TP - Opérateurs Temporels

Question 1 – Propriétés de temps logique

a) Écrire un module lustre **never : bool → bool** tel que la sortie **never(a)** vaut **true** à l'instant **i** si et seulement si l'entrée **a** n'a jamais eu la valeur **true** jusqu'au pas **i** inclus.

b) Écrire un module lustre **alt: bool x bool → bool** tel que **alt(a,b)** vaut **true** à l'instant **i** si, jusqu'à l'instant **i**, **a** et **b** ont été vrai en alternance. Note : il se peut qu'il y ait des pas auxquels ni **a**, ni **b** n'ont été vrais.

c) Écrire un module lustre **faster: bool x bool → bool** tel que **faster(a,b)** vaut **true** à l'instant **i** si, jusqu'à l'instant **i**, **a** a eu la valeur **true** plus souvent (ou autant) que **b**.

- Note : il se peut qu'il y ait des instants auxquels ni **a**, ni **b** n'ont la valeur **true**.
- On peut utiliser un compteur !

d) Écrire un module lustre **exclusive: bool x bool → bool** tel que **exclusive(a,b)** vaut **true** à l'instant **i** si, jusqu'à l'instant **i**, **a** et **b** n'ont jamais pris la valeur **true** au même instant.

e) Écrire un module lustre **subClock: bool x bool → bool** tel que **subClock(a,b)** vaut **true** à l'instant **i** si, jusqu'à l'instant **i**, **a** n'a jamais pris la valeur **true** quand **b** avait la valeur **false**.

f) Écrire un module lustre **sync: bool x bool → bool** tel que **sync(a,b)** vaut **true** à l'instant **i** si, jusqu'à l'instant **i**, **a** et **b** sont toujours vrais en même.

- Donner une autre définition de **sync** à partir de **subClock**
- Donner une autre définition de **sync** à partir de **faster** ?
- Écrire un observateur pour vérifier si les trois définitions sont équivalentes et le vérifier avec **lesar**.
- Construire le BDD pour les trois formes

Question 2 – Générateurs

a) Écrire un module lustre **periodic:bool x int → bool** tel que **periodic(c, p)** produit une horloge **p** fois plus lente que **c**. **p** est une constante !

	1	2	3	4	5	6	7	8	9
c	T	F	F	T	T	F	T	F	T
periodic(c,2)	T	F	F	F	T	F	F	F	T

1. Écrire un observateur pour s'assurer que **periodic(periodic(c, n), m)** est pareil que **periodic(c, n x m)**. Que dit **lesar** ?
2. Utiliser un observateur **lesar** pour vérifier que **periodic(c, n)** est une sous-horloge de **c**.
3. Faire une nouvelle version qui produit une horloge qui est absente quand **periodic(c,n)** vaut **false** et vaut **true** sinon.

b) Écrire un module lustre **delai: bool x int → bool** tel que **delai(c, d)** qui créé une sous-horloge de **c** qui a 2 ticks de retard sur **c**. **d** est une constante !

	1	2	3	4	5	6	7	8	9
c	T	F	F	T	T	F	T	F	T
delai(c,2)	F	F	F	F	T	F	T	F	T

c) Écrire un module lustre **union: bool x bool → bool** tel que **union(c1, c2)** vaut **true** si et seulement si soit **c1** soit **c2** valent tous les deux **true**. Quelles relations de sous-horloge pouvez-vous établir ? Les vérifier avec **lesar**.

d) Écrire un module lustre **intersection: bool x bool → bool** tel que **intersection(c1, c2)** vaut **true** si et seulement si **c1** et **c2** valent tous les deux **true** ! Quelles relations de sous-horloge pouvez-vous établir ? Les vérifier avec **lesar**.

e) Écrire un module lustre **inf_sup: bool x bool → bool x bool** tel que **inf_sup(c1, c2)** produit deux horloges **(inf, sup)** telles que **inf** est plus rapide que **c1** et **c2** et **sup** est plus lente que **c1** et **c2**.

1. Utiliser un observateur **lesar** pour vérifier que **sup** et **inf** ont la bonne propriété.
2. Quelles propriétés de sous-horloge pouvez-vous établir ? Utiliser lesar.

	1	2	3	4	5	6	7	8	9
c1	T	F	F	T	F	F	T	F	T
c2	F	T	F	T	T	F	T	T	F
union(c1,c2)	T	T	F	T	T	F	T	T	T
intersection(c1,c2)	F	F	F	T	F	F	T	F	F
inf(c1,c2)	T	F	F	T	T	F	T	F	F
sup(c1,c2)	F	T	F	T	F	F	T	F	T