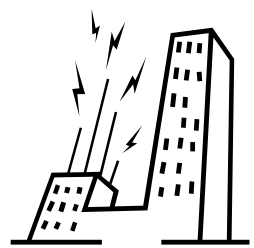
Architecture des ordinateurs III

Frédéric Boulanger

CentraleSupélec



Fondements

De l'électronique à la logique

- une tension électrique peut être nulle ou pas

Fondements

De l'électronique à la logique

- une tension électrique peut être nulle ou pas
- un transistor peut laisser passer un courant ou non

Fondements

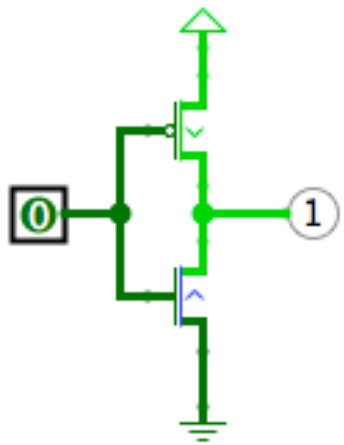
De l'électronique à la logique

- une tension électrique peut être nulle ou pas
- un transistor peut laisser passer un courant ou non
- on peut ainsi coder deux états, par exemple *vrai* et *faux*

Fondements

De l'électronique à la logique

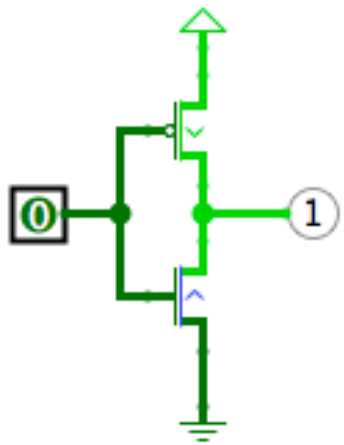
- une tension électrique peut être nulle ou pas
- un transistor peut laisser passer un courant ou non
- on peut ainsi coder deux états, par exemple *vrai* et *faux*



Fondements

De l'électronique à la logique

- une tension électrique peut être nulle ou pas
- un transistor peut laisser passer un courant ou non
- on peut ainsi coder deux états, par exemple *vrai* et *faux*

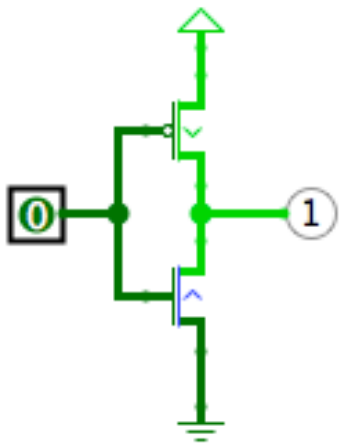


Porte NON

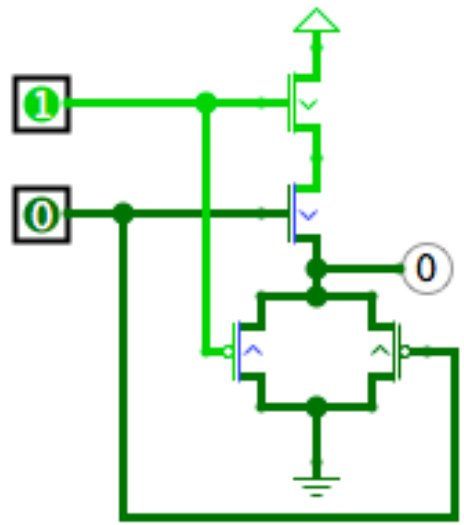
Fondements

De l'électronique à la logique

- une tension électrique peut être nulle ou pas
- un transistor peut laisser passer un courant ou non
- on peut ainsi coder deux états, par exemple *vrai* et *faux*



