Unité : Conception de systèmes numériques (CSN)

Outils EDA et script TCL

EDA : Electronic Design Automation

HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD

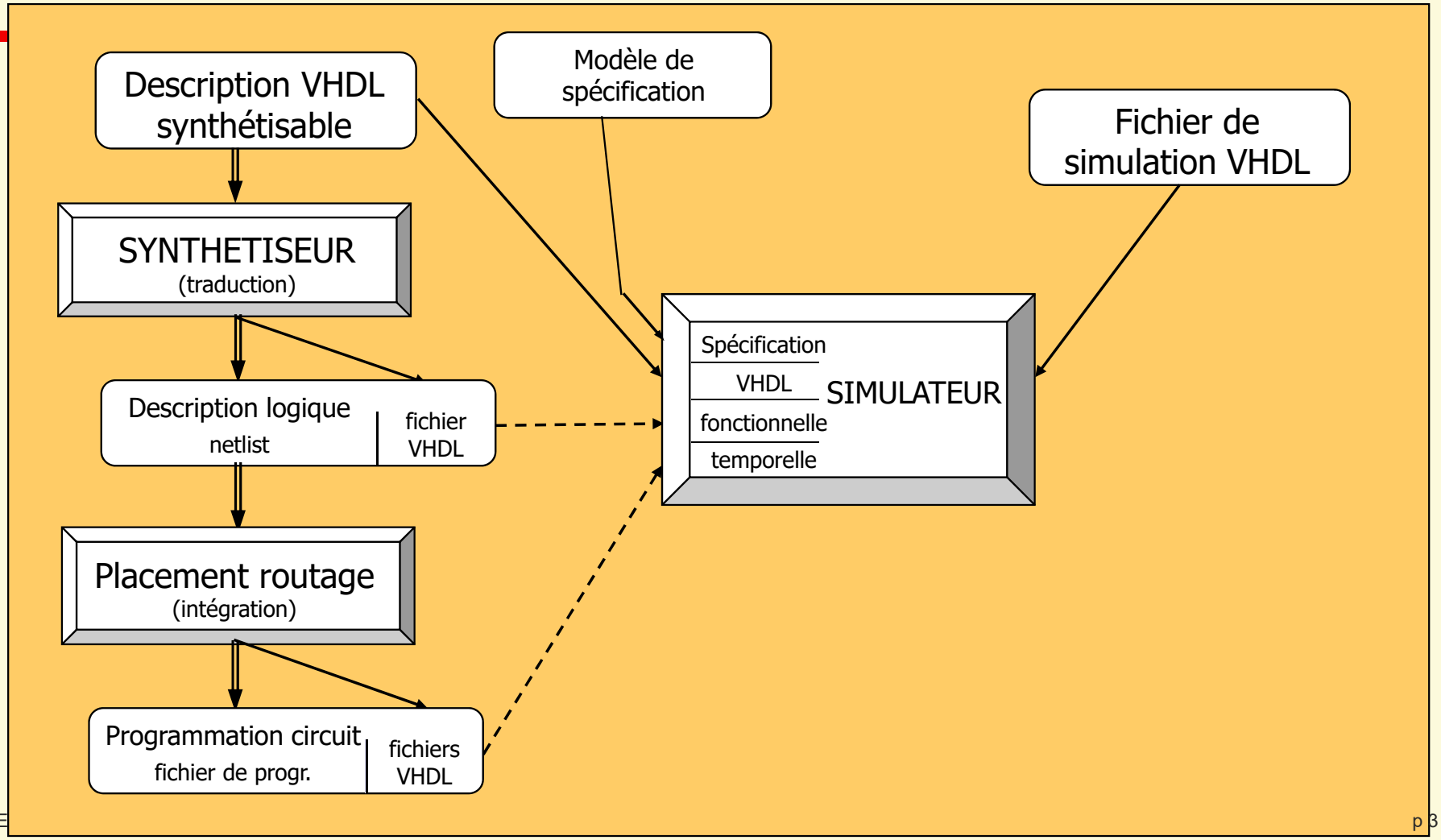
Etienne Messerli

Février 2023

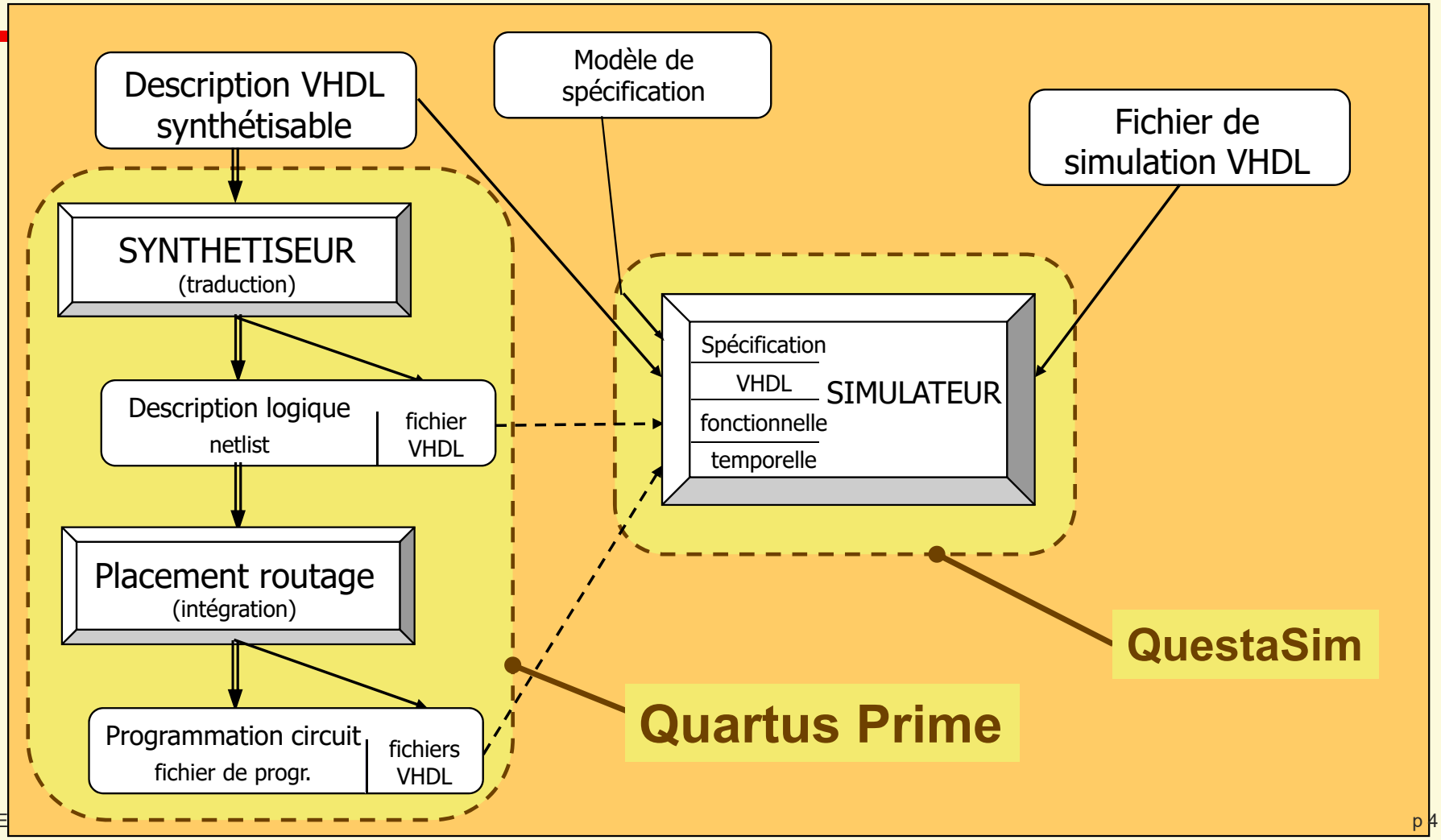
Contenu présentation

- Rappel: Design flow VHDL
- Les outils EDA:
  - catégorie, fonctionnalités, liste, ...
- Outils EDA au REDS
  - Projet graphique avec Logisim
  - Projet textuel VHDL
- Script QuestaSim/ModelSim

Réalisation d'un circuit : *Design flow*



Réalisation d'un circuit : *Design flow*



Outils EDA (Electronic Design Automation)

- Logiciels EDA modernes comprennent les catégories suivantes:
  - Outils d'aide à la gestion de projet IDE (Integrated Development Environment)
    - inclus saisie graphique, parfois génération auto de VHDL
  - Simulateur VHDL
  - Synthétiseur VHDL
- Spécifique à chaque fabricant de PLDs
  - Permet le placement-routage pour leurs PLDs
  - Inclus IDE, synthétiseur et plus …

Catégories de logiciels EDA …

- Exemple d'outil de gestion de projet et saisie graphique (IDE):
  - **Logisim-evolution** (outil libre)
    - Logisim initié par Carl Burch, Hendrix College, USA
    - collaboration REDS avec d'autres HES (BE, GE)
    - utilise logiciel vendeur de PLDs (Quartus)
  - Outils des vendeurs de PLDs
    - **Altera: Quartus Prime,**

… catégories de logiciels EDA …

- Simulateurs:
  - Modelsim — VHDL, Verilog, SystemC
  - **Questasim** (Mentor Graphics) — VHDL, VHDL 2008, Verilog, SystemC, SystemVerilog
  - Riviera-PRO 2010 (Aldec) — VHDL, **New** VHDL 2008, Verilog, SystemVerilog
  - VCS (Synopsys) — VHDL, Verilog, SystemC, SystemVerilog
- Synthétiseurs:
  - Nombreux outils d'entreprises
  - Logiciels de placement routage des fabricants des PLDs
    - inclus un synthétiseur efficace

Logiciels EDA des vendeurs de PLDs …

- Fabricant de PLDs propose une suite d'outils avec :
  - IDE pour la gestion du projet
  - Editeur graphique hiérarchique (saisie schéma)
    - Remarque: souvent sans génération de VHDL !
  - MegaWizard pour fonctions de base et étendue (licence, IP)
    - propre aux circuits du fabricant
  - Simulation simple ou simulateur externe (gratuit, prix réduit)
  - Synthétiseur
    - actuellement dispose d'excellent synthétiseur
  - **Placement/routage : indispensable et spécifique pour leur technologie**
  - Fréquemment : version de base gratuite

… logiciels EDA des vendeurs de PLDs …

- Utilisation dans l'industrie:
  - Fréquemment utilisé dans les PME
  - Outil abordable
    - version de base gratuite
    - version étendue à prix raisonnable (quelques KFr)
- Adéquat pour design simple
- Design de moyenne complexité
  - simulateur externe performant recommandé
  - version étendue à prix abordable

Outils EDA tiers pour PLDs …

- Liste des simulateurs:
  http://en.wikipedia.org/wiki/List_of_Verilog_simulators
- **Aldec** - http://www.aldec.com/
  HDL design entry and simulation software for programmable logic designers.
- **Altium** – http://www.altium.com/
  Altium Designer (IDE)
- **Cadence** - http://www.cadence.com/
  Allegro Design Entry HDL (IDE), Incisive (simulateur), Encounter RTL (synthèse)
- **Dolphin Integration** - http://www.dolphin.fr/
  SLED Schematic Editor (Verilog, VHDL)

… outils EDA tiers pour PLDs …

- **Mentor** – http://www.mentor.com/
  HDL Designer & Visual Elite HDL: design environnement
  Precision & LeonardoSpectrum: FPGA synthetis
  Modelsim: simulator VHDL/ Verilog/ PSL
  Questa: simulator VHDL/ Verilog/ SystemVerilog
  Catapult Synthesis: high level synthesis
- **Synopsys** - http://www.synopsys.com/
  Synplify Pro : Logic Synthesis for FPGA Implementation
  Synplify premier :Fast implementation of advanced FPGA
  VCS: High-performance simulation
- …

Autres outils EDA ...

