

# Le microcontrôleur

- Avec un sujet pareil, je suis bon pour être invité à des dîners de cons du mercredi
- Et pourtant ...



# Le microcontrôleur

- Au sommet de la chaîne du traitement de l'information, on trouve**
- **des gigantesques volumes de données**, essentiellement nourris par l'Internet mondial,
  - **des puissants systèmes informatiques**,
  - les logiciels les plus évolués comme **l'intelligence artificielle**, capable de discernement et d'apprentissage.

**A la base de cette chaîne et au plus près du matériel, on trouve les microcontrôleurs.**  
Ils sont à vos cotés tous les jours pour vous faciliter la vie.  
Pour autant ils restent discrets, cachés au sein de vos appareils domestiques.  
On les trouve aussi ...

- dans les appareils industriels,
- dans les machines qui fabriquent les appareils domestiques et industriels,
- dans les machines qui fabriquent les machines qui fabriquent les appareils ...

# Pourquoi utiliser un microcontrôleur ?

- Les microcontrôleurs sont préférés aux ordinateurs dans de nombreuses applications pour plusieurs raisons :
  - **Coût** : Les microcontrôleurs sont généralement moins chers que les ordinateurs. Ils sont programmés pour des tâches spécifiques et **ils intègrent donc uniquement les fonctionnalités nécessaires**
  - **Taille et intégration** : Les microcontrôleurs sont de petite taille et peuvent être intégrés directement dans des applications embarquées où l'espace est limité
  - **Consommation d'énergie** : Les microcontrôleurs sont conçus pour une faible consommation d'énergie, ce qui les rend adaptés aux applications alimentées par batterie ou nécessitant une bonne efficacité énergétique
  - **Temps de réponse rapide** : Les microcontrôleurs sont capables de réagir rapidement aux événements en temps réel en raison de leur architecture et de leur conception optimisées, ce qui les rend adaptés aux systèmes de contrôle et de surveillance en temps réel
  - **Fiabilité** : Les microcontrôleurs sont souvent conçus pour être robustes et fiables, ce qui les rend adaptés aux environnements industriels et automobiles exigeants
  - **Simplicité de conception** : Les microcontrôleurs sont plus simples à programmer et à utiliser que les ordinateurs, ce qui réduit le temps de développement



# Analogie fonctions humaines

- **Les gros systèmes informatiques sont plutôt dédiés au traitement de l'information pure.**  
**Ils seraient plutôt à mettre en parallèle avec les raisonnements conscients et élaborés traités dans le cerveau**
- **Les microcontrôleurs sont plutôt dédiés à des fonctions en interface avec le matériel.**  
**Dans l'analogie, ils seraient plutôt dédiés aux fonctions automatiques et décentralisées en interaction avec l'environnement extérieur**
  - battements du cœur,
  - Marche,
  - clignement des paupières,
  - adaptation des pupilles à la lumière,
  - tremblements de réaction au froid
  - ...



# Exemple de la cafetière

- **Besoin**
  - **Attendre l'appui sur le bouton « démarrage »**
  - **Activer la résistance de chauffage de l'eau**
  - **Attendre que le capteur de température indique que l'eau est chaude**
  - **Activer la pompe qui pousse l'eau chaude à travers la capsule**
  - **Attendre la durée nécessaire au remplissage de la tasse**
  - **Arrêter le chauffage et la pompe à eau**
  - **Pendant tout le cycle, surveiller le niveau d'eau et couper le chauffage et la pompe en cas de niveau bas**



# Exemple du poêle à granulés

- **Besoins :**
  - **faire tomber les granulés au fil de l'eau** dans le creuset,
  - **surveiller la température** de la pièce, adapter le rythme d'alimentation des granulés en fonction de la différence entre la consigne et la température de la pièce,
  - **souffler sur les braises** pour les maintenir rouges,
  - **ré allumer les granulés** après que le chauffage a été arrêté si une température excessive a conduit à un arrêt du chauffage,
- **Que diriez-vous si un petit être électronique faisait tout ça pour vous à longueur de journée et de nuit ?**
- Pendant ce temps, vous pouvez vous consacrer à autre chose (y compris dormir) en profitant du confort. **Petit microcontrôleur veille.**

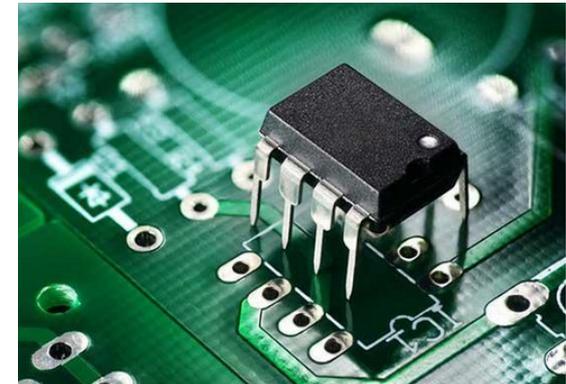
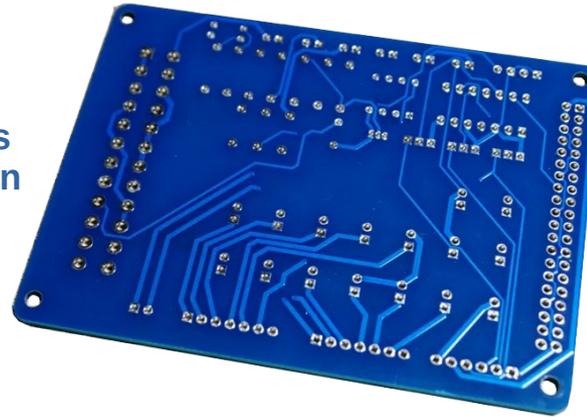
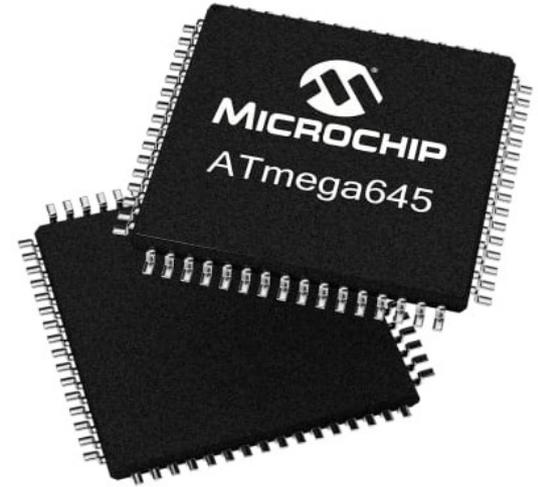
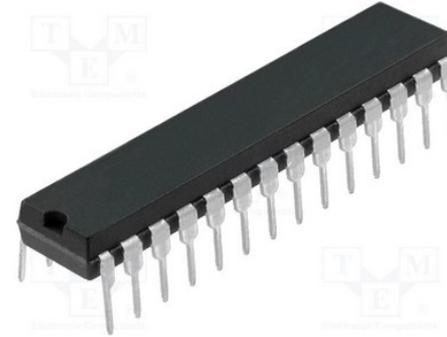


# Autres exemples d'utilisation

- **Les microcontrôleurs sont au cœur du fonctionnement de l'ABS des voitures**
  - **Ils surveillent en temps réel les vitesses de rotation des roues** à l'aide de capteurs de vitesse
  - **Lorsqu'un capteur détecte que la vitesse de rotation d'une roue diminue brusquement, ce qui indique un risque de blocage, le microcontrôleur réagit en relâchant puis en appliquant à nouveau rapidement la pression sur le frein de cette roue de manière pulsée.**
- **Les microcontrôleurs sont couramment utilisés dans les systèmes de climatisation modernes, qu'il s'agisse de climatisation résidentielle, commerciale ou automobile**
  - **Contrôle de la température**
  - **Gestion des modes de fonctionnement** : refroidissement, chauffage, déshumidification
  - **Contrôle de la vitesse du compresseur**
  - **Régulation de la circulation d'air** : Les microcontrôleurs contrôlent les moteurs des ventilateurs
  - **Optimisation de l'efficacité énergétique** : surveillance de la consommation d'énergie et ajustement des paramètres de fonctionnement

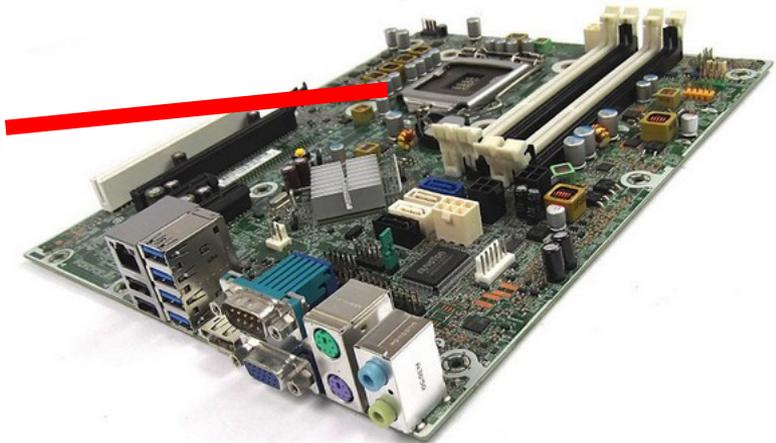
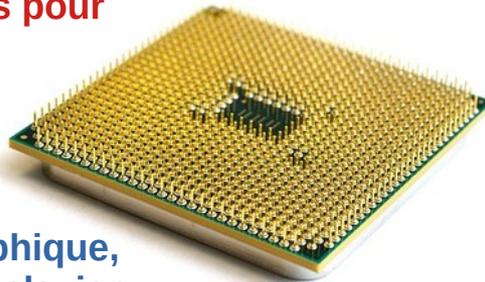
# Vocabulaire

