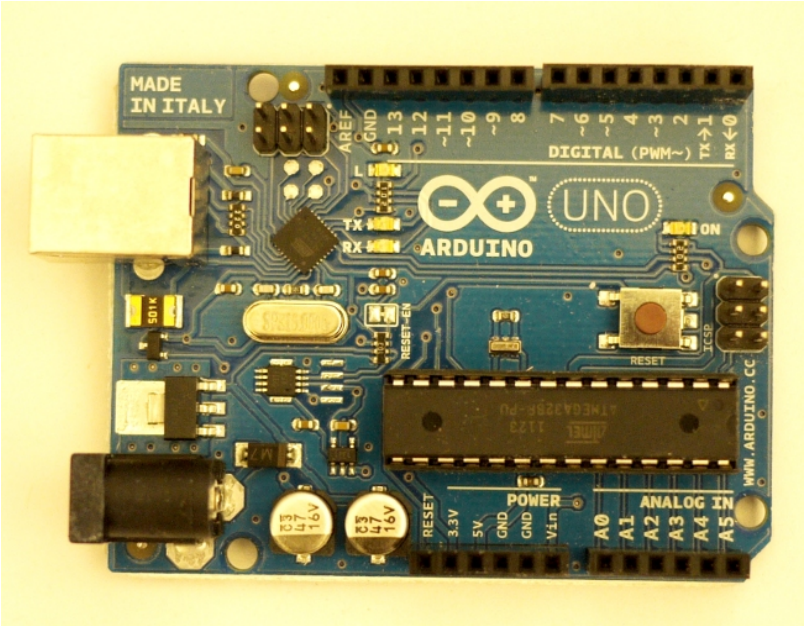
Retour aux sources avec Arduino

Rencontres Microtel-Multimedia, Noirmoutier

Samedi 25 Mars 2017

En une phrase ...

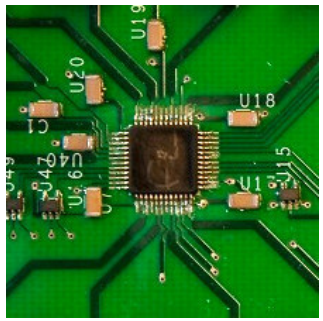


Arduino est une plate-forme de prototypage d'objets interactifs à usage créatif constituée d'une carte électronique et d'un environnement de programmation

Pourquoi cet atelier ? Pourquoi Arduino ?

L'informatique (dans le clubs) ne doit pas se réduire à cliquer dans des interfaces graphiques propriétaires et des pages web ...

- Revenir aux **bases de l'informatique** :
  - matériel (*hardware*)
  - programmation (*software*)
- Véritable **initiation** à l'informatique en tant que science et technique
- Interagir avec le **monde réél** : capteurs/actionneurs
- **S'amuser** en bricolant, bidouillage "*hacking*", ...



```c
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
   printf("Hello world!\n");
   return EXIT_SUCCESS;
}
```

Sommaire

1. Présentation de la plateforme
2. Interfaçage avec le monde réél
3. La programmation
4. Exemples simples de réalisations

Sommaire

1 Présentation de la plateforme

2 Interfaçage avec le monde réél

3 La programmation

4 Exemples simples de réalisations

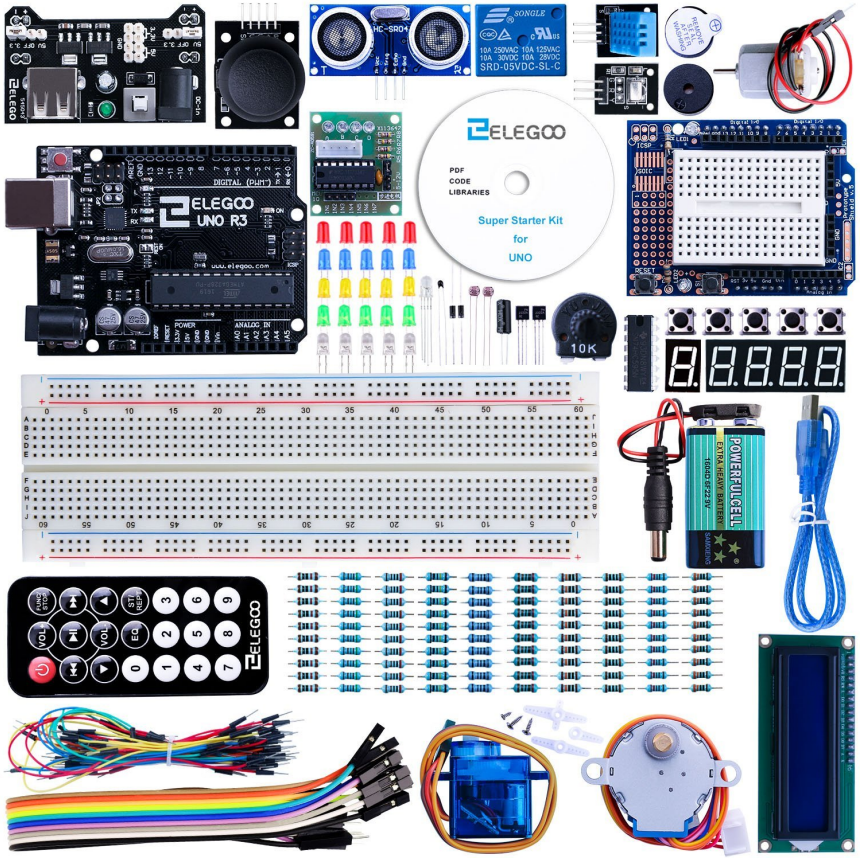
Origine et contexte

- Projet open source italien de carte à *microcontrôleur*
- Deux versions : Uno et Mega
- Programmation directe en mémoire Flash par port USB
- Environnement de développement intégré (IDE) convivial et multi-plateforme (Win/Mac/Linux), nombreuses bibliothèques
- Open source hardware : nombreux clones et dérivés, moins de 25 Euros
- Nombreux périphériques/extensions : "*shield*"

Applications

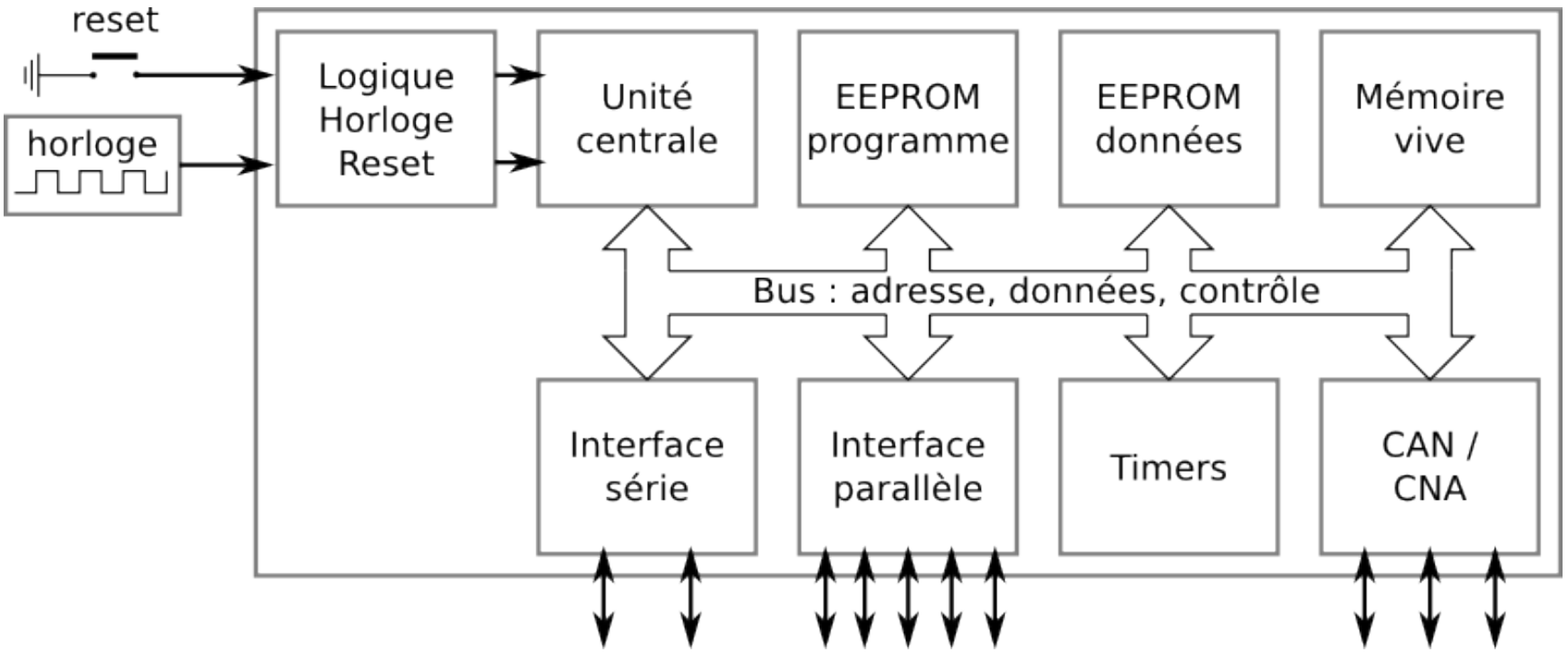
- Robotique, automatique (ex. pilotage imprimante 3D)
- Domotique (généralement via des relais)
- Instrumentation (ex. station météo)
- Enseignement (collèges, lycées, IUT, ...) : initiation programmation et électronique
- Objets connectés (Ethernet, WiFi), "*internet des objets*"
- Même des applis industrielles pouvant remplacer des API (PLC) !
- Interface "évoluée" entre un PC/Mac et son environnement (capteurs, actionneurs) au travers de l'USB