- Symphony EDA - http://www.symphonyeda.com/
  Offers a VHDL compiler/simulator with an integrated development environment. Supports VHDL'93, Vital, and SDF. Free command-line tools also available
- Translogic - http://www.translogiccorp.com/
  EASE and EALE provide HDL aware entry tools, both graphical and text based. Also providing Linux support.
- SynaptiCAD - http://www.syncad.com/
  Software tools for EDA, VHDL and Verilog model generation, simulation, and timing diagram editor.
- TransEDA - http://www.transeda.com/
  Provider of ready-to-use verification solutions for the SoC, ASIC and FPGA

Objectifs utilisation outils EDA

Cours CSN : sans outil IDE

- Former les étudiants aux nouvelles technologies
- Conception de systèmes numériques
- Maitriser la description en VHDL synthétisable (textuelle)
- Pratiquer la simulation de design pour PLDs
  - Utilisation de bancs de test automatique (test-bench) avec des scripts
  - Maitriser la chercher d'erreurs via le chronogramme
- Utilisation d'un outil de vendeur de PLD (synthèse et p-r)
- Test du design sur une cible (carte)

Les outils EDA au REDS

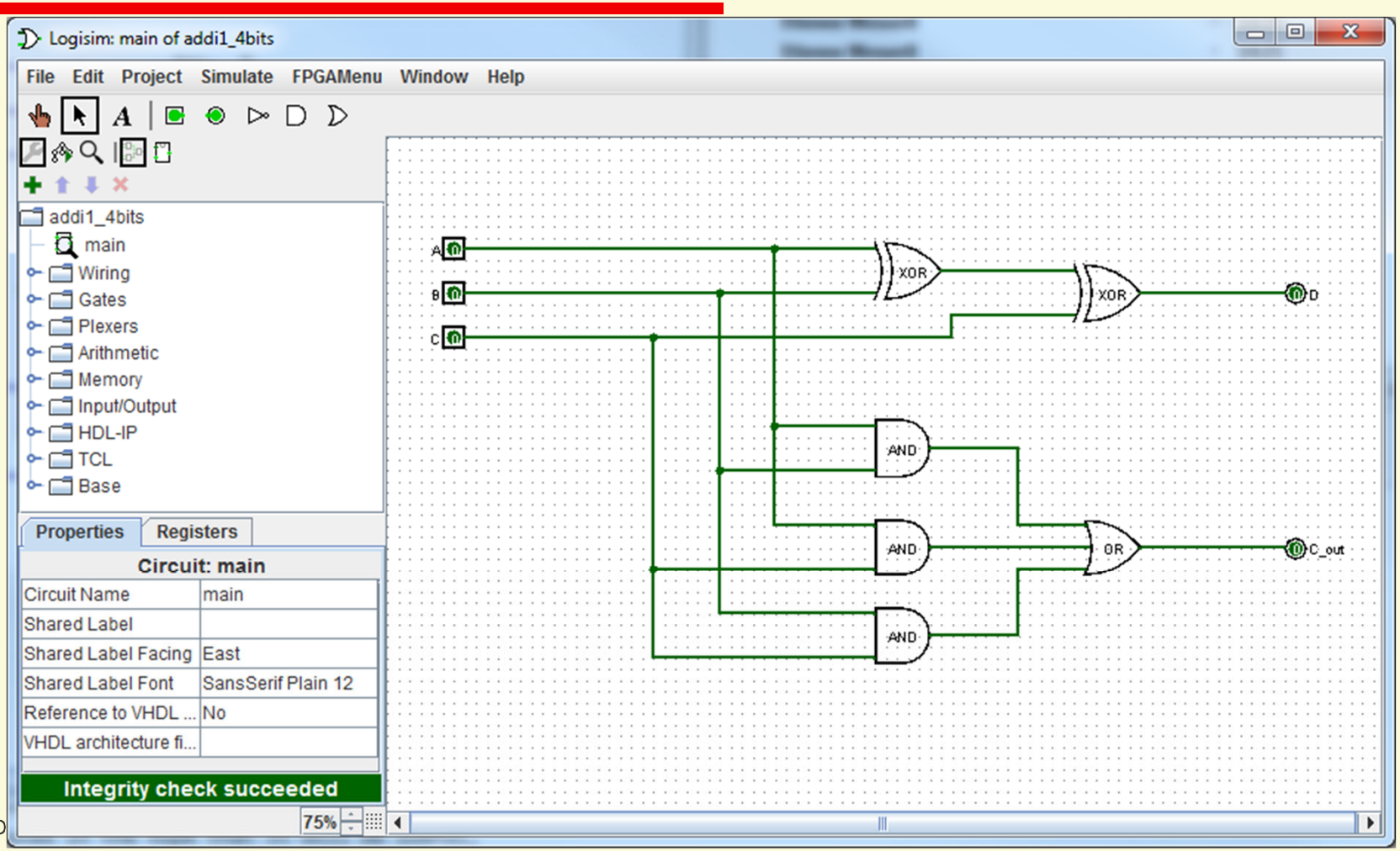
- Outil de gestion de projet et saisie graphique:
  - Logisim (utilisé en 1ère et 2ème année) avec Quartus
- Simulateur:
  - **QuestaSim 2020**
- Synthétiseur:
  - Intégré dans l'outil des fabricants de PLDs
- Synthétiseur et placement/routage :
  - **Intel-Altera: Quartus Prime 18.1**
  - Xilinx: Vivado 2019.2 et ISE 14.7

Logisim-evolution

Rappel fonctionnalités de Logisim-evolution

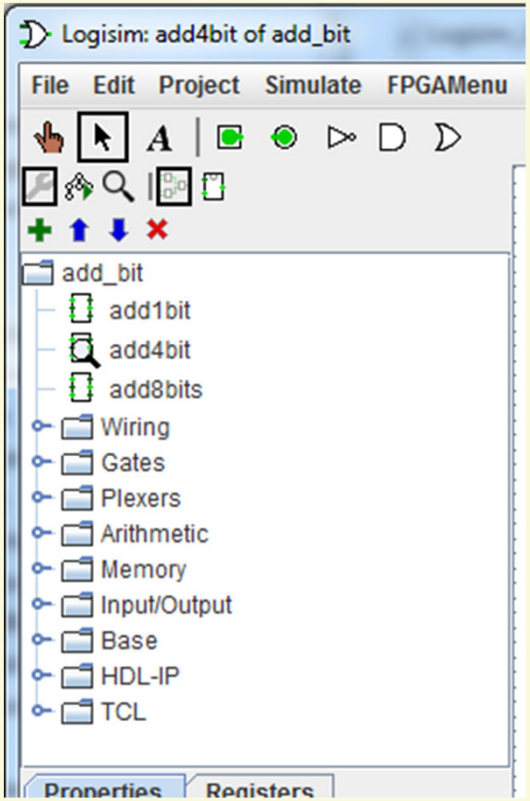
- Evolution assurée par le REDS avec HES BE & GE
- Saisie graphique de schéma et textuelle VHDL
- Librairie de composant de base
- Librairie d'éléments I/O
- Simulation : schéma & VHDL (avec Questasim)
  - visualisation état des signaux
  - chronogramme
- Intégration dans des cartes avec outil fabricants de PLDs

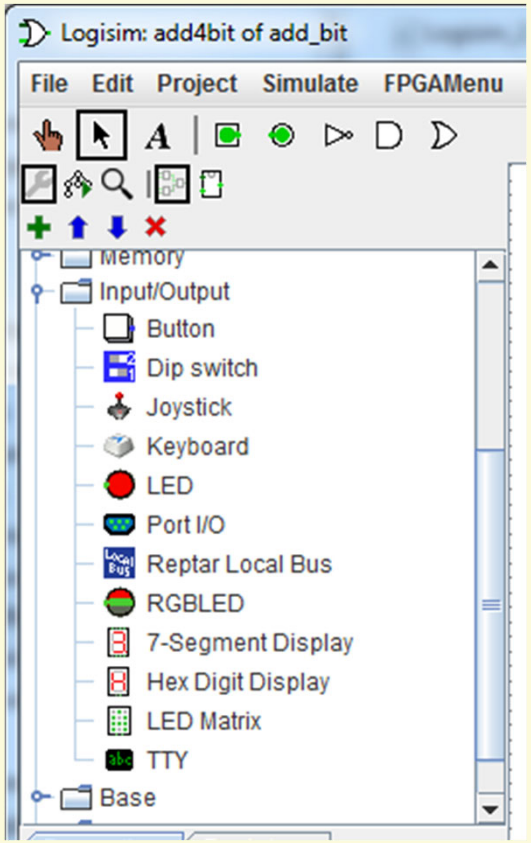
Logisim - GUI

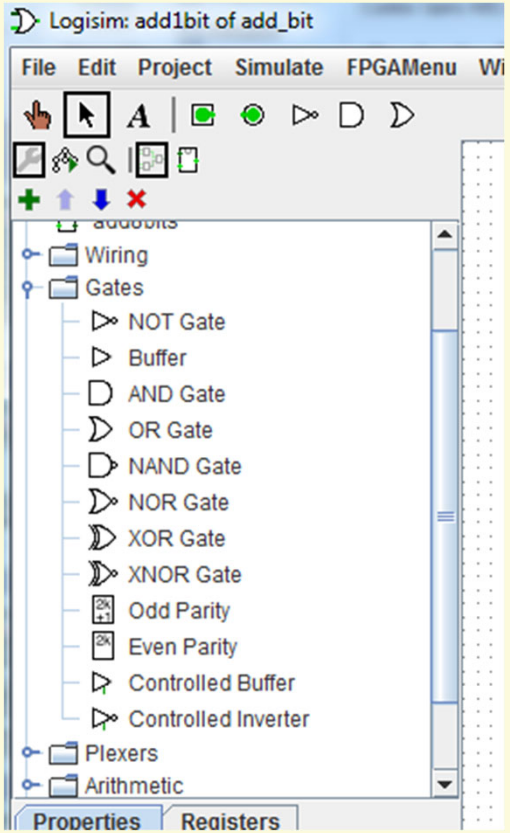

Logisim: main of addi1_4bits
File Edit Project Simulate FPGAMenu Window Help
addi1_4bits
main
Wiring
Gates
Plexers
Arithmetic
Memory
Input/Output
HDL-IP
TCL
Base
Properties
Registers
Circuit: main
Circuit Name main
Shared Label
Shared Label Facing East
Shared Label Font SansSerif Plain 12
Reference to VHDL ... No
VHDL architecture fi...
Integrity check succeeded
75%
A
B
C
XOR
XOR
D
AND
AND
AND
OR
C_out

