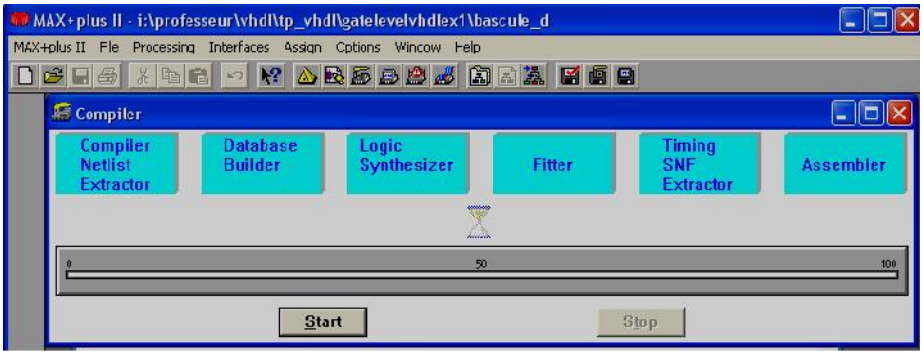
L'environnement de démarrage du logiciel Max+plus II est le suivant :

La création d'une description VHDL sous Max+plus II nécessite la génération, par le logiciel, d'un nombre important de fichiers. Tous les fichiers relatifs à une description portent le même nom, mais une extension différente.

III-2- Début d'un projet

Pour débuter un projet, il faut suivre les 4 étapes suivantes :

- **Etape de conception** : consiste à la création et l'activation d'un nouveau projet. Il faut sélectionner d'abord un nom de projet et une directory ou seront stockés les fichiers (pour nos TP, **C:\Max2work\Nom de projet**).
  - ✓ Créer un nouveau fichier ; sous l'éditeur de texte (**Text Editor file**).
  - ✓ Sauvegarder le fichier (**Save as**) ; avec le même nom de l'entité du programme VHDL et l'extension "**.vhd**".
  - ✓ Activer le projet (**Set projet to current file**). En effet, le logiciel ne traite qu'un projet à la fois ; il faudra vérifier que le projet en cours de traitement est bien celui désiré.
- **Etape de saisie du code VHDL** : une fois la première étape terminée, saisissez le code de description VHDL.
  - ✓ Déclaration des bibliothèques.
  - ✓ Déclaration de l'entité.
  - ✓ Déclaration de l'architecture.
- **Etape de compilation** : une fois la description faite, il faut vérifier que cette description est correcte et pour cela on utilise la phase de compilation (**Max+plus II → Compiler**). Vous avez à ce moment, accès à cette fenêtre :

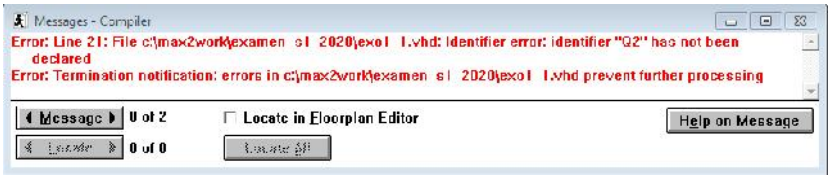


  - ✓ S'il y a des erreurs il faut les corrigées.
  - ✓ Lorsque la compilation passe sans erreurs, il est alors possible de vérifier le comportement logique du circuit.

Remarque :

Pendant l'étape de compilation, on distingue trois types de couleurs des messages affichées :