- Les **circuits intégrés** ont des **pattes**
  - J'utilise indifféremment les mots **pattes**, **broches** ou **pinoches**
  - Les circuits intégrés sont aussi appelés **puces électroniques**  
Donc quand vous appelez votre conjoint « ma puce » ...
- En utilisation, les circuits intégrés sont soudés sur des **circuits imprimés**
  - Circuit imprimé = pistes conductrices tracées sur la surface d'une plaque en époxy
  - Il existe des circuits imprimés multicouches
- Des circuits imprimés équipés de composants forment des **cartes électroniques**



# Microcontrôleur vs microprocesseur

- **Le microprocesseur est massivement optimisé pour la vitesse de calcul**
  - **Il est destiné à être implanté au sein d'une carte mère**
  - **Il a besoin**
    - **d'un dispositif de refroidissement**
    - **de suffisamment de broches pour communiquer avec**
      - la mémoire
      - les circuits d'interfaces
    - **de barrettes mémoire**
    - **de périphériques (carte graphique, périphériques USB tels que clavier, souris ...)**
- **Le microcontrôleur embarque sur un seul circuit intégré tout ce qu'il faut pour fonctionner**
    - **Un processeur modeste mais vaillant**
    - **De la mémoire**
    - **Des interfaces de communication**



# L'électronique veille ...

- **Il est aujourd'hui habituel que l'électronique reste sous tension pour attendre une action de la part de quelqu'un.**
  - Boutons tactiles d'ascenseur
  - Interrupteurs à effleurement de maison
  - Récepteurs des télécommandes
  - Boutons poussoirs (monostables)
- **Tous ces dispositifs nécessitent qu'une électronique surveille des changements d'état (capteur infrarouge pour la télécommande)**
- **C'est typiquement l'une des missions d'un microcontrôleur : attendre un changement d'état sur une de ses pattes pour effectuer un traitement (et peut-être même activer quelque chose si c'est le moment)**



# Un microcontrôleur, c'est quoi ?

- **C'est un circuit électronique microprogrammé**
- C'est à dire un circuit électronique capable d'exécuter un programme informatique (une séquence d'instructions)
- Une fois le programme implanté dans le microcontrôleur (depuis un ordinateur via un câble USB), il tourne en continu dès la mise sous tension
- Petit microcontrôleur prend vie
- Il en existe de différentes tailles de microcontrôleurs, de différentes capacités
- **Le programme : c'est là que tout le talent de l'artiste peut s'exprimer : faire simple, court et efficace**



# Un microcontrôleur, c'est quoi ?

- **C'est un circuit électronique qui dispose de pattes pour échanger des tensions électriques avec l'extérieur (entrées-sorties)**
- **Une entrée-sortie est une patte sur laquelle le programme peut lire ou forcer une tension électrique (typiquement 5 volts)**
- **Le programme peut lire une entrée pour récupérer une information (état d'un capteur, appui sur un bouton ...)**
- **Le programme peut écrire sur une sortie pour commander une action (allumer une lampe, démarrer un moteur ...) par l'intermédiaire d'un circuit de puissance**

Capteurs  
Boutons poussoirs  
Codeurs



Commandes  
séquencées

Moteurs  
Vérins  
Lampes  
Afficheurs



# Un microcontrôleur, c'est quoi ?

- **C'est un circuit électronique qui dispose d'interfaces de communications vers d'autres dispositifs électroniques**
  - USB (Universal Serial Bus)
  - UART (Universal Asynchronous Receiver Transmitter)
  - I2C (Inter Integrated Circuit)
  - ...
- **Ça sert d'abord à transférer le programme de l'ordinateur vers le microcontrôleur**
- **Ça sert aussi à communiquer avec des périphériques (horloge, afficheurs, GPS, capteurs ...)**
- **Ça peut aussi servir à faire communiquer des microcontrôleurs entre eux ou avec des ordinateurs**



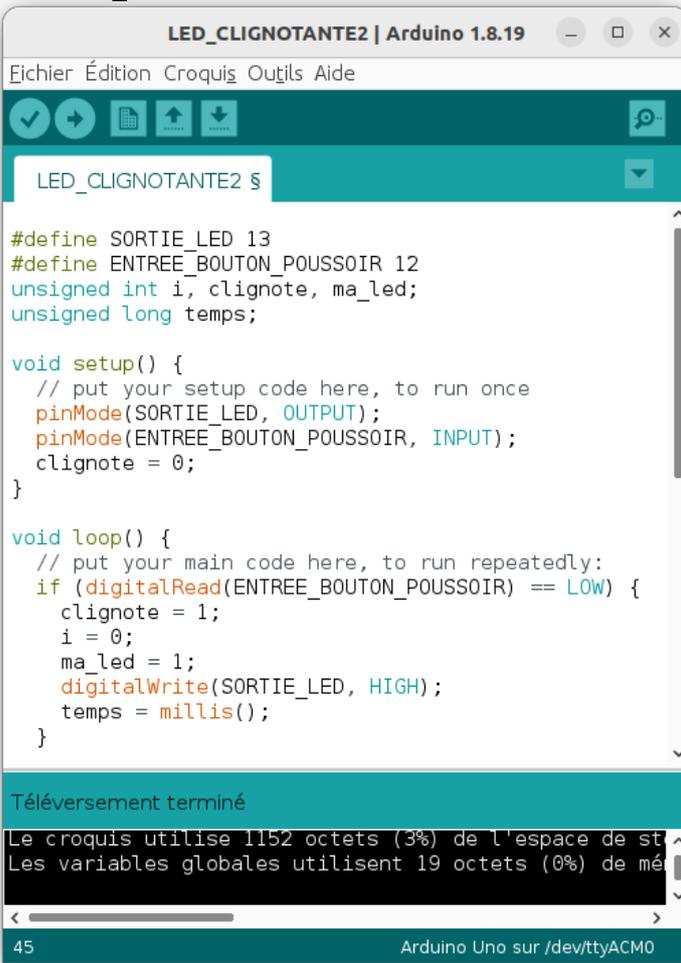
# Un programme c'est quoi ?

- **Un programme est une suite d'instructions** comparable à une recette de cuisine
- Parmi les instructions, on trouve
  - Des opérations arithmétiques
  - Des tests
  - ...
- Un test permet l'exécution conditionnelle d'instructions en fonction du résultat d'une comparaison de deux valeurs
- **Les tests permettent de faire des choix**
  - Par exemple, arrêt d'une résistance électrique de chauffage d'eau si la température est atteinte



# Un programme c'est quoi ?

- Il y a bien longtemps que personne n'écrit plus de programme en langage machine (binaire)
- Les programmes sont écrits sous forme de textes avec une syntaxe appelée langage de programmation (assembleur, C, C++, java ...)
  - Programme Arduino ci-contre : langage C
- Un programme appelé compilateur est lancé sur le PC, il transforme le texte en codes machine
  - Le programme en langage machine peut ensuite être transféré de l'ordinateur vers le microcontrôleur



```
LED_CLIGNOTANTE2 | Arduino 1.8.19
Echier Édition Croquis Outils Aide
LED_CLIGNOTANTE2 $

#define SORTIE_LED 13
#define ENTREE_BOUTON_POUSSOIR 12
unsigned int i, clignote, ma_led;
unsigned long temps;

void setup() {
  // put your setup code here, to run once
  pinMode(SORTIE_LED, OUTPUT);
  pinMode(ENTREE_BOUTON_POUSSOIR, INPUT);
  clignote = 0;
}

void loop() {
  // put your main code here, to run repeatedly:
  if (digitalRead(ENTREE_BOUTON_POUSSOIR) == LOW) {
    clignote = 1;
    i = 0;
    ma_led = 1;
    digitalWrite(SORTIE_LED, HIGH);
    temps = millis();
  }
}

Téléversement terminé
Le croquis utilise 1152 octets (3%) de l'espace de st
Les variables globales utilisent 19 octets (0%) de mé
45 Arduino Uno sur /dev/ttyACM0
```

# Analogie avec le livre

- **Valeur importante = matière immatérielle**
  - Papier et encre = support (pas si simple à l'époque de Néandertal ...)
  - L'organisation des dépôts d'encre sur le papier donne du sens
- **Coûts récurrents en baisse (liés au support de l'information)**
  - Avec en particulier le partage sur ordinateur
- **Pour nous : information immatérielle = programme du microcontrôleur**
  - **Variables / constantes**
  - **Programme lui-même**