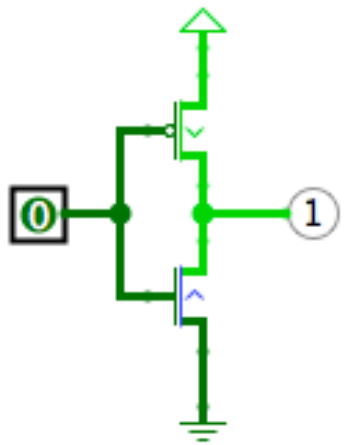
Porte NON



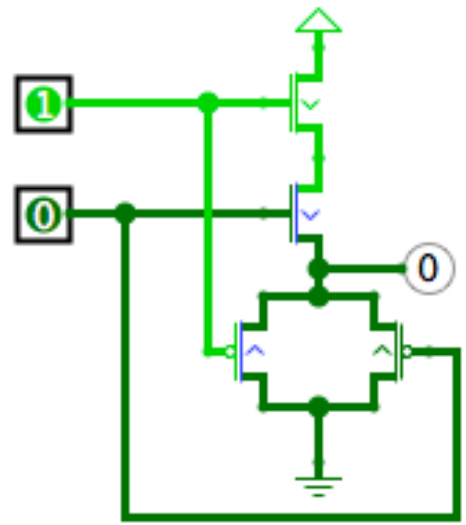
Fondements

De l'électronique à la logique

- une tension électrique peut être nulle ou pas
- un transistor peut laisser passer un courant ou non
- on peut ainsi coder deux états, par exemple *vrai* et *faux*



Porte NON

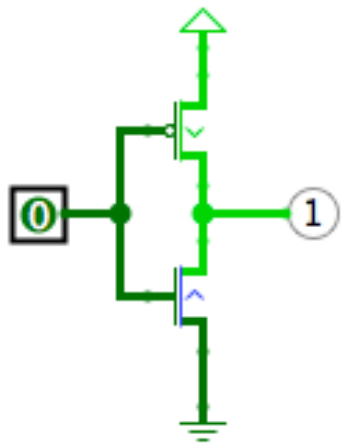


Porte ET

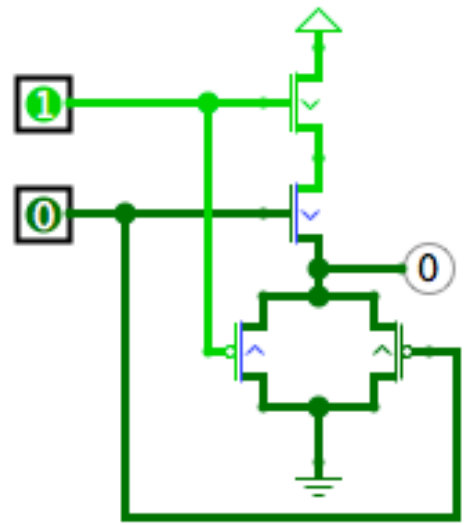
Fondements

De l'électronique à la logique

- une tension électrique peut être nulle ou pas
- un transistor peut laisser passer un courant ou non
- on peut ainsi coder deux états, par exemple *vrai* et *faux*



Porte NON



Porte ET

Circuit Logisim

wdi.supelec.fr/architecture/Info/Gates

Fondements

De la logique à l'arithmétique

- on compte en binaire : faux = 0, vrai = 1

Fondements

De la logique à l'arithmétique

- on compte en binaire : faux = 0, vrai = 1
- addition :

	0	1
0	00	01
1	01	10

De la logique à l'arithmétique

- on compte en binaire : faux = 0, vrai = 1
- addition :

	0	1
0	00	01
1	01	10

somme = OU exclusif

retenue = ET

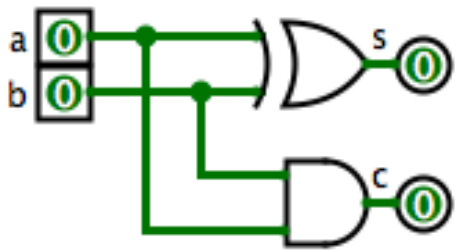
De la logique à l'arithmétique

- on compte en binaire : faux = 0, vrai = 1
- addition :

	0	1
0	00	01
1	01	10

somme = OU exclusif

retenue = ET



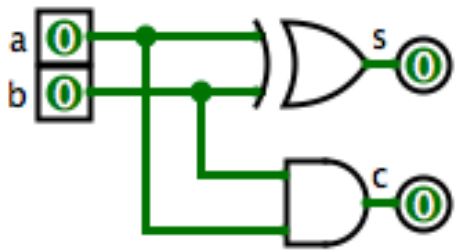
De la logique à l'arithmétique

- on compte en binaire : faux = 0, vrai = 1
- addition :

	0	1
0	00	01
1	01	10

somme = OU exclusif

retenue = ET



demi additionneur

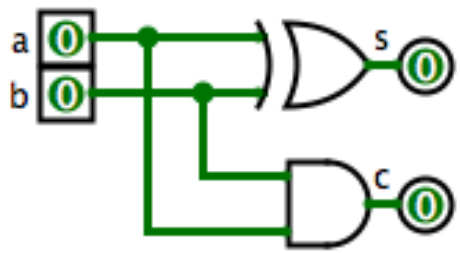
De la logique à l'arithmétique

- on compte en binaire : faux = 0, vrai = 1
- addition :

	0	1
0	00	01
1	01	10

somme = OU exclusif

retenue = ET



demi additionneur

Circuit Logisim

wdi.supelec.fr/architecture/Info/Combi

Fondements

Arithmétique binaire

- addition : comme en base 10, mais la table est plus simple...