Logisim - GUI

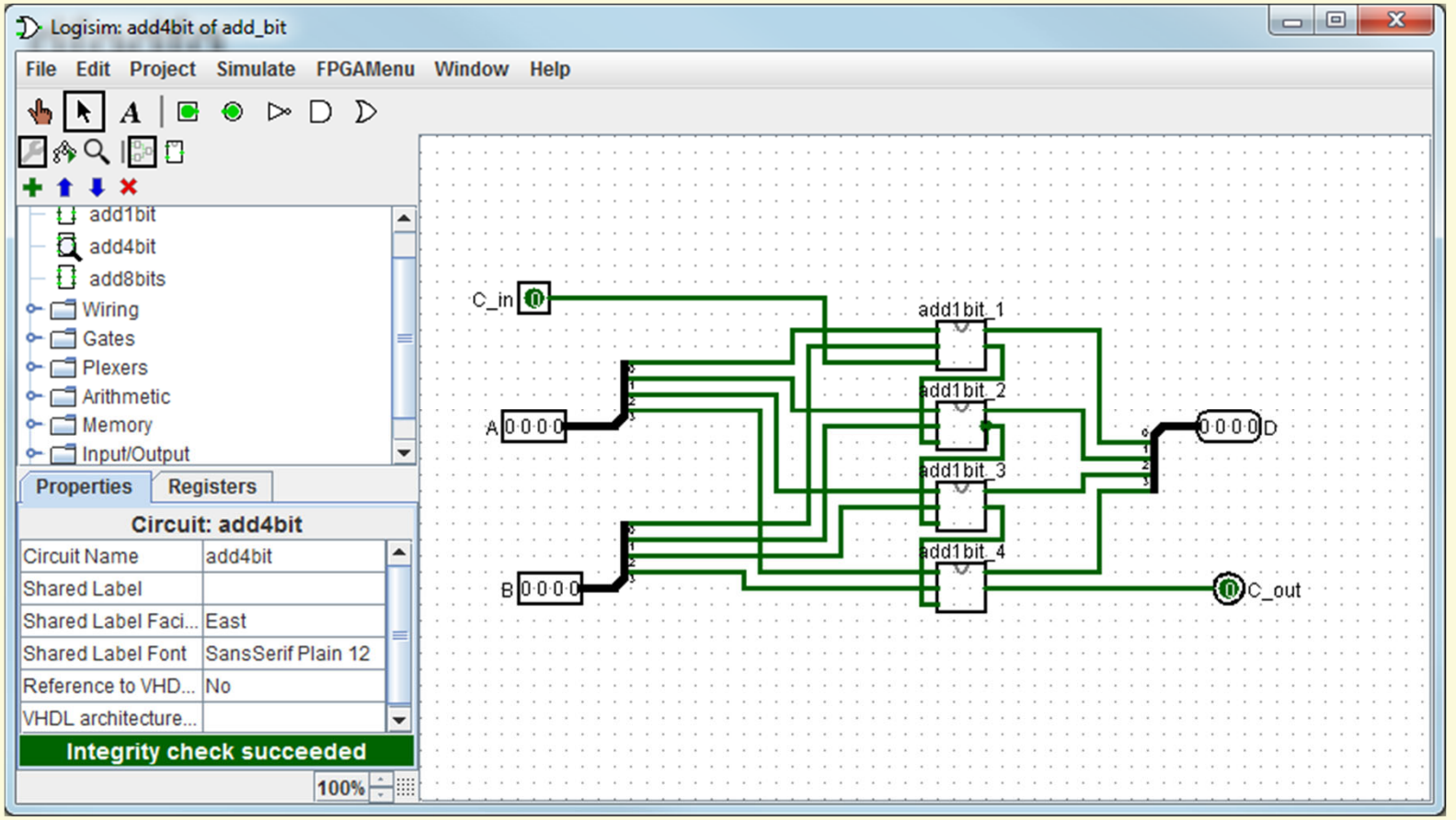
Liste des librairies



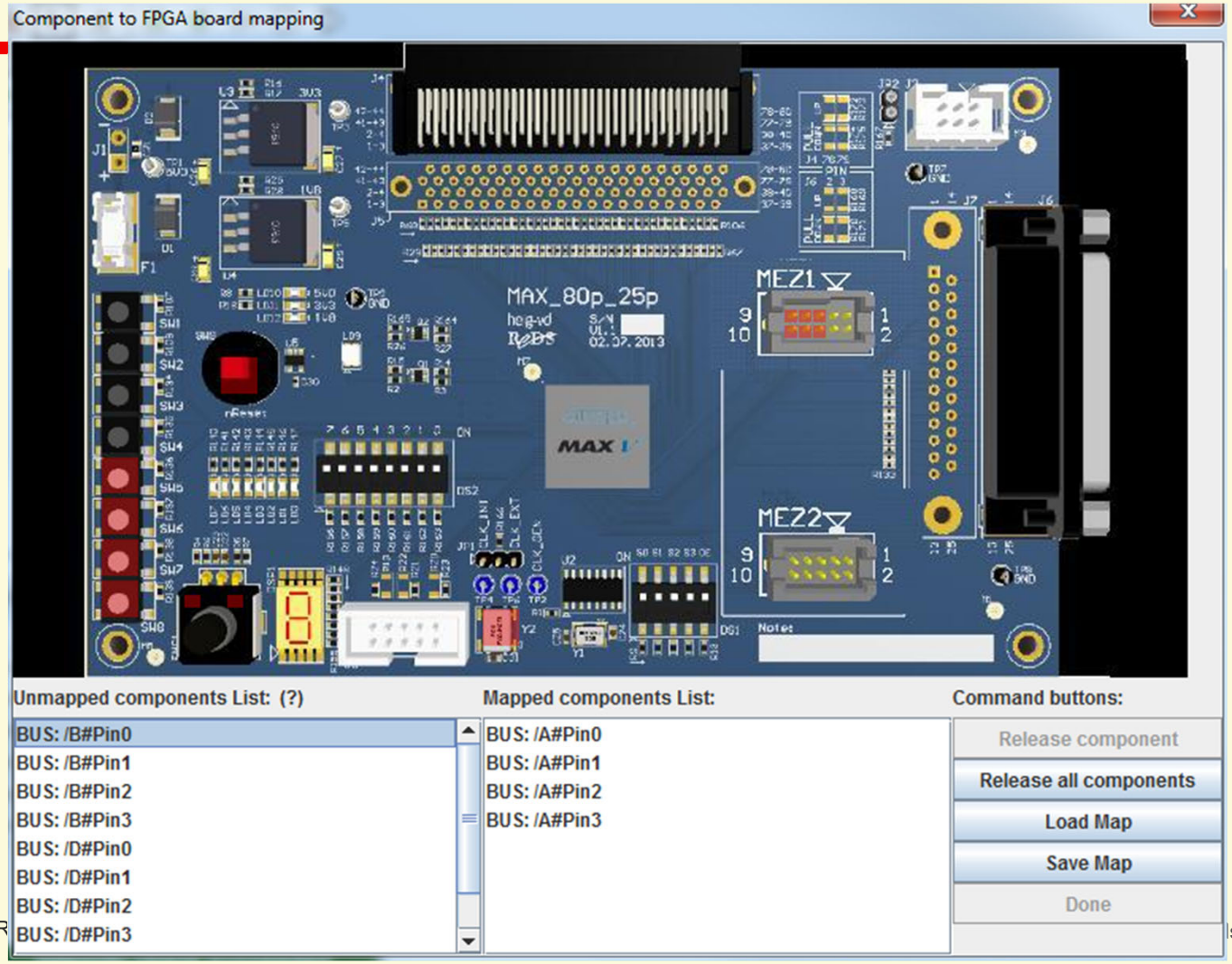




Logisim - Hiérarchie



Logisim - Intégration

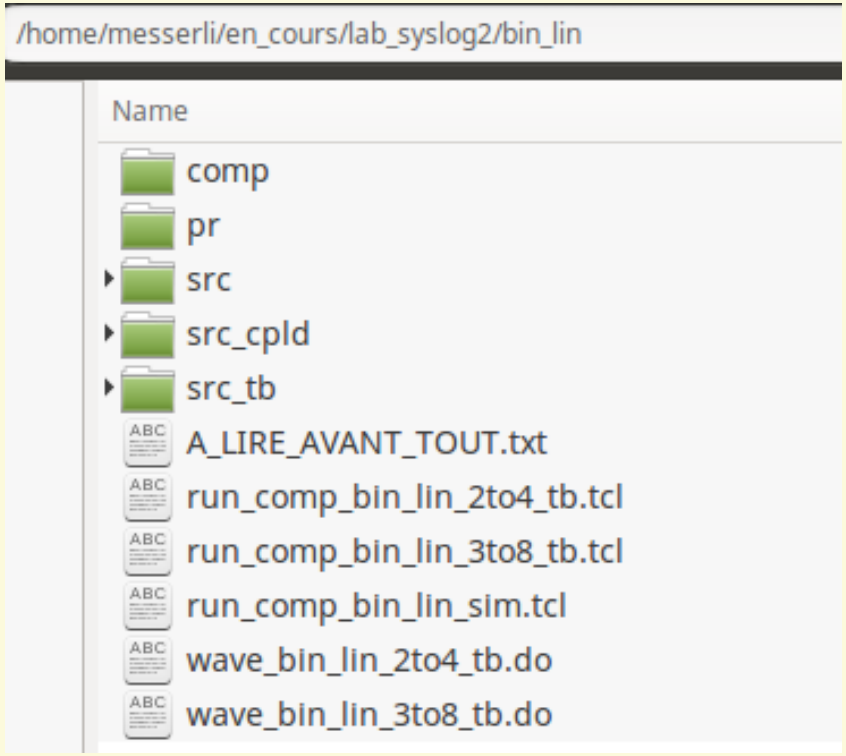


Réalisation de projet textuel

- Utilisation de scripts pour les différents outils
  - GUI des outils sont en Tcl/Tk, dispose d'un wish
  - Utilisation langage scipt Tcl/Tk (gratuit)
  - Automatise compilation, chargement, …
- Structure normalisée des répertoires:
  - racine: scripts, puis sous-répertoires:
    - comp compilation pour ModelSim/ Questasim
    - p_r infos pour le placement-routage par Quartus
    - src fichiers sources vhdl
    - src_tb fichiers sources pour la simulation
    - synth : fichiers pour la synthèse, Precision