Bien débuter

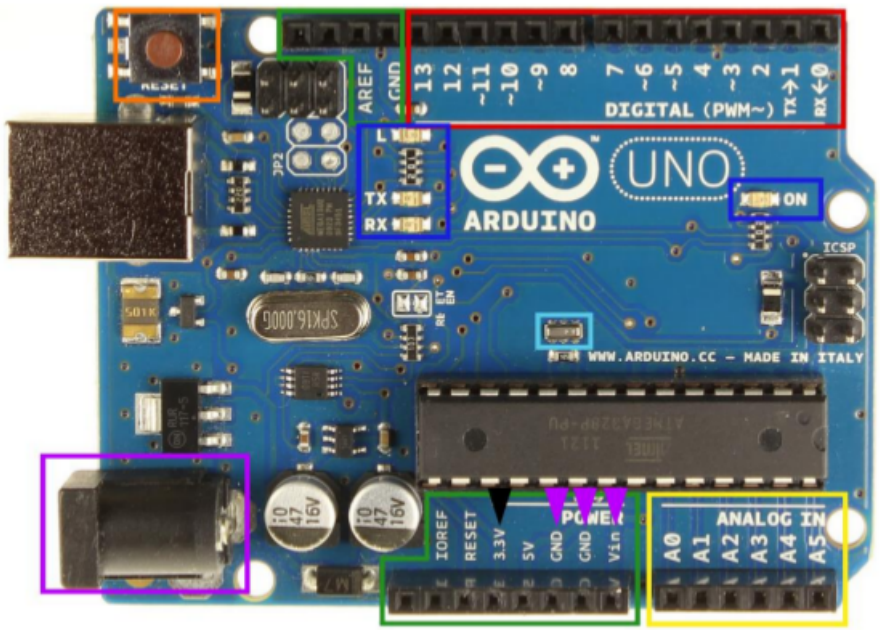


- Site officiel `www.arduino.cc`
- Commander un *kit de démarrage* sur Amazon, Selectronic, Conrad, Lextronic, Sparkfun, ... : de 30 à 100 Euros
- Livre de C. Tavernier, *Arduino, Maîtrisez sa progammation et ses cartes d'interface*, Dunod, 20 Euros
- MOOC Arduino sur FUN

Qu'est ce qu'un microcontrôleur ?

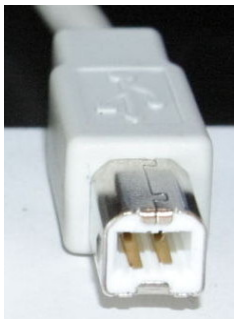


L'Arduino Uno (R3)



- Microcontrôleur 8 bits Atmel ATmega328P
- Horloge 16 MHz
- 32 ko mém. Flash, 2 ko RAM, 1 ko EEPROM
- Bus GPIO 15 E/S dont 6 sorties PWM
- 6 CAN 10 bits
- Port USB, ICSP, série i2c
- Régulateur tension 5 V et 3,3 V

L'alimentation



- USB 5 V : relié à PC/Mac ou chargeur USB
- Jack 2,1 mm ou broches Vin/GND :
  - entrée non-régulée 7 V mini à 12 V maxi
  - piles, batteries ou bloc transformateur-redresseur
- Ne jamais tenter de rétro-alimenter l'Arduino par la broche 5 V
- Broches 5 V et 3,3 V régulés dispos pour alimenter les périphériques, 500 mA maxi
- Sensible aux décharges d'éléctricité statique
  Prévoir un boîtier de protection (8 Euros)

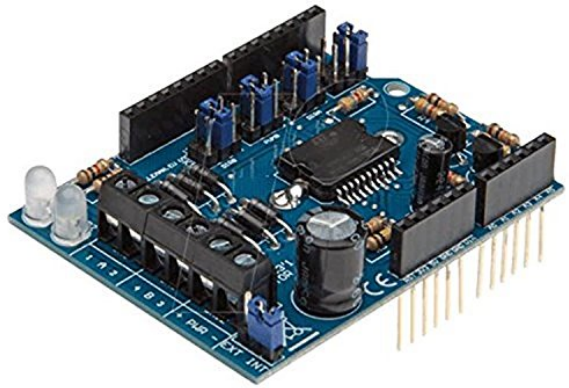
Ce qu'il n'y a pas sur l'Arduino

- Pas d'IHM du type clavier/souris/écran
  → shield LCD keypad
- Seulement 2 ko de RAM, il faut optimiser l'utilisation mémoire
- Pas d'OS, seulement un *bootloader* de 0,5 ko en Flash
- Pas de mémoire de masse
  → shield lecteur carte microSD
- Pas d'horloge temps réél
  → shield RTC
- FPU en simple précision seulement, pas de double

Les shields ou cartes d'extension empilables

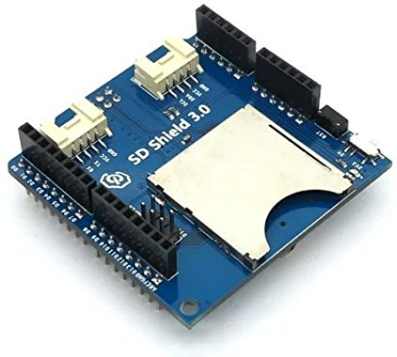
LCD keypad

Commande moteurs

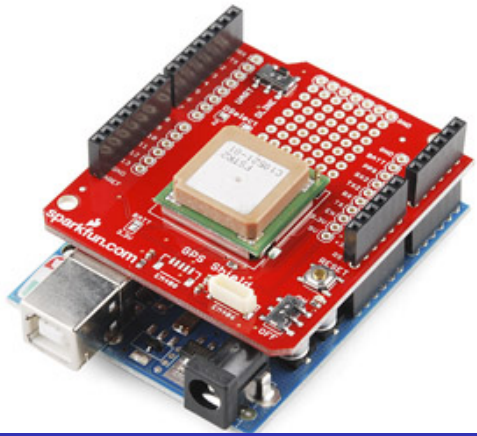


Ethernet

Lecteur SD



GPS



Sommaire

1 Présentation de la plateforme

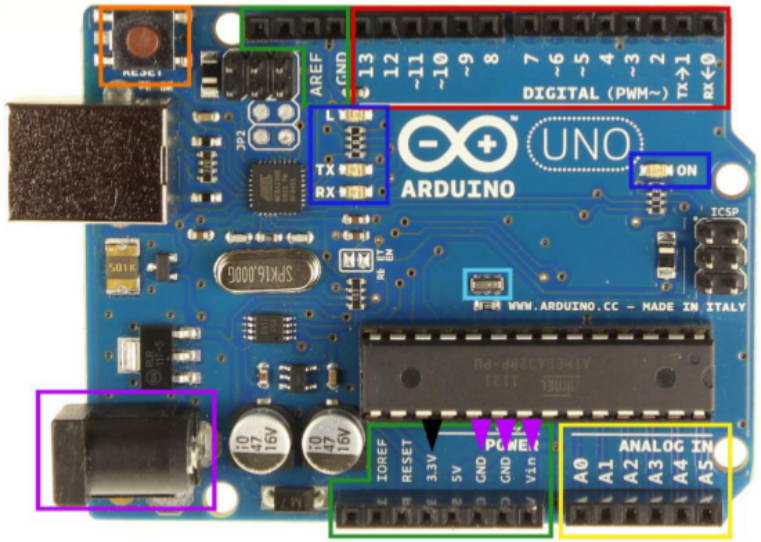
2 Interfaçage avec le monde réél

3 La programmation

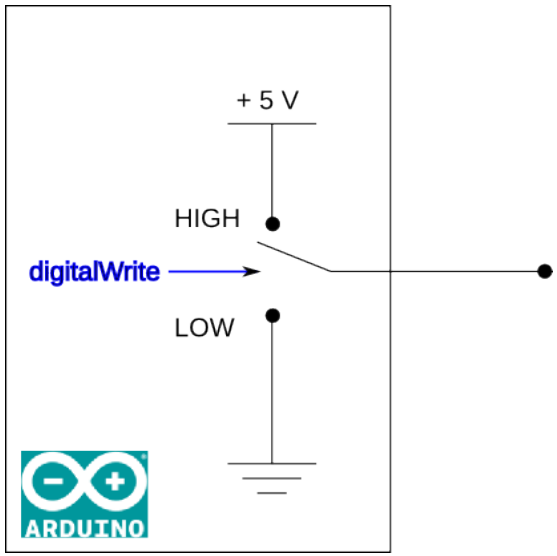
4 Exemples simples de réalisations

Les entrées-sorties : s'interfacer avec le monde réél

nombre d'E/S	numériques	analogiques
entrées	12	6
sorties	12	6



Sorties numériques



- 12 sorties numériques TTL (n = 2 à 13)
- initialisation :

```
pinMode(n, OUTPUT)
```