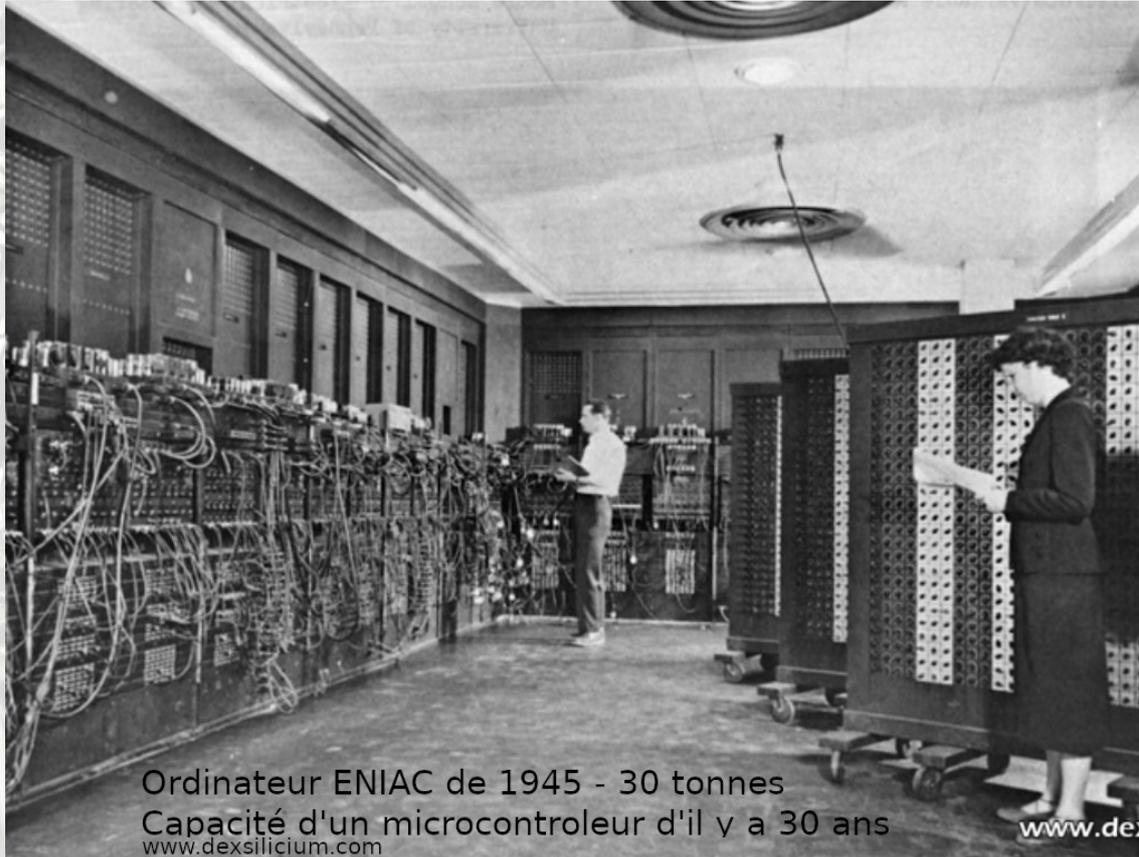


# Dans un produit informatique

- **Il y a deux types de composants :**
  - Des composants matériels
  - Des composants logiciels
- **Matériel = coûts récurrents**
  - Cartes électroniques
  - Boîtier, alimentation
  - Périphériques ...
- **Logiciel = coûts non récurrents**
  - Système d'exploitation
  - Logiciels applicatifs ...
- **Les endroits où on trouve du logiciel se multiplient**
- **Les logiciels sont de plus en plus complexes au fur et à mesure que les capacités augmentent (mémoire, vitesse de calcul, nombre d'entrées-sorties, débit des interfaces de communication)**



# Les capacités informatiques évoluent

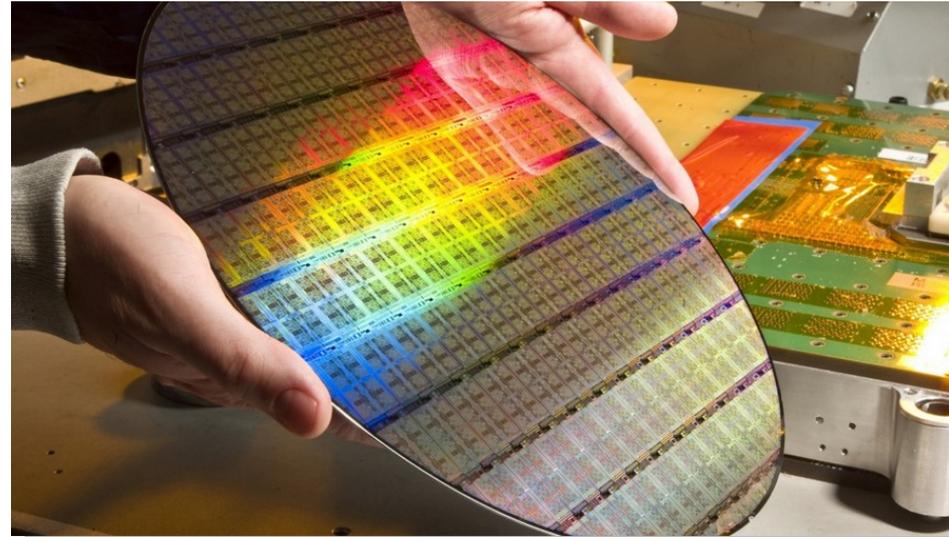


Ordinateur ENIAC de 1945 - 30 tonnes  
Capacité d'un microcontrôleur d'il y a 30 ans  
[www.dexsilicium.com](http://www.dexsilicium.com)

[www.dexsilicium.com](http://www.dexsilicium.com)

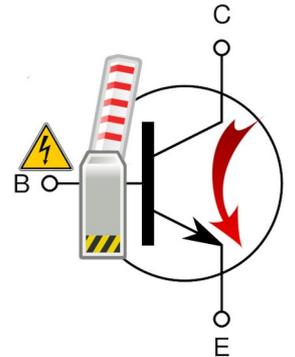
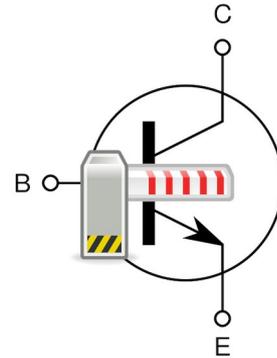
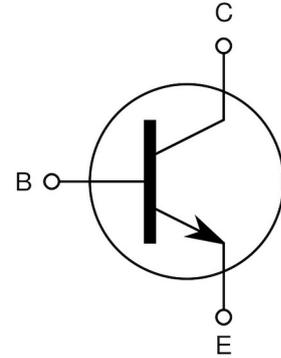
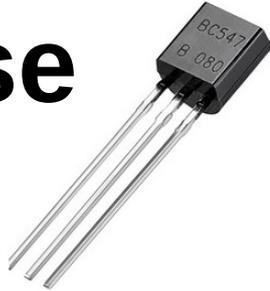
# Le semi conducteur

- **Le semi conducteur**
  - **ni vraiment conducteur**
  - **ni vraiment isolant**
  - **Dopé avec des atomes choisis, il peut être plus ou moins conducteur, voire isolant**
- **Suivant le dopage et le type de jonction entre zones dopées, il peut devenir conducteur sous l'effet d'une tension électrique**



# Le transistor : la brique de base

- Il en existe plusieurs types de transistors
  - On s'en moque, on n'est pas là pour faire de l'électronique
  - Le transistor est le plus petit élément d'un dispositif électronique (= jonctions entre zones de semi-conducteurs dopés)
  - Il peut fonctionner en étant passant ou bloqué ou entrouvert. **En numérique, il est passant ou bloqué.**
- **Un microcontrôleur est constitué d'un ensemble organisé d'un grand nombre de transistors**



