Advanced Logic Synthesis for Electronics
www.ALSE.fr

# SystemVerilog for Design & Verification

## Formation de 1+3 jours

DOULOS

## Présentation

**SystemVerilog** (IEEE 1800™) est un « super-langage » basé sur le langage de description matériel **Verilog** qu'il étend sur de nombreux aspects pour en faire le langage universel du présent et du futur, aussi bien pour la conception que pour la vérification des circuits et systèmes numériques. Pour la conception (synthèse, RTL), les améliorations sont conséquentes. Mais c'est surtout dans les domaines de l'Architecture, la Modélisation et la Vérification de systèmes complexes que ce langage montre toute sa puissance (Types de données, Objets et Classes, Assertions SVA, Couverture fonctionnelle, Cover-driven Constrained Random, etc).

***SystemVerilog for Design & Verification*** est une Formation *dense* et *intense* qui permet d'acquérir toutes les compétences pour utiliser ce langage avec profit dans le contexte d'une utilisation de Conception et Vérification. Elle est nécessaire et suffisante pour un usage de **Conception Avancée** (Conception et Vérification + Modélisation). De plus , elle offre les bases indispensables aux ingénieurs de Vérification qui souhaiteraient aborder ensuite les compléments : Classes de Vérification et Méthodologie **UVM**.

Grâce au premier jour de mise à niveau en **Verilog**, elle est adaptée aux Ingénieurs qui ne maîtrisent que le langage VHDL.

Les participants peuvent utiliser au choix différents outils de Conception, Synthèse et Simulation (FPGA ou ASIC) durant les exercices pratiques qui occupent environ 50 % du temps de la formation. Ce sont des éléments-clés du succès de ces formations, *indispensables* pour un *réel* apprentissage. Ils assurent également un contrôle continu de l'acquisition des compétences.

Les Instructeurs ALSE sont aussi et surtout des **Experts** en Conception qui utilisent à journée entière les langages qu'ils enseignent pour concevoir et vérifier des IPs et des systèmes complexes. Ils savent partager leur forte expérience et leur savoir-faire avec passion, et sont particulièrement appréciés des participants.

| | |
|---|---|
| **Lieu** | Enseignement en ligne par un Instructeur qui dispense le cours interactivement, répond aux questions en temps réel, et surveille ensuite la bonne exécution des exercices en fournissant l'aide individuelle éventuellement nécessaire. |
| **Matériel** | Requiert un PC Windows ou Linux (ou Mac) avec un écran confortable pour les exercices.<br>Pour suivre le cours : soit une salle équipée en vidéo conférence, soit un accès stable à Internet et un micro (+ casque si possible).<br>Un environnement calme et non perturbé est indispensable. |
| **Exercices** | Tous les exercices pourront être exécutés dans le cloud (pas d'outils à installer). |

## Objectifs pédagogiques

- Maîtriser très rapidement le langage **Verilog** RTL en tirant profit des acquis VHDL.
- Comprendre les évolutions et les besoins qui ont conduit au Langage SystemVerilog.
- Apprendre et maîtriser l'**ensemble du langage SystemVerilog** hormis la partie objet (classes)
- Maîtriser la **Vérification par Assertions** (ABV) et se préparer aux **outils formels**.
- Comprendre et mettre en pratique la **Couverture Fonctionnelle**.
- Savoir mettre en œuvre le **Random-Contraint** ainsi que le **Cover-driven**.
- Savoir utiliser efficacement le langage pour la **Conception** et pour la **Vérification**.
- Évoluer des techniques de tests unitaires vers des méthodes plus sophistiquées permises par les nombreuses extensions du langage.

Les responsables de groupes pourront préparer efficacement les transitions méthodologiques.

© ALSE SAS - RCS DijonB 392 822 763 - SIRET 392 822 763 00034 - Code APE 7112B - TVA FR72392822763

## Qu'apprendrez vous ?

Le cours est structuré en différentes sections :

- ***Verilog Primer*** enseigne rapidement et efficacement le langage *Verilog*.
- ***Fundamentals of SystemVerilog for Design*** apprend à utiliser SystemVerilog pour la conception RTL (synthèse), la vérification unitaire et les modèles comportementaux simples.
- ***SystemVerilog Assertions*** enseigne la partie du langage qui est dédiée aux différents layers des Assertions, et permet d'en tirer avantage pour construire des modèles et des règles de vérification.
- ***SystemVerilog Verification*** montre comment utiliser le **Random contraint** et la **Couverture Fonctionnelle** pour adresser les challenges de la vérification des designs actuels dont la complexité exige des bancs de test et des modèles sophistiqués.

## Connaissances requises

Une connaissance préalable sérieuse du langage VHDL est indispensable.

# Programme [1]

## Le Langage Verilog (Jour 1)

### Différences entre VHDL et Verilog

◆ "Philosophie", ligne rouge, Typing fort, Déterminisme ◆ Abstraction des données ◆ Nets & Registers
◆ Structure du Langage – architecture, packages, configurations, fichiers ◆ Identifieurs ◆ Output ports
◆ Les Wires Implicits ◆ Arrays ◆ Aggrégats ◆ Signedness ◆ Opérateurs ◆ Signal vs variables / nets
◆ Process vs initial / always ◆ if, case, loop : différences ◆ E/S Fichiers ◆ Noms Hiérarchiques

