

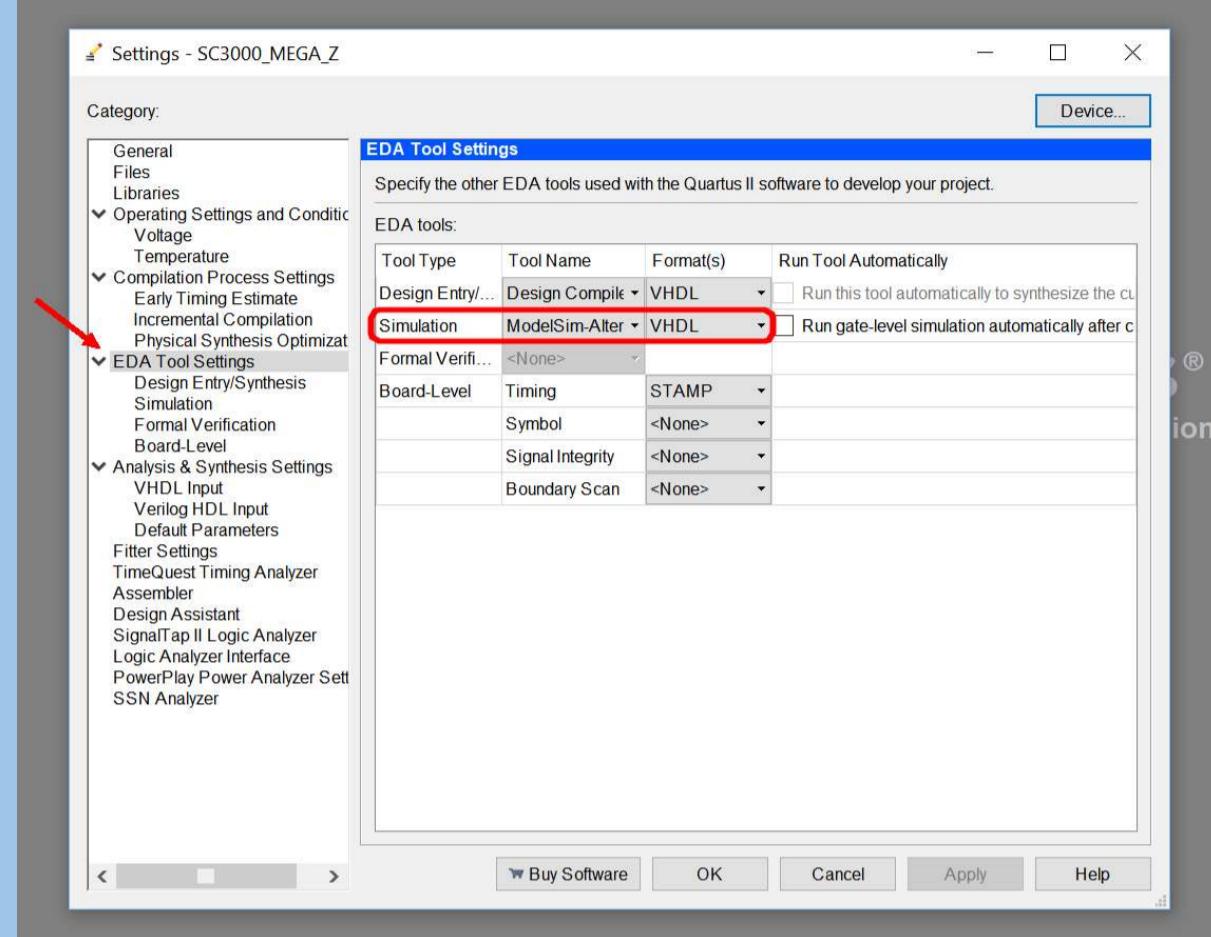
Altera Quartus Modelsim

Ce "tuto" n'aborde que la partie simple de la simulation, ca permet de vérifier le bon fonctionnement du VHDL et de contrôler que les signaux soit bien conformes pendant les divers changements d'états.

En premier lieu il faut bien configurer son projet Quartus.

pour ce dans les paramètres du projet vérifier qu'il est bien indiqué dans EDA tool>Simulation = ModelSim-Altera (à modifier selon la version que vous utilisez de Quartus).