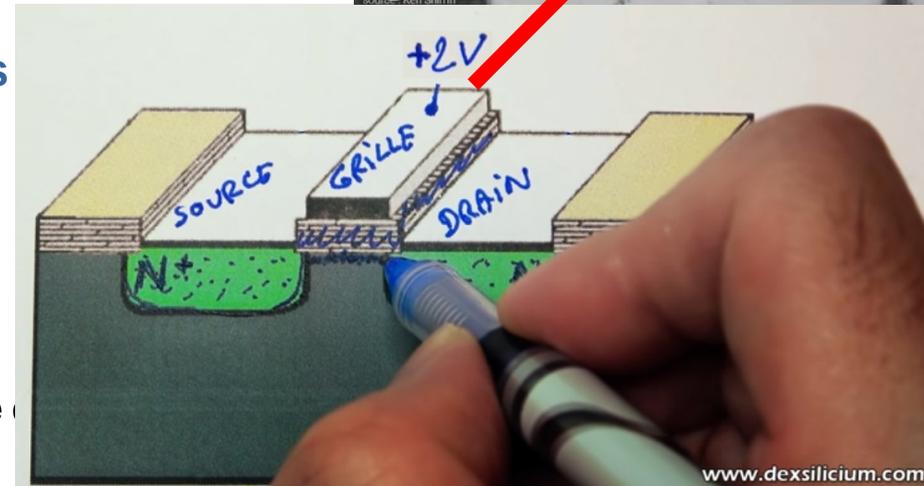
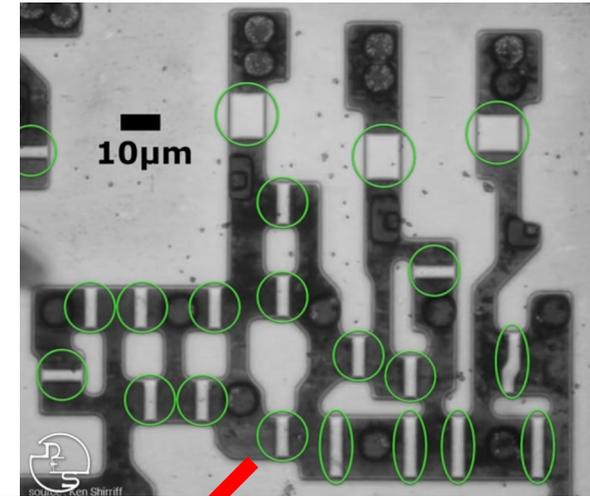
# Combien de transistors ?

- **Se compte aujourd'hui en milliards**
- **Circuit intégré sans le boîtier**
  - **plaquette de silicium sur laquelle sont « gravés » les transistors**
  - **connections périphériques vers les pattes du circuit intégré**



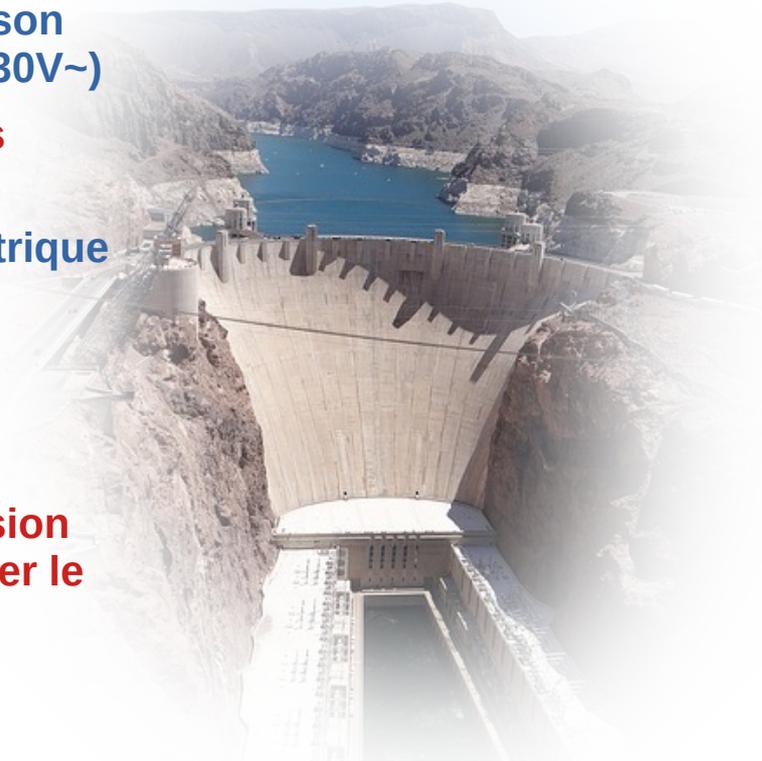
# La fabrication en deux mots

- Structures 10 à 100 fois plus petites qu'un cheveu
- Plaque de base en silicium fondu
- Cycle de dépôts de matériau dopé, masquages et gravures en salle blanche
  - Masquages par lithographie optique
  - Connexions entre transistors par dépôts de cuivre
- C'est le monde du minuscule, de l'optique, du bombardement électronique, de la propreté ...



# La tension, le courant, pour rappel

- **Volt = l'unité de tension électrique** (notre maison est en général alimentée électriquement en 230V~)
- **Tension = différence de potentiels électriques entre deux points**
- **Pour simplifier, on peut voir un potentiel électrique comme une accumulation d'électrons**
- **Un potentiel électrique est comparable à une différence de pressions ou une différence de hauteurs d'eau**
- **Si on compare la tension électrique à la pression de l'eau dans une tuyauterie, on peut comparer le courant au débit d'eau correspondant.**  
**Unité de courant électrique : l'Ampère**

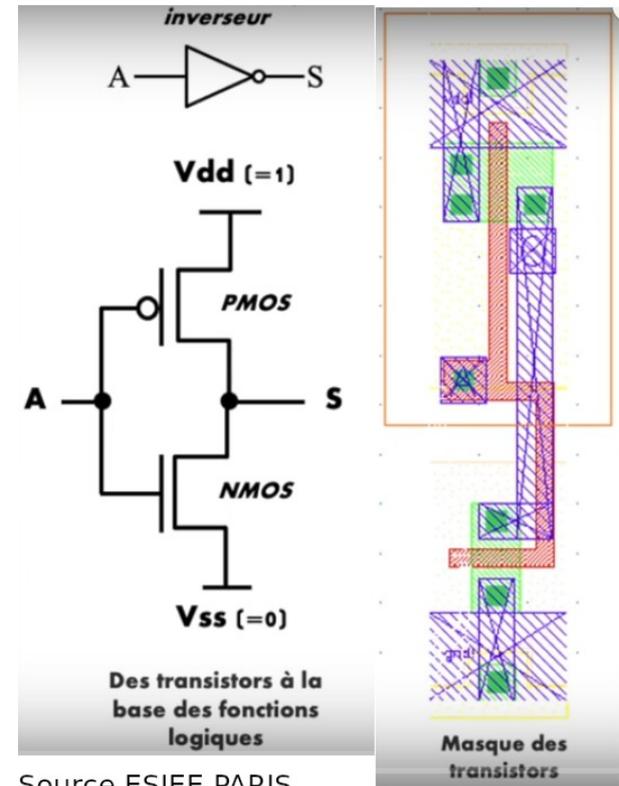


# Le binaire c'est quoi ?

- **L'informatique manipule des 0 et des 1**
  - 0 = bas / faux
  - 1 = haut / vrai
- **En général, en électronique numérique, on prend :**
  - 0 = 0 Volt
  - 1 = 5 Volts (ou 3,3 Volts suivant circuits)
- **Pourquoi des 0 et des 1 ?**
  - **Parce que c'est plus simple**
    - tension présente ou absente,
    - transistor passant ou bloqué
  - **Parce que c'est tolérant** : si je présente 3,5 Volts au lieu de 5V sur une entrée (= patte déclarée en entrée), le circuit numérique comprend 1 quand même et l'information reste intègre

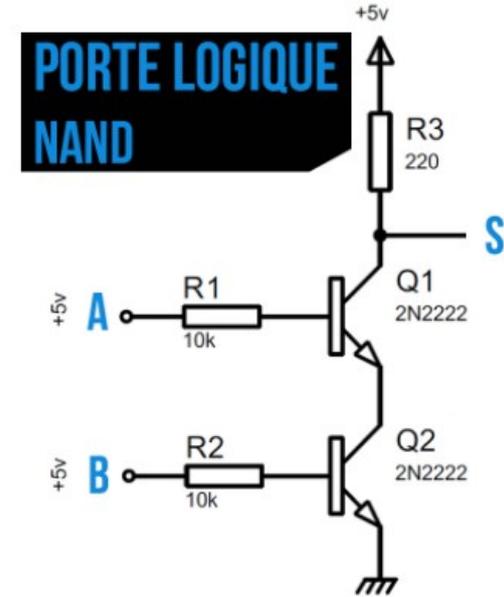
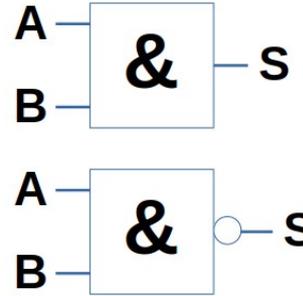
# Du transistor à la porte logique

- **Un circuit intégré numérique :**
  - est une **plaque de semi conducteur constitué en couches**
  - dans laquelle sont gravés des transistors
  - Les transistors sont organisés en **portes logiques**
  - à partir des portes logiques on construit les **fonctions binaires** (mémoires, compteurs, registres, bascules ...)
- **Le tout est placé dans un boîtier en plastique**
  - L'ensemble constitue un **composant matériel**
  - Dans le cas du microcontrôleur, le petit bout de silicium devient capable d'exécuter un **programme**

