

Logique



Exercice 1

On définit une fonction `mux` qui prend trois paramètres `a`, `b` et `c` booléens. Si `a` est faux, alors `mux(a,b,c)` vaut `b`; si `a` est vrai alors `mux(a,b,c)` vaut `c`.

1. Dresser la table de la fonction `mux`.
2. Dresser la table de la fonction qui à `a`, `b` et `c` associe `(not(a) and b) or (a and c)`.
Comparer le résultat avec celui de la table précédente.
3. Dessiner alors un circuit réalisant la fonction `mux`.

Note d'information : On appelle multiplexeur cette fonction `mux`.

Exercice 2

Le ou exclusif, ou `xor` est une fonction logique qui prend deux arguments booléens `a` et `b`, et qui vaut vrai si et seulement si un seul des deux booléens est vrai.

1. Établir la table de vérité du ou exclusif.
2. En déduire une formule logique pour « `a xor b` » à l'aide des fonctions logiques `not`, `or` et `and`.
3. Écrire une fonction (Python) `xor` qui prend en paramètres deux booléens `a` et `b` et qui renvoie la valeur « `a xor b` ».
4. Nous avons vu le demi-additionneur et donné sa table de vérité. Vérifier que celle-ci est la même que celle du `xor`, puis écrire une fonction `demi` qui prend en paramètres deux booléens `a` et `b` et renvoie le couple `(s,r)` (somme et retenue) dans le cas d'un additionneur 1 bit.
5. Écrire une fonction `addit1` qui prend en paramètres trois booléens `a`, `b` et `re`, et renvoie le couple `(s,rs)` (somme et retenue) dans le cas d'un additionneur complet 1 bit.