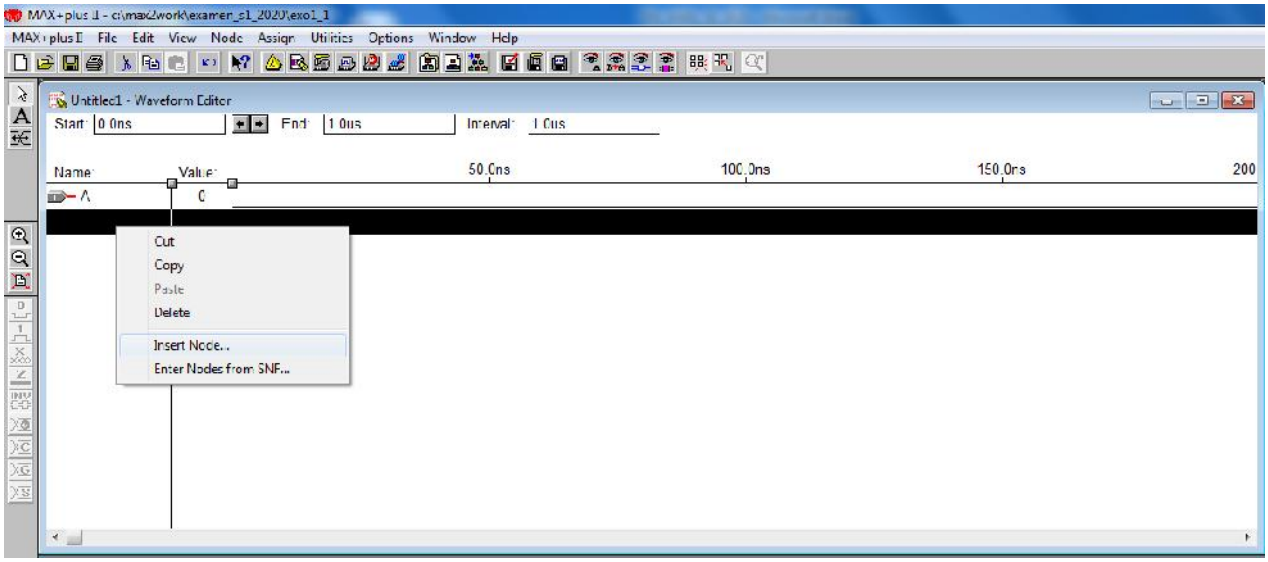
- **Messages de couleur rouge** : erreurs obligatoires à corriger (erreurs de syntaxe ou de synthèse). Dans ce cas, le code VHDL ne peut pas être compilé et ne peut pas être synthétisé (code VHDL non exécutable).



- **Messages de couleur bleu** : erreurs non obligatoires à corriger (erreurs d'avertissement ou des "warnings"). Dans ce cas, le code VHDL peut être compilé et peut être synthétisé (code VHDL exécutable).

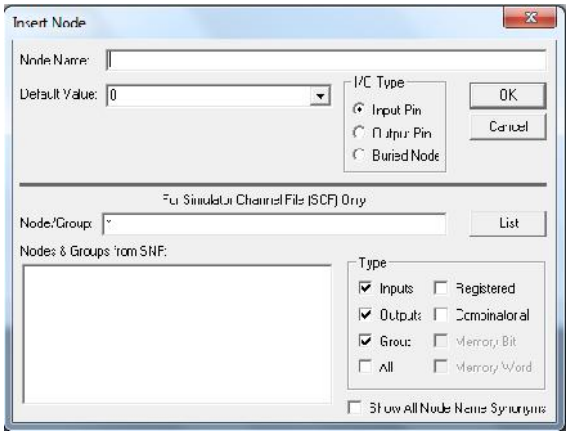
- **Messages de couleur verte uniquement** : pas d'erreurs. Le code VHDL est parfaitement exécutable.

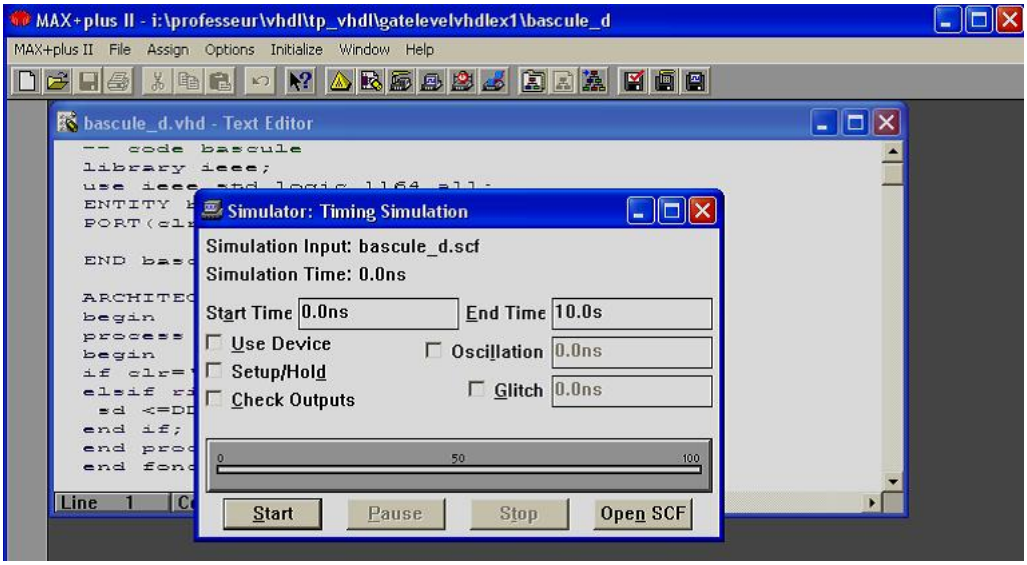
- **Etape de vérification** : comporte la simulation de type **Waveform Editor**, pour visualiser les chronogrammes.



Sur le fichier de type Waveform, nous affectons des valeurs pour les entrées de l'entité et on examine les résultats des sorties.