- sortie à 5 V :

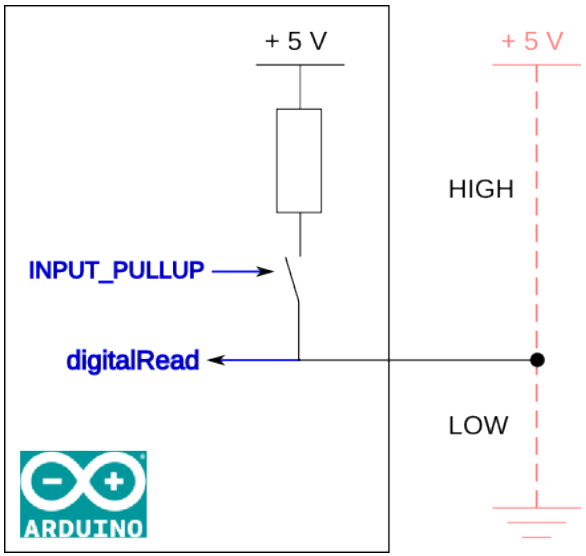
```
digitalWrite(n, HIGH)
```

- sortie à 0 V :

```
digitalWrite(n, LOW)
```

- 40 mA maxi par broche, 200 mA au total

Entrées numériques



- 12 entrées numériques TTL (n = 2 à 13)
- initialisation :

  `pinMode(n, INPUT)`

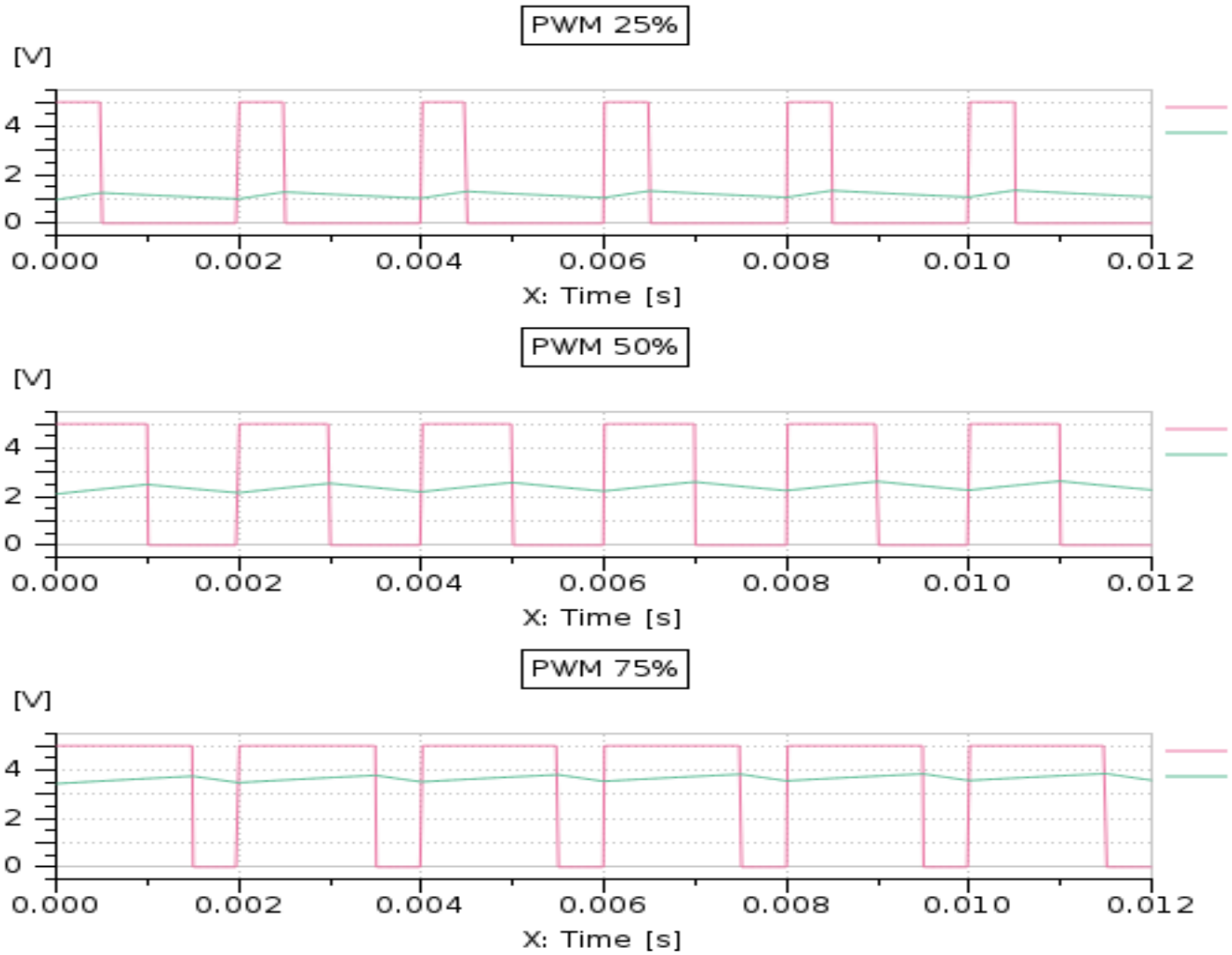
  ou

  `pinMode(n, INPUT_PULLUP)`
- si entrée < 2 V :

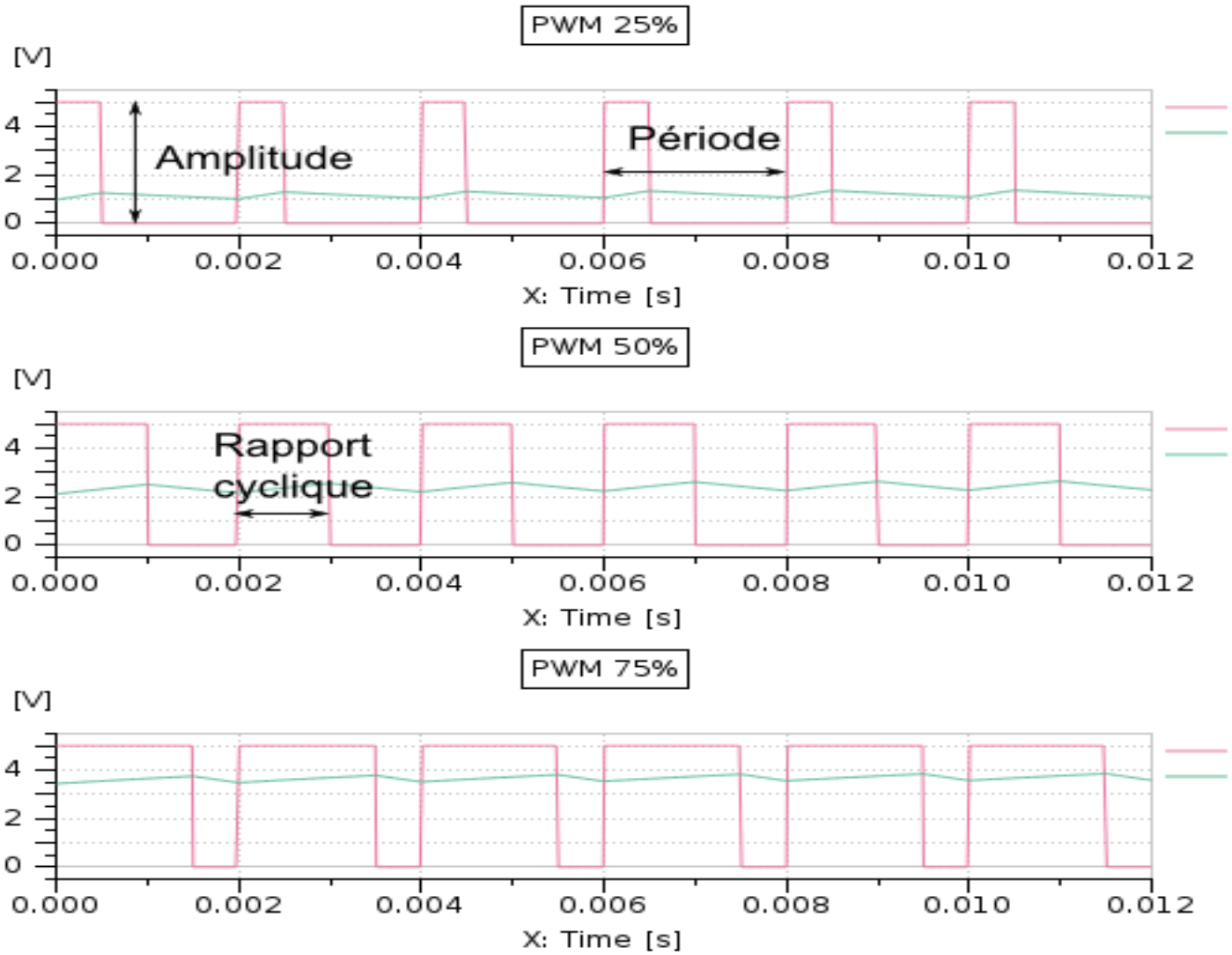
  `digitalRead(n)==LOW`
- si entrée > 3 V :