	1	0	1	0	(10)
+	0	0	1	1	(3)
		1			
=	1	1	0	1	(13)

Fondements

Arithmétique binaire

- addition : comme en base 10, mais la table est plus simple...

	1	0	1	0	(10)
+	0	0	1	1	(3)
		1			
=	1	1	0	1	(13)

- soustraction : addition de l'opposé

Fondements

Arithmétique binaire

- addition : comme en base 10, mais la table est plus simple...

	1	0	1	0	(10)
+	0	0	1	1	(3)
		1			
=	1	1	0	1	(13)

- soustraction : addition de l'opposé
- représentation des nombres négatifs : complément à 2
  Sur k bits, $-n$ est représenté par $2^k - n$

Fondements

Arithmétique binaire

- addition : comme en base 10, mais la table est plus simple...

	1	0	1	0	(10)
+	0	0	1	1	(3)
		1			
=	1	1	0	1	(13)

- soustraction : addition de l'opposé
- représentation des nombres négatifs : complément à 2
  Sur k bits, $-n$ est représenté par $2^k - n$
- exemple : sur 4 bits, -3 est représenté par $16 - 3 = 13$

Fondements

Arithmétique binaire

- addition : comme en base 10, mais la table est plus simple...

	1	0	1	0	(10)
+	0	0	1	1	(3)
		1			
=	1	1	0	1	(13)

- soustraction : addition de l'opposé
- représentation des nombres négatifs : complément à 2
  Sur k bits, $-n$ est représenté par $2^k - n$
- exemple : sur 4 bits, -3 est représenté par $16 - 3 = 13$
- $5 - 3$ devient $5 + (16 - 3) = 5 + 13 = 18 = 16 + 2$

Fondements

Arithmétique binaire

- addition : comme en base 10, mais la table est plus simple...

	1	0	1	0	(10)
+	0	0	1	1	(3)
		1			
=	1	1	0	1	(13)

- soustraction : addition de l'opposé
- représentation des nombres négatifs : complément à 2
  Sur k bits, $-n$ est représenté par $2^k - n$
- exemple : sur 4 bits, -3 est représenté par $16 - 3 = 13$
- $5 - 3$ devient $5 + (16 - 3) = 5 + 13 = 18 = 16 + 2$
- $2^k - n = (2^k - 1) - n + 1 =$ complément à 1 plus 1

Fondements

Arithmétique binaire

- addition : comme en base 10, mais la table est plus simple...

	1	0	1	0	(10)
+	0	0	1	1	(3)
		1			
=	1	1	0	1	(13)

- soustraction : addition de l'opposé
- représentation des nombres négatifs : complément à 2
  Sur k bits, $-n$ est représenté par $2^k - n$
- exemple : sur 4 bits, -3 est représenté par $16 - 3 = 13$
- $5 - 3$ devient $5 + (16 - 3) = 5 + 13 = 18 = 16 + 2$
- $2^k - n = (2^k - 1) - n + 1 =$ complément à 1 plus 1
- $-3 \rightarrow \overline{0011} + 1 = 1100 + 1 = 1101$

Fondements

Arithmétique binaire

- addition : comme en base 10, mais la table est plus simple...

	1	0	1	0	(10)
+	0	0	1	1	(3)
		1			
=	1	1	0	1	(13)

- soustraction : addition de l'opposé
- représentation des nombres négatifs : complément à 2
  Sur k bits, $-n$ est représenté par $2^k - n$
- exemple : sur 4 bits, -3 est représenté par $16 - 3 = 13$
- $5 - 3$ devient $5 + (16 - 3) = 5 + 13 = 18 = 16 + 2$
- $2^k - n = (2^k - 1) - n + 1 =$ complément à 1 plus 1
- $-3 \rightarrow \overline{0011} + 1 = 1100 + 1 = 1101$
- $5 - 3 \rightarrow 0101 + 1101 = 1\,0010 \rightarrow 0010 = 2$

Fondements

Arithmétique binaire

- addition : comme en base 10, mais la table est plus simple...

