

## V1 - Les bases du langage VHDL

### *Programmation et simulation de FPGA avec VHDL*

#### Objectifs

- Comprendre les différentes possibilités offertes par le langage VHDL
- Découvrir le flux de conception complet
- Comprendre les notions de synthèse logique
- Implémenter la logique combinatoire et séquentielle
- Développer des machines à états finis
- Apprendre à écrire des bancs de test efficaces pour la simulation
- Vérifier les timings
- Réutiliser et configurer les composants

#### Environnement du cours

- Cours théorique
  - Support de cours au format PDF (en anglais) et imprimé
  - Le formateur répond aux questions des stagiaires en direct pendant la formation et fournit une assistance technique et pédagogique
- Activités pratiques
  - Les activités pratiques représentent de 40% à 50% de la durée du cours
  - Elles permettent de valider ou compléter les connaissances acquises pendant le cours théorique.
  - Un PC par binôme pour les activités pratiques
  - Pour Xilinx:
    - ▶ Xilinx Vivado IDE
    - ▶ Carte Nexys-3 (basée sur Xilinx Spartan6) ou Nexys-4 (basée sur Xilinx Artix7)
  - Pour Lattice :
    - ▶ Diamond avec Synplify Pro et Active-HDL
    - ▶ Carte ECP2 ou MachXO
  - Pour Actel :
    - ▶ Libero avec Synplify Pro
    - ▶ ModelSim + Actel Fusion
  - Exemples de code, exercices et solutions
- Au début de chaque demi-journée une période est réservée à une interaction avec les stagiaires pour s'assurer que le cours répond à leurs attentes et l'adapter si nécessaire

#### Pré-requis

- Connaissance de la technologie numérique
- Notions d'algèbre booléenne
- Quelques notions de programmation sont souhaitables (quel que soit le langage)

#### Audience visée

- Tout ingénieur ou technicien en systèmes embarqués possédant les prérequis ci-dessus.

# Plan du cours

## Premier Jour

### From the logic gate to the FPGAs

- Reminder on digital electronic
  - Combinational Logic
  - Sequential (Synchronous) Logic
- Schematics / Hierarchical representation
- Structure of an Integrated Circuit
  - SSI (small scale integration), TTL
  - MSI (medium scale integration), PALs, GALs, PLDs
  - LSI (large scale integration), CPLDs
  - VLSI (very large scale integration), ASICs, ASSPs, FPGAs
- Development of logical architectures
- Technology constraints
  - Interconnection methods (SRAM, Fuse, AntiFuse, Flash)
  - Clock distribution
  - Logic element types
  - Look Up Table
  - Basic logic cell
  - I/O modules
  - Timing issues
- VHDL Contributions
  - Benefits of VHDL programming
  - The VHDL Design Flow
  - Programming
  - Simulation
  - Synthesis
  - Mapping
  - Place and Route
  - Timing Analysis
  - Bitstream generation

### VHDL Basic concepts

- The Entity / architecture concept
  - Entity declaration
  - Ports
  - Different styles of architecture
- Libraries and context
- Component instantiation
  - Port map
- Simulation flow and environment
  - The Testbench
- Getting started with the IDE
- Creating a project from scratch
- Synthesis / Translate / Map / Place and Route (PAR) /BitGen
- Report Analysis
- Assigning I/O locations using PlanAhead (editing constraint file)
- Schematics Views
- Analyzing the placement
- Flashing with Impact

**Exercise :** Understanding the steps of design and programming

**Exercise :** Getting started with the simulator, waveform generation and analysis

## Deuxième Jour

### VHDL Syntax

- Lexical items
  - Comments
  - Identifiers and keywords
  - Characters, Strings, Numbers, Bit strings
- Constants
- Signals
- Variables and aliases
- Data types
  - Scalar types:
    - Integer
    - Real
  - Enumerated type
  - Physical types
  - Composite types:
    - Array
    - Record
    - Special types
- Library and Packages
  - Standard package
  - IEEE packages
  - Std\_logic\_1164 package
  - Multi-valued types
  - Multi-driver and resolved types
  - Numeric types
- Type conversion
- Aggregates
- Attributes
  - Type attributes
  - Signal attributes

**Exercise :** Importing a predefined hardware definition in the project, instantiating a component

### Combinational logic in VHDL - 1st part

- Concurrent instructions
  - Component instantiation
  - Signal affectation
  - Simple affectation
  - With... Select... When statement
  - When... Else statement
  - Unaffected keyword
  - Variable aggregates
  - Relational operators
  - Arithmetic operators
  - Concatenation / Slicing

**Exercise :** Coding, simulating and synthesizing a bounds enforcer

**Exercise :** Designing a 7-segment decoder

**Exercise :** Designing a 4-bit adder

## Troisième Jour

### Combinational logic in VHDL - 2nd part

- Sequential instructions
  - Processes
  - Sensitivity list, Wait statement
  - Potential interpretation incoherencies between logical synthesis and simulation
  - Signal affectation
  - Transparent Latch
  - Use of variables
  - If... Then... Else statement
  - Case... When statement
  - Null statement
  - Iterative statements:
    - For loop
    - While loop
    - Conditional Iteration
- Numeric\_std / Numeric\_bit packages
  - Defined Types and Operators
  - Conversion functions
- Ambiguity about the types and the « use » clause

**Exercise :** Coding, simulating and synthesizing a bounds enforcer

**Exercise :** Designing a 7-segment decoder

**Exercise :** Designing a 4-bit adder

### Synchronous logic in VHDL

- Limits of asynchronous designs
- Synchronous Design, Registers and Timing
- Pipeline notion
- D Flip-flop description
- Use of Variable for synchronous process
  - Variable Synthesis
- Reset and Set management
- Clock Enable
- Tri-state buffers description
- Synchronous design methodology
- Memory Synthesis
  - Asynchronous RAM
  - Synchronous RAM
- Single port
- Double port
- Pipelined
  - ROM
  - IP generator introduction

**Exercise :** Designing a counter/decounter

**Exercise :** Designing a FIFO

## Quatrième Jour

### Hierarchical Design

- Hierarchical division
- Analysis and Elaboration
- Components and Configurations

- Components
- Configuring components instances
- Direct instantiation
- Basic configurations
- Configuration declaration
- Default binding
- Configuration specification
- Port map and Generic map
  - Genericity and automatic configuration of re-usable modules
- Packages
  - Package Declarations
  - Package Bodies
  - Using package
- Libraries

**Exercise :** Designing a generic 4-digits BCD-counter and displaying it on a 7-segment display

**Exercise :** Enhancing a 4-bit BCD-counter/decounter to create a generic one

**Exercise :** Working with configurations

**Exercise :** Designing a n digits BCD-counter/decounter and displaying it on a 7-segment display

## Synthesis and Testbenches

- Synthesis
  - Syntactic and Semantic Restriction
  - Creating synthesizable Designs
  - Inferring Hardware elements
  - Initialization and Reset
  - Pragmas
- Testbenches
  - A few basic rules for the writing of an efficient test bench
  - Potential incoherencies between logical synthesis and simulation: how to avoid it
  - VHDL instructions specific to simulation
  - Testbench with File I/O

**Exercise :** Designing and testing a logical address decoder

**Exercise :** Simulation of sequential processes

**Exercise :** Advanced simulation techniques Text files