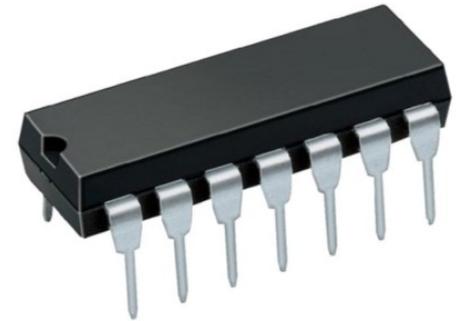
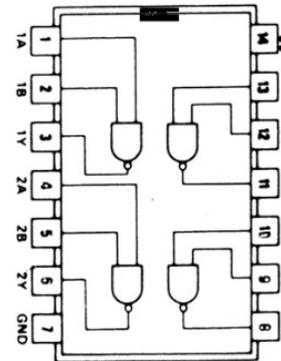


# Les portes logiques

- Il existe plusieurs sortes de portes logiques
  - La porte « ou »  
La sortie est à 1 si l'une OU l'autre des entrées est à 1
  - La porte « et »  
La sortie est à 1 si les deux entrées sont à 1
  - L'inverseur  
la sortie est l'inverse de l'entrée (entrée à 1 = sortie à 0 et vice versa)
- On trouve aussi les porte inversées « non-et » et « non-ou » (NAND ou NOR pour les anglo-saxons)
  - Ce sont les mêmes portes la sortie inversée
- Un microcontrôleur est constitué d'un ensemble organisé d'un grand nombre de portes logiques

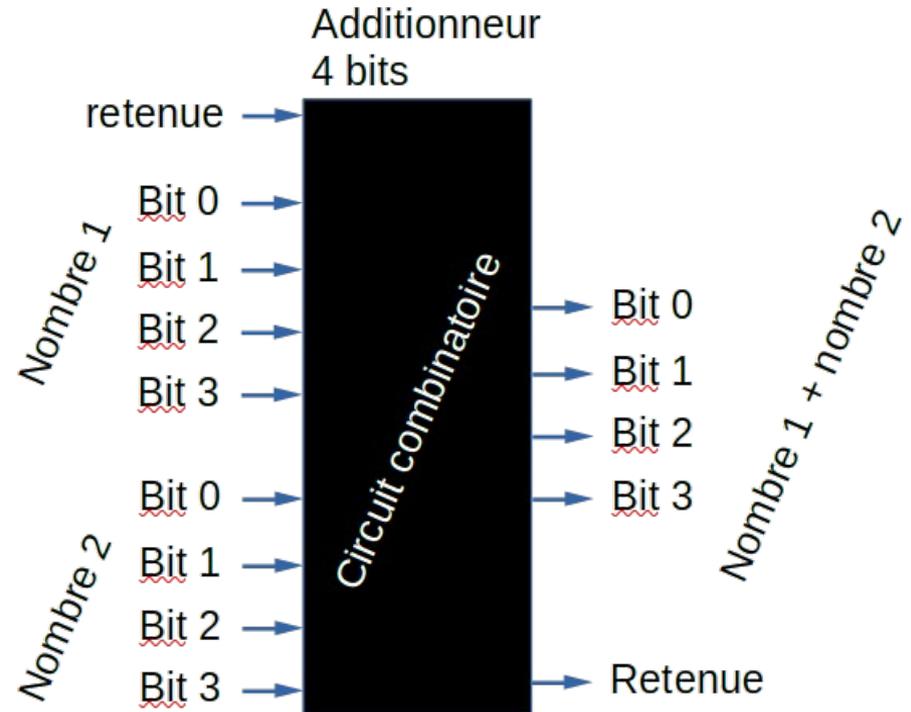


Circuit Intégré TTL 74026 Quadruple Porte NAND À 2 Entrées Et Sortie À Collecteur Ouvert



# Combinatoire vs séquentiel

- **Combinatoire :**
  - Un jeu d'entrées donné génère toujours la même sortie
  - Exemple : additionneur 4 bits
- **Séquentiel :**
  - Les changements d'état sont cadencés par un signal qui varie
    - Bouton poussoir
    - Horloge (quartz)
    - Barrière infrarouge ...
  - Les systèmes microprogrammés exécutent les instructions au rythme d'une horloge
- **Le microcontrôleur utilise les deux modes**



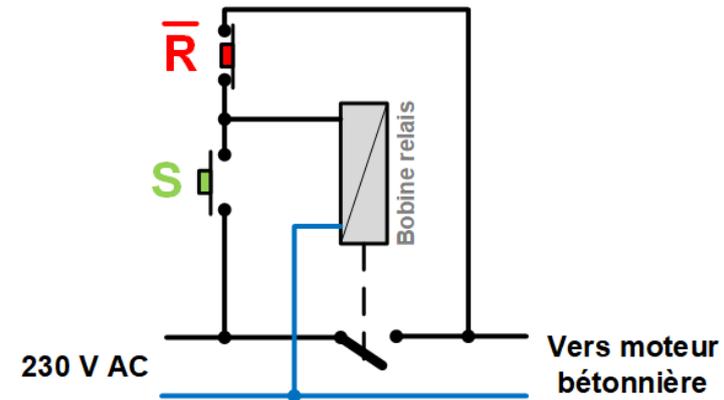
# L'interrupteur de bétonnière

- **Mise en situation**

- Un opérateur utilise une bétonnière électrique (par exemple pour faire du béton)
  - Survient une coupure de courant !
  - L'opérateur pense que ça va durer, il vide le béton et se lance dans le nettoyage de sa bétonnière
  - Pendant qu'il passe le Mirror à l'intérieur, le courant revient.
- Il ne faut pas que la bétonnière se remette en route, sinon ça lui casse les bras et ça met du Mirror partout**

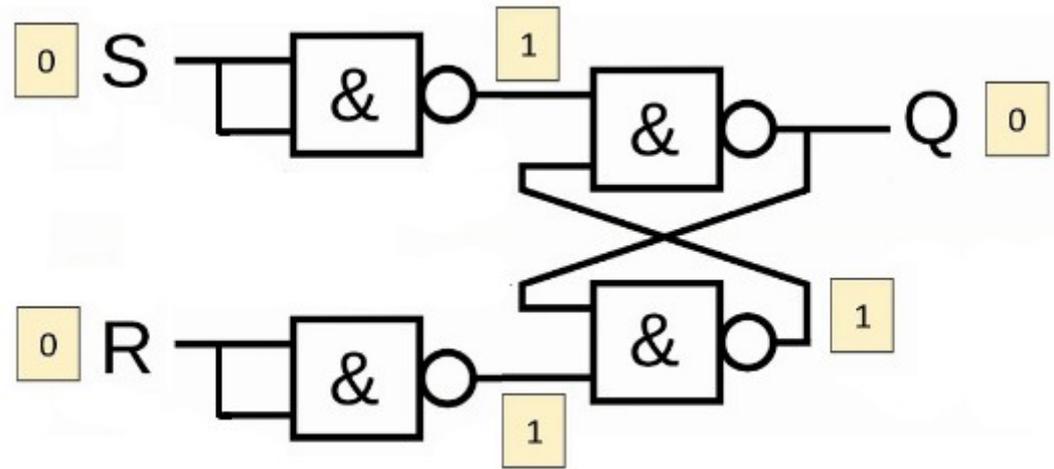
- **L'interrupteur de sécurité ...**

- Présente deux boutons poussoirs :  
marche et arrêt
- **Appui sur marche : ça tourne**  
**Appui sur arrêt : ça s'arrête**  
**Coupure de courant : ça tourne plus**  
**Retour de courant : ça tourne pas**
- **Fonctionnement avec un relais**