	1	0	1	0	(10)
+	0	0	1	1	(3)
		1			
=	1	1	0	1	(13)

- soustraction : addition de l'opposé
- représentation des nombres négatifs : complément à 2
  Sur k bits, $-n$ est représenté par $2^k - n$
- exemple : sur 4 bits, -3 est représenté par $16 - 3 = 13$
- $5 - 3$ devient $5 + (16 - 3) = 5 + 13 = 18 = 16 + 2$
- $2^k - n = (2^k - 1) - n + 1 =$ complément à 1 plus 1
- $-3 \rightarrow \overline{0011} + 1 = 1100 + 1 = 1101$
- $5 - 3 \rightarrow 0101 + 1101 = 1\,0010 \rightarrow 0010 = 2$

Fondements

Arithmétique binaire

- sur 4 bits, le plus grand entier positif est $0111 = 7$

Fondements

Arithmétique binaire

- sur 4 bits, le plus grand entier positif est $0111 = 7$
- sur 4 bits, le plus petit entier négatif est $1000 = -8$

Fondements

Arithmétique binaire

- sur 4 bits, le plus grand entier positif est $0111 = 7$
- sur 4 bits, le plus petit entier négatif est $1000 = -8$
- sur k bits, on peut représenter l'intervalle $[-2^{k-1}, 2^{k-1} - 1]$

Fondements

Arithmétique binaire

- sur 4 bits, le plus grand entier positif est $0111 = 7$
- sur 4 bits, le plus petit entier négatif est $1000 = -8$
- sur *k* bits, on peut représenter l'intervalle $[-2^{k-1}, 2^{k-1} - 1]$
- débordement :
  - $$\begin{array}{rccccc} & 5+4 & = & 0101+0100 & = & 1001(9) \\ \rightarrow & \overline{1001}+1 & = & 0110+1 & = & 0111(-7) \end{array}$$

Fondements

Arithmétique binaire

- sur 4 bits, le plus grand entier positif est $0111 = 7$
- sur 4 bits, le plus petit entier négatif est $1000 = -8$
- sur k bits, on peut représenter l'intervalle $[-2^{k-1}, 2^{k-1} - 1]$
- débordement :
  - $$\begin{array}{rcccc} & 5+4 & = & 0101+0100 & = & 1001(9) \\ \rightarrow & \overline{1001}+1 & = & 0110+1 & = & 0111(-7) \end{array}$$
  - $-5 + -4 = 1011 + 1100 = 0111(7)$

Fondements

Arithmétique binaire

- sur 4 bits, le plus grand entier positif est $0111 = 7$
- sur 4 bits, le plus petit entier négatif est $1000 = -8$
- sur k bits, on peut représenter l'intervalle $[-2^{k-1}, 2^{k-1} - 1]$
- débordement :
  - $$\begin{array}{rcccc} & 5+4 & = & 0101+0100 & = & 1001(9) \\ \rightarrow & \overline{1001}+1 & = & 0110+1 & = & 0111(-7) \end{array}$$
  - $-5 + -4 = 1011 + 1100 = 0111(7)$

Octal et hexadécimal

Fondements

Arithmétique binaire

- sur 4 bits, le plus grand entier positif est $0111 = 7$
- sur 4 bits, le plus petit entier négatif est $1000 = -8$
- sur k bits, on peut représenter l'intervalle $[-2^{k-1}, 2^{k-1} - 1]$
- débordement :
  - $$\begin{array}{rcccc} & 5+4 & = & 0101+0100 & = & 1001(9) \\ \rightarrow & \overline{1001}+1 & = & 0110+1 & = & 0111(-7) \end{array}$$
  - $-5 + -4 = 1011 + 1100 = 0111(7)$

Octal et hexadécimal

- octal (base 8) :
  $179 = 128 + 32 + 16 + 2 + 1 = 10\,110\,011_2 = 263_8$

Fondements

Arithmétique binaire

- sur 4 bits, le plus grand entier positif est $0111 = 7$
- sur 4 bits, le plus petit entier négatif est $1000 = -8$
- sur *k* bits, on peut représenter l'intervalle $[-2^{k-1}, 2^{k-1} - 1]$
- débordement :
  - $$\begin{array}{rcccc} & 5+4 & = & 0101+0100 & = & 1001(9) \\ \rightarrow & \overline{1001}+1 & = & 0110+1 & = & 0111(-7) \end{array}$$
  - $-5 + -4 = 1011 + 1100 = 0111(7)$

