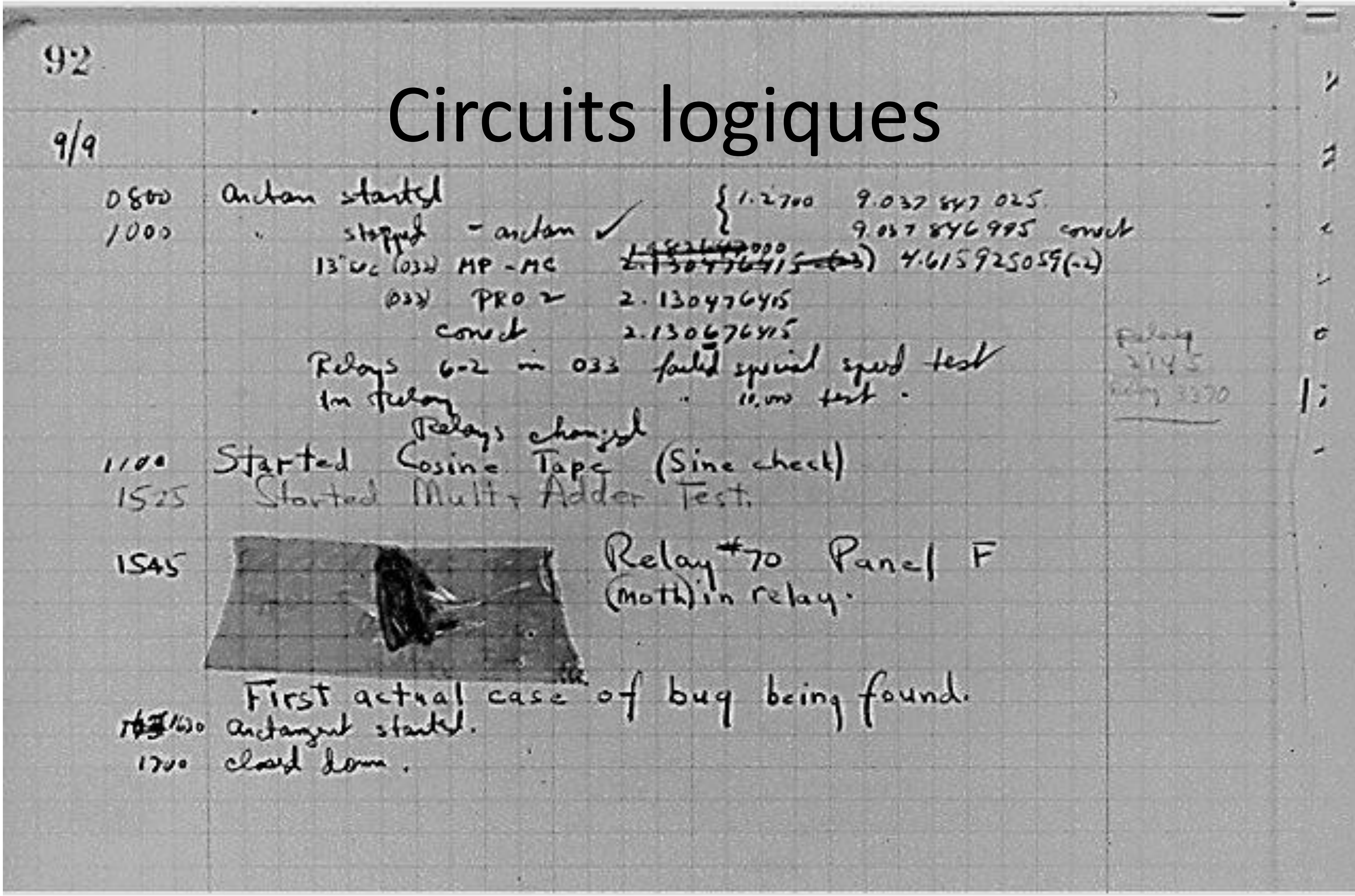
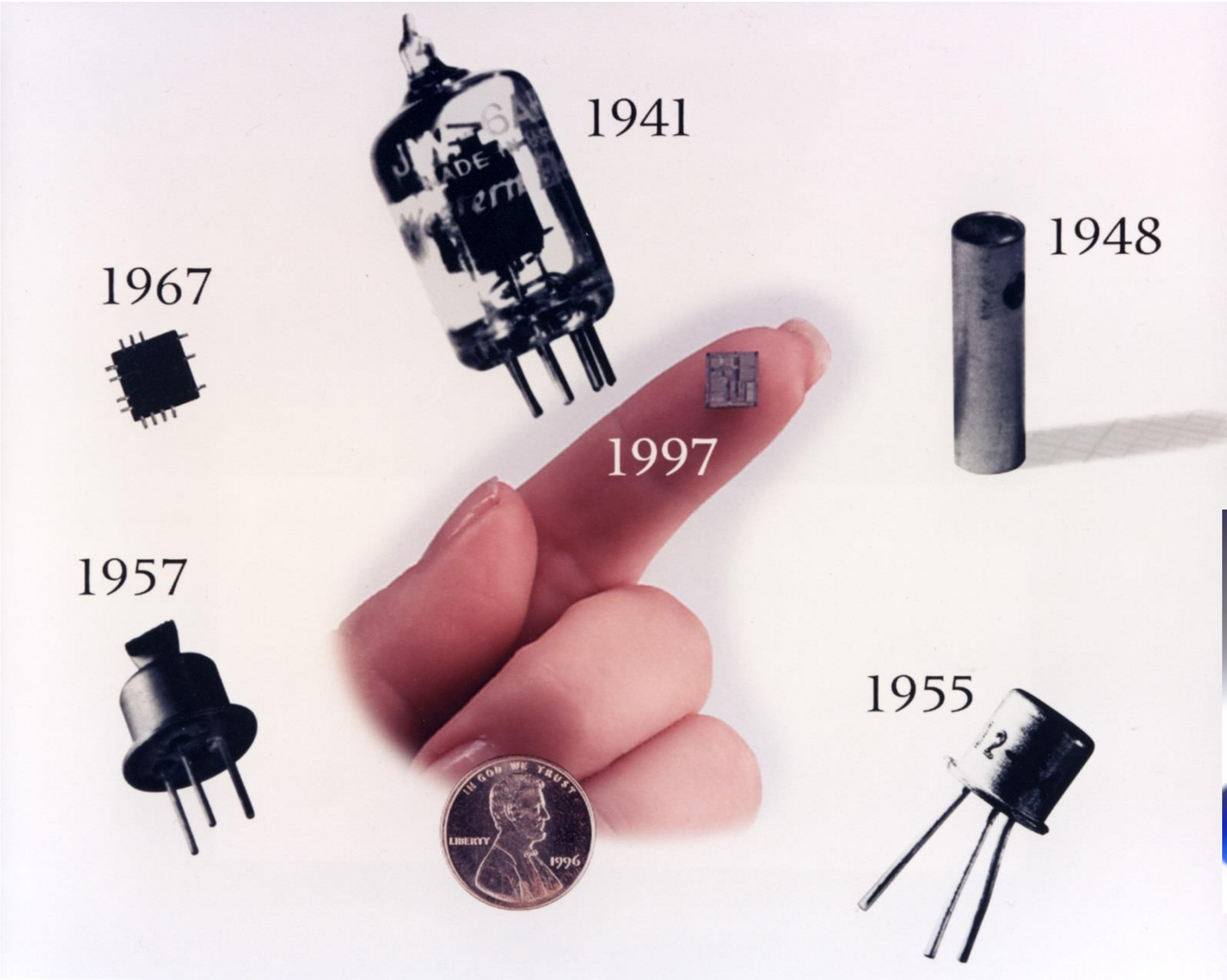
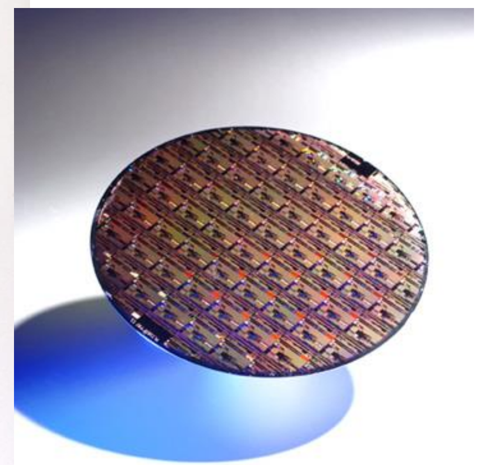
Circuits logiques