  `digitalRead(n)==HIGH`

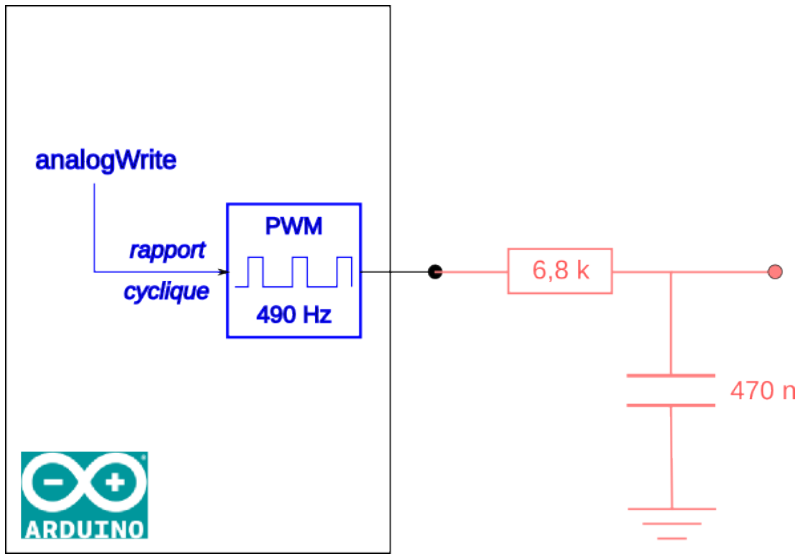
Modulation de largeur d'impulsion (PWM)



Modulation de largeur d'impulsion (PWM)

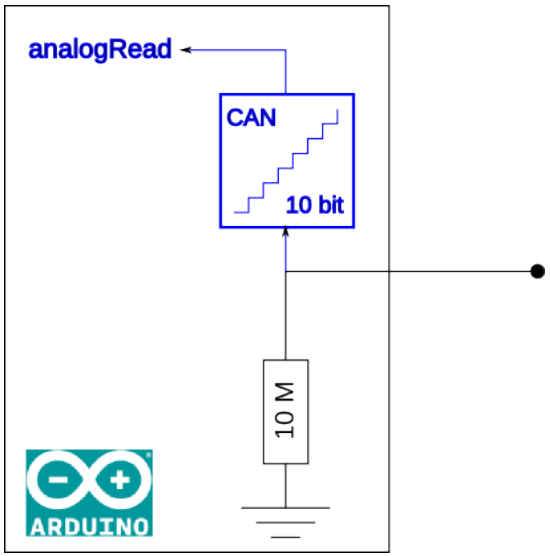


Sorties analogiques



- 6 sorties PWM
  (n = 3, 5, 6, 9, 10, 11)
- amplitude 5 V
- fréquence 490 Hz $\Rightarrow$ période $\simeq$ 2 ms
- initialisation :
  `pinMode(n, OUTPUT)`
- définition du rapport cyclique :
  `analogWrite(n, cr)`
  $0 \leq cr \leq 255$
  ($0 \leftrightarrow 0\,\%$ et $255 \leftrightarrow 100\,\%$)
- ajouter un filtre passe-bas RC,
  fréquence de coupure à 50 Hz

Entrées analogiques



- 6 entrées analogiques (n = A0 à A5)
- CAN 10 bits, 10 kHz maxi
- lecture :

  `analogRead(n)`

  valeur entre 0 et 1023
- impédance d'entrée $\geq$ 10 MΩ

Sommaire

1 Présentation de la plateforme

2 Interfaçage avec le monde réél

3 La programmation

4 Exemples simples de réalisations

Programmation Arduino

- Environnement de développement intégré (IDE) *Arduino*, disponible pour Windows / Mac / Linux
- Langage *Wiring* : sous-ensemble du C++ avec des bibliothèques spécifiques pour l'accès au matériel
- Motif (*pattern*) de programmation unique : *setup* et *loop*
- Compilation croisée (*cross compilation*) sur le PC/Mac, puis téléchargement du binaire AVR par le port USB
- Le binaire une fois chargé, l'Arduino redémarre et exécute le programme
- La liaison USB reste disponible pour communiquer avec le PC/Mac, si nécessaire

L'IDE Arduino

led_blink_dec12b | Arduino 1.6.13

File Edit Sketch Tools Help

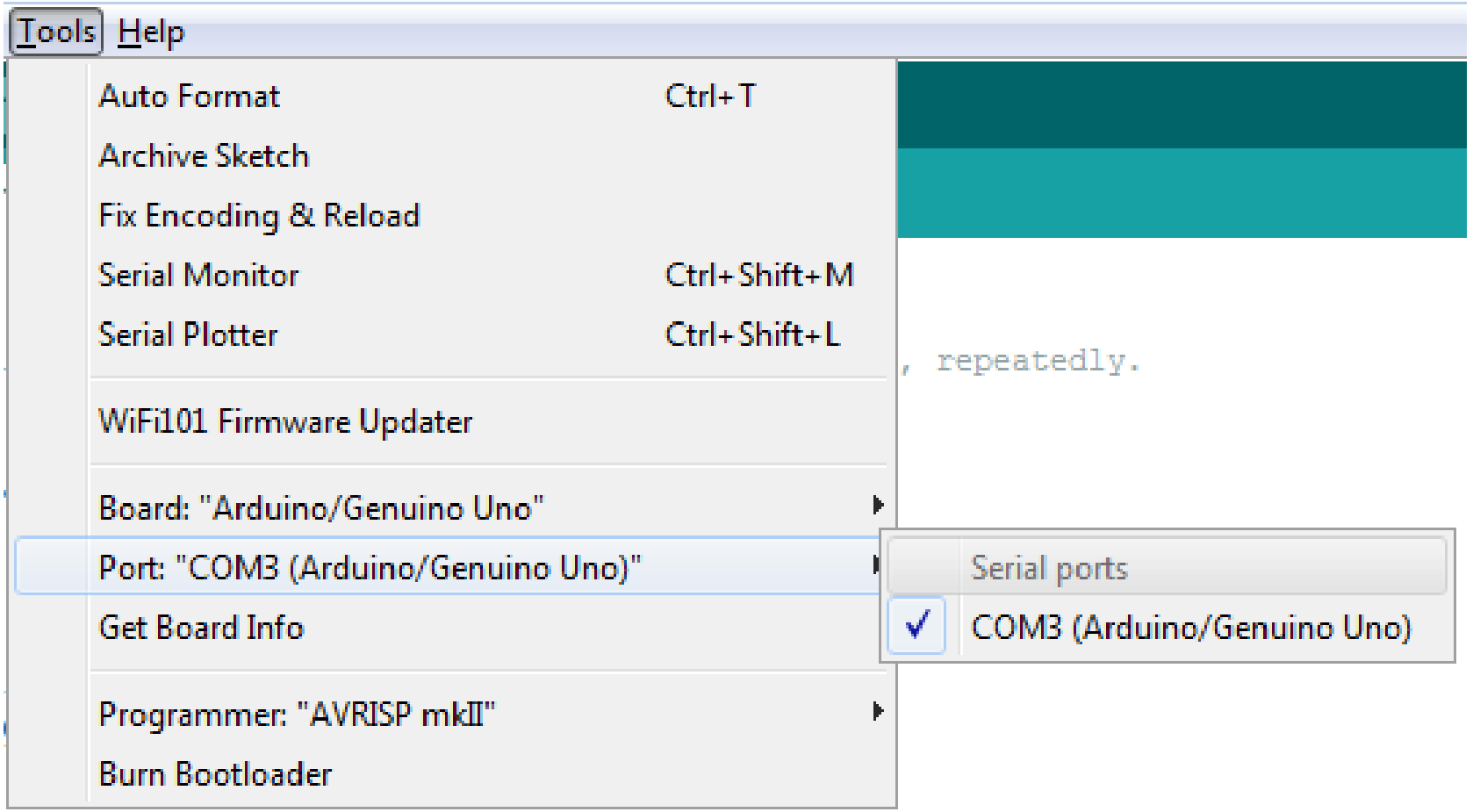
led_blink_dec12b

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.
*/

#define LED_PIN 13

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_PIN as an output.
  pinMode(LED_PIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_PIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_PIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Done Saving.