Exemple sur mon projet SC3000/ORIC

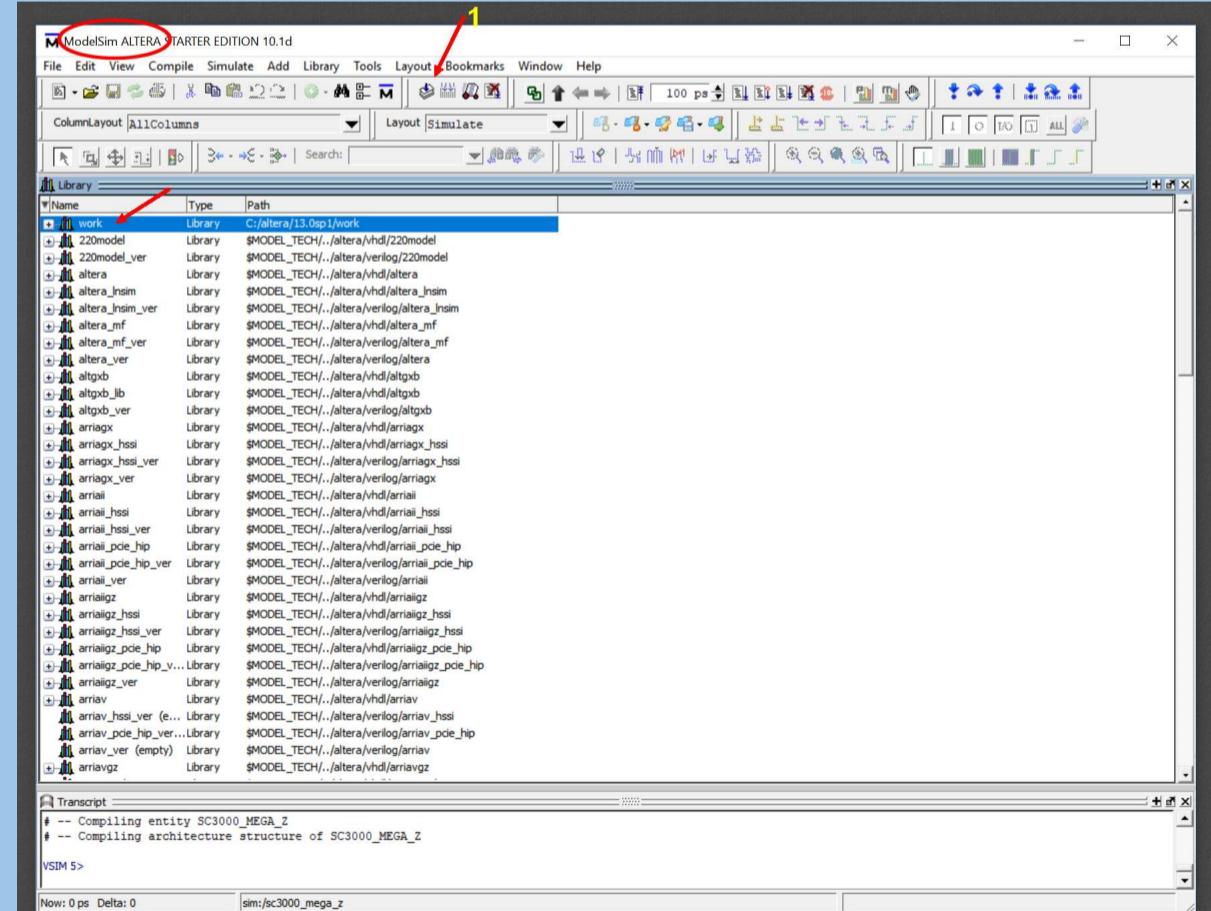


Ensuite pensez à faire une compilation du projet sous Quartus pour que les fichiers nécessaires à la simulation soit générés.

Ouvrir ModelSim-Altera soit par le menu windows soit par l'icone "RTL simulation" dans Quartus.

Une fois ModelSim ouvert vous devrais voir dans la liste des librairies un dossier "work", c'est dans celui la que vous devrais trouver tous vos projets.

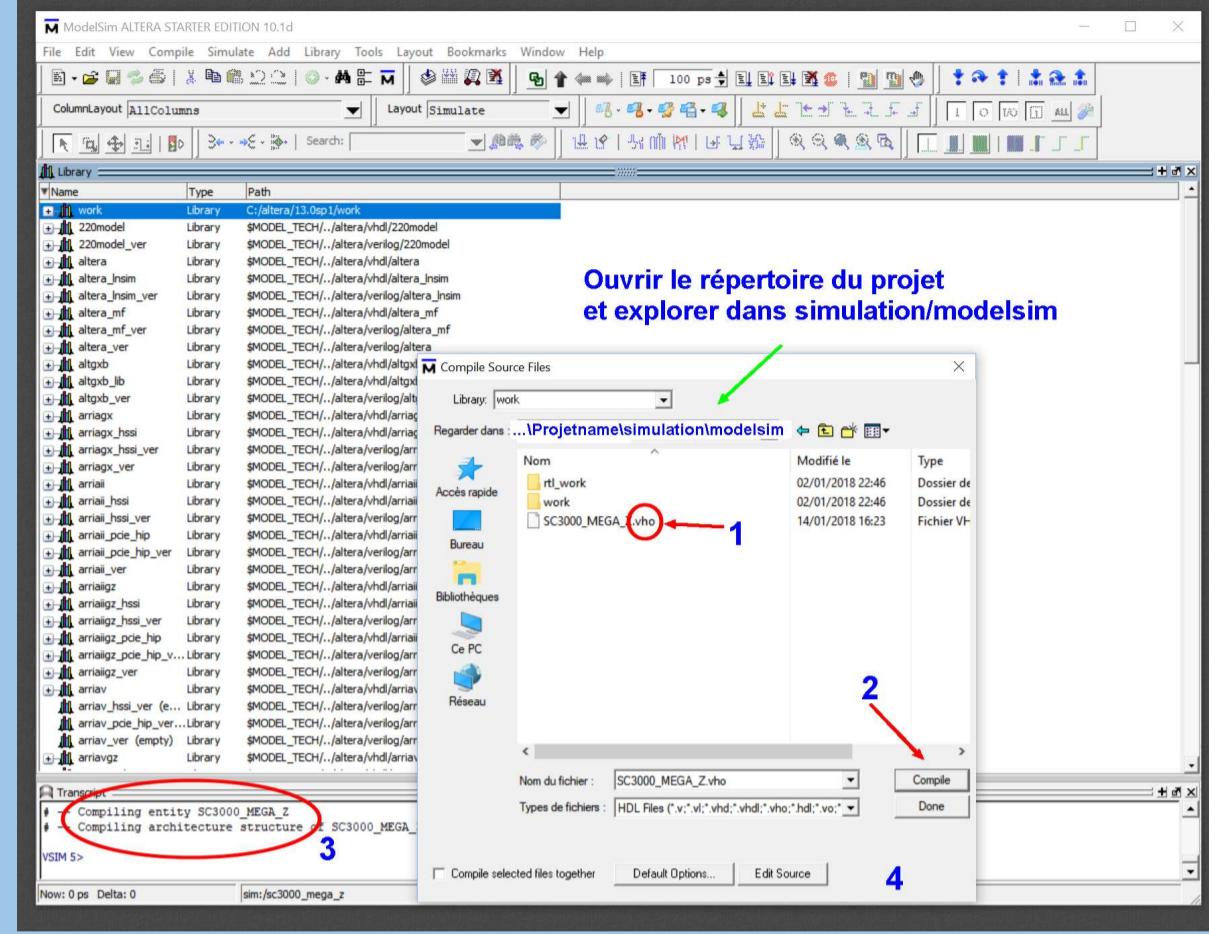
On va compiler dans modelsim le projet, on appuie sur l'icone compile (1).