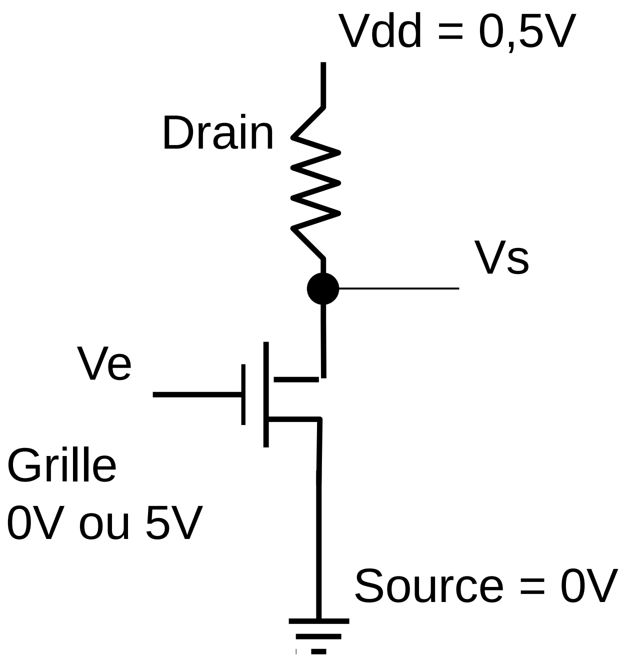
L'historique du transistor



Découvert en 1947
au Bell Labs



Principe de fonctionnement



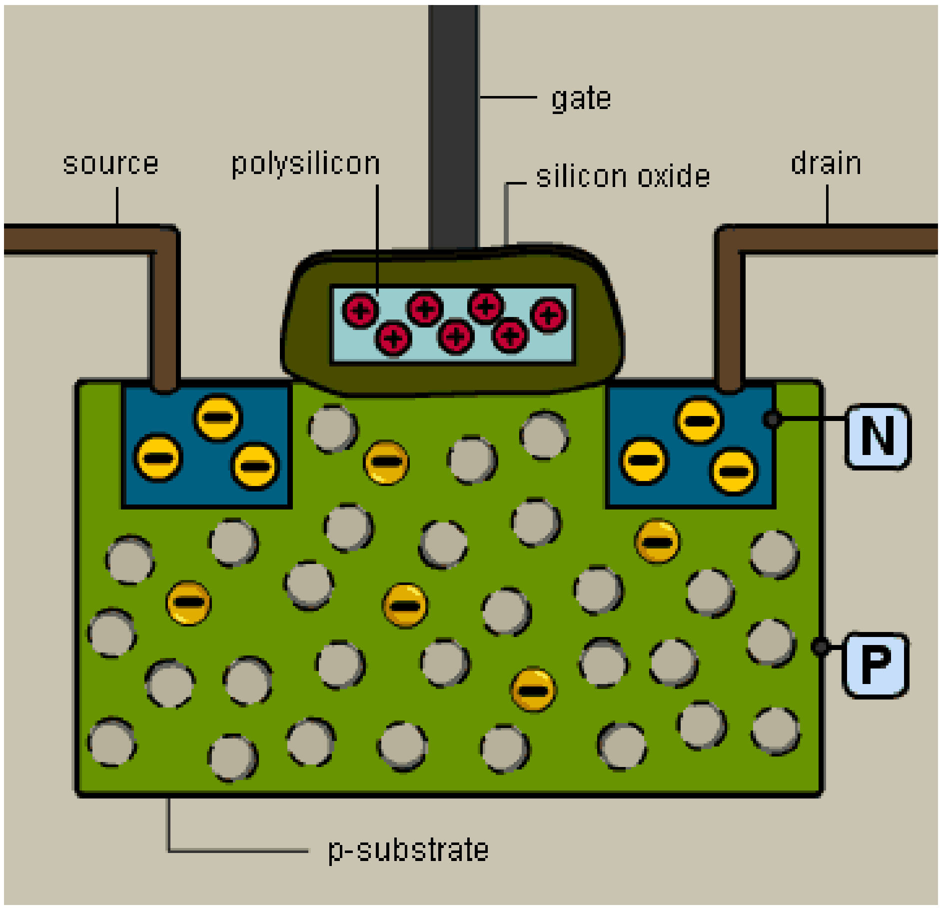
- Si Ve < tension de seuil (0,6V)
  - Le transistor est bloquant
  - Interrupteur ouvert (Vs ≅Vcc)

- Si Ve > tension de seuil
  - Le transistor est passant
  - Interrupteur fermé

Comportement Inverseur :

Ve	Vs
haut (1)	bas (0)
bas (0)	haut (1)

Principe de fonctionnement



Transistor NMOS dit à canal N

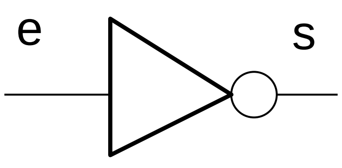
Le transistor à canal P (PMOS) inverse les polarités

La technologie actuelle utilise des transistors CMOS qui met en jeu à la fois des transistors P et N.
Il sont plus rapides et consomment moins en électricité.