Exemple structure de répertoires

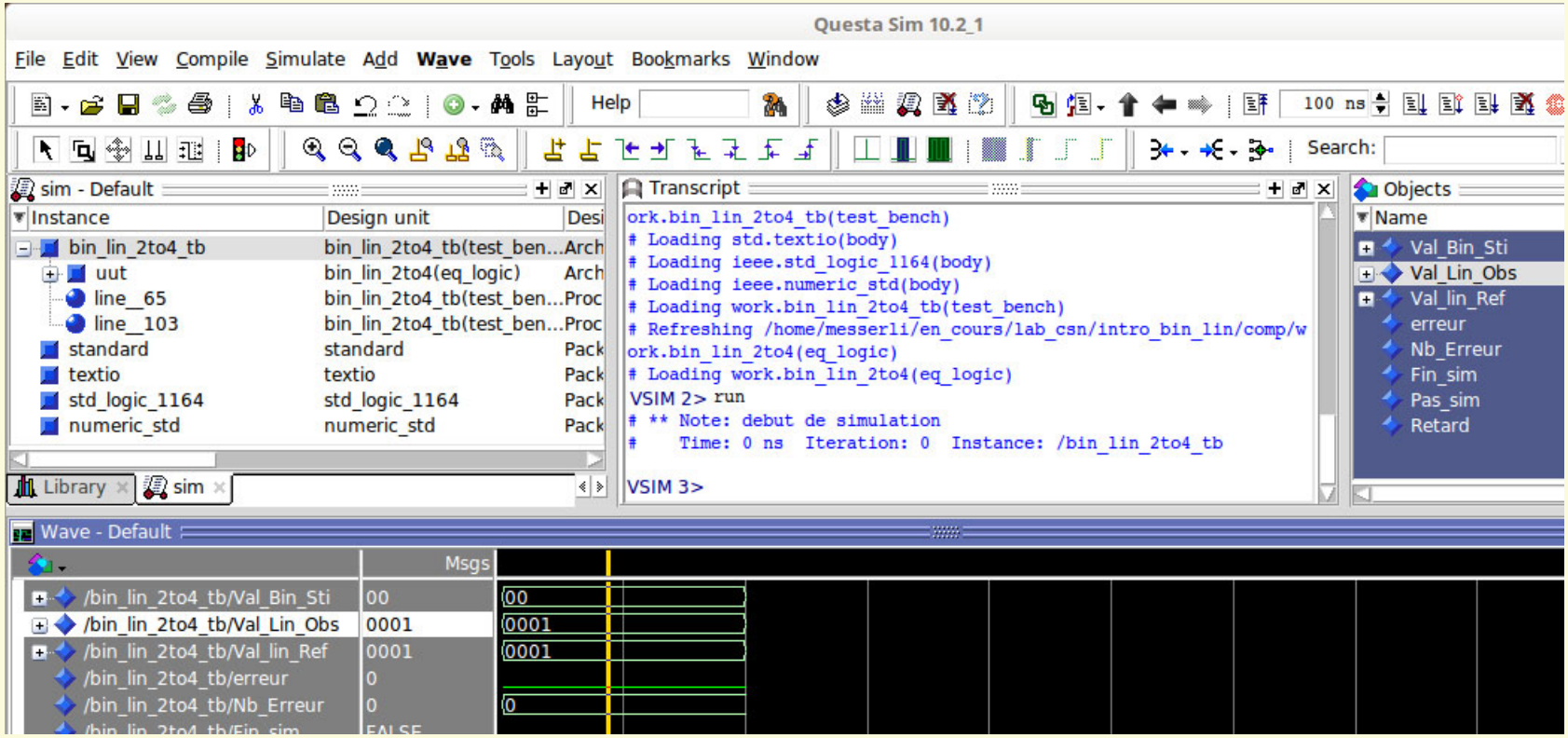
- Structure des répertoires du projet "bin_lin"



comp	compilation ModelSim/QuestaSim
pr	synthèse du design top
pr_cpld	placement/routage du design avec interface CPLD
src	source du design avec fichier top
src_cpld	fichier source maxv_top.vhd (interface carte)
src_tb	fichiers pour la simulation

Simulateur QuestaSim

- QuestaSim : VHDL, *Verilog, SystemC, SystemVerilog*

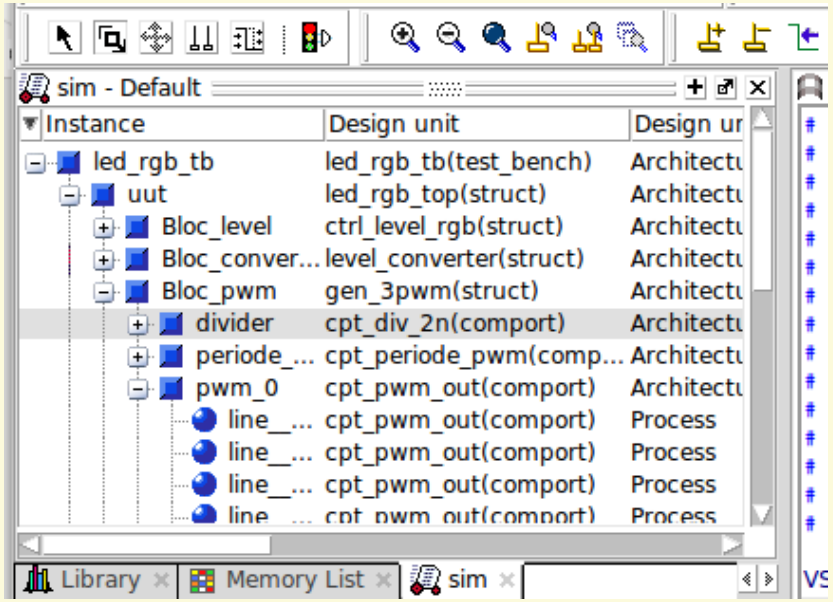


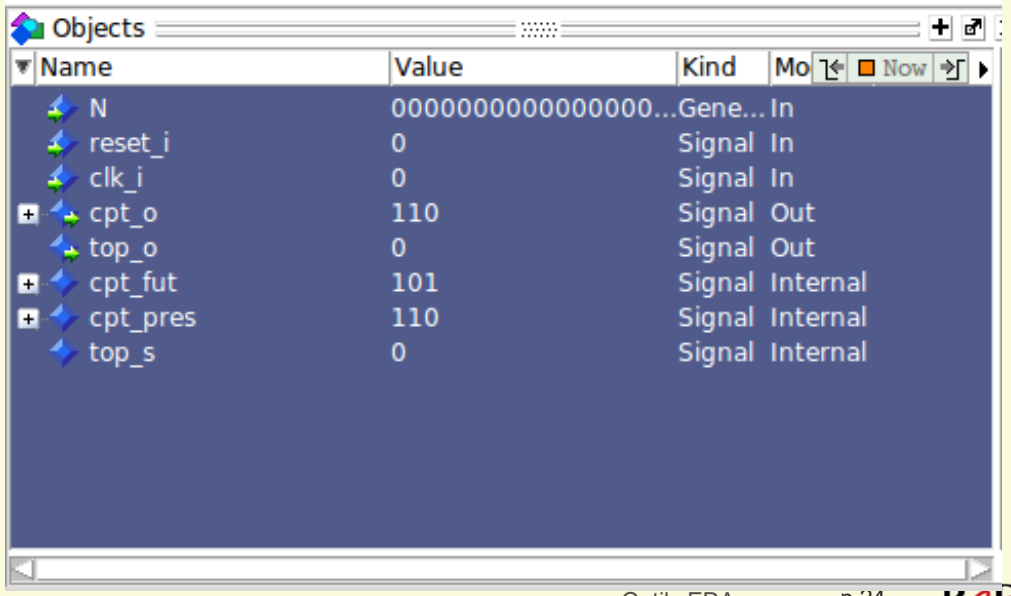
Fonctionnalités d'un simulateur …

- Exécute la description VHDL
  - visualise le comportement dans le temps de la description
- VHDL permet de décrire des bancs de test
  - envoi de stimuli et possibilité de vérifications des sorties
- Chronogramme:
  - Permet de visualiser n'importe quel signal du design
  - Indispensable pour le debug d'un design
- Encore beaucoup d'autres fonctionnalités …