- ✓ Sauvegarder le fichier sous le même nom de l'entité avec l'extension "**.scf**".
- ✓ Sur la fenêtre affichée, avec la souris, faite entrer les nœuds (**Insert Node**) pour ajouter les signaux voulus.
- ✓ Configurer le temps de simulation des signaux (**File → End Time**).
- ✓ Configurer le pas de visualisation des signaux (**Options → Grid Size**).
- ✓ Entrer des valeurs de votre choix pour les entrées déclarés.
- ✓ Cliquer sur l'icône **Simuler** pour la simulation, puis cliquer sur **Start** et observer les chronogrammes (à l'aide du bouton **OpenSNF**) et vérifier vos résultats.





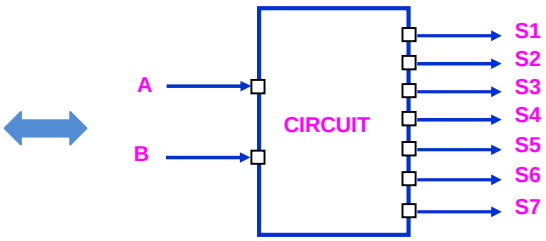
IV- Exercice d'application

Soit le code de description VHDL du circuit numérique suivant :

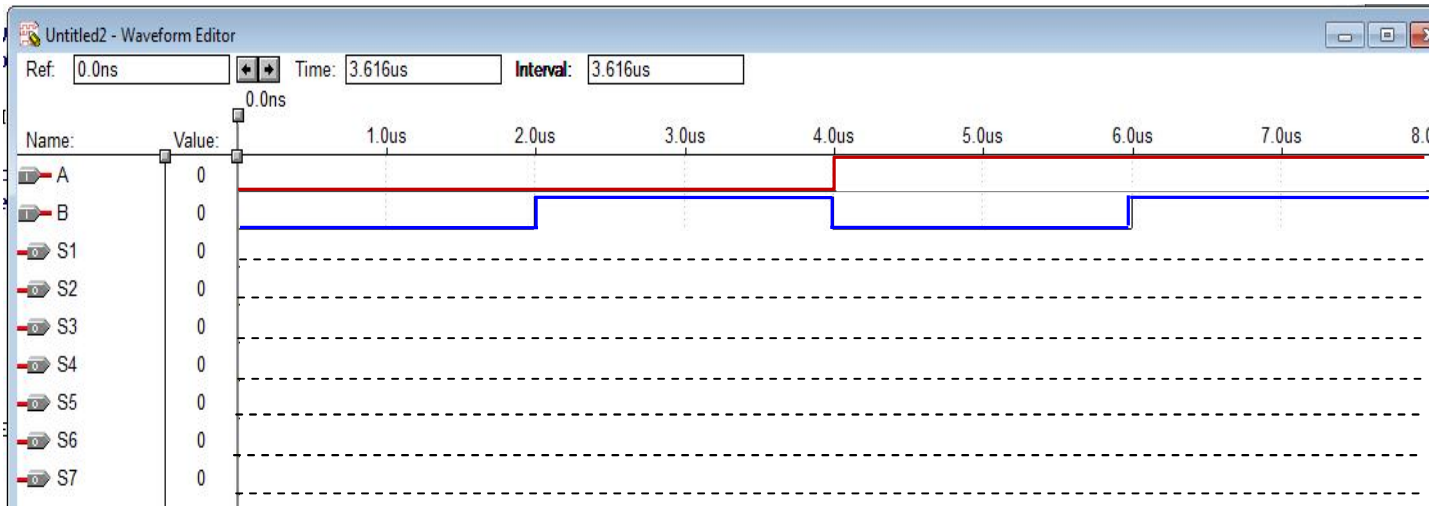
```
library ieee;
use ieee.std_logic_1164.all;

entity CIRCUIT is
    port (A,B :in std_logic;
          S1,S2,S3,S4,S5,S6,S7:out std_logic);
end CIRCUIT;

architecture DESCRIPTION of CIRCUIT is
    begin
      S1 <= A and B;
      S2 <= A or B;
      S3 <= A xor B;
      S4 <= not A;
      S5 <= A nand B;
      S6 <= A nor B;
      S7 <= not(A xor B);
end DESCRIPTION;
```



- Faire les étapes mentionnées au paragraphe III-2 précédent : Saisir le code VHDL du circuit donné, sauvegarder, compiler, corriger les erreurs si nécessaire, simuler et vérifier les résultats via Waveform editor ?
- Tracer les chronogrammes des sorties pour le cas ci-dessous (simulation de 0 à 8 us avec un pas de 1 us) ?



- Répondez aux questions dans le rapport d'évaluation donné à la salle des TP.

Présenté par : Dr. S. ABADLI.