

TD10: Lustre

Equipe Enseignante PC3R

15/04/2021

1 Méthode: Simuler les flots

Voici un programme lustre :

```
node udtqc (x : int) returns (y : int);
  let y = 1 -> pre (2 -> pre (3 -> pre (4 -> 5)));
tel;
```

1. Quelle est la valeur à chaque instant de $udtqc(x)$? Présenter une simulation du flot dans un tableau.
2. Donner un noeud qui produit le flot des entiers.

2 Compteur dans un musée

On souhaite comptabiliser le nombre de visiteurs présents dans le musée à chaque instant.

1. Compléter le noeud suivant pour compter les entrées et les sorties.

```
node ces (in, out : bool) returns (compteur : int);
```

On suppose qu'à chaque instant, **in** (resp. **out**) simule l'entrée (resp. sortie) d'un visiteur. Les **in** et **out** sont ici simulés manuellement. Ils pourraient être générés à chaque instant par les portiques.

3 Fibonacci

On rappelle la définition des nombres de Fibonacci:

$$F_0 = 1, F_1 = 1, F_{n+2} = F_{n+1} + F_n.$$

1. Proposer plusieurs solutions générant le flot des nombres de Fibonacci.
2. Même question, mais avec une borne **max** pour les valeurs ?
3. Même question, mais avec un indice **max n**.

4 Horloges

1. Compléter le tableau suivant :

X	1	2	3	4	5	6	7	8	9
Y	true	false	true	false	true	false	true	false	true
Z	true	true	false	false	true	false	false	true	false
A = X when Z									
B = Y when Z									
C = A when B									
current(C)									

2. Compléter le nœud `node modulo_n (n : int) returns (max_modulo : int; nombre : int);` permettant de générer le `nombre` qui s'incrémente à chaque instant de 1 et le `max_modulo` correspondant au plus grand nombre inférieur ou égal à `nombre` et modulo `n` (voir l'affichage ci-dessous).

```
$ modulo_n
## STEP 1 #####
n (integer) ? 5
max_modulo = 0
nombre = 0
## STEP 2 #####
n (integer) ? 5
max_modulo = 0
nombre = 1
## STEP 3 #####
n (integer) ? 5
max_modulo = 0
nombre = 2
## STEP 4 #####
n (integer) ? 5
max_modulo = 0
nombre = 3
## STEP 5 #####
n (integer) ? 5
max_modulo = 0
nombre = 4
## STEP 6 #####
n (integer) ? 5
max_modulo = 5
nombre = 5
## STEP 7 #####
n (integer) ? 5
max_modulo = 5
nombre = 6
```

```
## STEP 8 #####
n (integer) ? 5
max_modulo = 5
nombre = 7
## STEP 9 #####
n (integer) ? 5
max_modulo = 5
nombre = 8
## STEP 10 #####
n (integer) ? 5
max_modulo = 5
nombre = 9
## STEP 11 #####
n (integer) ? 5
max_modulo = 10
nombre = 10
## STEP 12 #####
n (integer) ? 5
max_modulo = 10
nombre = 11
## STEP 13 #####
n (integer) ? 5
max_modulo = 10
nombre = 12
## STEP 14 #####
n (integer) ?
```

3. On suppose que le nœud suivant est donné :

```
node foo (n : int; inc : int) returns (resultat : int);
let
  resultat = inc + n - n;
tel;
```

L'opération `+ n - n` est inutile. On veut juste imposer la même horloge pour `inc` et `n` sans quoi l'opération `inc + n - n` est impossible.

4. Compléter le nœud `node modulo_n_clock (n : int) returns (max_modulo : int; nombre : int);` pour générer le même résultat que la question précédente. On souhaite utiliser le test avec `mod n = 0` pour créer une horloge
5. Pourquoi n'a-t-on pas besoin de faire le test `(inc mod n) = 0` dans le nœud `foo()` ?