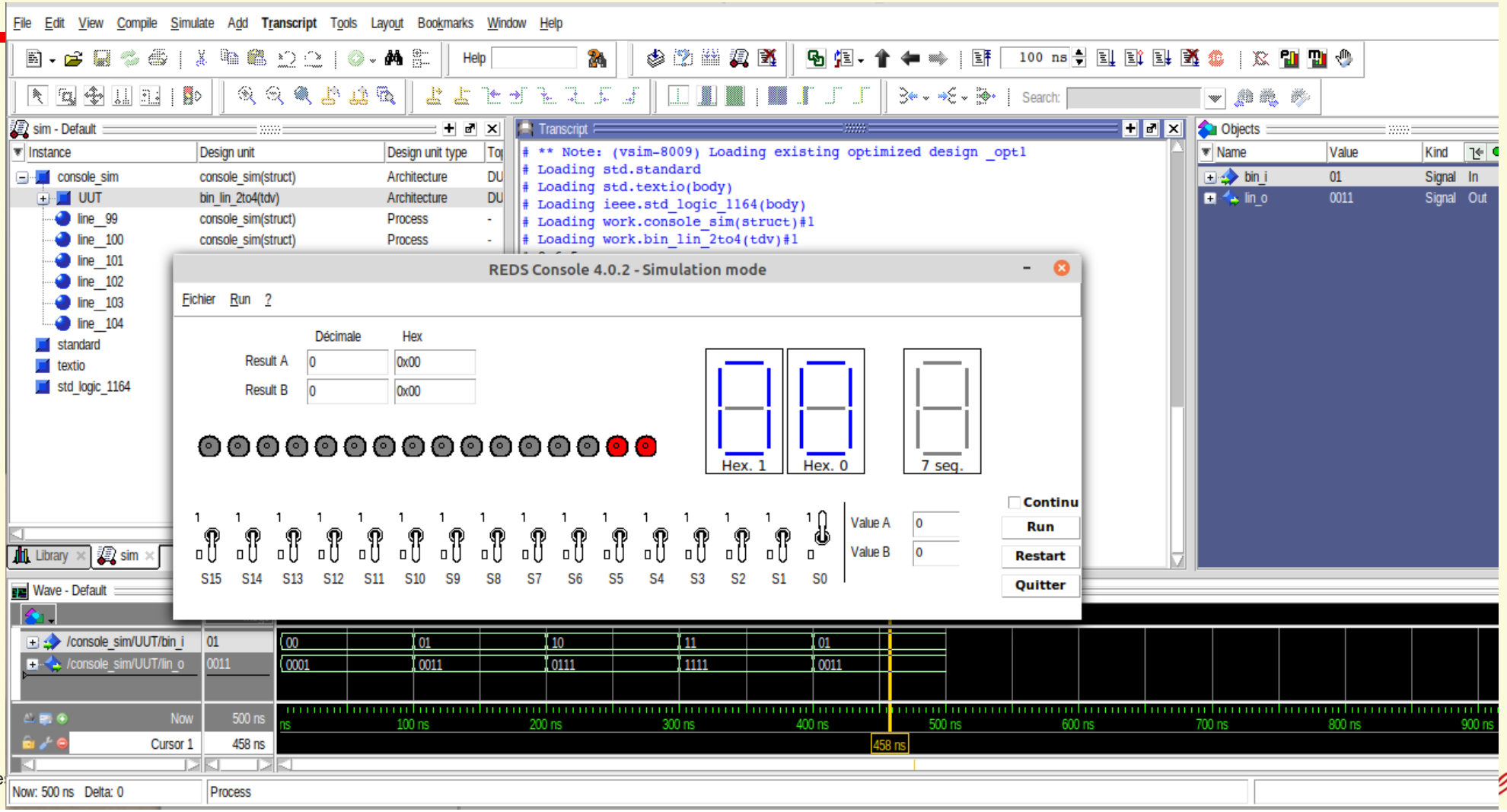
… fonctionnalités d'un simulateur

- Analyseur logique virtuel
- Permet d'analyser **tous** les signaux du design
  - Naviguer dans la hiérarchie, puis sélectionner les signaux





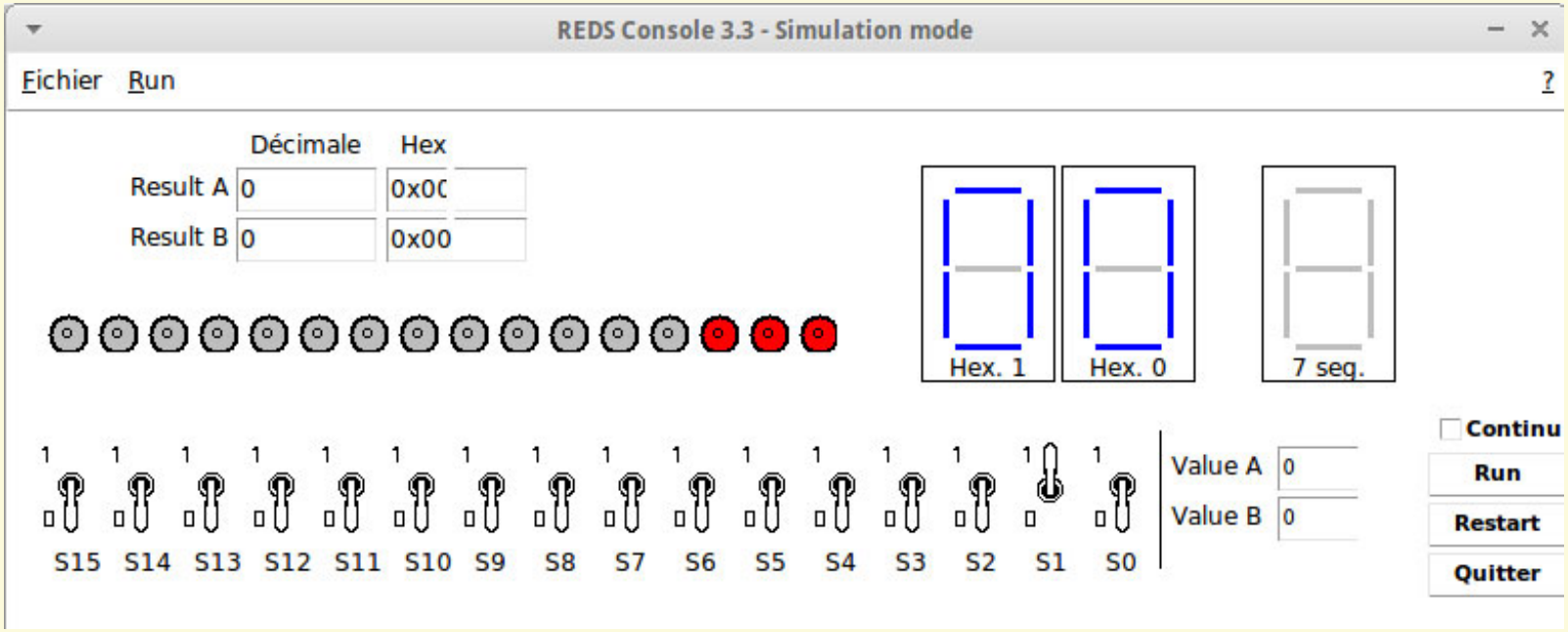
Simulation manuelle avec console REDS



Console interactive REDS

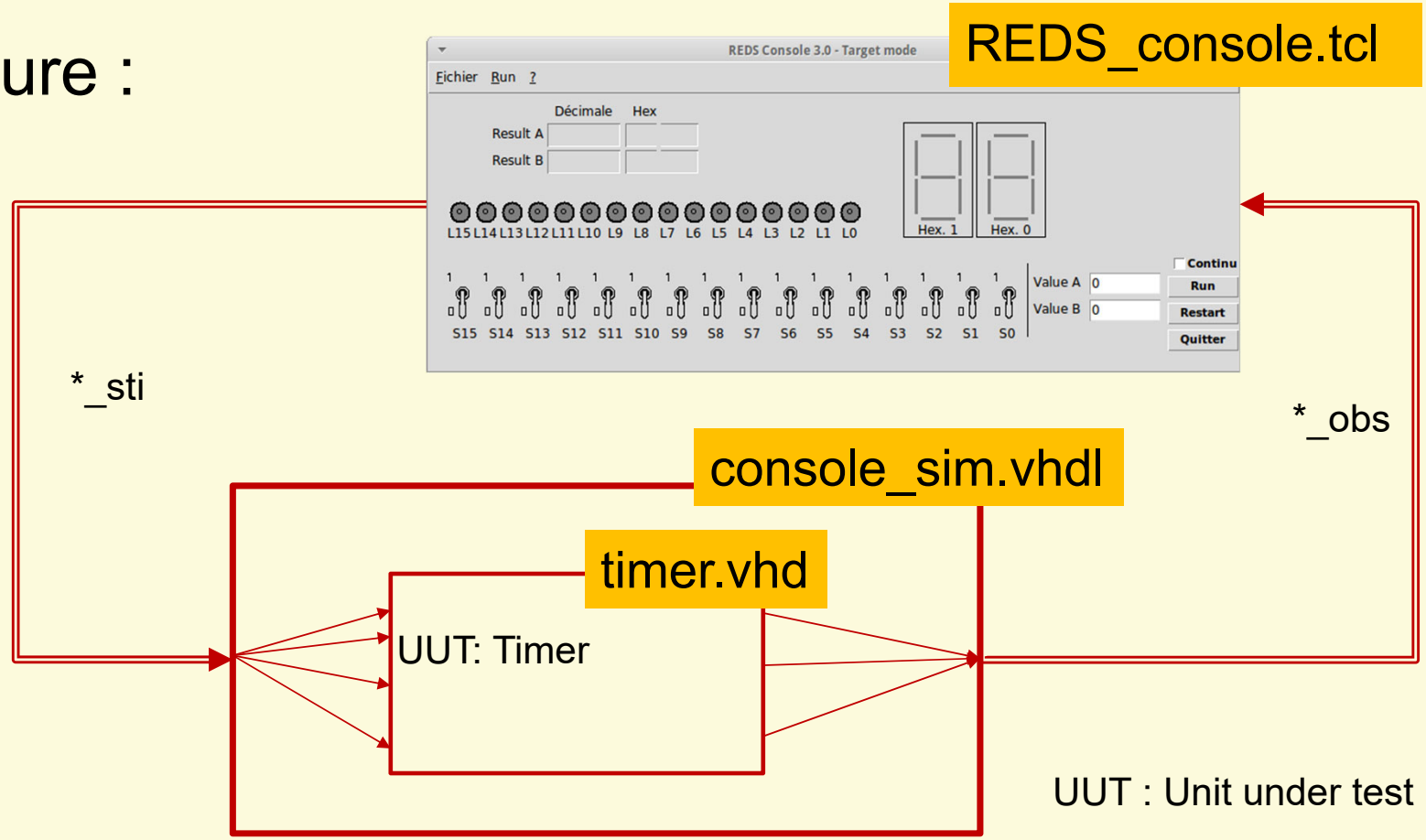
Console utilisée pour simulation manuelle :

- Composant à simuler connecté via `console_sim.vhd`



Console interactive du REDS

- Structure :



Fichier `console_sim.vhd` ...