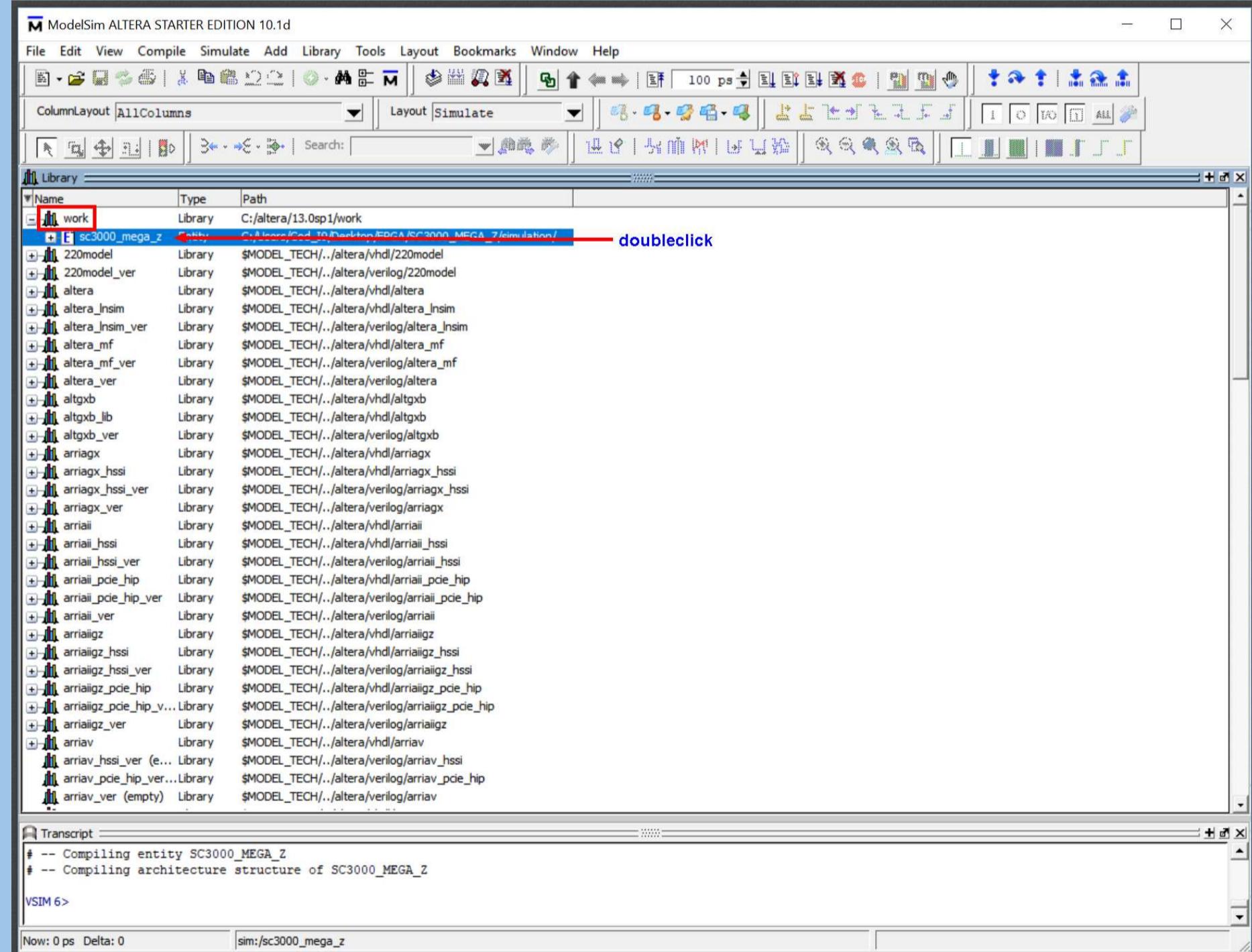
On recherche le dossier du projet et on ouvre les sous dossiers simulation et modelsim, pour trouver le fichier .VHO.

VHO correspond à la simulation du projet complet, mais on peut aussi avoir besoin de simuler qu'une partie du projet dans ce cas ou compilera directement un fichier VHDL.

Une fois le fichier choisi on click sur compile (2), dans la fenêtre du bas on peut voir les résultats de la compilation si tout c'est bien passé on click sur done.