2. Portes logiques

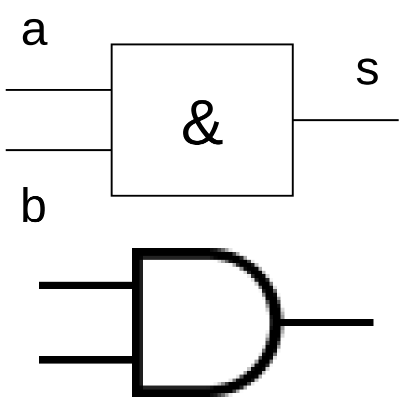
Portes logiques de base

NON

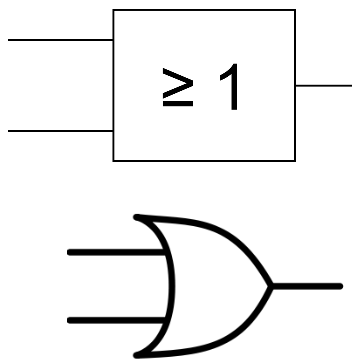


e	s
0	1
1	0

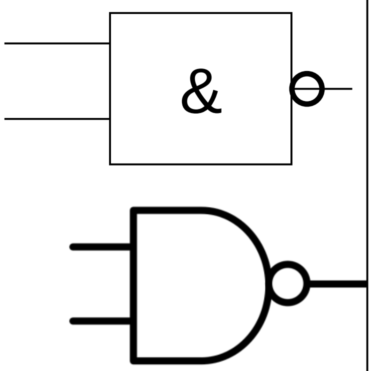
ET



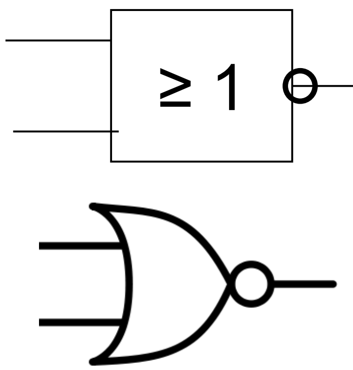
OU



NON-ET

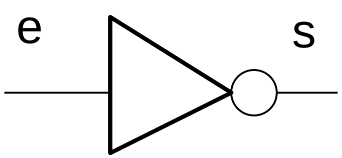


NON-OU



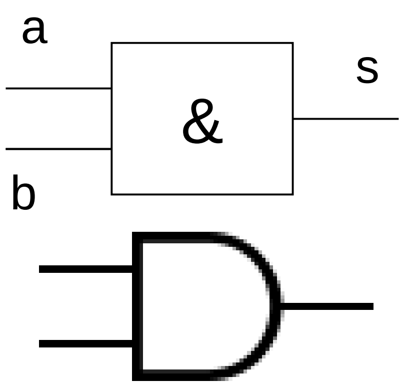
Portes logiques de base

NON



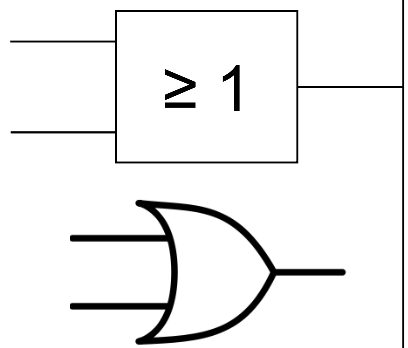
e	s
0	1
1	0

ET

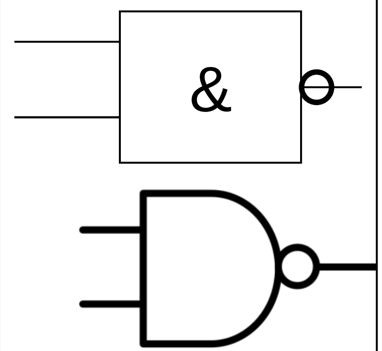


a	b	s
0	0	0
0	1	0
1	0	0
1	1	1

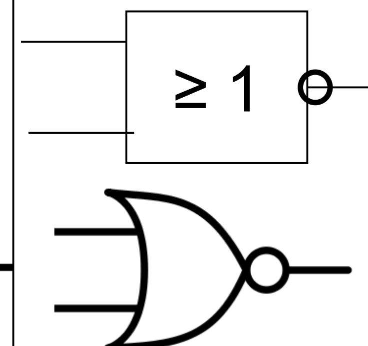
OU



NON-ET

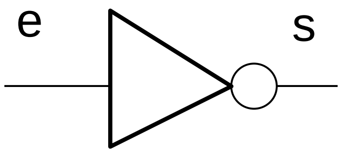


NON-OU



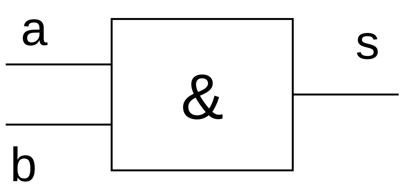
Portes logiques de base

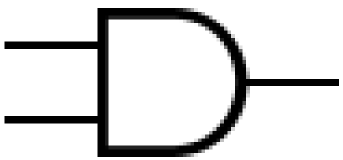
NON



e	s
0	1
1	0

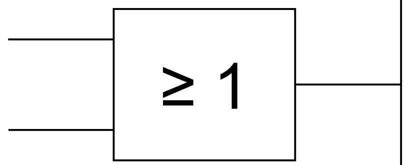
ET

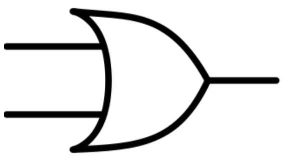




a	b	s
0	0	0
0	1	0
1	0	0
1	1	1

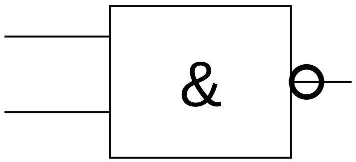
OU

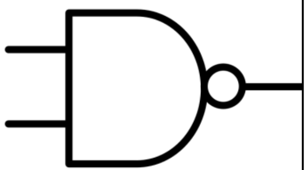




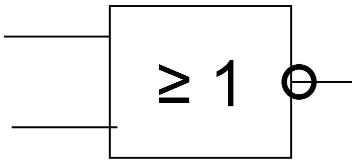
a	b	s
0	0	0
0	1	1
1	0	1
1	1	1

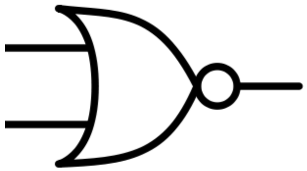
NON-ET





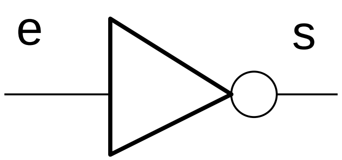
NON-OU





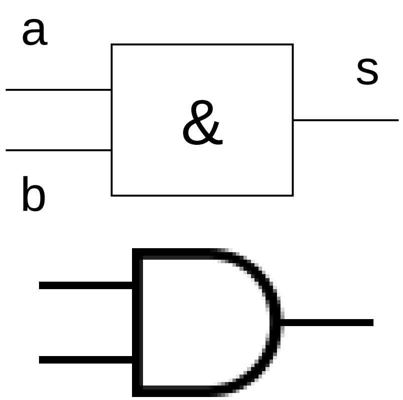
Portes logiques de base

NON



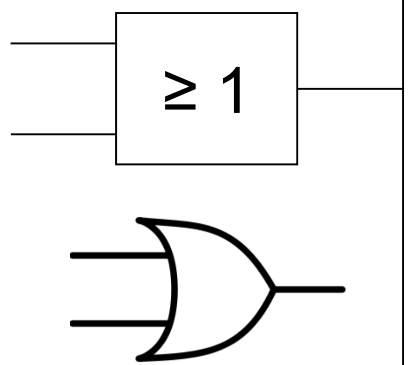
e	s
0	1
1	0

ET



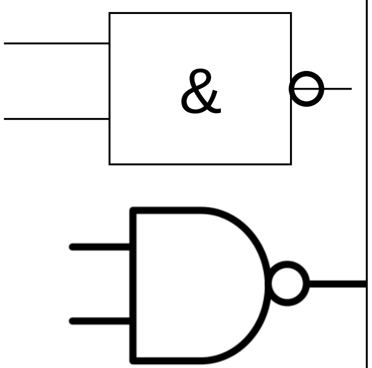
a	b	s
0	0	0
0	1	0
1	0	0
1	1	1

OU



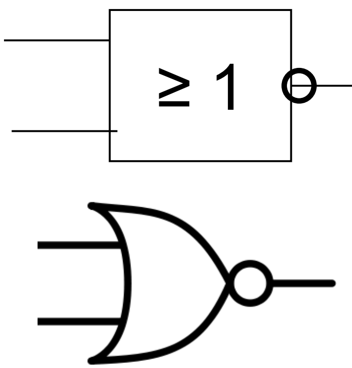
a	b	s
0	0	0
0	1	1
1	0	1
1	1	1

NON-ET



a	b	s
0	0	1
0	1	1
1	0	1
1	1	0

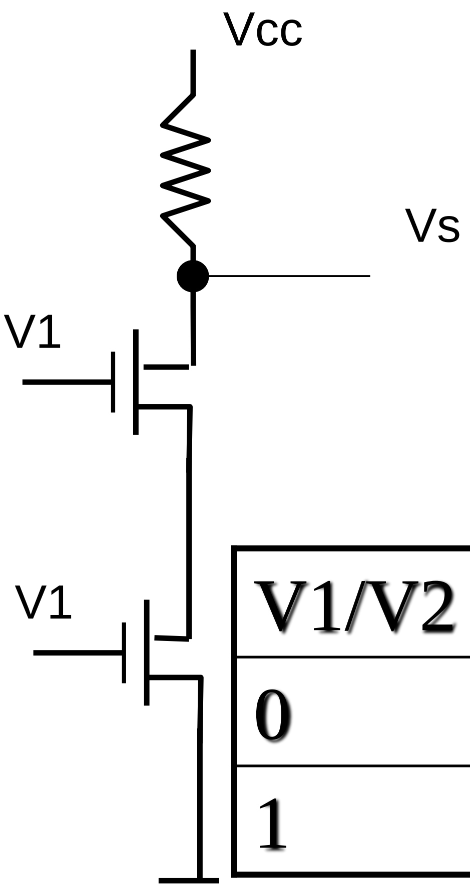
NON-OU



a	b	s
0	0	1
0	1	0
1	0	0
1	1	0

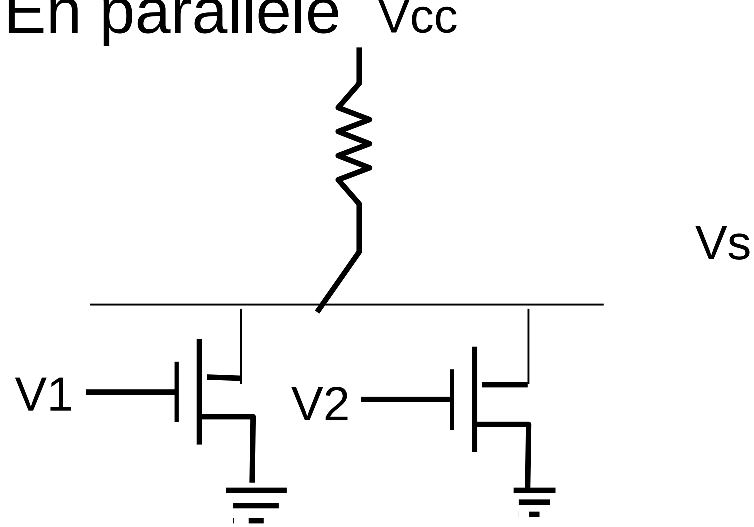
Réalisation de portes en technologie NMOS

En série



V1/V2	0	1
0	?	?
1	?	?