- Permet une simulation manuelle
- Connexion de la console *REDS_Console* avec la description à simuler (UUT : unit under test)
- Modèle d'un générateur d'horloge dans le fichier *console_sim.vhd*

- Preuve de la simulation: chronogramme

… console_sim.vhd …

```
entity console_sim is
   port(
      --16 switchs (interrupteurs)
      S0_sti        : in     Std_Logic;
      …
      S15_sti       : in     std_logic;
      --Valeurs 16 bits Val_A et Val_B
      Val_A_sti     : in     std_logic_vector (15 downto 0 );
      Val_B_sti     : in     std_logic_vector (15 downto 0 );
      --16 Leds
      L0_obs        : out    std_logic;
      …
      L15_obs       : out    std_logic;
      --2 Resultats sur 16 bits
      Result_A_obs  : out    std_logic_vector (15 downto 0 );
      Result_B_obs  : out    std_logic_vector (15 downto 0 );
      --2 Affichage 7 segments avec valeurs en hexadecimal (4 bits)
      Hex0_obs      : out    Std_Logic_Vector (3 downto 0);
      Hex1_obs      : out    Std_Logic_Vector (3 downto 0);
      --1 Affichage 7 segments
      seg7_A_obs    : out    std_logic;
      …
      seg7_G_obs    : out    std_logic                    );
end console_sim ;
```

… console_sim.vhd …

```
architecture struct of console_sim is
   -- Declraration de signaux internes
   signal B_s : std_logic_vector(3 downto 0);
   signal P_s : std_logic_vector(3 downto 0);

   -- Declaration du composant a simuler
   component Master_I2C_timer
      port(
        Clock_i  : in  std_logic;
        Reset_i  : in  std_logic;
        Start_i  : in  std_logic;
        Done_o   : out std_logic     );
   end component ;
   for all : Master_I2C_timer
                  use entity Work.Master_I2C_timer(Comport);

  -- Declaration de constantes et signaux internes
  constant Periode_c : Time := 100 ns;
  signal Horloge_s : Std_Logic;

begin
```

… console_sim.vhd

```
begin

--¦process de generation d'une horloge interne à console_Sim
  process
  begin
    Horloge_s <= '0', '1' after Periode_c/3,
                      '0' after (Periode_c/3)+(Periode_c/2);
    wait for Periode_c;
  end process;

--¦ Instanciation du composant a simuler
   uut : Master_I2C_timer
      port map (
        Clock_i  => Horloge_s,  --connecter sur horloge
        Reset_i  => S15_sti,
        Start_i  => S0_sti,
        Done_o   => L0_obs     );

end struct;
```

Utilisation d'un script Tcl

- Permet d'automatiser les commandes nécessaires pour les différentes étapes d'une simulation, soit:
  - création de librairie : Work ou spécifique
  - compilation de l'ensemble des fichiers du projet inclus le test-bench
  - chargement du test-bench
  - ouverture des fenêtres et placement de celles-ci
  - ajout des signaux à visualiser ou chargement du format de la fenêtre wave préalablement sauvée
  - lancement de la simulation

Script de compilation *console_sim*

```
# ------------------------------------------------
# -- HEIG-VD, institut REDS
# -- File : comp_master_i2c_timer_sim.tcl
# ------------------------------------------------

# Complation des paquetages
vcom -reportprogress 300 -work work  ../lib/LOG_pkg.vhd
vcom -reportprogress 300 -work work  ../src_Master_I2C/Master_I2C_pkg.vhd

# Complation des fichiers du I2C master
vcom -reportprogress 300 -work work  ../src_Master_I2C/Master_I2C_Timer.vhd

# Compilation du top_sim pour la simulation manuelle
vcom -reportprogress 300 -work work  ../src_Master_I2C_tb/ console_sim.vhd
```

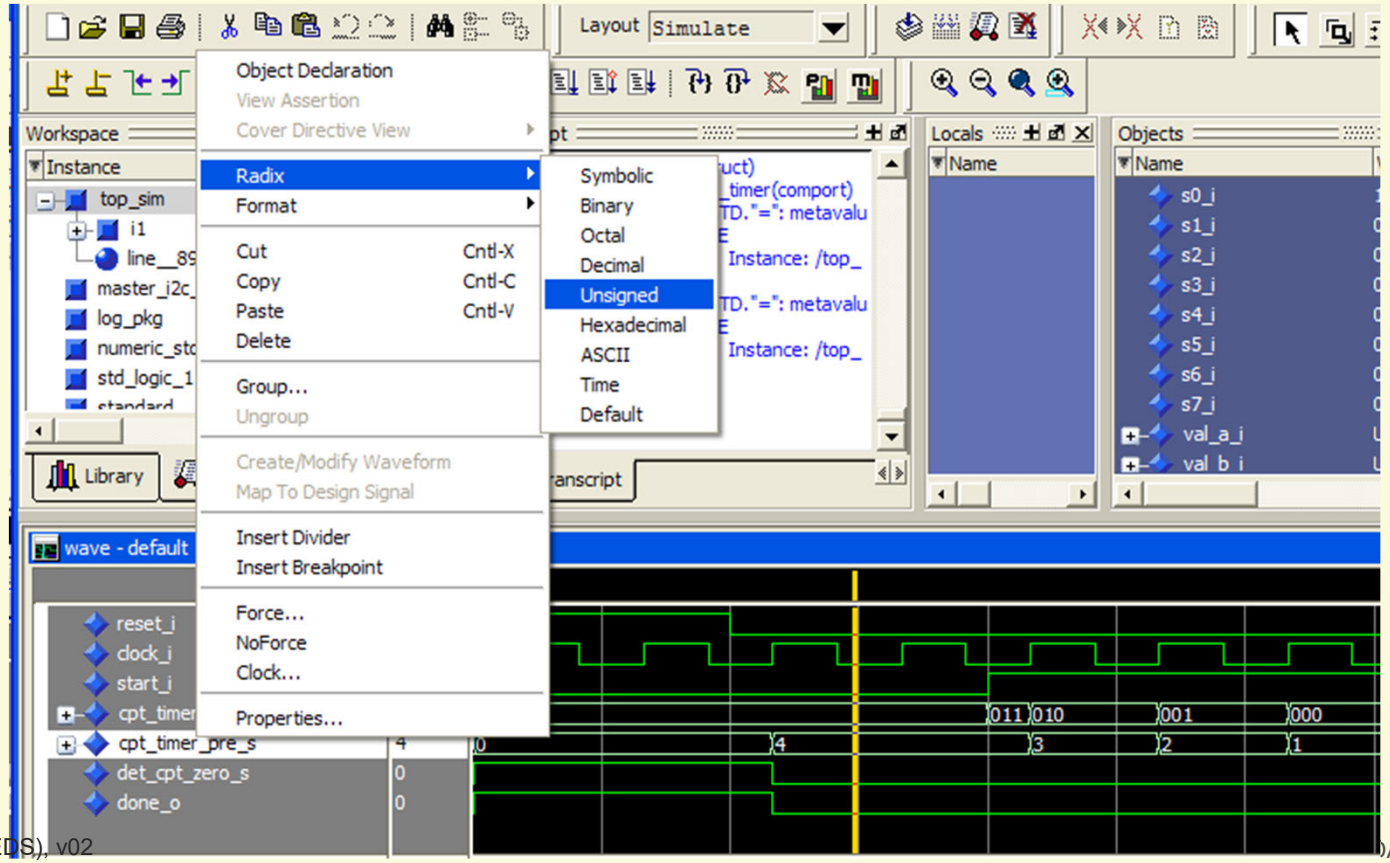
Script pour lancer simulation

```
# ------------------------------------------------
# -- HEIG-VD, institut REDS
# -- File :   run_master_i2c_timer_sim.tcl
# ------------------------------------------------

vlib work
vmap work work

#appel fichier de compilation de Top_Sim_Timer
do comp_master_i2c_timer_sim.tcl

#Charge le projet dans QuestaSim
vsim -voptargs="+acc" work.console_sim

#Affiche les signaux dans la fenetre wave
do wave_master_i2c_timer_sim.tcl
```