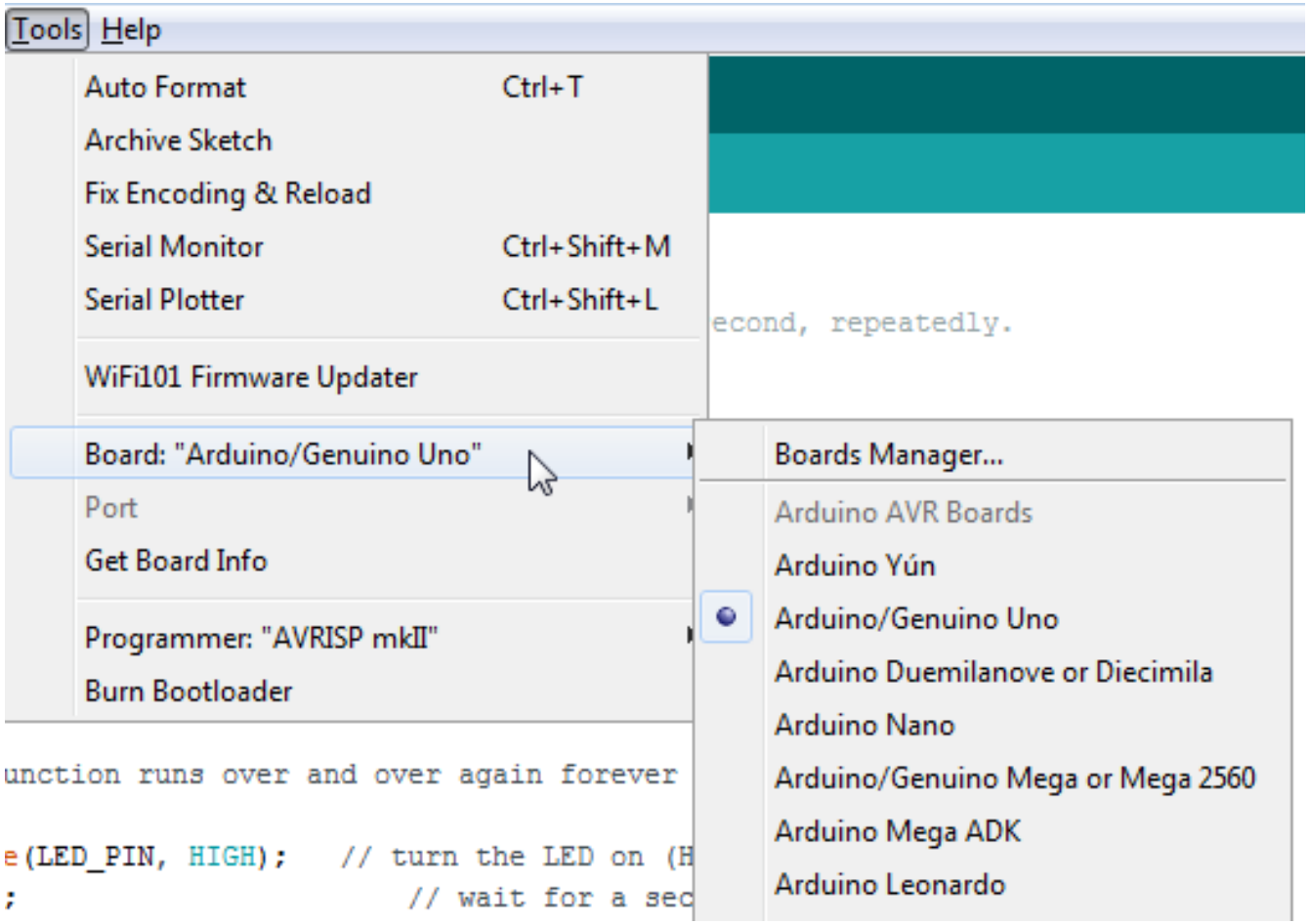
Configuration de l'IDE (1/3)

- Choisir le port



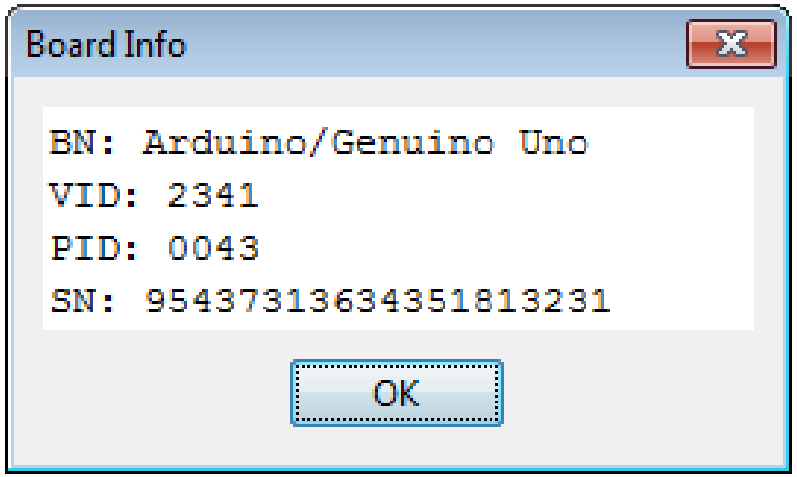
Configuration de l'IDE (2/3)

- Choisir le modèle de carte



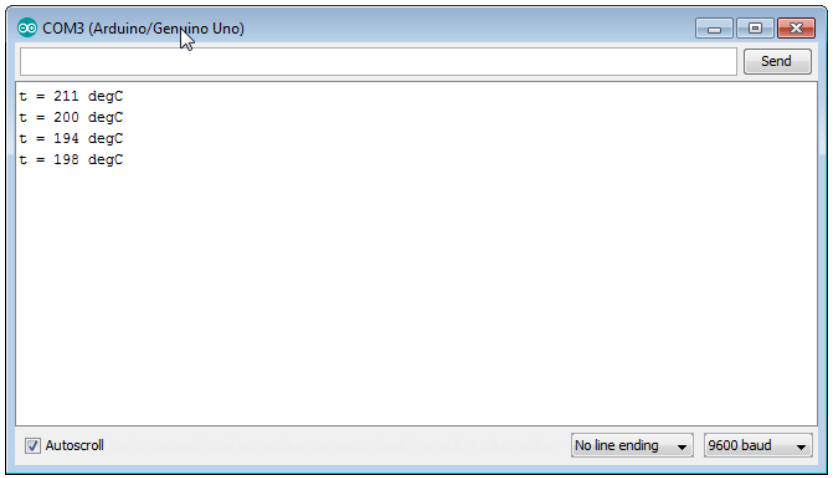
Configuration de l'IDE (3/3)

- Vérifier la communication sur le port USB
- *Tools* → *Get Board Info*



Le moniteur série

- L'Arduino peut communiquer avec le PC/Mac au travers du port USB
- Dans l'IDE, *Tools* → *Serial Monitor*



- Utilisation de la bibliothèque `Serial`

Design pattern unique

- Deux fonctions à définir :

```
setup() {
  ...
}
loop() {
  ...
}
```

- `setup()` est exécuté une seule fois après *reset*, pour initialiser les variables et les E/S
- `loop()` est exécuté en boucle infini, jusqu'au prochain *reset*

Sommaire

1 Présentation de la plateforme

2 Interfaçage avec le monde réél

3 La programmation

4 Exemples simples de réalisations

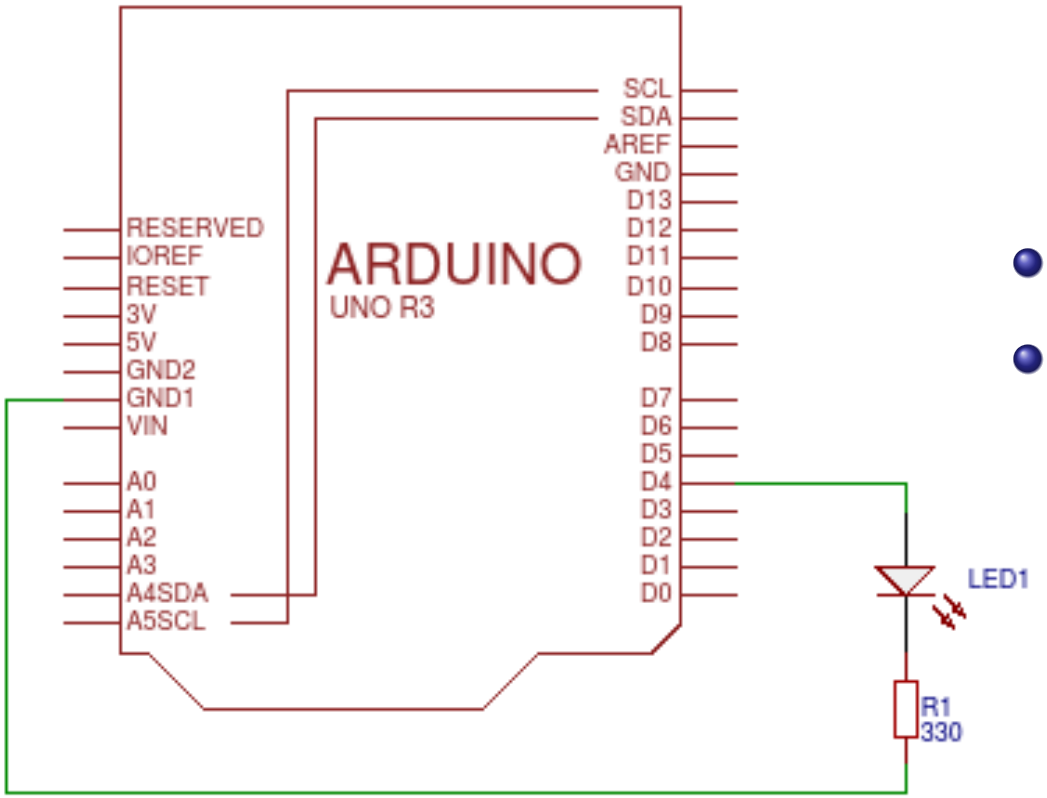
Le «hello world» des micro-contrôleurs

Faire clignoter une LED (connectée à la sortie numérique 13) :

```
setup() {
    pinMode(13, OUTPUT);
}

loop() {
    digitalWrite(13, HIGH);
    delay(1000);
    digitalWrite(13, LOW);
    delay(1000);
}
```

