Simulateur de netlist Sim

Cette archive contient les sources du programme `sim`, un simulateur de netlist.

Installation

Prérequis

L'installation nécessite make, ocamlopt, ocamldep et menhir. Le plus simple est d'utiliser Linux.

Sous Linux

make est sûrement déjà installé, ocamlopt et ocamldep sont disponibles dans le paquet ocaml et menhir peut également se trouver dans le gestionnaire de paquets.

(Testé sous Ubuntu 12.04)

Sous Windows

Pour installer make :
http://gnuwin32.sourceforge.net/packages/make.htm

Pour installer ocaml (contenant ocamlopt et ocamldep) :
http://protz.github.io/ocaml-installer/

Pour installer menhir, il faut compiler les sources se trouvant ici :
http://gallium.inria.fr/~fpottier/menhir/menhir.html.fr

(Testé sous Windows 7 avec mingw32-make au lieu de GNUmake)

Sous MacOS

1. Installer Linux sur une machine virtuelle
2. Suivre les instructions pour Linux

(J'ai pas de Mac sous la main pour tester)

Compilation

- Taper `make sim` après s'être placé dans le dossier contenant les sources pour uniquement compiler le programme `sim`.
- Taper `make all` (ou juste `make`) pour compiler puis lancer un exemple d'utilisation.

Le programme `sim` ainsi créé est autonome.

Utilisation

Fonctionnement général

Sim prend en entrée les fichiers suivants :

- Un fichier netlist contenant la description du circuit (non ordonnée)
- Un fichier d'entrées sur n cycles
- Un fichier décrivant la ROM (facultatif)

Le fichier ROM est facultatif. S'il n'existe pas, un avertissement est affiché dans la console mais la simulation est tout de même effectuée. La ROM est alors identiquement nulle.

Le programme écrit ensuite les fichiers suivants :

- Un fichier contenant les sorties sur n cycles correspondant aux entrées données et avec l'éventuel état de la ROM indiqué.
- Un fichier netlist ordonné (sur demande)

Syntaxe des différents fichiers

La syntaxe de la netlist peut se trouver ici : (en seconde partie)
http://www.di.ens.fr/~bourke/minijazz.pdf

La syntaxe des fichiers d'entrée, de sortie et de description de la ROM est la suivante :

- Les valeurs sont représentées par des caractères : `1` représente un bit activé tandis qu'un `0` représente un bit désactivé.
- Les espaces et tabulations sont ignorées.
- Les valeurs des nappes sont données entre crochets.
- Les retours à la ligne annoncent un nouveau cycle, de sorte que chaque ligne corresponde à un cycle.
- Des commentaires peuvent être ajoutés en fin de ligne après le caractère `#`.
- Une ligne commençant directement par le caractère `#` n'est pas considérée comme correspondant à un cycle.

Pour la rom, les retours à la ligne ne sont pas interprétés. Les valeurs données sont enregistrées dans la ROM par ordre croissant d'adresse, en partant de l'adresse 0 et par pas de 1. Une adresse peut contenir un bit ou une nappe.

Syntaxe de la commande sim

Le seul paramètre obligatoire est le fichier netlist à simuler. Sim s'appelle alors comme suit :

`sim [filename].net`

Par défaut, le Sim utilise les fichiers suivants :

- `input.sim` pour les entrées
- `rom.sim` pour la rom

- `output.sim` pour les sorties
- `sch.net` pour la netlist ordonnée

Les options doivent être indiquées avant le nom du fichier netlist.

Pour modifier le fichier d'entrées, utiliser l'option `-input [filename].sim`.

Pour modifier le fichier de description de la rom, utiliser l'option `-rom [filename].sim`.

Pour modifier le fichier de sortie, utiliser l'option `-output [filename].sim`.

Pour demander à ce que le fichier `sch.net` soit écrit, utiliser l'option `-sch`. Le nom du fichier ne peut être modifié.

Lorsque l'on travaille sur plusieurs fichiers simultanément, l'utilisation des options `-input`, `-rom` et `-output` à chaque appel de la commande sim peut se révéler lourd, c'est pourquoi l'option `-p` (pour « préfixe ») a été ajoutée. Cette option permet de mofifier les valeurs par défaut de l'entrée, la description de la rom et la sortie en respectivement `[filename]_input.sim`, `[filename]_rom.sim` et `[filename]_output.sim` où `[filename]` est le nom (sans extension) du fichier netlist à simuler.

Une dernière option, `-n [n]`, permet de ne simuler que `[n]` cycles sur ceux indiqués dans le fichier d'entrée. Si cette valeur dépasse le nombre de cycles prévus par le fichier d'entrée, cette option est ignorée et tout le fichier est alors simulé.

Informations complémentaires

Bugs connus

Aucun n'a été détecté pour le moment

Historique de développement

Sim a été développé par Elie Michel dans le cadre du cours « Système digital, de l'algorithme au circuit » dispennsé par Jean Vuillemin à l' École Normale Supérieure.

Le développement de Sim est la première partie du projet coordonné par Timothy Bourke qui doit être rendu pour ce cours.