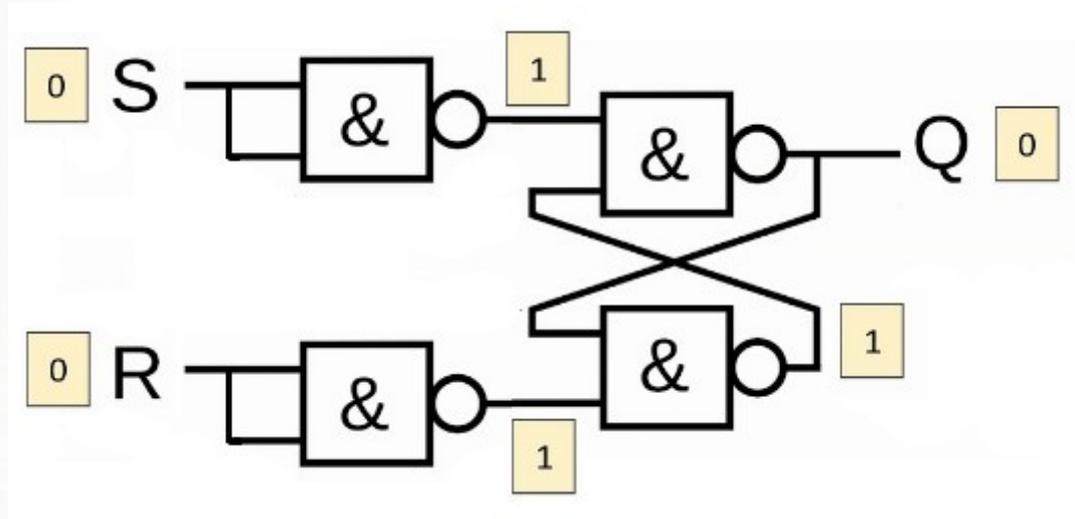


# Exemple d'utilisation des portes

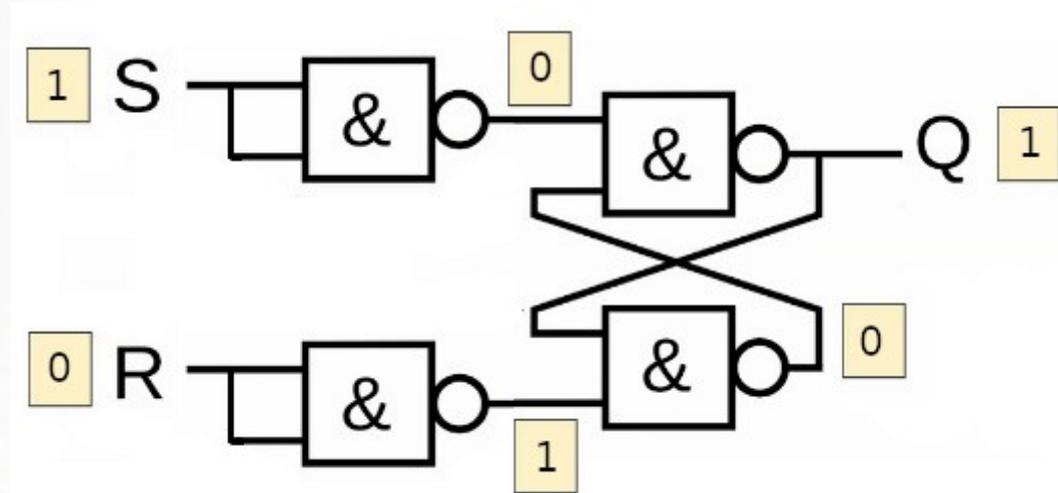
- **La bascule RS**, un exemple remarquable qui se comporte comme l'interrupteur de bétonnière
- **Sa sortie dépend de l'état des entrées mais pas seulement. Elle dépend aussi de la séquence d'application des entrées.**
- **Ce circuit préfigure la réalisation de mémoires, de registres à décalage, de compteurs ...**



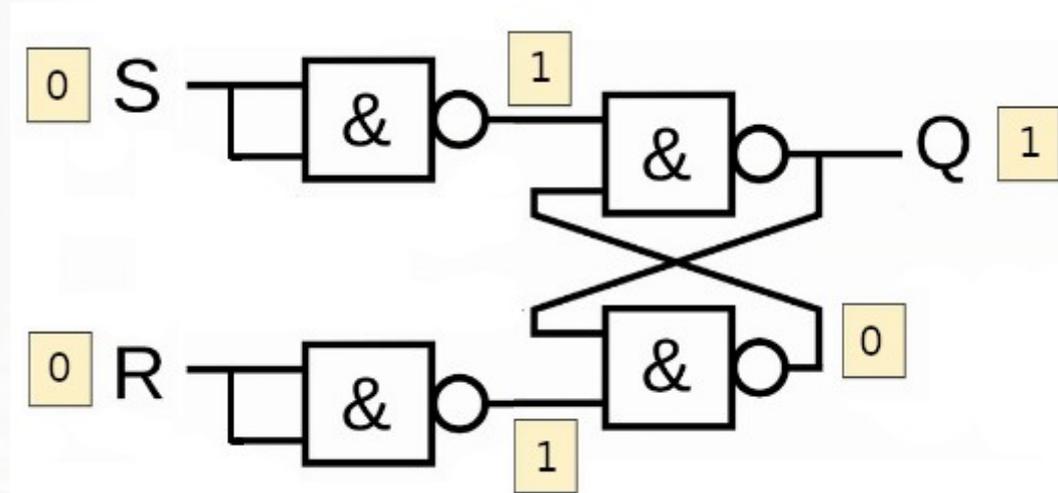
# La bascule RS



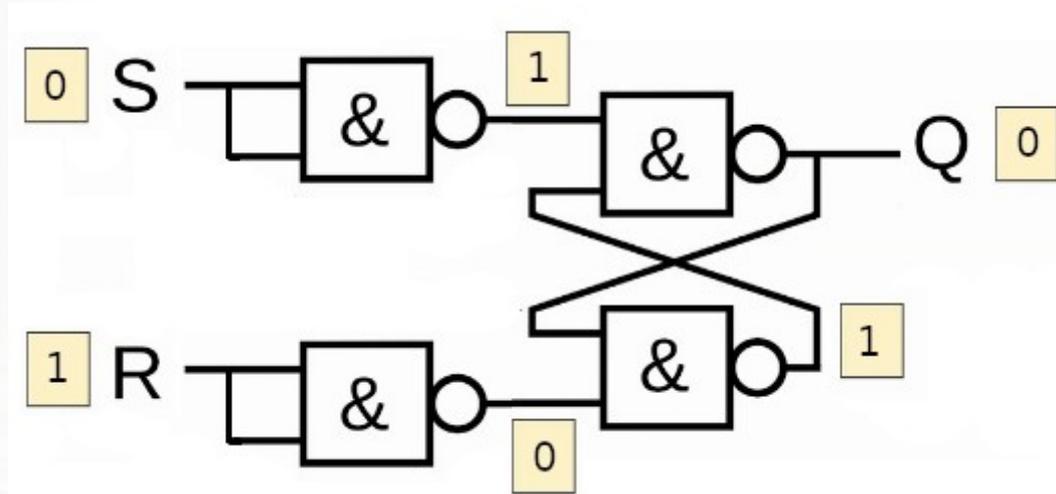
# La bascule RS (suite)



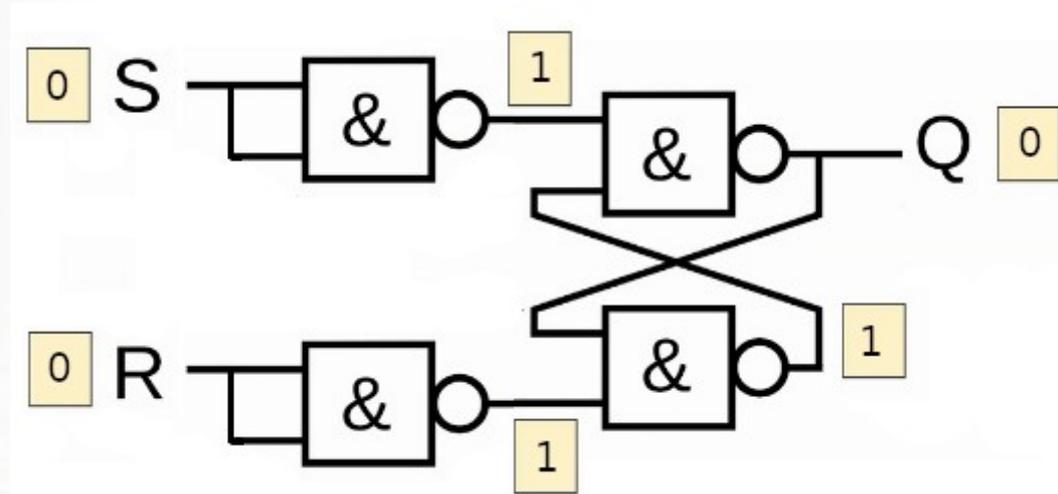
# La bascule RS (suite)