Clignotement de LED : hardware

Idem avec une LED sur la sortie numérique 4 :



- Tensions sortie Arduino et seuil LED :

$$U_A = 5\,\mathrm{V} \qquad U_{LED} = 1,7\,\mathrm{V}$$

- Courant maxi : $I = 10\,\mathrm{mA}$
- Résistance protection LED :

$$\begin{aligned} R_1 &= \frac{U_A - U_{LED}}{I} \\ &= \frac{5 - 1,7}{0,01} \\ &= 330\,\Omega \end{aligned}$$

Clignotement de LED : software

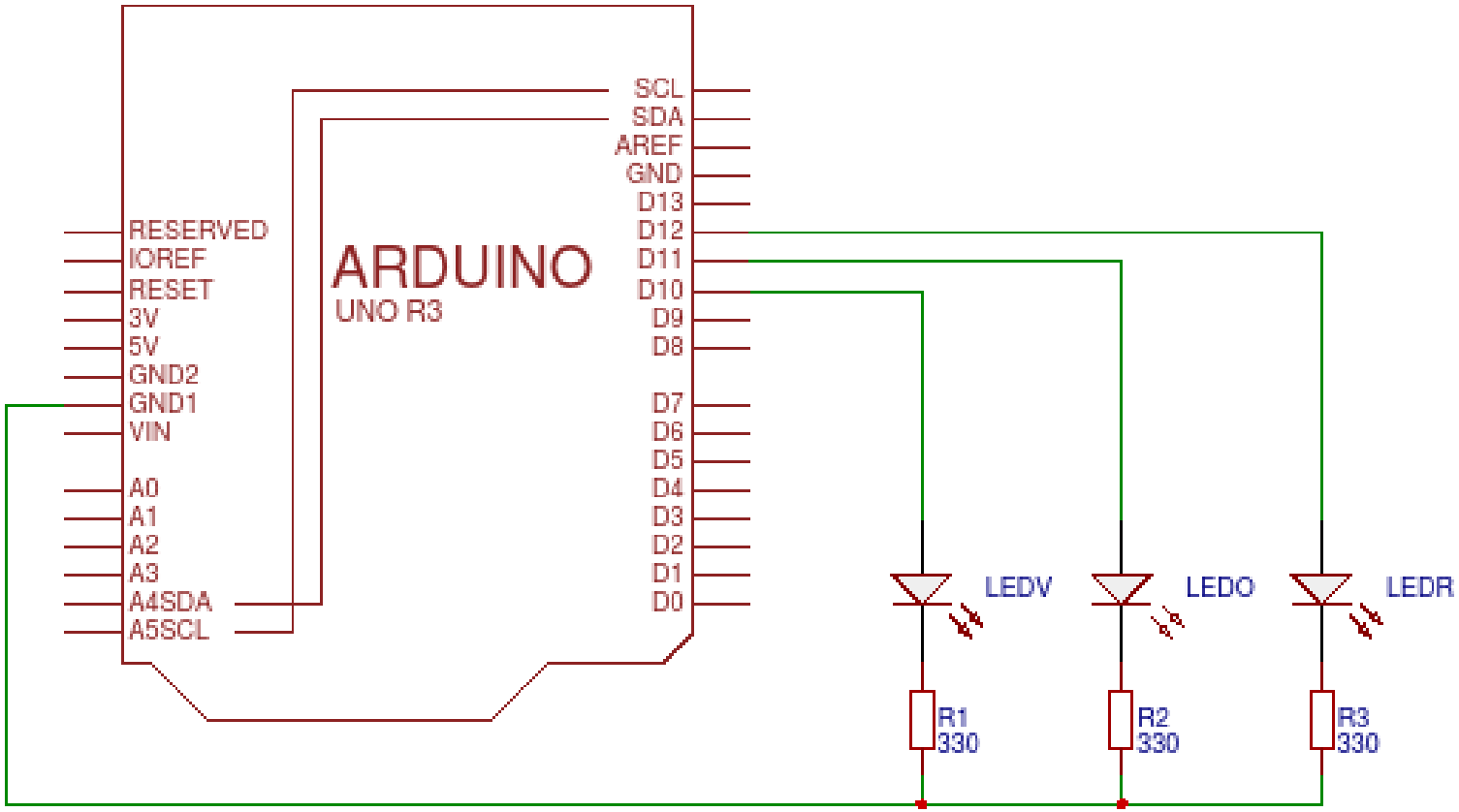
Idem avec une LED sur la sortie numérique 4 :

```
#define LED_NUM 4
setup() {
    pinMode(LED_NUM, OUTPUT);
}
loop() {
    digitalWrite(LED_NUM, HIGH);
    delay(1000);
    digitalWrite(LED_NUM, LOW);
    delay(1000);
}
```

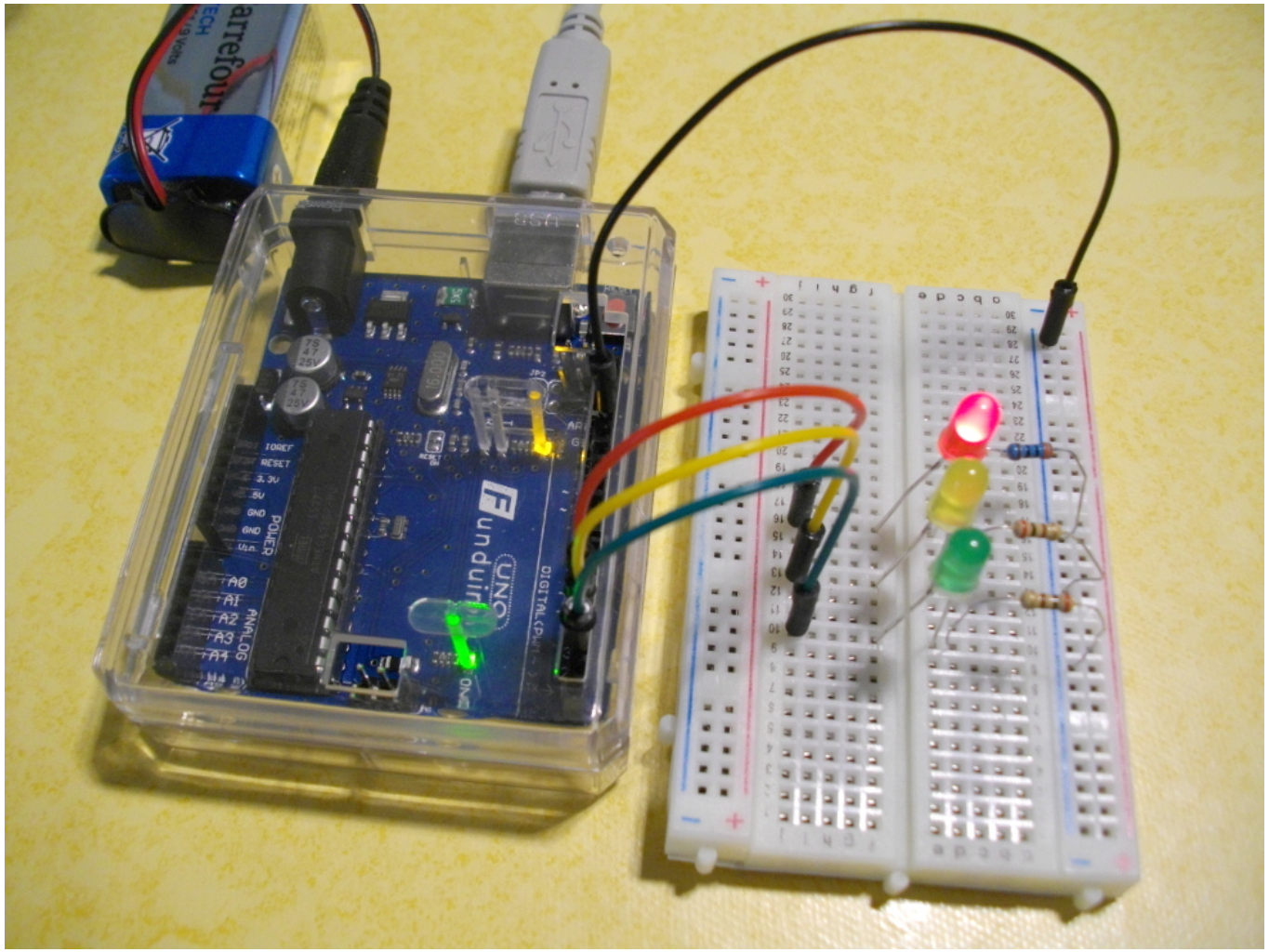
Feux tricolores : cahier des charges

- Une LED orange (broche 11) allumée pendant 1 seconde (c'est la première LED à s'allumer)
- Puis une LED rouge (broche 12) allumée pendant 3 secondes
- Enfin, une LED verte (broche 10) allumée pendant 3 secondes
- Et on recommence