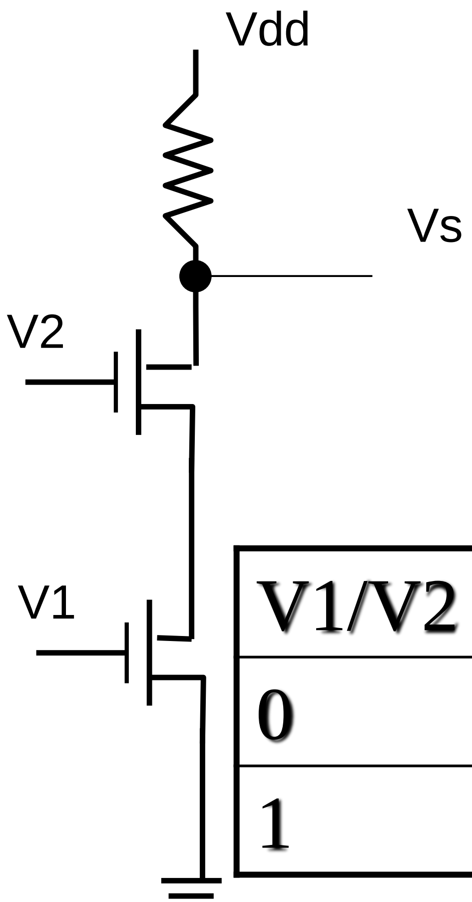
En parallèle



V1/V2	0	1
0	?	?
1	?	?