- Suite script Tcl: voir fin présentation

Configuration du chronogramme

- Possible de changer format d'affichage

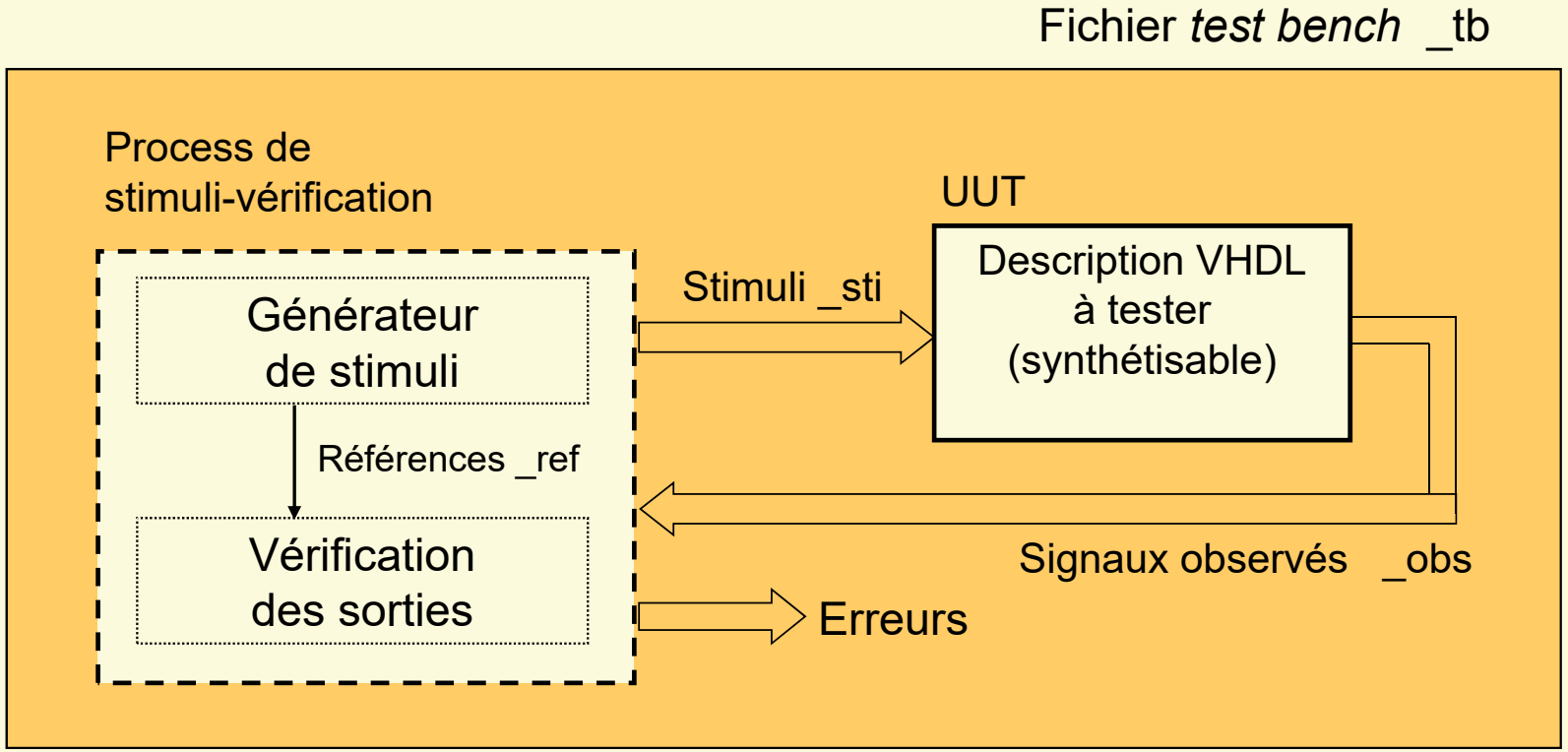


Simulation automatique d'un design …

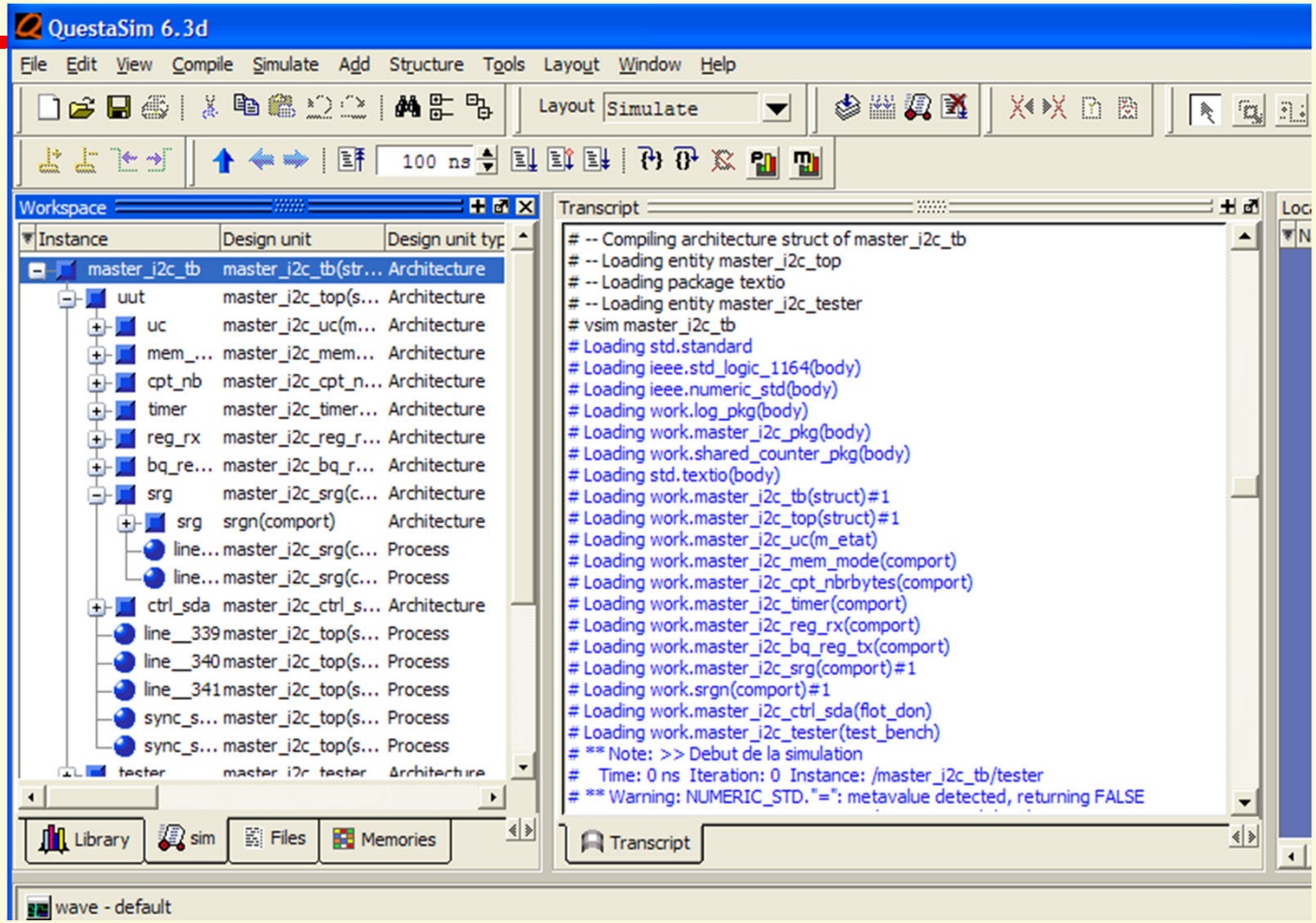
- Description en VHDL d'un banc de test automatique:
  - Génère des stimuli sur les entrées
  - Modélise comportement système externe au design
    - Génération de valeur de référence
  - Vérification des sorties avec le model (référence)
  - Résultat Go/ no Go
    - Possibilité de génération de fichiers de report
  - Affichage des erreurs
  - Fenêtre « wave » pour le debug du design
  - Chronogramme pour la documentation

… simulation automatique d'un design

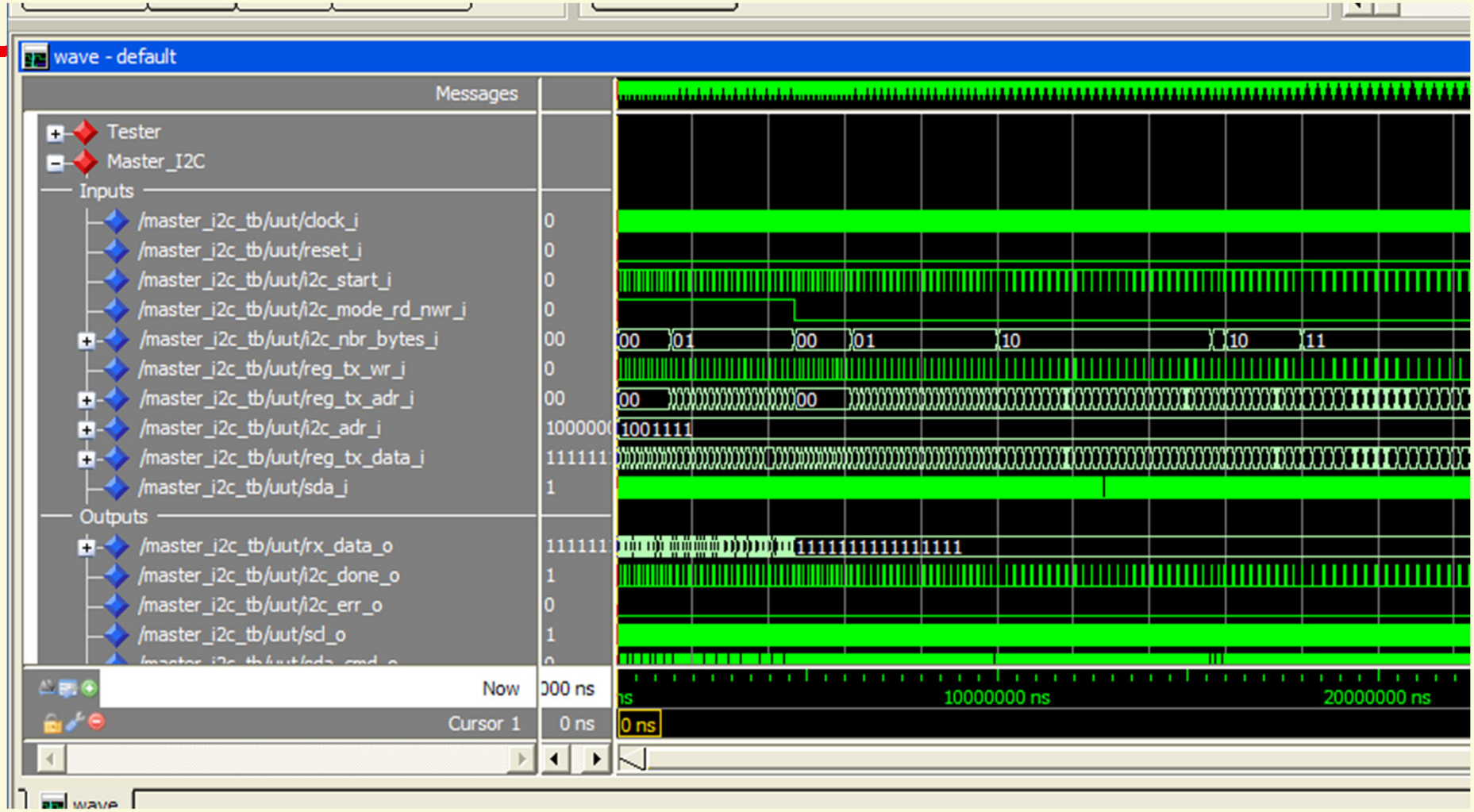
- Structure d'un banc de test :



