Les opérateurs logiques

https://www.youtube-nocookie.com/embed/7mLNb8XIRHw

> Cette matière étant globalement la même que celle vu en math, je vous renvois donc vers ma synthèse de math sur le sujet : (Math) Les connecteurs logiques de base.

Symbole électronique	Nom électronique	Formule mathématique
$ \overline{a} $	`NOT` a	$ \neg a $
$ ab $ ou $ a * b $	a `AND` b	$ a \wedge b $
$ a + b $	a `OR` b	$ a \vee b $
$ a \oplus b $	a `XOR` b	$ a \oplus b $

Ensuite il y a la négations des portes précédentes :

Symbole électronique	Nom électronique	Formule mathématique
$ a \downarrow b $ ou $ \overline {a + b} $	a `NOR` b	$ \neg (a \vee b) $
$ a \uparrow b $ ou $ \overline {ab} $	a `NAND` b	$ \neg (a \wedge b) $
$ \overline {a \oplus b} $ ou $ a \iff b $	a `XNOR` b	$ a \iff b $

Et voici à quoi correspondent ces portes dans des schémas électroniques :

symboles électroniques
Image not found or type unknown

Comment construire les portes logiques avec des transistors

Pour en savoir plus, j'ai trouvé 2 vidéos en anglais qui expliquent comment fonctionnent les transistors et les résistances :

- Une vidéo avec animation 3D
- Une vidéo qui explique toutes les portes logiques avec un schéma électronique

2 petites informations pour mieux comprendre les vidéos :

- Un transistor se comporte comme un interrupteur mais activé de manière électronique
- Une résistance crée une différence de tension. Donc dans un circuit fermé, avant la résistance la tension serait de 5V et après elle serait de 0V.

Les formes normales

Les formes normales permettent de représenter toute fonction logique avec uniquement des AND, OR et NOT.

La première forme normale (disjonctive)

1. On construit la table de vérité de la fonction

a	b	xor
0	0	0
0	1	1
1	0	1
1	1	0

2. On se concentre uniquement sur les fois où la fonction vaut 1

a	b	xor
0	1	1
1	0	1

3. On met un opérateur `AND` entre les deux inputs et on remplace le 0 par une négation

> $ \overline{a} b $ | $ a \overline{b} $

4. On sépare les différents résultat de l'étape précédente par des `OR`

> Première forme normale de XOR : $ \overline{a} b + a \overline{b} $

La deuxième forme normale (conjonctive)

1. On construit la table de vérité de la fonction

a	b	xor
0	0	0
0	1	1
1	0	1
1	1	0

2. On se concentre uniquement sur les fois où la fonction vaut 0

a	b	xor
0	0	0
1	1	0

3. On met un opérateur `OR` entre les deux inputs et on rempalce le 1 par une négation

> $a + b$ | $\overline{a} + \overline{b}$

4. On sépare les différents résultats de l'étape précédente par des `AND`

> Deuxième forme normale de XOR : $ (a + b)(\overline{a} + \overline{b}) $

Revision #1
Created 27 April 2023 06:15:07 by SnowCode
Updated 27 April 2023 06:27:26 by SnowCode