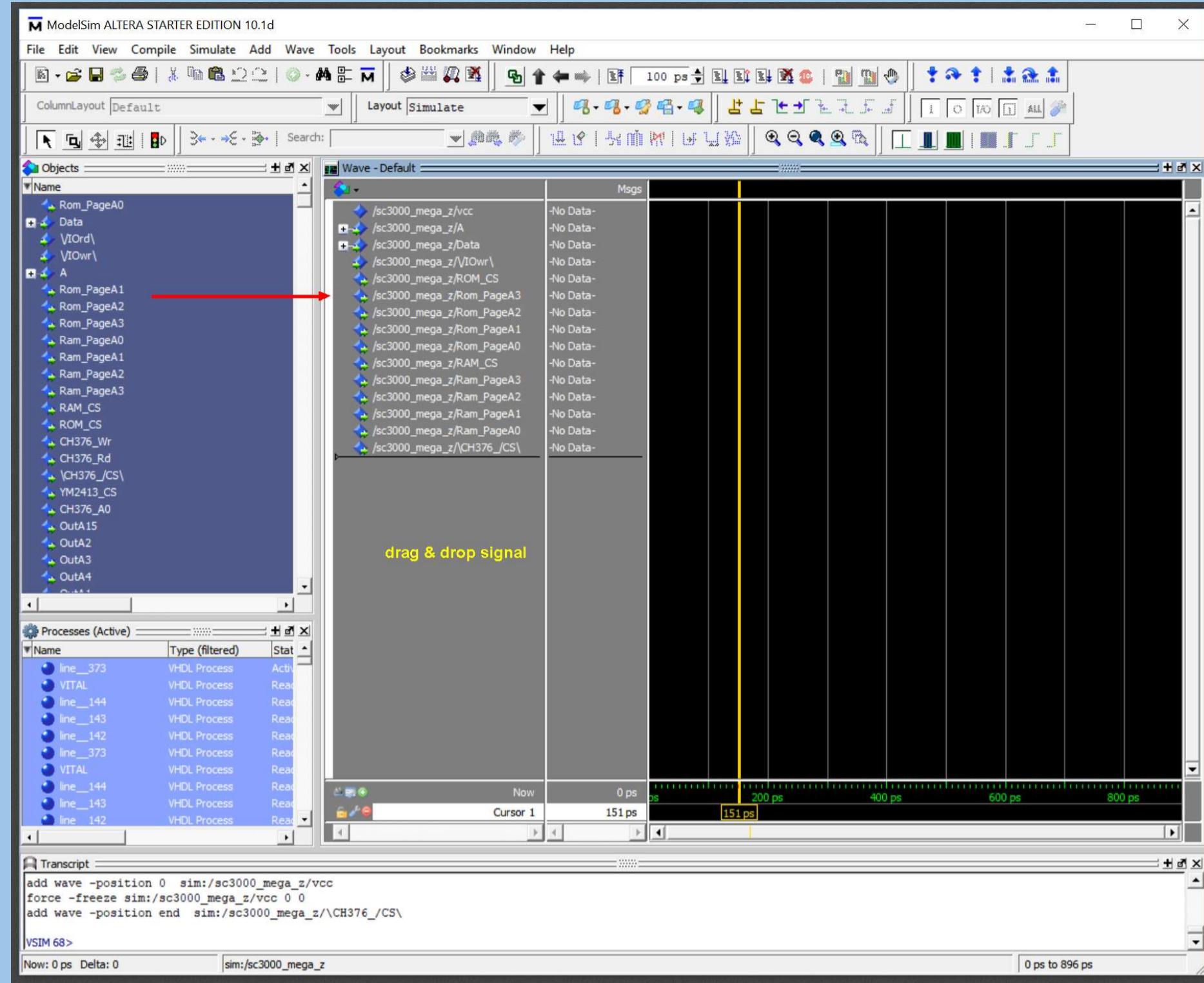
On click sur le + à côté de work si le dossier n'est pas déjà déplié, puis on double click sur le projet choisi.



De nouvelles fenêtres doivent s'ouvrir avec les signaux disponibles(objects), les processus (processes), et la partie analyse (wave).

Pour commencer vous devez faire glisser les signaux que vous voulez analyser vers la fenêtre wave.

Il est possible de rajouter ou d'enlever des signaux au fur et à mesure de l'exécution de la simulation.



Une fois tous les signaux sélectionnés il faut les configurer pour pouvoir les simuler et analyser leurs comportements.

Il y a plusieurs type de signaux en entrées (IN), en sorties (OUT) et bidirectionnel (BIDIR), un icône devant chaque nom de signaux indique le sens des signaux.