### Les Bases du Langage Verilog

◆ Modules et ports ◆ Assignations Continues ◆ Commentaires ◆ Noms ◆ Nets et strengths
◆ Hiérarchie du Design ◆ Module instances ◆ Primitive instances ◆ Text Fixtures ◆ $monitor
◆ Blocs Initial ◆ Logic values ◆ Vecteurs ◆ Registers ◆ Nombres ◆ Formatage des sorties
◆ Timescale ◆ Blocs Always ◆ $stop et $finish ◆ Utilisation correcte des nets et des variables

### Logique Combinatoire

◆ Event control ◆ Instructions If ◆ Begin-end, Assignation Incomplète et latches ◆ Unknown et don't care
◆ Opérateur Conditionnel ◆ Tristates ◆ Case, casez et casex ◆ full_case & parallel_case
◆ Les boucles for, repeat, while et forever ◆ Entiers ◆ Blocs Self-disabling ◆ Synthèse de Combinatoire

### Logique Séquentielle

◆ Synthèse des flip-flops & latches ◆ Eviter les aléas de simulation ◆ Assignations Non Bloquantes (NBA)
◆ Resets Asynchrones & Synchrones ◆ Clock Enables ◆ Template des Always Synthesizables
◆ Conception avec Machines d'états (FSMs) ◆ Architectures et implémentation des Machines d'états
◆ Etats inaccessibles et pratiques de conception sécurisée.

### Autres caractéristiques du langage Verilog

◆ Opérateurs Verilog ◆ Part selects ◆ Concatenation et réplication ◆ Registres à décalage
◆ Compilation Conditionelle ◆ Paramétrisation et generate ◆ Noms Hiérarchiques
◆ Opérateurs Arithmetiques et synthèse ◆ Signed et unsigned ◆ Memory arrays
◆ RAM : modélisation & synthèse ◆ $readmemb et $readmemh

### Tasks et Functions

◆ Comprendre les tasks ◆ Arguments des tasks
◆ Synchronisation ◆ Tasks et synthèse
◆ Functions

[1] Contenu non contractuel susceptible d'être mis à jour ou adapté par l'instructeur

© ALSE SAS - RCS DijonB 392 822 763 - SIRET 392 822 763 00034 - Code APE 7112B - TVA FR72392822763

## Fondamentaux de SystemVerilog pour Conception (Jours 2 & 3 matin)

**Les types de Données du SystemVerilog**

- enum • typedef • struct • union • packed/unpacked
- Packages et $unit • Utilisation des arrays en SystemVerilog
- array et structure literals, assignment patterns

**Nets et variables**

- Changements en Verilog-2005 et SystemVerilog
- Assignations continues des variables • Règles modifiées pour drivers et connexions
- Data types pour les ports et les nets

**Modules et Processes**

- Port connexions abrégées • Type parameters
- Idiômes de Synthèse des processes pour une meilleure expressivité
- Diverses améliorations du langage

**Application des Interfaces pour le design**

- Principe et utilité des Interfaces
- Interfaces pour encapsulation de la communication • modports
- Synthèse des interfaces et des modports
- Fonctions importées et conception RTL

## SVA : les Assertions en SystemVerilog (Jour 3)

**Introduction aux Assertions**

- Assertions, Properties, Sequences
- Clocking et sampling
- Property implication
- Utilisation des Assertions
- Simulation des Assertions
- Outils Formels

**Méthodologie des Assertions**

- ABV : concepts, champs d'application et intérêt
- Assertion et assumption
- Benéfice de l'utilisation des assertions pour le concepteur
- Vérifieurs de Protocole

**Introduction à la syntaxe SVA**

- Ecriture d'assertions simples
- Séquences et opérateur ##
- Répétition et time ranges
- Sequence Fusion
- Revue des opérateurs temporels
- Variables Locales et actions dans les assertions

**Packager les Assertions**

- Assertions dans les Interfaces et les Modules
- *bind* construct
- Déployer les *Verification IPs*, et en particulier les Assertion-Based IPs.

© ALSE SAS - RCS DijonB 392 822 763 - SIRET 392 822 763 00034 - Code APE 7112B - TVA FR72392822763

## Module-based SystemVerilog Verification (Jour 4)

**Vérification pour les groupes de conception**

- Les BFMs : Bus Functional Models
- Architecture des Testbench en Verilog traditionnel
- Fonctions traditionnelles de distribution en Verilog
- Timing des Stimuli et Réponses
- SystemVerilog pour construire des testbenches niveau module
- Clocking et Program blocs ◆ Application aux Testbenches et Interfaces
- Construire des librairies de patterns de simulation (séquences)
- Ecrire les *test cases* pour contrôler le testbench

**Vérification avancée : Constraint Random stimuli,**

- Utilité de la génération aléatoire de stimuli.
- Génération pseudo-aléatoire en SystemVerilog
- Randomisation des données de stimuli grâce à std::randomize
- Définir des contraintes pour contrôler la génération pseudo-aléatoire
- Graines (seeds) et stabilité de génération
- Fonctions avancée de génération : randcase & randsequence

**Vérification avancée : Functional Coverage + Cover-driven Testbenches**

- Introduction au concept d'automatisation
- Couverture fonctionnelle : covergroups et coverpoints
- Bins
- Transitions coverage
- Cross coverage
- Cover-driven testbenches

**Les Types de Données Dynamiques**

- Strings
- Queues
- Dynamic arrays
- Associative arrays
- Méthodes Queue et array
- Boucle *foreach*

**Sujets avancés**

- $root, $unit
- Pass-by-reference
- Bitstream casting
- Array querying & search functions
- Introduction à DPI

-=oOo=-

© ALSE SAS - RCS DijonB 392 822 763 - SIRET 392 822 763 00034 - Code APE 7112B - TVA FR72392822763