# La bascule RS (suite)



# La bascule RS (suite)

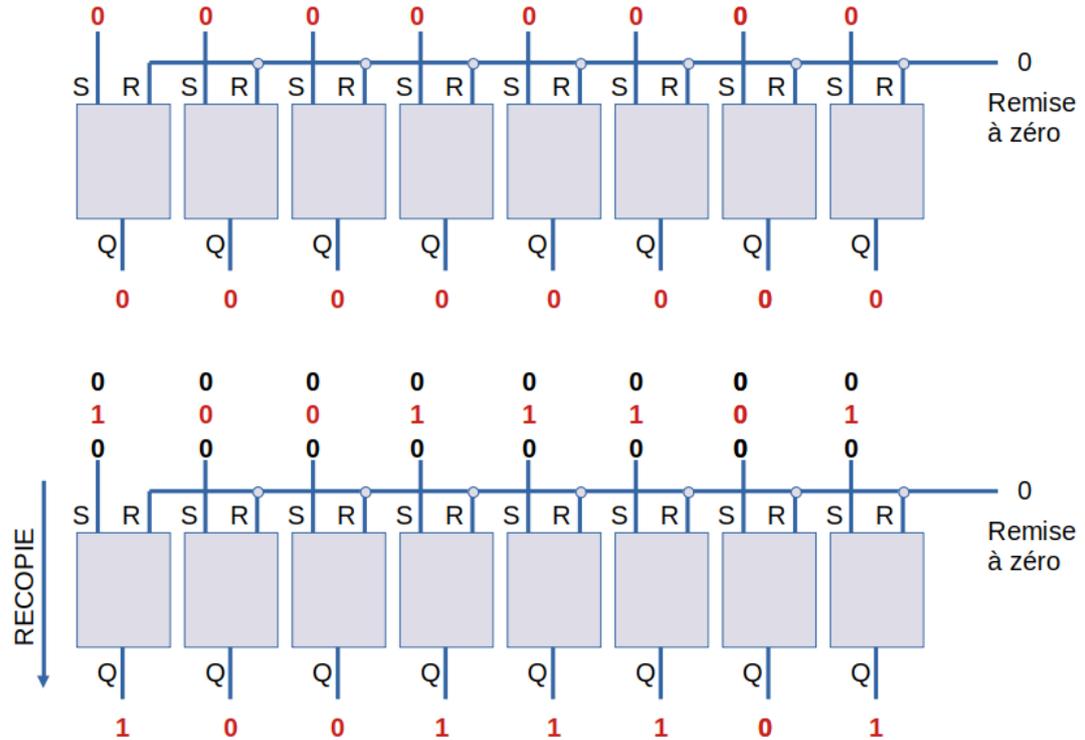


# Le bit, l'octet

- **0 ou 1**
  - Une information binaire
  - **1 bit** (masculin SVP)
- **Octet**
  - L'usage est de grouper les informations binaires par 8
  - Pourquoi 8 : parce que c'est une puissance de deux bien adaptée aux premiers processeurs
  - **8 bits = 1 octet**
  - **01100010** : octet

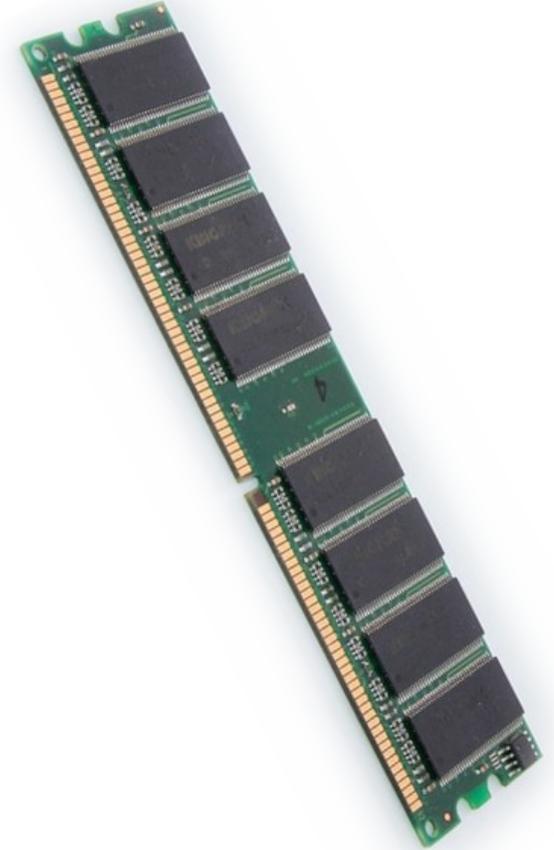
# La mémoire - principe

- **Si on met 8 bascules RS en parallèle, on a fait une mémoire capable de retenir un octet**
- **Principe illustration :**
  - Mise à zéro de la mémoire en passant les entrées R des bascules RS à 1 puis à 0
  - Recopie d'un octet des entrées S des bascules RS vers les sorties Q
  - Les sorties Q restent mémorisées bien que les entrées S soient revenues à 0
- **Pour faire une mémoire, il faut aussi « adresser » les cases mémoire**
  - = pouvoir sélectionner chacune d'elles avec une adresse



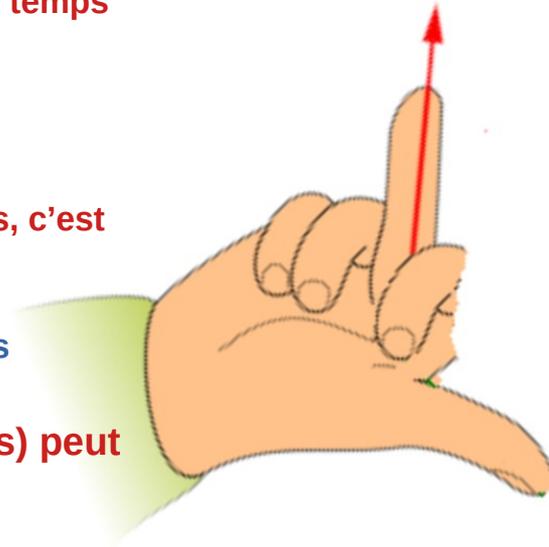
# Capacité mémoire

- Aujourd'hui, la capacité mémoire d'un ordinateur personnel est de l'ordre de quelques Go
  - Avec des adresses sur 32 bits, on peut adresser 4294967295 ( $2^{32}-1$ ) cases mémoire (4 Go)
- Pour rappel
  - kilo =  $\times 1000 = \times 10^3$
  - Mega =  $\times 1000000 = \times 10^6$
  - Giga =  $\times 1000000000 = \times 10^9$
- 1 Go = 1 milliard d'octets
- La capacité mémoire d'un microcontrôleur est de l'ordre de quelques ko
  - L'espace est compté pour le programme et les données



# On représente quoi avec des octets ?

- **Ce geste technique peut avoir plusieurs significations**
  - **Ami automobiliste, je suis passé devant parce que mon temps est précieux ...**  
= affirmation d'un statut
  - **Ami motard en panne, j'ai pas d'outil ...**  
= aveu d'impuissance ...
  - **J'ai pas tellement envie de faire ce que tu me demandes, c'est peut-être mieux si tu te débrouilles ...**  
= suggestion
  - **geste à manier avec précaution, tout le monde n'est pas capable de comprendre**
- **Les octets c'est pareil, un même (groupe d') octet(s) peut avoir plusieurs significations**
  - **des caractères**
  - **un entier**
  - **un réel**
  - **une instruction**



# Représentation d'un entier

- **Un ou plusieurs octets peuvent représenter un entier**
  - En général sur plusieurs octets
  - **On compte en base 2 :**  
2 signes '0' et '1'
- **Pour pouvoir représenter les nombres négatifs, on réserve un bit pour le signe**
  - Complément à deux : pour qu'une soustraction puisse être réalisée avec une addition
- **Exemple**
  - 125 en décimal s'écrit 1111101 en binaire

Décimal - base 10

1 x  $10^2$  = 1 x 100  
+ 2 x  $10^1$  = 2 x 10  
+ 5 x  $10^0$  = 5 x 1

0 0 0 0 1 2 5  
10 signes de '0' à '9'

Binaire - base 2

1 x  $2^6$  = 1 x 64  
+ 1 x  $2^5$  = 1 x 32  
+ 1 x  $2^4$  = 1 x 16  
+ 1 x  $2^3$  = 1 x 8  
+ 1 x  $2^2$  = 1 x 4  
+ 0 x  $2^1$  = 0  
+ 1 x  $2^0$  = 1

0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 1  
2 signes '0' et '1'

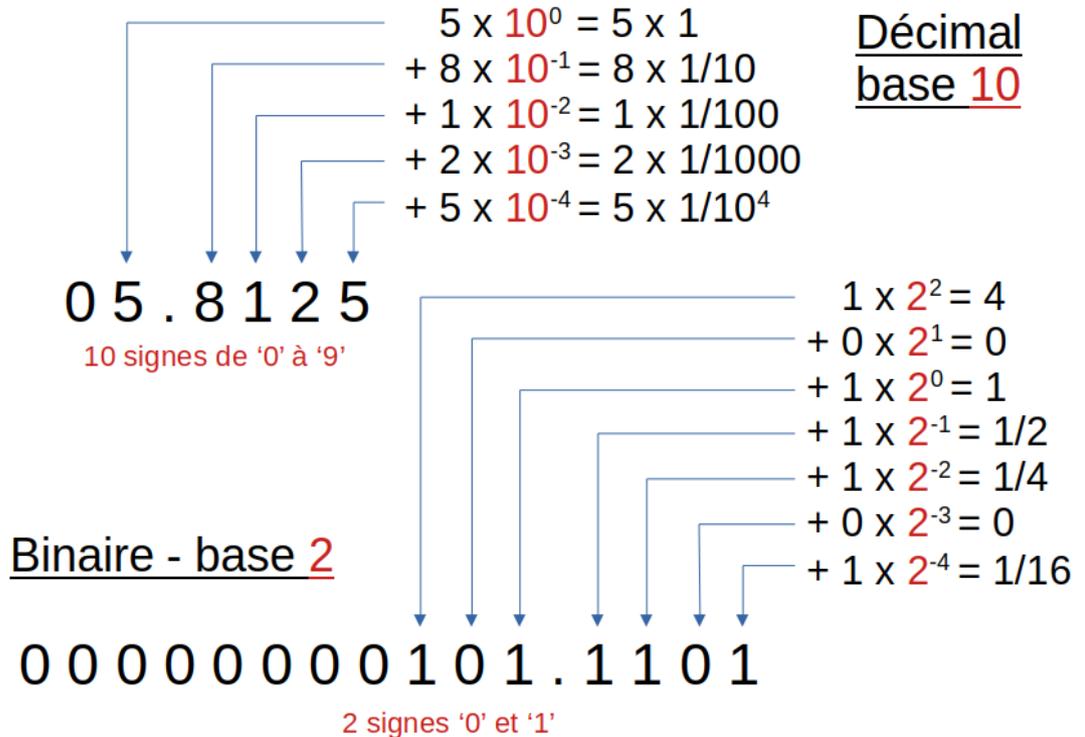
# Un octet peut représenter un caractère