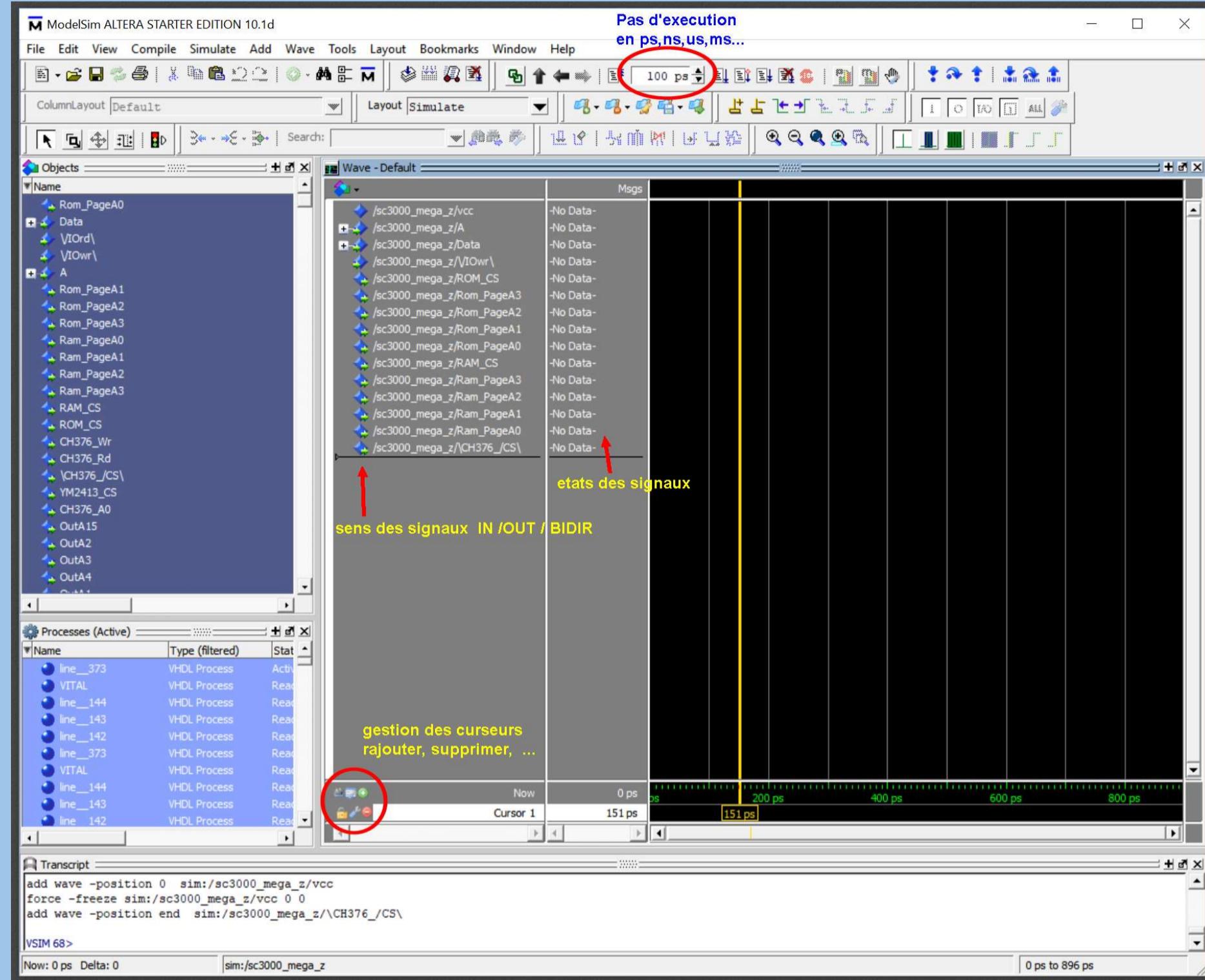
Je conseil de toujours paramétrer le signal VCC à 0 pour le début de la simulation, ca permet de commencer la simulation en simulant l'arrêt de la machine.

Ensuite paramétrez tous les signaux IN et laisser les signaux OUT dans l'état il seront mis à jour par le simulateur.

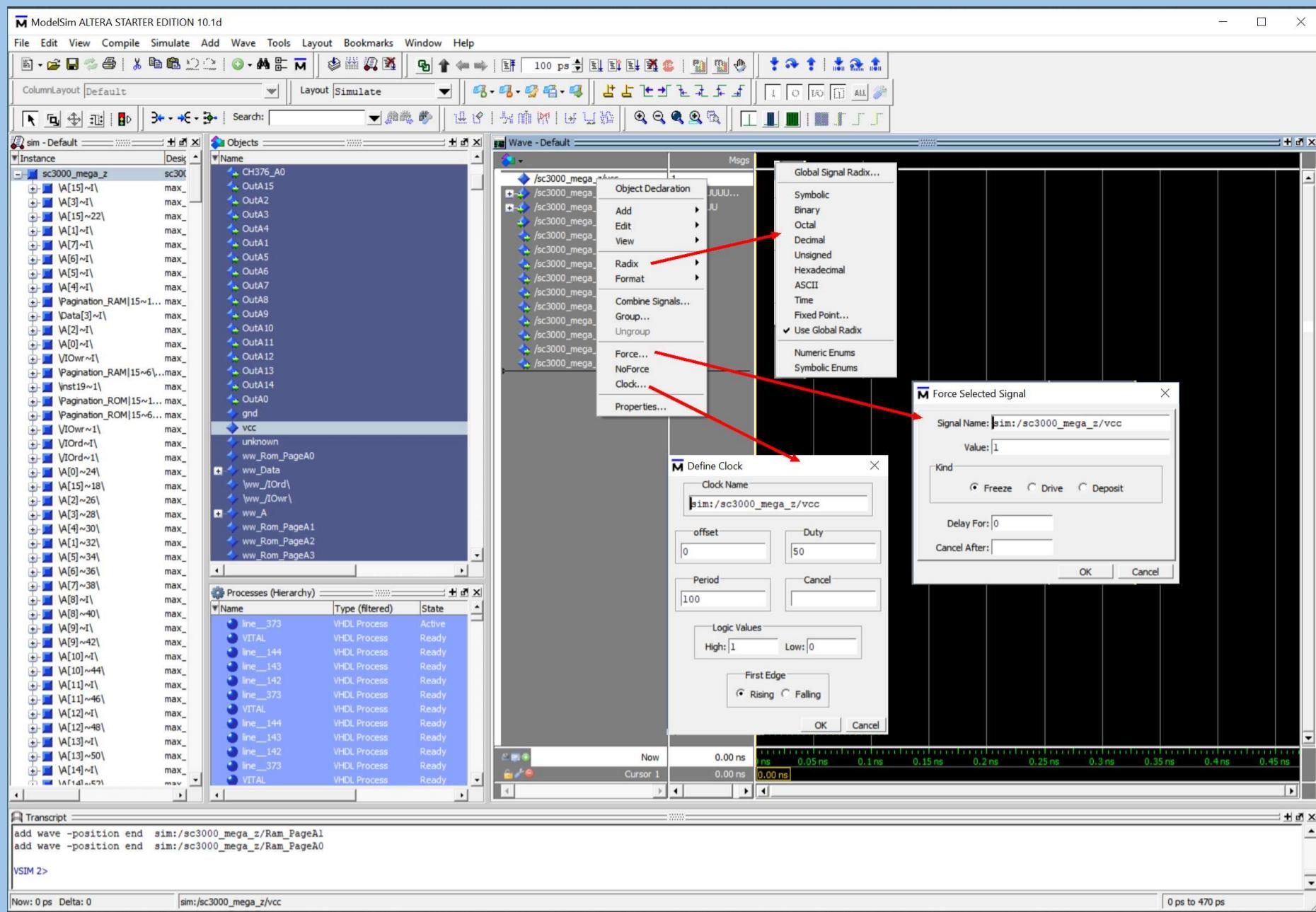
Je conseil d'utiliser le mode pas à pas qui est à mon avis le plus pratique pour de petits montages (décodage d'adresse, bascule, ...)