Octal et hexadécimal

- octal (base 8) :
  $179 = 128 + 32 + 16 + 2 + 1 = 10\,110\,011_2 = 263_8$
- hexadécimal (base 16) :
  $179 = 128 + 32 + 16 + 2 + 1 = 1011\,0011_2 = \mathrm{B3}_{16}$

Matt Parker half-adder

Logique séquentielle

- Les circuits vus jusqu'ici calculent des fonctions

Matt Parker half-adder

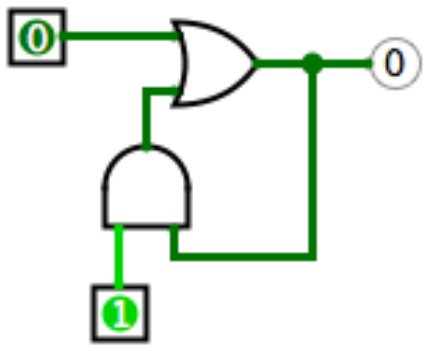
Logique séquentielle

- Les circuits vus jusqu'ici calculent des fonctions
- Ils n'ont pas d'état et ne mémorisent pas de valeurs

Fondements

Matt Parker half-adder

Logique séquentielle

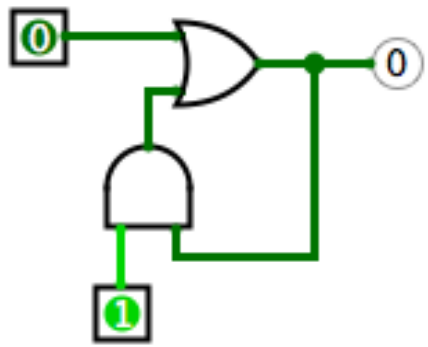
- Les circuits vus jusqu'ici calculent des fonctions
- Ils n'ont pas d'état et ne mémorisent pas de valeurs
- Mémoriser une valeur : 

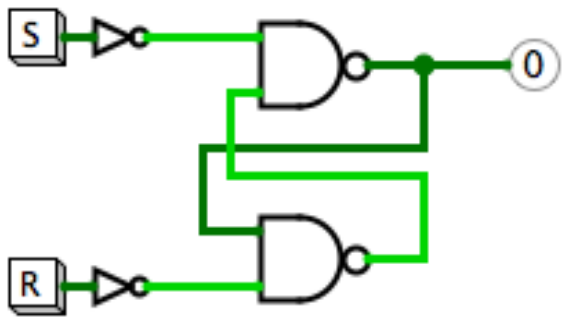
circuit

Fondements

Matt Parker half-adder

Logique séquentielle

- Les circuits vus jusqu'ici calculent des fonctions
- Ils n'ont pas d'état et ne mémorisent pas de valeurs
- Mémoriser une valeur : 

circuit
- Bascule SR : 

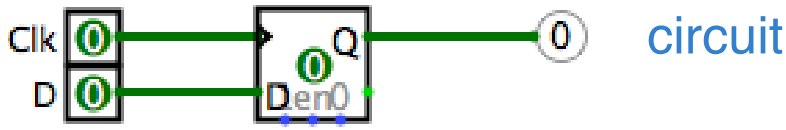
circuit

Logique séquentielle

- la bascule SR mémorise la dernière entrée passée à 0

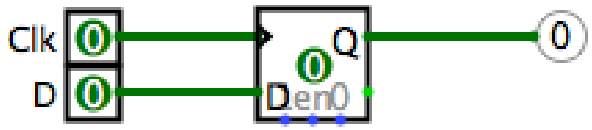
Logique séquentielle

- la bascule SR mémorise la dernière entrée passée à 0
- en combinant plusieurs étages de mémorisation, on peut réaliser un circuit qui mémorise la valeur d'une de ses entrées (D) lorsque l'autre (Clk) passe de 0 à 1 : bascule D maître-esclave



Logique séquentielle

- la bascule SR mémorise la dernière entrée passée à 0
- en combinant plusieurs étages de mémorisation, on peut réaliser un circuit qui mémorise la valeur d'une de ses entrées (D) lorsque l'autre (Clk) passe de 0 à 1 : bascule D maître-esclave



circuit

- plusieurs bascules D en parallèle constituent un registre exemple

Suite...

Ordinateurs