Tutorial Simili VHDL

A – Installation du logiciel de simulation VHDL Simili

1. Télécharger le programme sur le site de l'éditeur, disponible sous Linux et Windows :

http://www.symphonyeda.com/products.htm

2. Lancer l'interface graphique du programme : sonata (c'est la version gratuite, contrairement aux versions standards et professionnelles)

File Edit Project Comple Simulate Window Help □ 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
□ 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
Workspace: No top level
■ Modules
Scope = None Note: License checkout for 'sonatafree' succeeded Note: You are using the FREE edition of Sonata. Please consider purchasing a license. Warning: Some functionality may be disabled or restricted. Note: You can also request a 'Trial License' to experience the full product Note: using the Licensing Wizard (use Help->Licensing Wizard to launch the wizard) * Console / Command Log / Code Coverage /

B – Gestion d'un projet VHDL

<u>1^{ère} étape – création d'un workspace</u>

File -> New Workspace

Choisir un répertoire pour les fichiers VHDL et donner un nom à l'espace de travail

2^{ème} étape – édition des fichiers VHDL

Dans l'éditeur de texte, éditer vos fichiers VHDL. Un nom de fichier VHDL doit avoir le même nom que le composant qu'il décrit et l'extension ".vhd" ou ".vhdl".

3ème étape – compilation d'un fichier VHDL

Compile -> Compile File 'mon_composant.vhd'

Répondre 'Yes' à la question posée qui vous propose de rajouter le composant au workspace. Il apparaît alors dans la partie gauche



4^{ème} étape – Simulation

- Pour simuler un composant, il faut qu'il soit autonome. Si ce n'est pas le cas, faire un composant de test (Test Bench) qui fournira un jeu de valeurs pour les entrées du composant à tester.
- Sélectionner le composant de plus haut niveau à simuler (celui qui contient les autres)

Simulate -> Select Top Level

Lancer le simulateur

Simulate -> Restart

Temps total de simulation

portes_ou - Symphony EDA Sonata	3.1 - [Waveform0.wfs]			
	им пер 1 🕼 🎊 📢 🔝 🂦 💌	🚟 🖽 🖹 💷 🔶 🗍 us	- - - - - - - - - - - - - - - - - - -	
Workspace: Work = portes_ou	Signals	* Values	2	
ice statandard: ice s				
composants	Source: Simulator Cursor: Ops	os 500	Ons lus 1.5u	s 2us 2.5us
Files		▶ ↓ ↓ ▲ b_or2.vhd / ➡ or_2.vhd /		
Scope = 11 or 2:	Reading	portes_ou.sym\or_2\prim.	var	
Y₂ Signals Y > Lues JT x_in U' JT y_in U' JT fout U'	signaux	bortes ou.sym/or_2/_or_2 = = 6 ents = 1 ses = 2 tion Elapsed T	_arch2.var ime: 00h:00m:00s:016ms	
	8	· ·		
Signals Variables		/ Command Log / 🖹 Code Co	verage /	

- Sélectionner les signaux à visualiser (en bas à gauche). On peut naviguer dans les composants (fenêtre en haut à gauche) pour sélectionner des signaux internes (mais pas les variables dans la version free).
- Régler le temps de simulation
- Lancer la simulation <u>Simulate -> R</u>un

👑 portes_ou - Symphony EDA Sonata 3.1 - [Waveform0.wfs *]					
File Edit Project Compile Simulate Window Help					
]D ☞ 🖬 🗿 🐒 % 🖎 🖾 🖙 🖾 🖾 🚺 📜 🗼	: □来花刻11,2,2,2,2,1,1,2,2,2,1,4,4,5,×,2,1,1,2,2,2,1,4,4,2,2,2,1,2,2,2,1,2,2,2,1,2,2,2,1,2,2,2,1,2,2,2,1,2,2,2,1,2,2,2,1,2,2,2,1,2,2,2,1,2,2,2,2,1,2,2,2,2,1,2				
Workspace: Work = portes_ou Toplevel = tb_or2	als // Values // 461783.4496 // // // // // // // // // // // // //				
Gristdistandard: Griece:std_logic_1164: Eref: tb_or2: Source: Cursor:	Image: Simulator S38216.56ps 30 100ns 500ns 700ns 900ns 1.1us 1.3us				
Files Modules	eform0.wfs* 🖹 tb_or2.vhd / 🖹 or_2.vhd /				
Scope = :tb_or2: Signals X Values Image: X_in '1' Image: Y_in '0' Image: Y_on '0' Image: Y_on '1'	<pre># of Processes = 2 Design Load/Elaboration Elapsed Time: 00h:00m:00s:015ms % run {1 us} Warning: Free edition - running at reduced performance levels! Simulation stopped at: 1 us Elapsed Time: 00h:00m:00s:063ms %</pre>				
	×				
Gonzole / Command Log / Code Coverage /					
	Ln 1, Col 1 Status: Stopped 1 us+2				

- Vous pouvez ensuite :
 - o zoomer/dézoomer
 - $\circ \quad$ prendre des mesures sur les signaux ou des écarts de temps.
 - Modifier le format de visualisation des données (binaire, décimal, héxa, ...)