Réalisation de portes en technologie NMOS

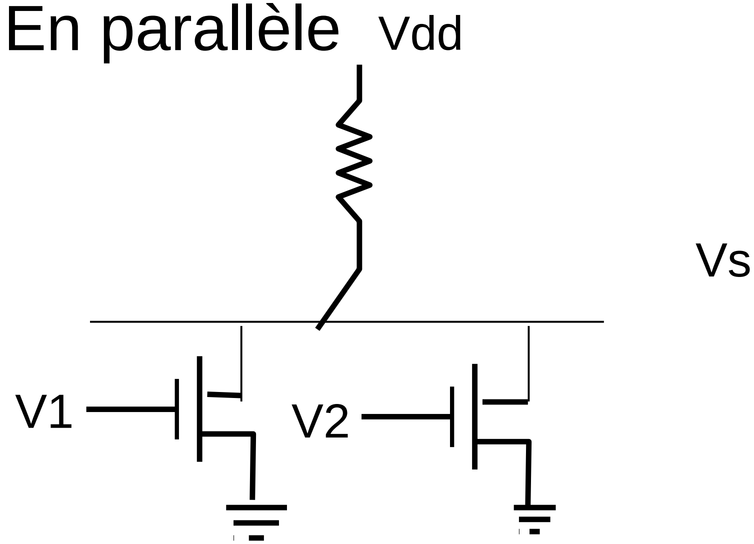
En série



V1/V2	0	1
0	1	1
1	1	0

NON-ET

En parallèle

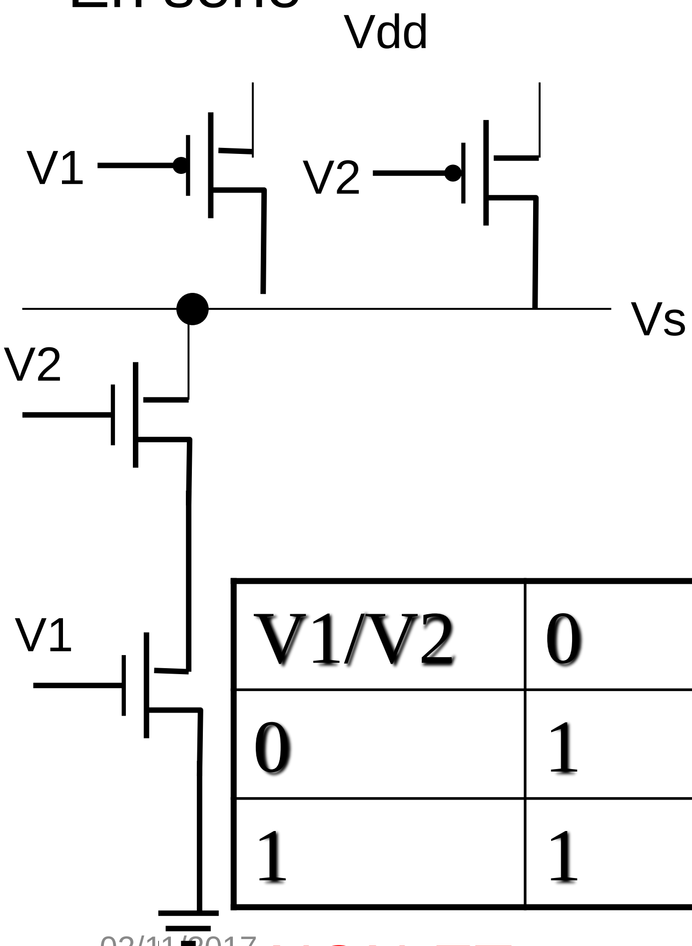


V1/V2	0	1
0	1	0
1	0	0

NON-OU

Réalisation de portes en technologie CMOS

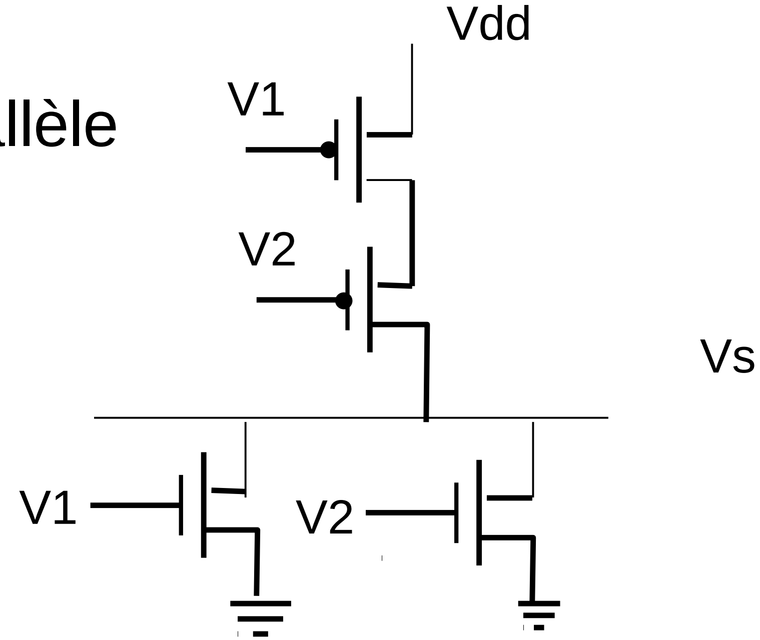
En série



V1/V2	0	1
0	1	1
1	1	0

NON-ET

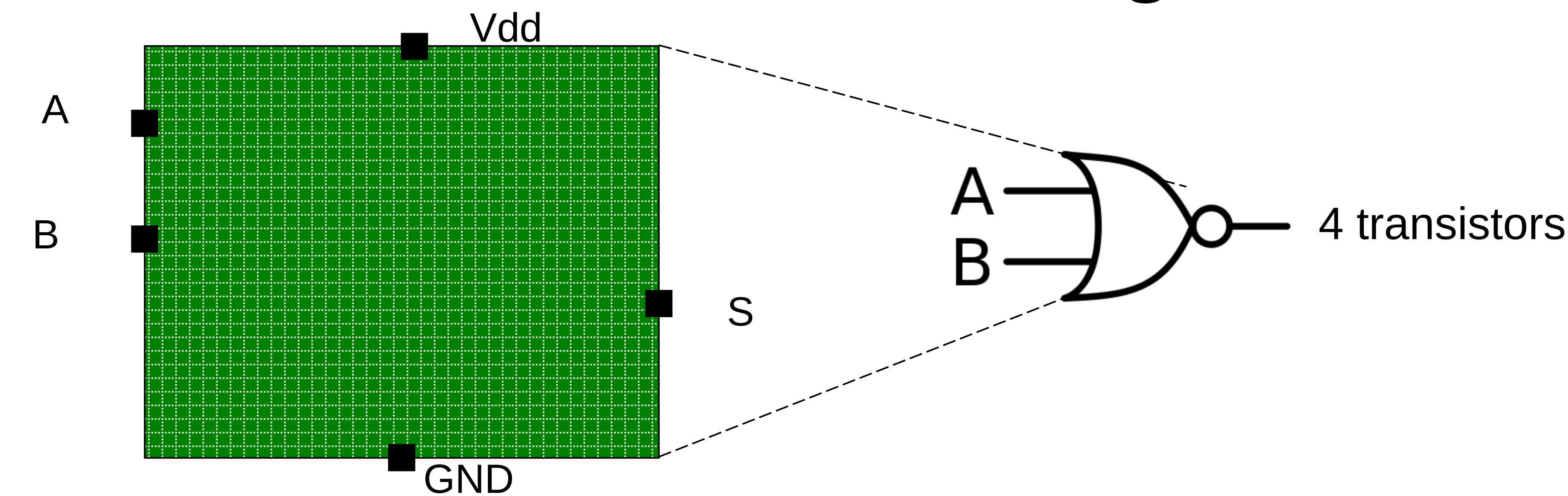
En parallèle

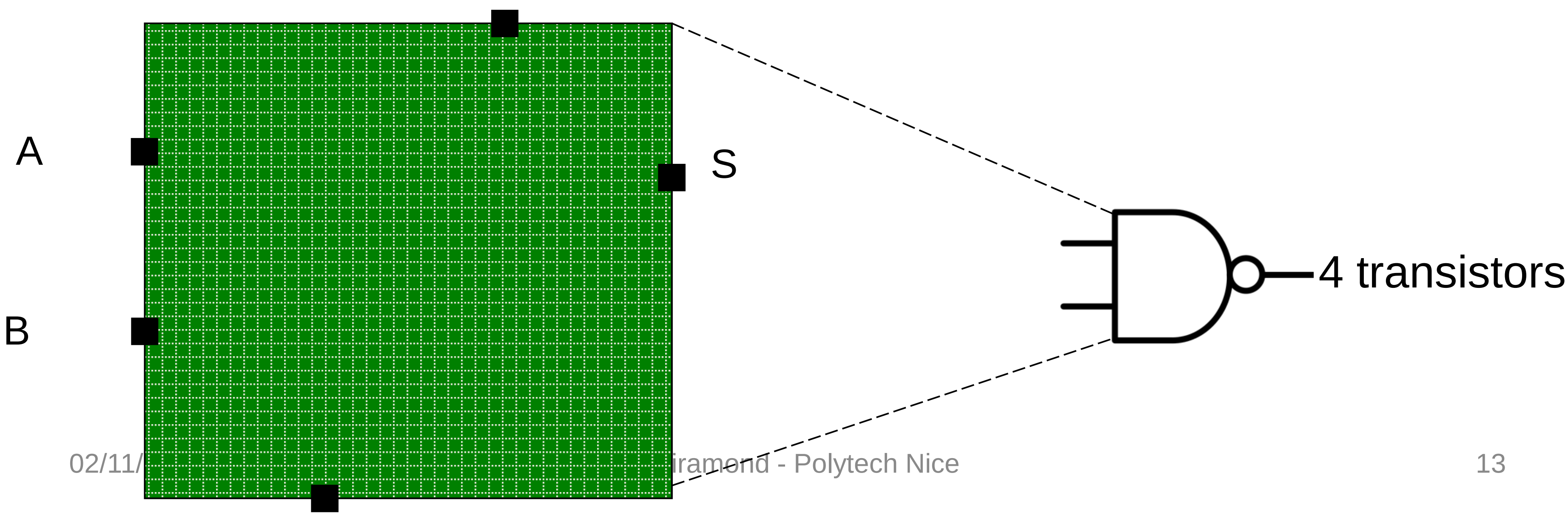


V1/V2	0	1
0	1	0
1	0	0

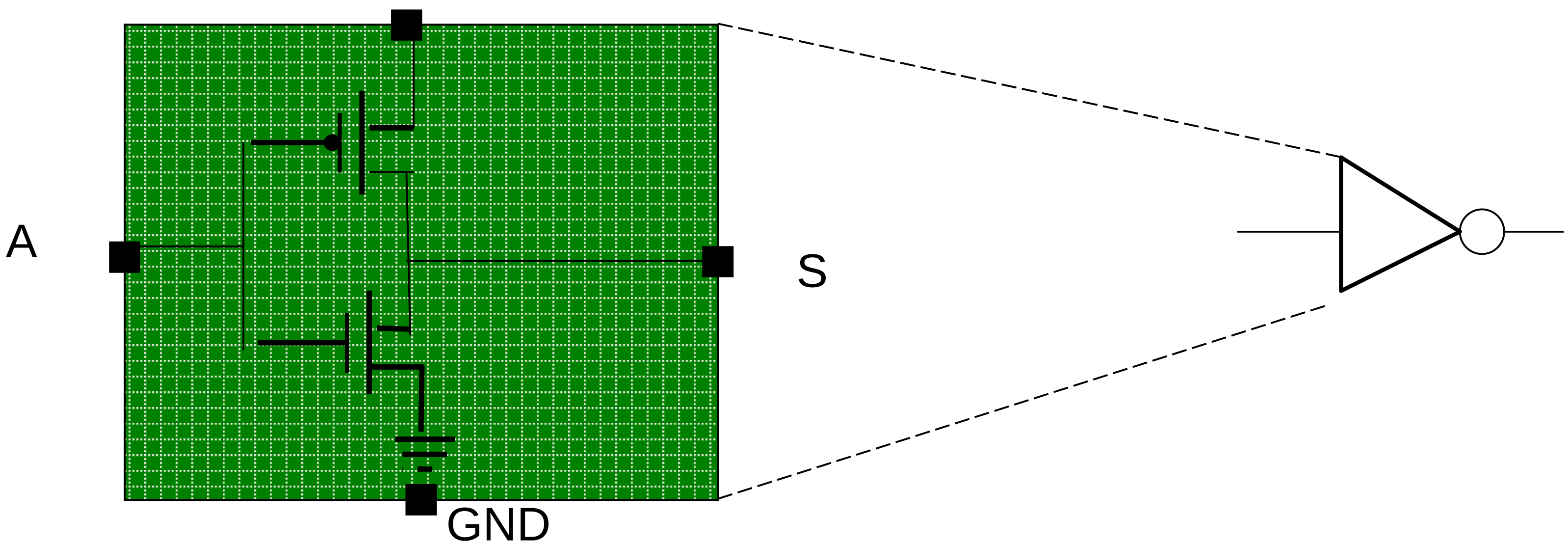
NON-OU

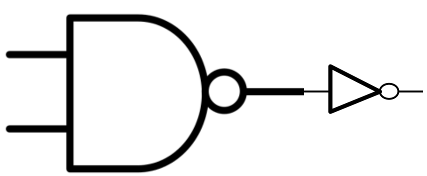
Portes de bases en technologie CMOS

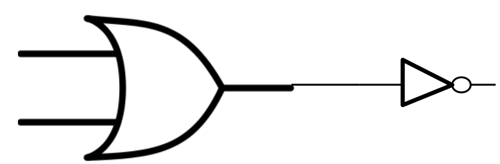


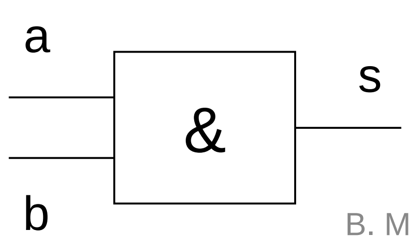