Feux tricolores : hardware



Feux tricolores : montage



Feux tricolores : software (1/2)

```
#define LED_VERTE  10
#define LED_ORANGE 11
#define LED_ROUGE  12
setup () {
    pinMode(LED_VERTE,  OUTPUT);
    pinMode(LED_ORANGE, OUTPUT);
    pinMode(LED_ROUGE,  OUTPUT);
    digitalWrite(LED_ROUGE,  LOW);
}
```

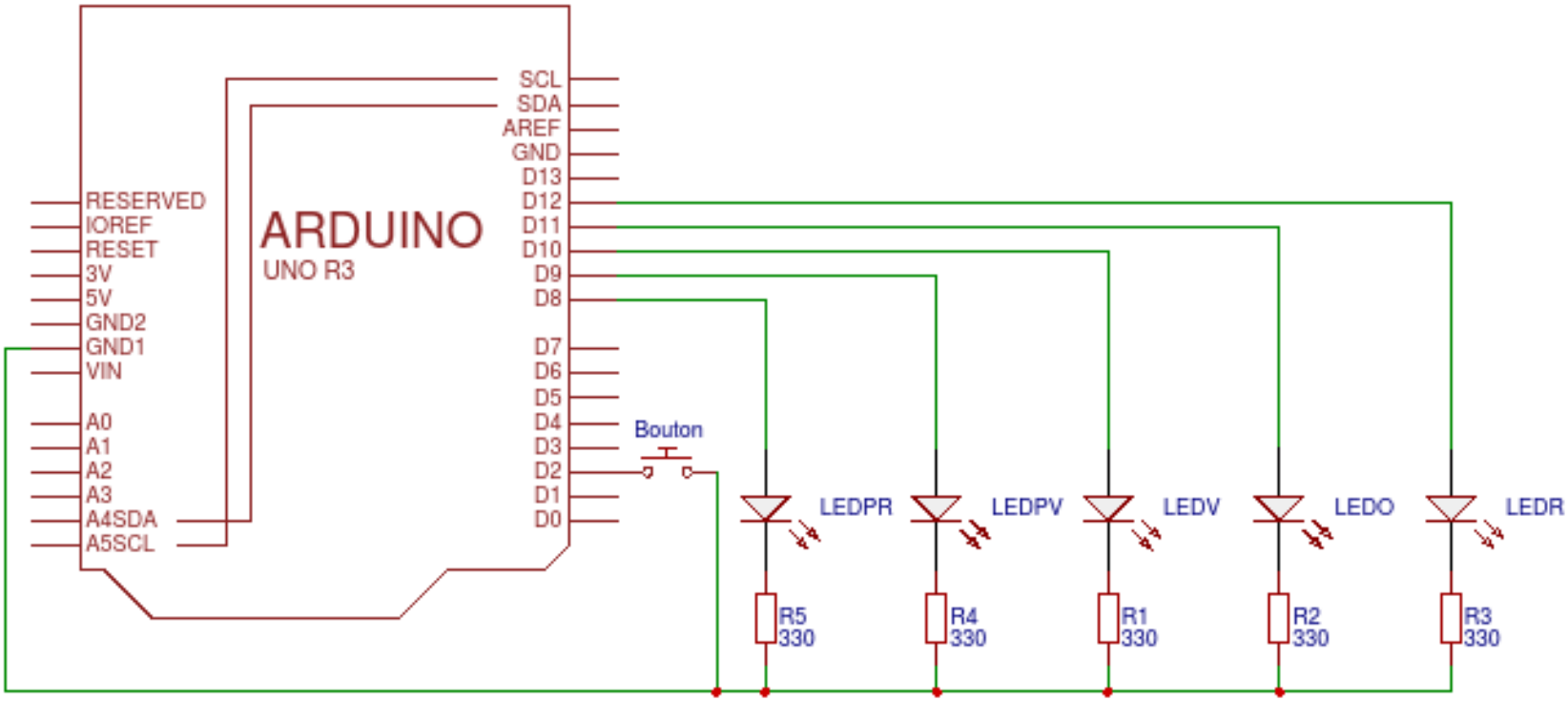
Feux tricolores : software (2/2)

```
loop () {
    digitalWrite(LED_VERTE,   LOW);
    digitalWrite(LED_ORANGE,  HIGH);
    delay(1000);
    digitalWrite(LED_ORANGE,  LOW);
    digitalWrite(LED_ROUGE,   HIGH);
    delay(3000);
    digitalWrite(LED_VERTE,   HIGH);
    digitalWrite(LED_ROUGE,   LOW);
    delay(3000);
}
```

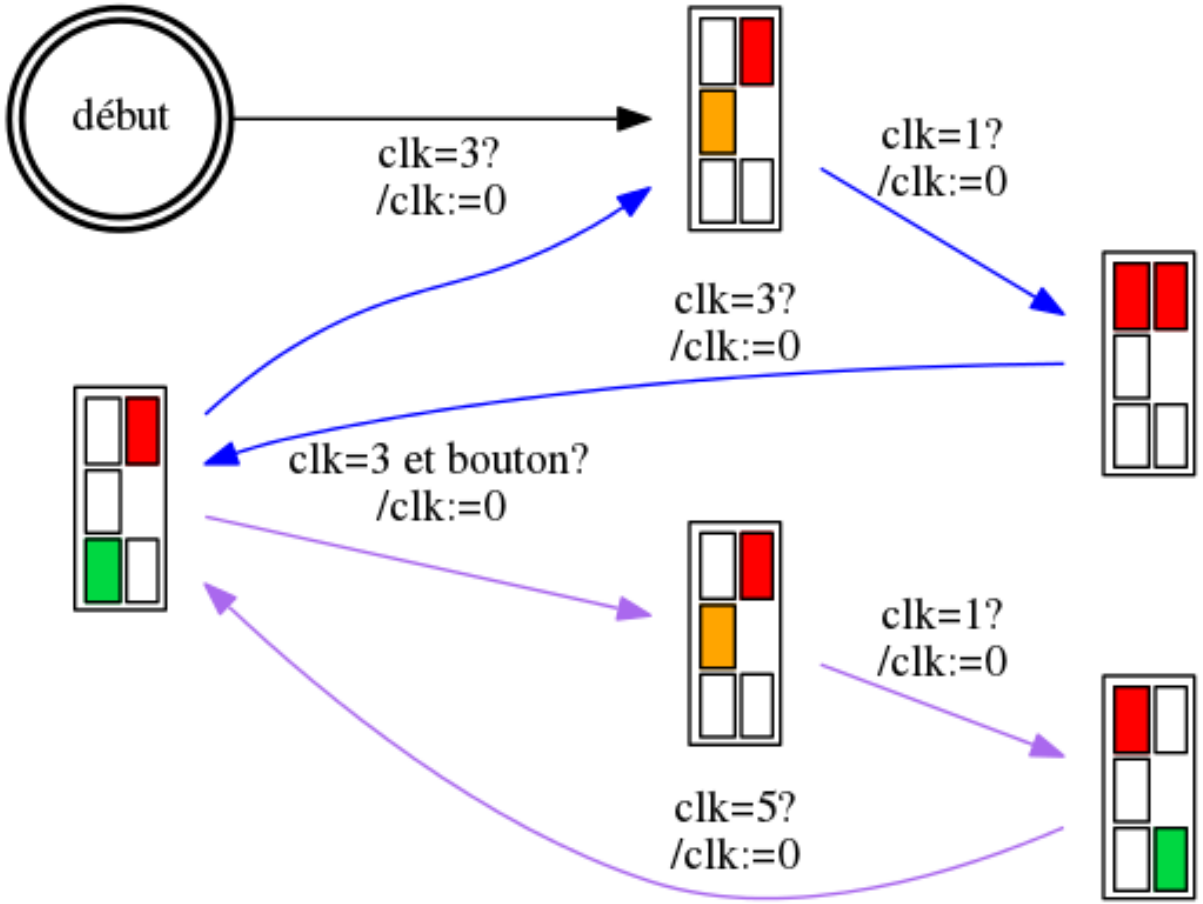
Feux voitures et piétons : cahier des charges

- Feu tricolore comme précédemment :
  orange 1 s $\rightarrow$ rouge 3 s $\rightarrow$ vert 3 s
- Le feu piéton est constamment rouge (rouge broche 9 et vert broche 8) tant que le bouton (broche 2) n'est pas appuyé
- Si le bouton est maintenu appuyé pendant que le feu voiture est au vert, alors le feu tricolore passe au orange allumé pendant 1 seconde puis rouge pendant 5 secondes durant lesquelles le feu vert piéton passe au vert
- Après ces 5 secondes, le feu piéton repasse au rouge et le fonctionnement normal reprend

