Systèmes embarqués et temps-réel
(SMB204/USRS2A)

TP : Compilateur lus2ada

Le but de ce TP est d'implanter un compilateur de Lustre vers Ada qui réalise la compilation classique dite en « boucle simple ». Le compilateur sera écrite en Java et les sources correspondant à la définition de la syntaxe abstraite de Lustre (plus exactement du format `ec` produit par l'outil standard `lus2ec`) sont fournis, ainsi que l'analyseur lexical et syntaxique.

Le sources fournis compilent avec Java 8, et nous conseillons d'utiliser Eclipse comme environnement de développment.

Pour compléter le compilateur, en partant du code fourni, vous devez :

- Ecrire tout d'abord un *printer* simple, qui réaffiche un programme avec la syntaxe de Lustre.
- Ecrire un autre *printer* qui génére du code impératif en « boucle simple » en respectant la syntaxe Ada. Les principes sont les suivants :
  - les variables de flot deviennent des variables impératives (pour les entrées, les sorties et les variables locales), chaque variable est accompagnée d'une nouvelle variable préfixée par `pre_` (cette variable sert à stocker la valeur du tour précédent de la boucle). Toutes ces variables doivent être déclarées avec le bon type.
  - les flèches d'initialisation « `->` » sont traitées ainsi : la valeur de gauche est utilisée quand on affiche le code d'initialisation (le code avant la boucle) et la valeur de droite est utilisée quand on affiche le corps de la boucle.
  - les équations deviennent des affectations.
  - des entrées/sorties sont réalisées à chaque tour de boucle pour permettre de simuler l'execution du code Lustre.
- Ecrire une fonction auxiliaire qui reordonne les équations du noeud lustre de manière à ce que le code Ada généré soit correct dans tous les cas: dans le code impératif, une variable doit être affectée avant de pouvoir être lue (seules les variables en entrées, et les `pre_` sont supposés déjà initialisées).

Un programme d'exemple, source Lustre « `stable.lus` » et code Ada généré « `stable.ada` » vous est aussi fourni.