Simulation d'un projet complexe …

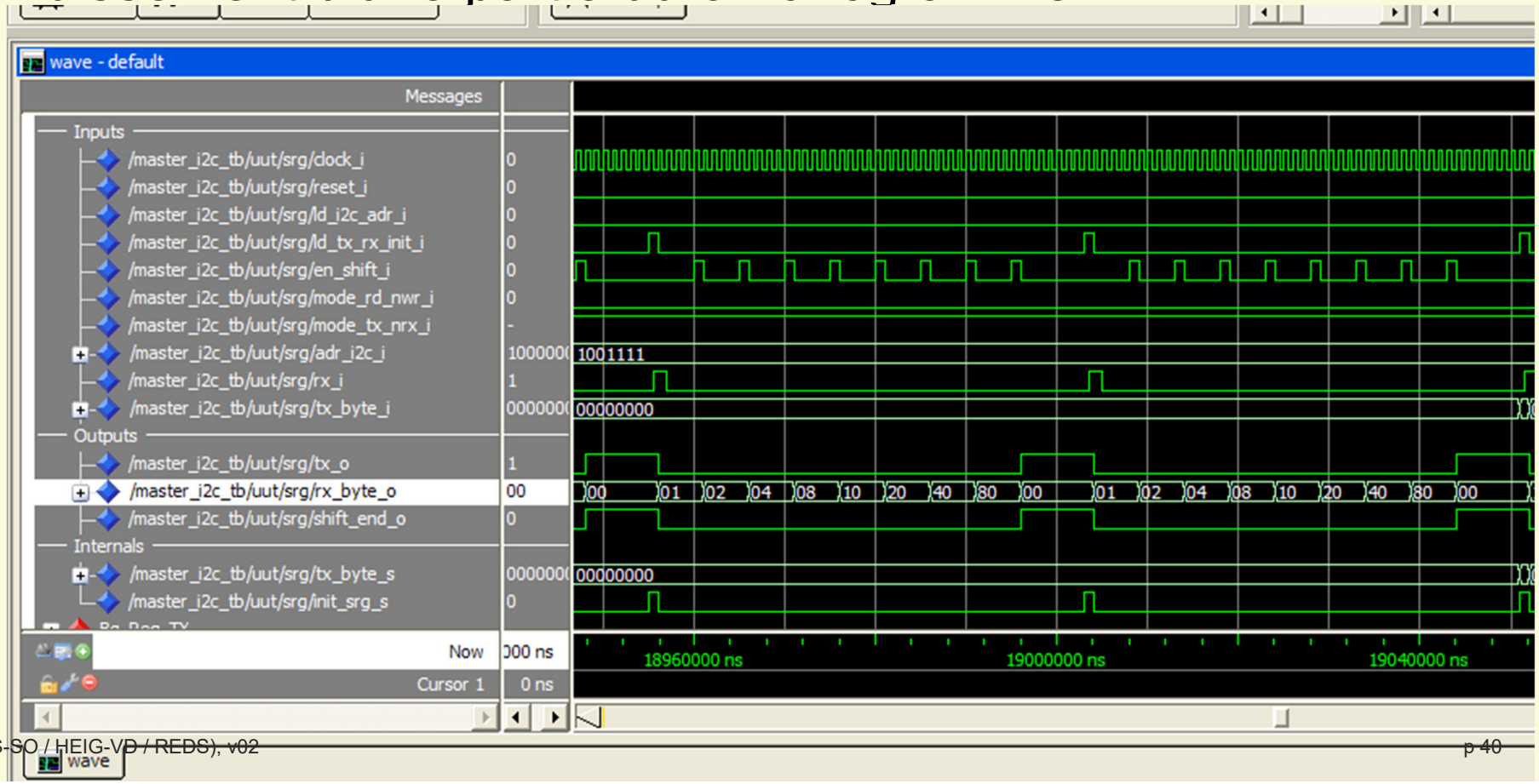


… simulation d'un projet complexe …



… simulation d'un projet complexe

- Agrandissement d'une partie du chronogramme



Simulation Go/ no GO

Copie de la fenêtre log du simulateur

```
# ** Note: >> Debut de la simulation
#    Time: 0 ns  Iteration: 0  Instance: /master_i2c_tb/tester
…
# ** Warning: NUMERIC_STD."=": metavalue detected, returning FALSE
#    Time: 0 ns  Iteration: 0  Instance: /master_i2c_tb/uut/timer
# ** Warning: NUMERIC_STD."=": metavalue detected, returning FALSE
#    Time: 0 ns  Iteration: 0  Instance: /master_i2c_tb/uut/cpt_nb
…
#    Time: 6696502 ns  Iteration: 0  Instance: /master_i2c_tb/tester
# ** Note: >> Debut d'une transmission donnee, stimuli (I=48)
#    Time: 6871502 ns  Iteration: 0  Instance: /master_i2c_tb/tester
# ** Note: >> Debut d'une transmission donnee, stimuli (I=49)
…
# ** Note: >> Debut d'une transmission donnee, stimuli (I=165)
#    Time: 33048502 ns  Iteration: 0  Instance: /master_i2c_tb/tester
# ** Note: ------------------------------------------------
#          >>Nombre d'erreur(s) détectée(s) = 0
#          >>Le design correspond au cas simulés
#          >>Fin de la simulation
#    Time: 33173502 ns  Iteration: 0  Instance: /master_i2c_tb/tester
```

Synthétiseur

- Traduction de la description VHDL en net-liste
- Synthèse adaptée selon la technologie utilisée
  - configurer la famille de PLD utilisé
  - utilisation d'algorithme d'optimisation différencié
- Possible configurer optimisation
  - surface/vitesse Fmax
  - signal particulier
- Fourni en sorti une net-list pour logiciel de placement et routage

Outil de placement et routage

- Spécifique aux vendeurs de PLDs
- Connait structure interne des circuits
  - configuration des chemins de routage dans le circuits
  - calcul des temps de propagation
  - calcul de la fréquence max en full synchrone
- Génère un fichier de programmation du circuit
- Programmation du circuit sur la carte via un module USB-JTAG

Terminologie

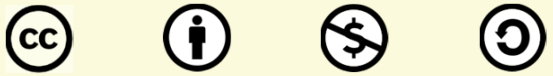
- CPLD : Complex Programmable Logic Device
- EDA : Electronic Design Automation
- FPGA : Field Programmable Gate Array
- HDL : Hardware Description Language
- IDE : Integrated Design Environment
- PLD : Programmable Logic Device
- SoC : System on Chip
- SoC-FPGA : System on Chip on FPGA
- *SoPC : System on Programmable Chip*

Script Tcl pour Questa & ModelSim

HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD

Etienne Messerli

Institut REDS, HEIG-VD



Utilité d'un script

- Permet d'automatiser les commandes nécessaires pour les différentes étapes d'une simulation, soit:
  - création des librairie Work ou spécifique
  - compilation de l'ensemble des fichiers du projet inclus le test-bench
  - chargement du test-bench
  - ouverture des fenêtres et placement de celles-ci
  - ajout des signaux à visualiser ou chargement du format de la fenêtre wave préalablement sauvée
  - lancement de la simulation

Questa / ModelSim et Tcl

- QuestaSim/ModelSim sont écrit en Tcl/Tk
- La fenêtre log est un *wish* Tcl
- Disponibilité immédiate du Tcl
- Lien immédiat avec les signaux VHDL et le Tcl
  - forcer des signaux, ou lire leur état
- Possibilité de réaliser des fenêtre graphique avec Tk
  voir exemple de la console REDS
- d'où :
  intégration complète entre le simulateur et Tcl/Tk

Intérêts pour script Tcl

- Outils gratuit
- Adapter à la gestion de fichier
- Disponible avec de nombreux outils EDA

Contrôle de Questa/ModelSim

- avec un script Tcl toutes les commandes et options de configuration de Questa/ModelSim sont disponibles
- via interface GUI seul une partie des options de configuration sont accessibles
- les commandes réalisées via le GUI sont aussi affichées dans la fenêtre log avec la commande Tcl correspondante
  - solution pour créer un script :
    copier les commandes de la fenêtre log

Commandes Tcl pour Questa/ModelSim

- Principales commandes seront présentées
- Arguments courants seront expliqués
- pour plus :
  voir documentation de ModelSim (menu Help)

Remarque :

Seul les principaux arguments des commandes seront présentés

Utilisation de scripts avec Questa/ModelSim

- Extension des fichiers script :
  - standard : *. tcl, spécifique Questa/ModelSim *.do
- Lancement d'un script dans Questa/ModelSim :
  - Lancer QuestaSim ou ModelSim
  - Sélectionner le répertoire de travail via le menu :

    File → Change Directory…
  - Lancer le script Tcl en tapant dans la fenêtre log :

    QuestaSim> do Nom_Srcipt.tcl

Lancement de script : do

- Commande **do**
  - The **do** command executes commands contained in a macro file (script Tcl).
- Synthaxe

  do <filename> [<parameter_value>]

Exemples :

- lancement de script de compilation

  do Nom_Srcipt.tcl
- Chargement d'un format pour la fenêtre wave (voir ci-après)

  do wave_NomDesign.do

Création de librairie : vlib

- Commande **vlib**
  - The **vlib** command creates a design library. Create a library directory.
- Synthaxe

  vlib <Lib_Name>
- Exemple :
  - création d'un répertoire pour la librairie par défaut work
    vlib Work

Mapper une librairie : vmap

- Commande **vmap**
  - The **vmap** command defines a mapping between a logical library name and a directory
- Synthaxe

  vmap <Lib_Name> <dir_Name>
- Exemple :
  - mapper une librairie spécifique avec work
    vmap PCI Work

Rem : commande superflue vmap Work Work

Compilation de fichier : vcom

- Commande **vcom**
  - The **vcom** command compiles VHDL source code into a specified working library (or to the **work** library by default)
- Synthaxe

```
vcom    [-87] [-93] [-2008]          --choix norme VHDL
        [-work <library_name>]       --spécifie le nom de la librairie
                                     --à utiliser
                                     --par défaut Work est utilisé
        <filename>
```

Exemple :

```
vcom -93 –work work  parite.vhd parite_tb.vhd
```

Chargement d'un design : vsim …

- Commande **vsim**
  - The **vsim** command is used to invoke the VSIM simulator
  - Le design (entité) spécifié est chargé dans Questa/ ModelSim
- Synthaxe

  vsim [-t [<multiplier>]<time_unit>] [-novopt] <library_name>.<design_unit>

  <library_name> nom de la librairie, par défaut *work*

  <design_unit> nom de l'entité du test-bench à simulé

  -t option permettant de changer le pas de simulation par défaut 1 ns

  -novopt option pour ne pas optimiser les signaux/var.

… chargement d'un design : vsim

Exemples :

- chargement simple d'un design :

  vsim -voptargs="+acc" work.parite_tb

- chargement avec choix d'un pas de simulation de 100 ps :

  vsim -voptargs="+acc" –t 100 ps work.parite_tb

Ouverture de fenêtres : view

- Commande **view**
  - The **view** command opens a ModelSim window
- Synthaxe

  view <window_type>

  <window_type> : nom d'une fenêtre de Questa/ModelSim,
  soit memory, process, signals, source, structure,
  variables, wave, …

  Exemple :

  view signal --ouvre la fenêtre des signaux

  view wave --ouvre la fenêtre wave

Ajout de signaux : add wave

- Commande **add wave**
  - The **add wave** command adds the following items to the List window: VHDL signals and variables
- Synthaxe

  add wave [*] [<item_name>]

  <item_name> nom des signaux avec noms des entités

  Exemples :

  add wave *

  add wave /uut/reg/clk

Chargement d'un "format" :

- Possible de sauver un format de la fenêtre Wave
  - Configurer une fenêtre Wave avec tous les signaux souhaités (évtl de plusieurs hiérarchies)
  - Sauvez le format depuis la fenêtre Wave :
    - menu File → save → format "wave_NomDesign.do"
- Commande **do wave_NomDesign.do**
  - The **do** command executes commands contained in a macro file.

Exemple :

do wave_Parite.do

Lancement de la simulation : run

- Commande **run**
  - The **run** command advances the simulation by the specified number of timesteps
- Synthaxe

  run [<timesteps>[<time_units>]] | [-all]

  Exemple :

  run 100 ns

  run –all

Exemple de script : Parite ...

```
#script pour design Parite

#create library work
vlib work
#map library work to work
#vmap work work

#compilation des fichiers vhd
vcom -93 -work work   Parite.vhd
vcom -93 -work work   Parite_tb.vhd

...
```

… exemple de script : Parite

```
...

#chargement du tb pour simuler
vsim -voptargs="+acc" work.Parite_tb

# Ouvre les fenetres.
view signals
view wave

#ajout des fichiers dans la fenetre wave
add wave sim:/parite_tb/*

#lancement de la simulation
run -all
```

Questions !