pour ça vous devez si besoin modifier la valeur du pas d'exécution de la simulation dans la partie entourée en rouge dans l'image ci dessous.

Dans mon exemple j'ai simulé un décodage d'adresses et une pagination sur de la RAM et de la ROM en utilisant les I/O (A8-A0) pour le Z80, la page 2 (A15-A0) pour le 6502.



On peut gérer plusieurs curseurs pour marquer différents états/temps, ainsi que de modifier l'affichage de la partie "wave" (unité de temps, taille de la grille,...).



La configuration des signaux permet de définir leurs états et leurs représentations ainsi qu'un comportement "scripté" (clock, valeur pendant x delay, ...), ainsi que définir des couleurs différentes pour une meilleure visibilité.

Pour commencer je conseil de définir les radix sur hexadécimal pour les adresses et datas c'est plus pratique à mon sens pour suivre les résultats, mais vous pouvez laisser sur binaire si cela vous arrange.

Ensuite on peut forcer un signal à une valeur donné ou configurer une valeur pendant un délai défini.

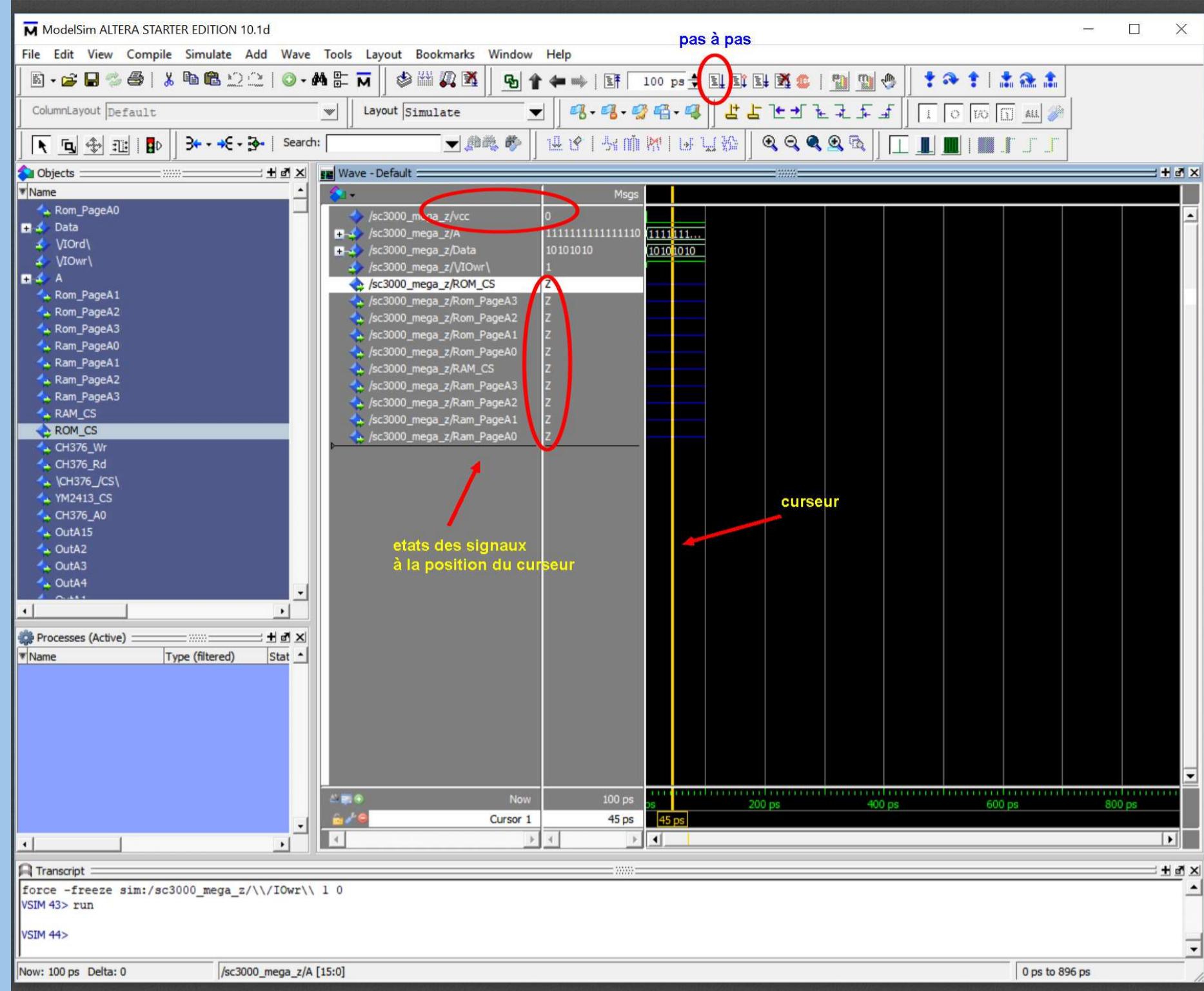
On peut aussi attribuer une horloge (clock) à un signal et définir les delay et le sens du front de démarrage.