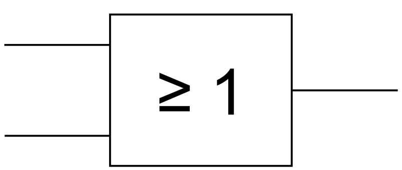
Portes de bases en technologie CMOS



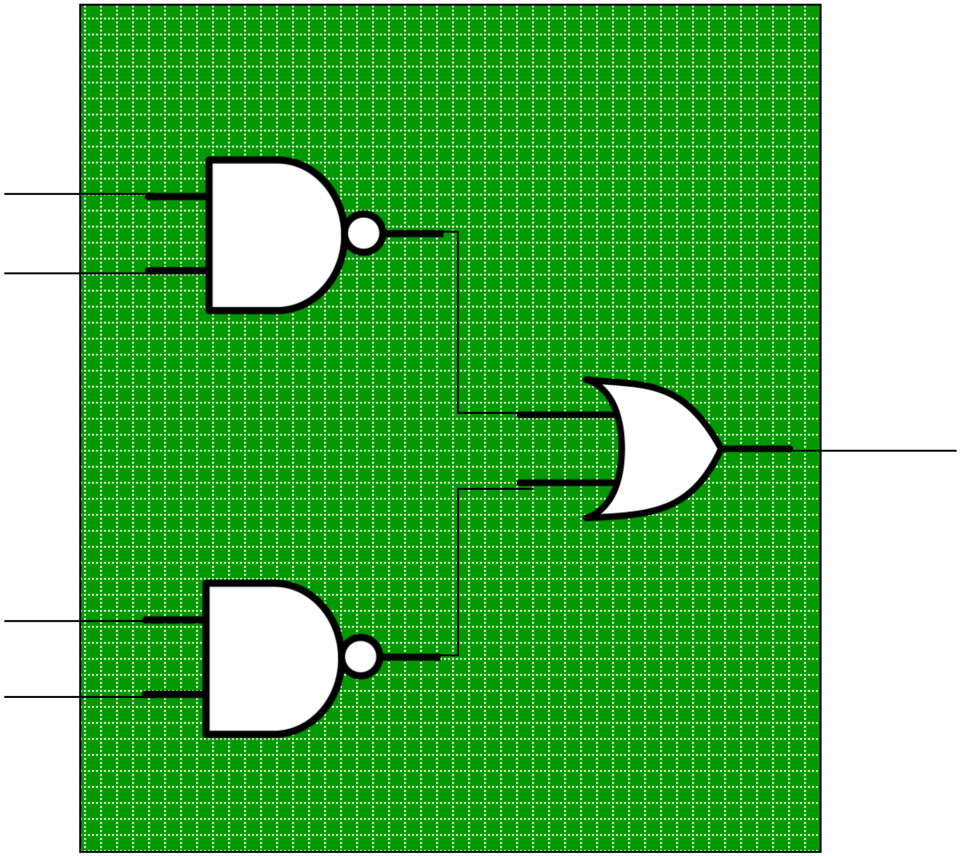






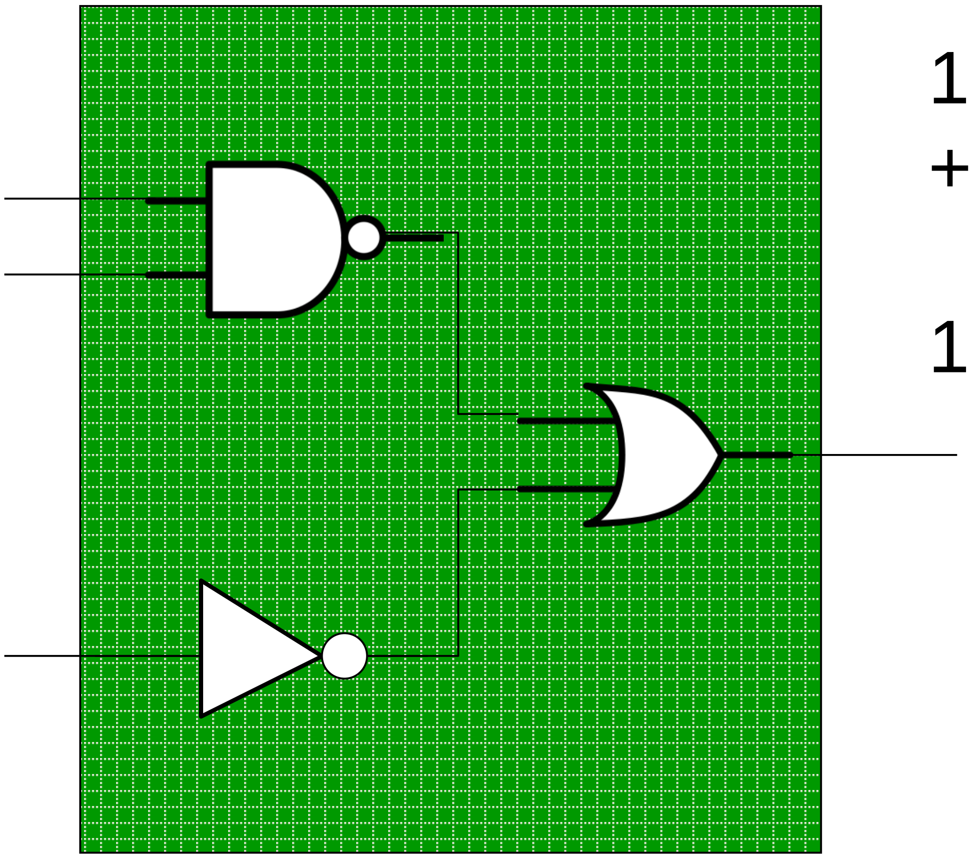


NAND à 4 entrées



2 NAND_2 + 1 OR_2
14 transistors

NAND à 3 entrées



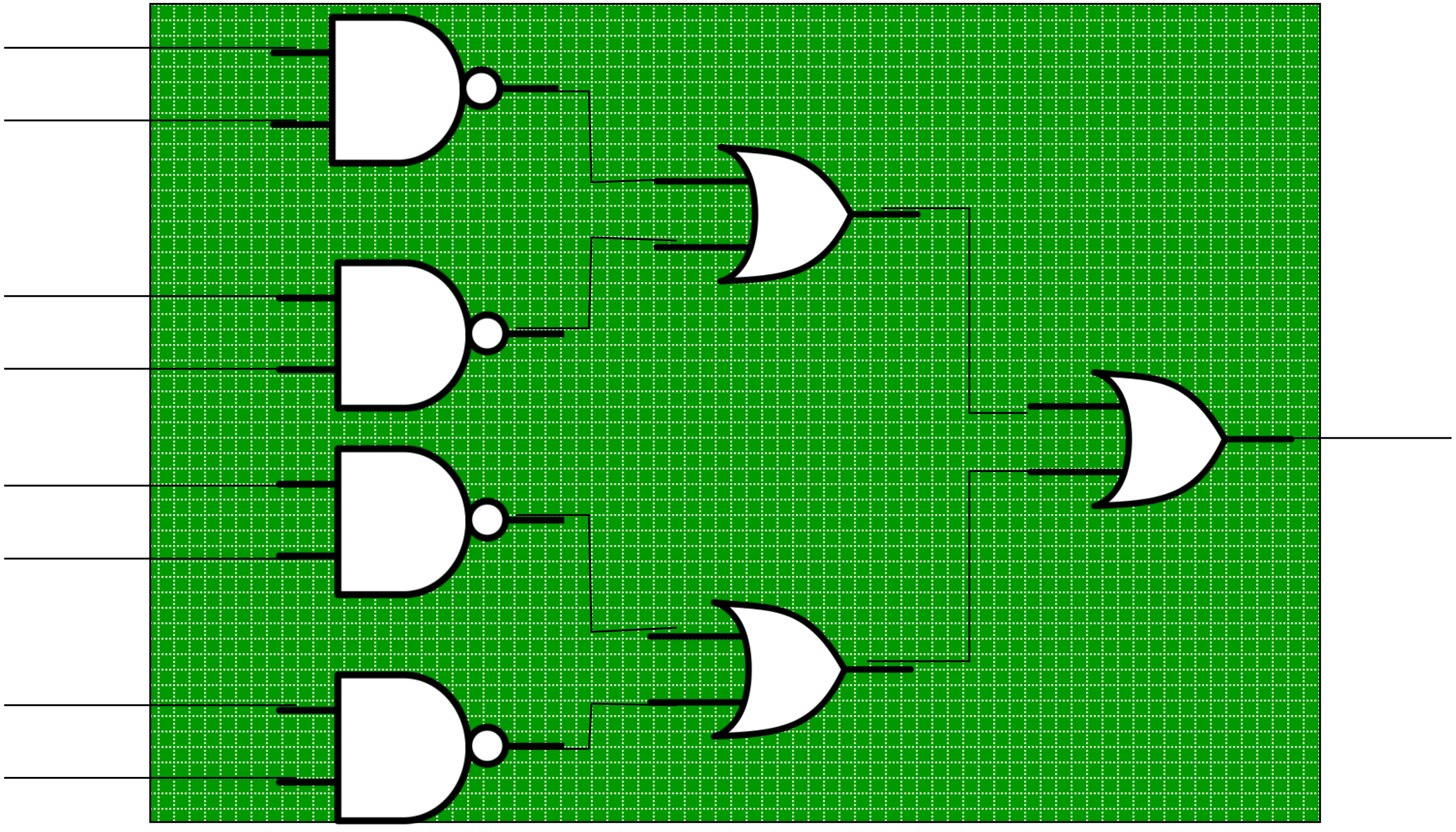
1 NAND_2 + 1 OR_2
+ 1 NOT_1

12 transistors

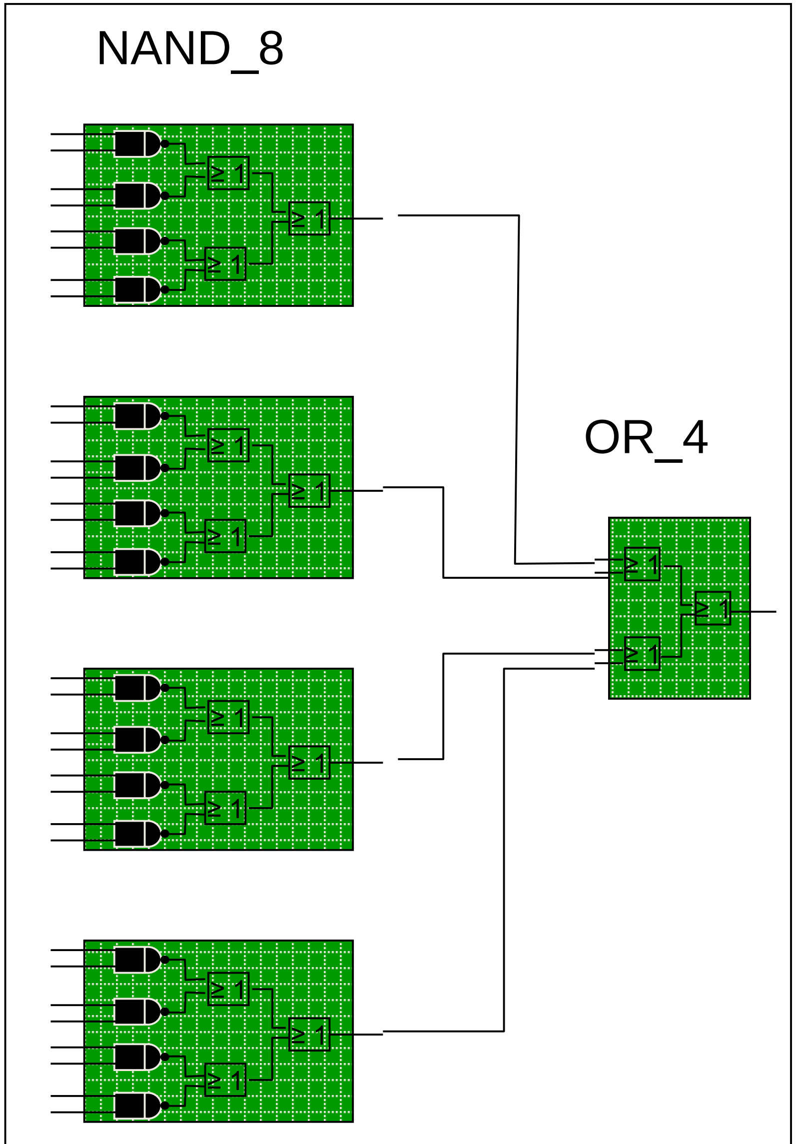
NAND à 8 entrées

NAND à 8 entrées

4 NAND_2 + 3 OR_2
34 transistors



NAND à 32 entrées



16 NAND_2 + 15 OR_2
154 transistors

Une porte NAND à *n* entrées demande *(n/2)* portes NAND et *(n/2) – 1* portes OR à 2 entrées , et donc *5n-6* transistors

Réalisation de fonctions logiques