- **La table ASCII**
  - American Standard Code for Information Interchange
  - Convention qui date des années 60
  - Convention qui fait correspondre chaque valeur que peut prendre un octet avec la représentation d'un caractère
- **Exemples :**
  - 0000010 : CTRL-C
  - 00100000 : espace
  - 01000001 : 'A'
- **Les 31 premières valeurs sont des caractères de contrôle en général non affichables**

Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	\$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-`	63	3F	?	95	5F	`	127	7F	DEL

# Représentation d'un réel en binaire

- Réel = nombre à virgule
- Ils sont utilisés pour les calculs numériques
- Les chiffres derrière la virgule correspondent aux puissances négatives
  - Pour rappel :  $a^{-b} = 1 / a^b$
- Dans la pratique, un nombre réel est représenté par un entier multiplié par une puissance de 2
  - Mantisse + exposant (base 2)
  - $101.1101 = 1011101 \times 2^{-4}$



# Octets = instruction

- **Le pointeur d'instruction pointe sur une case mémoire où une instruction est attendue**
- **L'octet qui se présente est interprété comme une instruction**
- **Le microcontrôleur comporte plusieurs registres (= cases mémoire de travail)**
- **Les instructions ...**
  - **peuvent copier des contenus de cases mémoire vers des registres**
  - **peuvent modifier les contenus de ces registres**
  - **peuvent copier les contenus des registres vers des cases mémoire**
- **Le pointeur d'instruction est augmenté de ce qui est prévu par l'instruction pour aller pointer sur l'instruction suivante**
- **Toutes ces opérations fonctionnent au rythme de plusieurs millions de tops par seconde**

Adresse n+9	10011111	
Adresse n+8	01010111	
Adresse n+7	11101010	
Adresse n+6	11100101	
Adresse n+5	11001011	
Adresse n+4	0000001	←
Adresse n+3	0000000	← Pointeur d'instruction
Adresse n+2	01001110	
Adresse n+1	11011101	
Adresse n	01111001	

# Exemple d'instruction



- Addition 4 bits de deux registres avec bit de retenue
- Le résultat est placé dans un registre destinataire

## 5. ADC – Add with Carry

### 5.1. Description

Adds two registers and the contents of the C Flag and places the result in the destination register Rd.

Operation:

(i)  $Rd \leftarrow Rd + Rr + C$

Syntax:

(i) ADC Rd,Rr

Operands:

$0 \leq d \leq 31, 0 \leq r \leq 31$

Program Counter:

$PC \leftarrow PC + 1$

16-bit Opcode:

0001	11rd	dddd	rrrr
------	------	------	------

# Mémoire limitée, donc ...

- **Programmation économe en occupation de mémoire**

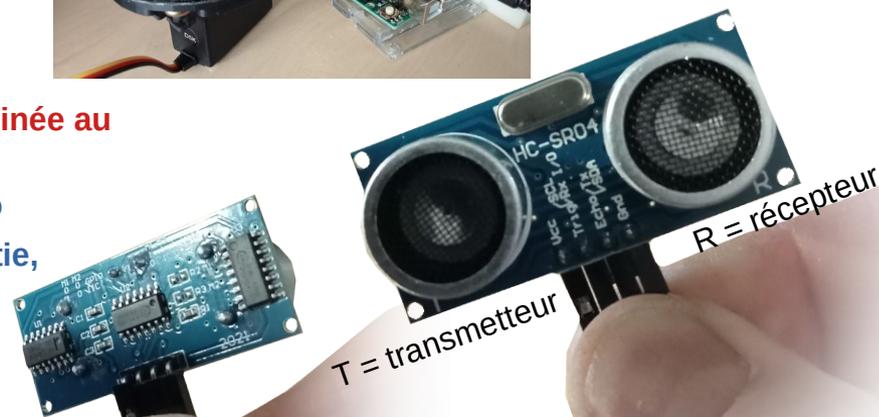
- Données (variables)
- Programme exécutable minimaliste (instructions)

- **Mise en place d'un contexte collaboratif**

- Délocaliser les traitements qui peuvent l'être

- **Exemple : radar ultrason**

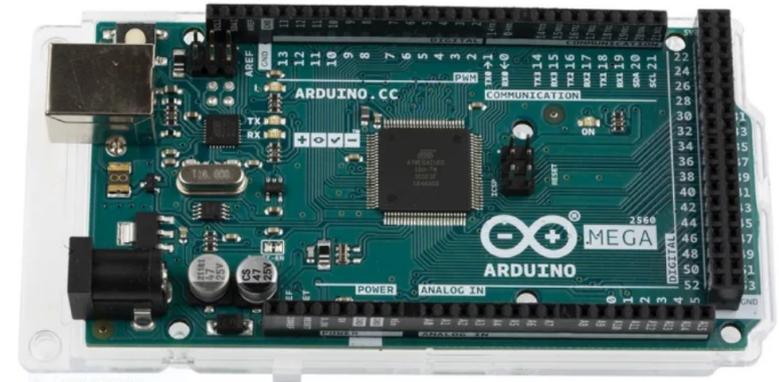
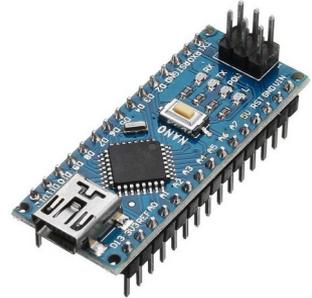
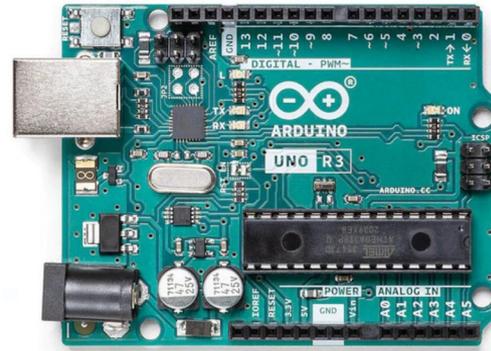
- Le programme du microcontrôleur active la sortie destinée au capteur pour initier une mesure de distance
- Le capteur lance un train d'ultrasons et attend un écho
- **Quand l'écho est reçu**, le capteur change d'état sa sortie, qui peut être lue par le microcontrôleur
- Le microcontrôleur n'a plus qu'à calculer le temps mis par le son pour faire l'aller-retour à 340 m/s
- **Commande du moteur** par le microcontrôleur pour voir s'il y a un obstacle dans une autre direction
- Et ça recommence ...
- **Commande moteur + capteur : typiquement une fonction qui peut être délocalisée pour ne récupérer que la direction et la distance de l'obstacle le plus proche**



Le transmetteur gueule tout ce qu'il peut  
Le récepteur tend l'oreille pour écouter s'il y a un écho

# Arduino : un monde magique

- **C'est pas cher**
- **C'est accessible aux jeunes et aux moins jeunes.**  
C'est mieux que de se droguer
- **C'est accessible aux débutants**
- **L'offre périphérique est riche (cartes microcontrôleur, capteurs...)**
- **La documentation est pléthorique et très bien faite**
  - Documentation Arduino
  - Documentations sur les périphériques compatibles
- **Je n'ai pas d'actions chez Arduino**



# Conclusion

- **Le micro contrôleur** gagne à être connu parce que c'est **intellectuellement satisfaisant** tout en n'étant **pas si compliqué**
  - C'est comme l'auberge espagnole, **toute l'intelligence qui est dedans, c'est vous qui l'y aurez mise**
- **Il permet de commander et faire bouger des choses de manière automatique**
- **Les capteurs, les relais, moteurs ... sont nombreux et peu chers**
  - Capteurs de température
  - Capteurs d'intensité lumineuse
  - Capteurs de présence
  - GPS
  - Capteurs d'humidité
  - Capteurs de proximité infrarouges, ultrasons, inductifs, capacitifs ...
  - Émetteurs – récepteurs radio
  - Moteurs pas à pas
  - Moteurs à courant continu
  - Moteurs brushless
  - Servos
  - Relais
  - Boutons poussoirs, à affleurement ...
- **Les bus et moyens de communication permettent de faire travailler tout ce petit monde ensemble**