Chaque signal doit être paramétrés au cours de l'exécution de la simulation si l'on veut pouvoir simuler les changements d'états des divers signaux d'entrées et visualiser les comportements des signaux de sorties.

Donc pour commencer forcer le signal VCC à 0.

Et configurez tous les signaux IN comme vous le souhaitez .

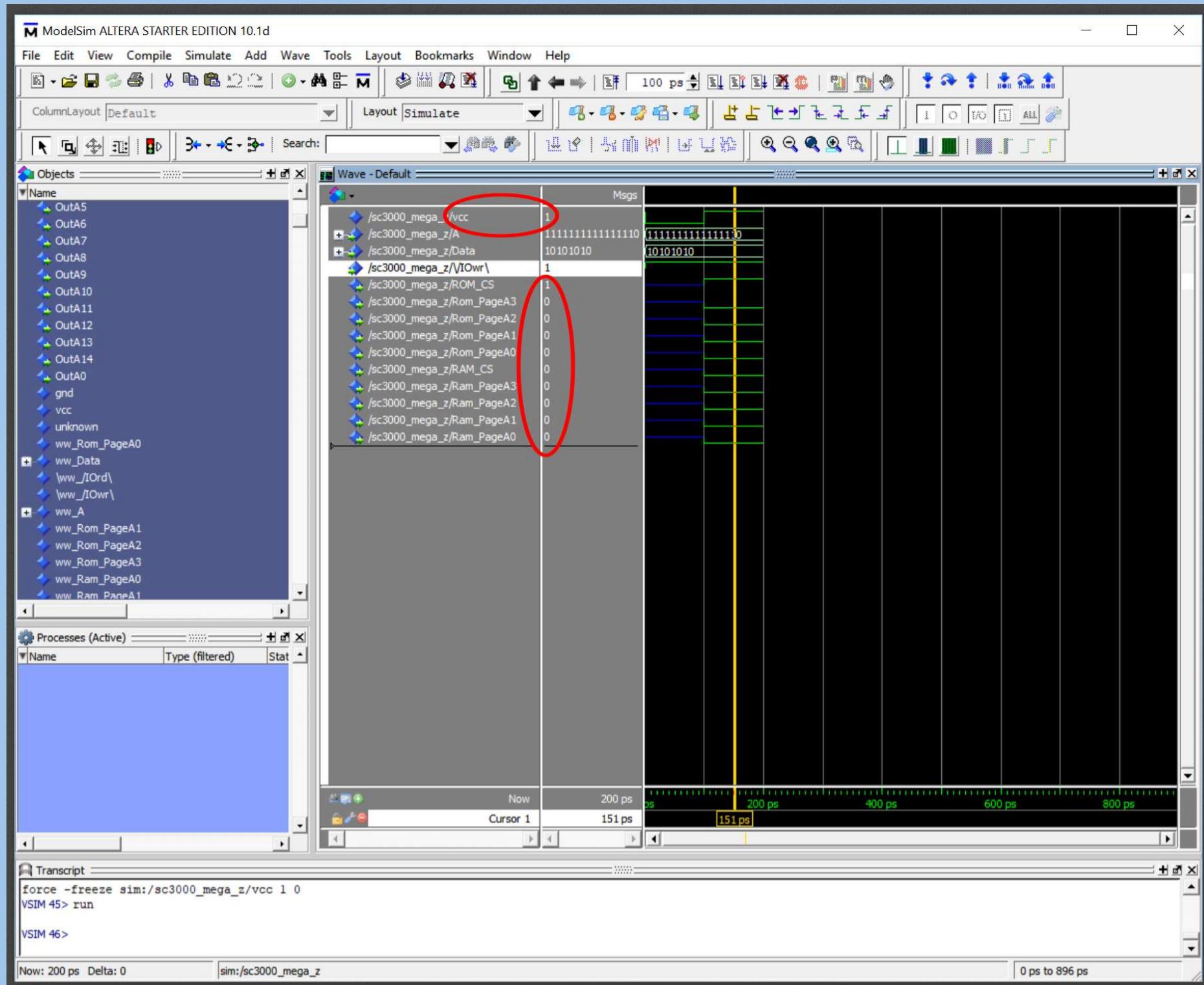
Commençons la simulation en appuyant sur l'icône pas à pas.