Pour définir chacune des fonctions logiques, on utilise plusieurs **représentations** :

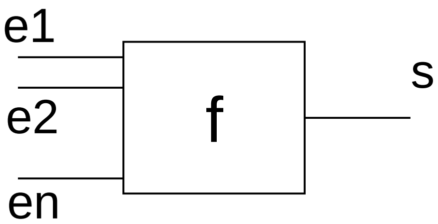
- une représentation électrique : schéma à contacts
- une représentation algébrique : **équation**
- une représentation arithmétique: **table de vérité**
- une représentation temporelle : **chronogramme**
- une représentation logique : **symbole logique**

Fonctions logiques

algèbrique

s = f(e1,e2,…en)
B^n -> B

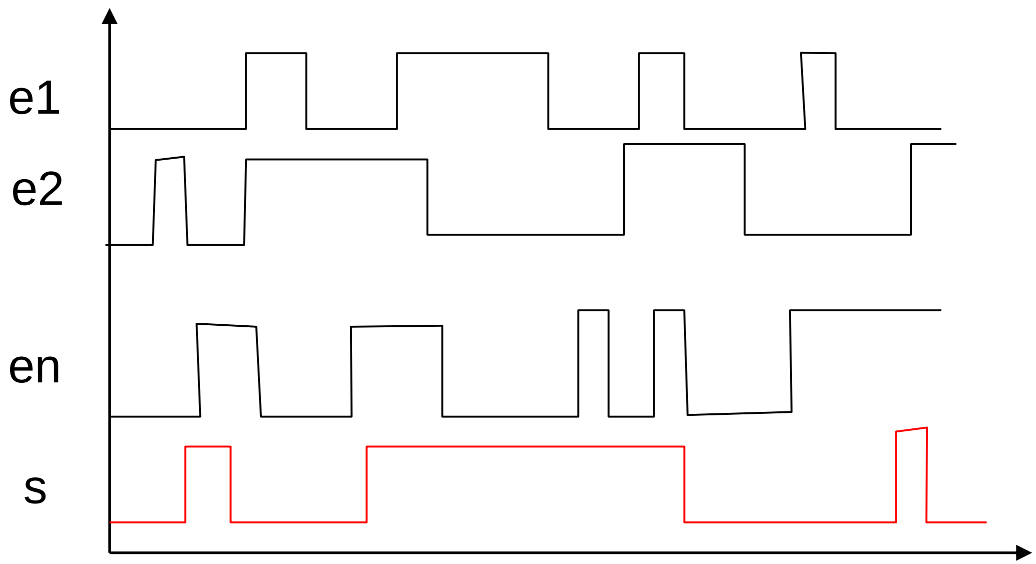
logique



arithmétique

a	b	…	s
0	0		v0
0	1		v1
1	0		v2
…			
1	1		vn

temporelle



Fonctions booléennes

- On peut décrire complètement une fonction booléenne de n variables avec un table de vérité de 2^n lignes