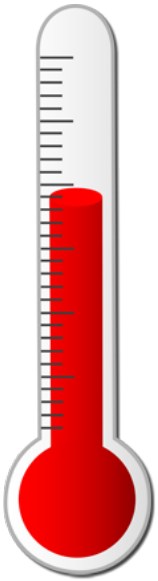
Feux voitures et piétons : hardware



Feux voitures et piétons : automate temporisé

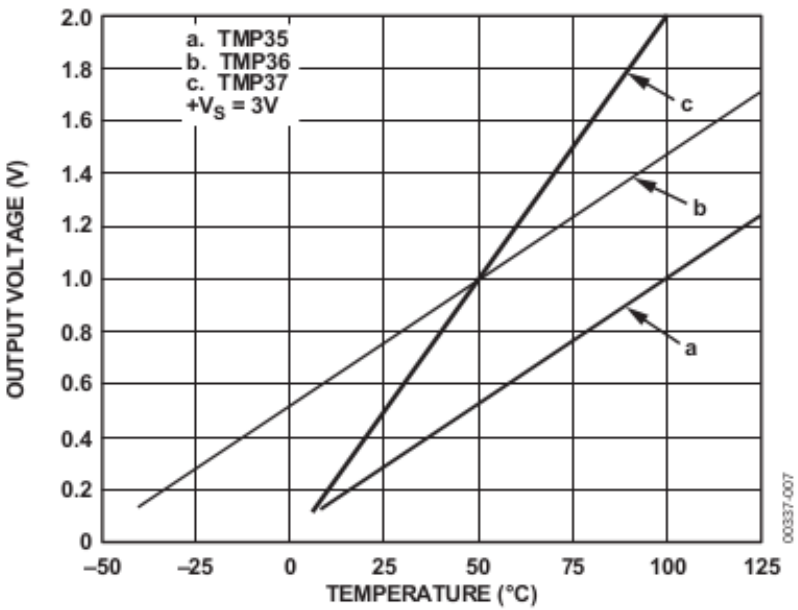


Mesure de température : cahier des charges



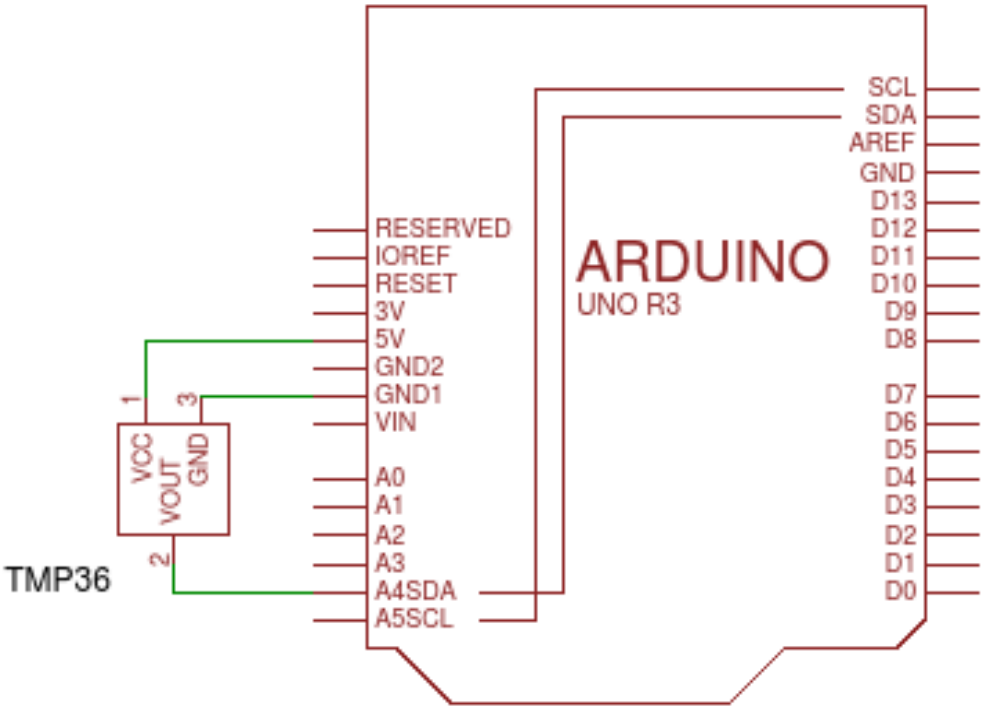
- Mesure température ambiante, toutes les 3 s
- Envoi de la valeur mesurée au PC/Mac par le port USB
- Lecture sur le moniteur série

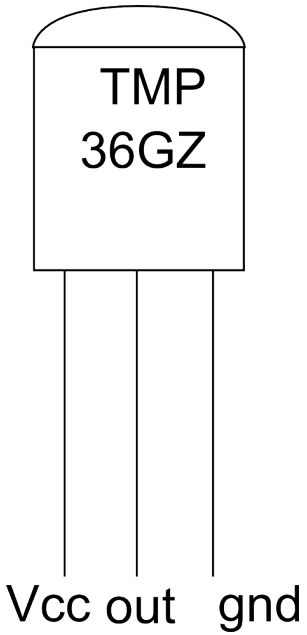
Mesure de température : instrumentation



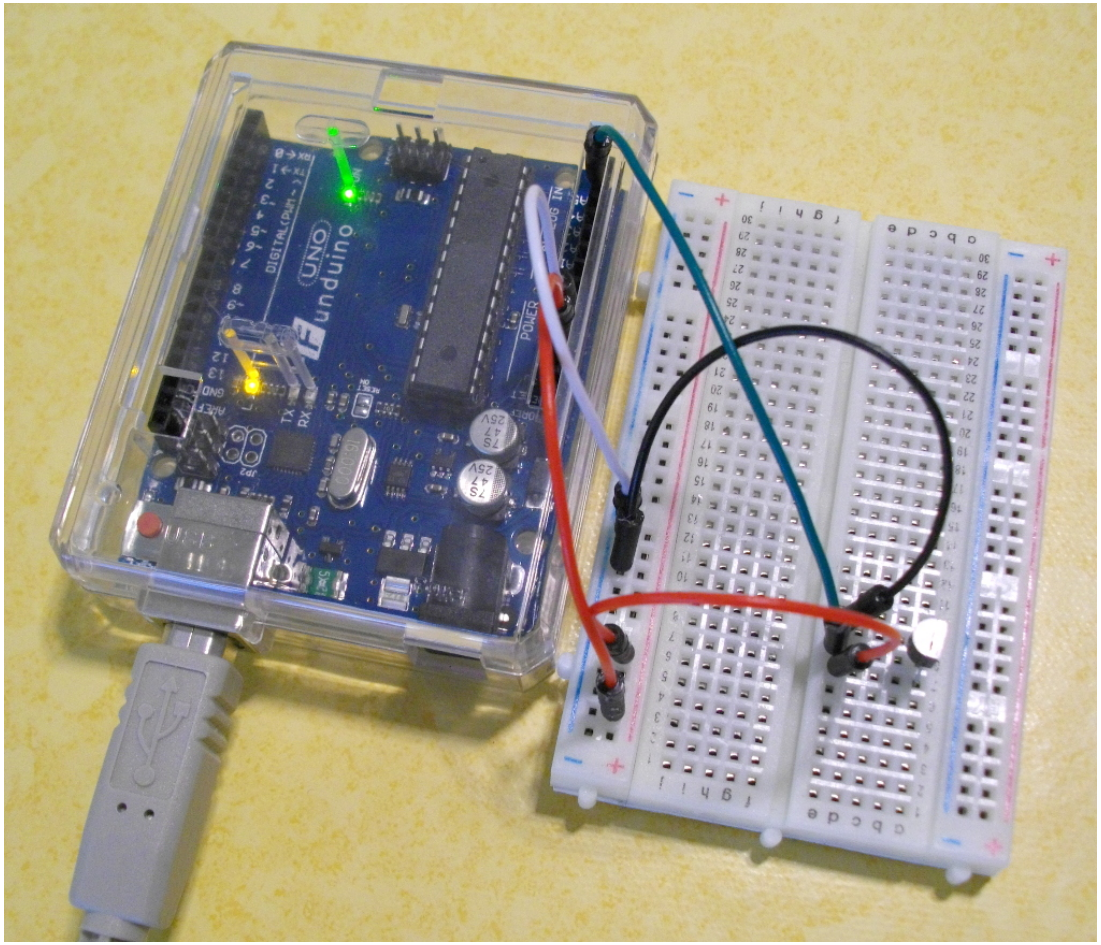
- On utilise un capteur de température à état solide TMP36GZ fabriqué par Analog Devices
- Coût : $\leq$ 2 Euros
- Gain : 10 mV/°C
- Offset : 500 mV $\leftrightarrow$ 0 °C
- Précision : environ 1 °C

Mesure de température : hardware





Mesure de température : montage



Mesure de température : software

```
setup() {
    Serial.begin(9600);
}
loop() {
    float tension = 5000./1023.*analogRead(A4),
          temp = (tension-500.) / 10.;
    Serial.println(temp);
    delay(3000);
}
```

Références I

Eskimon

Tutoriels Arduino

http://eskimon.fr/category/arduino

FUN MOOC Mines Telecom

Programmer un objet avec Arduino

http://www.fun-mooc.fr

EasyEDA

An Easier EDA Experience

http://easyeda.com

Références II

C. Tavernier

Arduino, Maîtrisez sa programmation et ses cartes d'interface

Dunod, 2ème éd., 2014

Collectif FLOSS

Arduino

http://www.flossmanualsfr.net/arduino