Comme nous avons défini VCC à 0 les signaux en sorties sont dans un états indéfinis ce qui est normal ;)

Le curseur peut être placé n'importe où sur la base de temps, dans la partie gauche vous pouvez lire l'états des signaux correspondant au curseur.

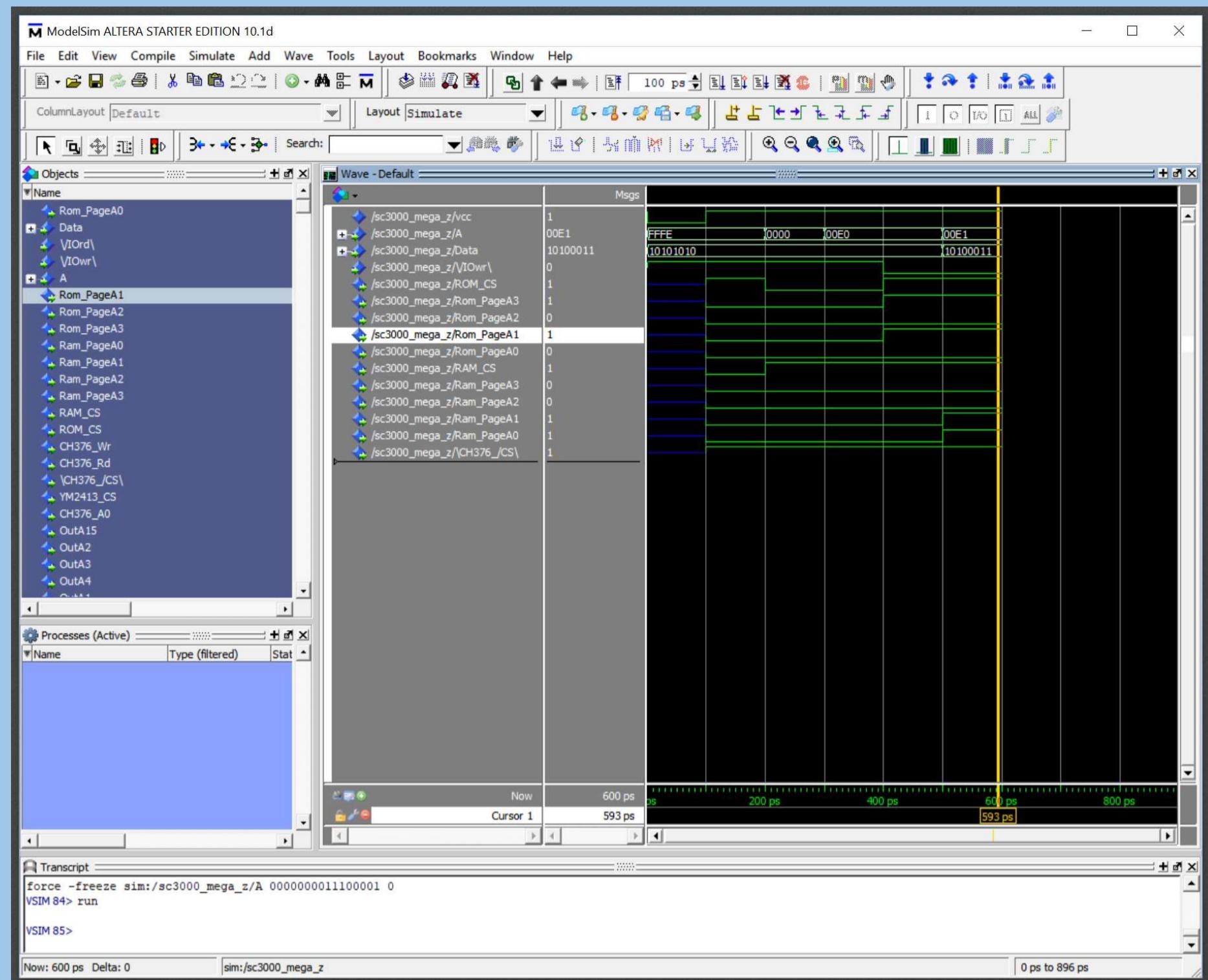
Maintenant forcez le signal VCC à 1, et cliquez sur l'icône pas à pas pour faire avancer la simulation d'un pas.



On peut voir que tous les signaux en sorties ont changés d'états, ça nous permet de vérifier que lors du démarrage les signaux sont bien aux niveaux attendus, sinon il faut modifier le code VHDL ou le schémas et recompiler dans Quartus et ModelSim et relancer une simulation .

Le principe est simple on modifie 1 ou plusieurs signaux et on clique sur pas à pas pour visualiser le comportement des divers signaux.

Voici un petit exemple en ayant modifié plusieurs fois l'état des signaux d'entrées, on voit bien que les sorties réagissent conformément aux signaux d'entrées :



Maintenant on peut modifier chaque signal pour vérifier tous les cas possibles et vérifier que les signaux se comportent de manières attendu.

Ce n'est pas une simulation absolue, mais ça permet de vérifier le bon fonctionnement et les différents cas possible.

Dans l'exemple choisi j'ai pris des timing évidemment pas réaliste mais le but est de vérifier que les changements d'états soi conformes, évidemment on pourrait utiliser des timing réaliste en ns ou μ s, mais c'est très difficile de connaître le timing exacte de la machine cible en exécution.

Il y a beaucoup d'option à vous de découvrir celle qui peuvent vous être utile selon vos projets ;)