n=2 variables

a	b	s
0	0	v0
0	1	v1
1	0	v2
1	1	v3

- 2 valeurs pour v0
- 2 valeurs pour v1
- 2 valeurs pour v2
- 2 valeurs pour v3

2*2*2*2 = 16 fonctions booléennes de deux variables

Fonctions booléennes

n variables

2^n lignes

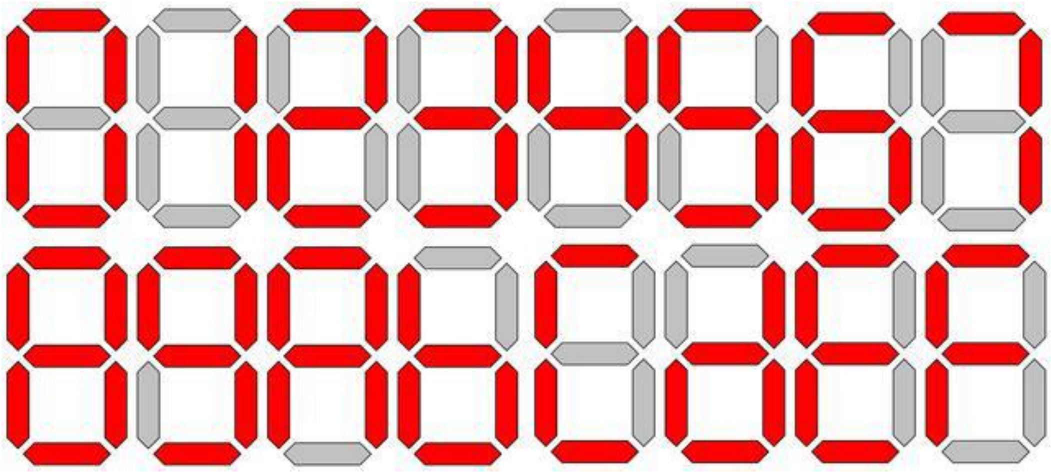
a	b	…	s
0	0		v0
0	1		v1
1	0		v2
…			
1	1		vn

- 2 valeurs pour v0
- 2 valeurs pour v1
- 2 valeurs pour v2
- …
- 2 valeurs pour v3

2^{2^n} fonctions booléennes de n variables

Exemple de circuit – le décodeur 7 segments

- L'afficheur 7 segments permet d'afficher l'ensemble des chiffres hexadécimaux de 0 à F
- 7 segments de contrôle de A à F
- Nécessite un décodeur pour établir la conversion du code hexadécimal sur 4 bits au contrôle de l'afficheur sur 7 bits



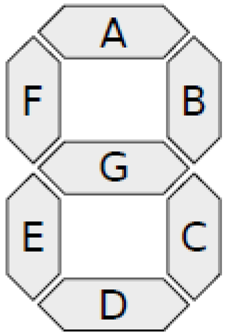
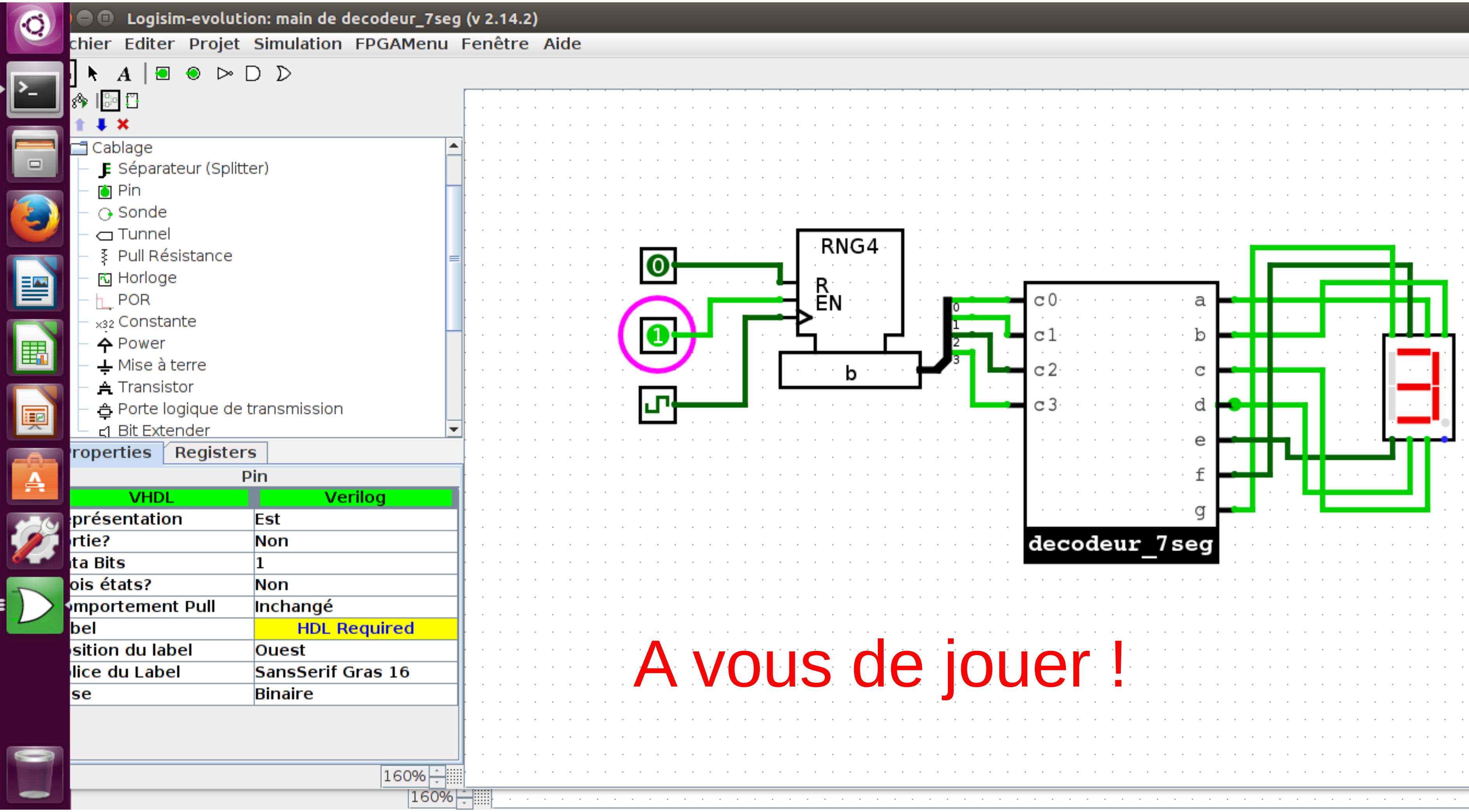


Table de vérité du décodeur

- Elle dispose de 1-lignes
- Les entrées sont les 4 bits codant le chiffre hexa
- Les sorties sont les 7 segments

	Individual Segments						
Display	A	B	C	D	E	F	G
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	1	1	0	1	1	0	1
3	1	1	1	1	0	0	1
4	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1
6	1	0	1	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1
A	1	1	1	0	1	1	1
B	0	0	1	1	1	1	1
C	1	0	0	1	1	1	0
D	0	1	1	1	1	0	1
E	1	0	0	1	1	1	1
F	1	0	0	0	1	1	1

Réalisation sous Logisim



A vous de jouer !

Rappels – Equivalence et simplifications de fonctions

- On démontre que toute fonction logique peut se décrire à l'aide des trois opérations de base grâce au théorème de De Morgan
- OU
- ET
- NOT

Théorème de **De Morgan**

- $\overline{a+b} = \overline{a}.\overline{b}$
  - Dans les deux cas, l'expression ne sera VRAIE que si a et b sont fausses.

- $\overline{a.b} = \overline{a} + \overline{b}$
  - Dans les deux cas, l'expression ne sera VRAIE que si a ou b sont fausses

Propriétés de la logique

Associativité

- Certaines parenthèses sont inutiles:
  $(a + b) + c = a + (b + c) = a + b + c$
  $(a \cdot b) \cdot c = a \cdot (b \cdot c) = a \cdot b \cdot c$

Commutativité

- L'ordre est sans importance.
  $a + b = b + a$
  $a \cdot b = b \cdot a$

Distributivité

- Comme avec les opérations habituelles, il est possible de distribuer:
  $a \cdot (b + c) = a \cdot b + a \cdot c$

Idempotence

- $a + a + a$ [...] $= a$
  $a \cdot a \cdot a$ [...] $= a$

Propriétés

	Forme OU	Forme ET
Loi d'identité	$a+0 = a$	$a.1 = a$
Loi de nullité	$a+1 = 1$	$a.0 = 0$
Loi d'idempotence	$a+a = a$	$a.a = a$
Loi d'inversion	$a+\overline{a} = 1$	$a.\overline{a} = 0$
Loi d'absorption	$a+a.b = a$	$a.(a+b) = a$