Kit de base pour UNO GT012

code 35110

Kit de base pour débutant livré avec une carte GoTronic Uno R3, une plaque de montage rapide et le nécessaire au prototypage.

Liste des chapitres

Présentation et installation du logiciel

Liste des composants livrés :

Merci de vérifier l'ensemble des composants livrés.

L'ensemble est livré dans une boite de rangement :

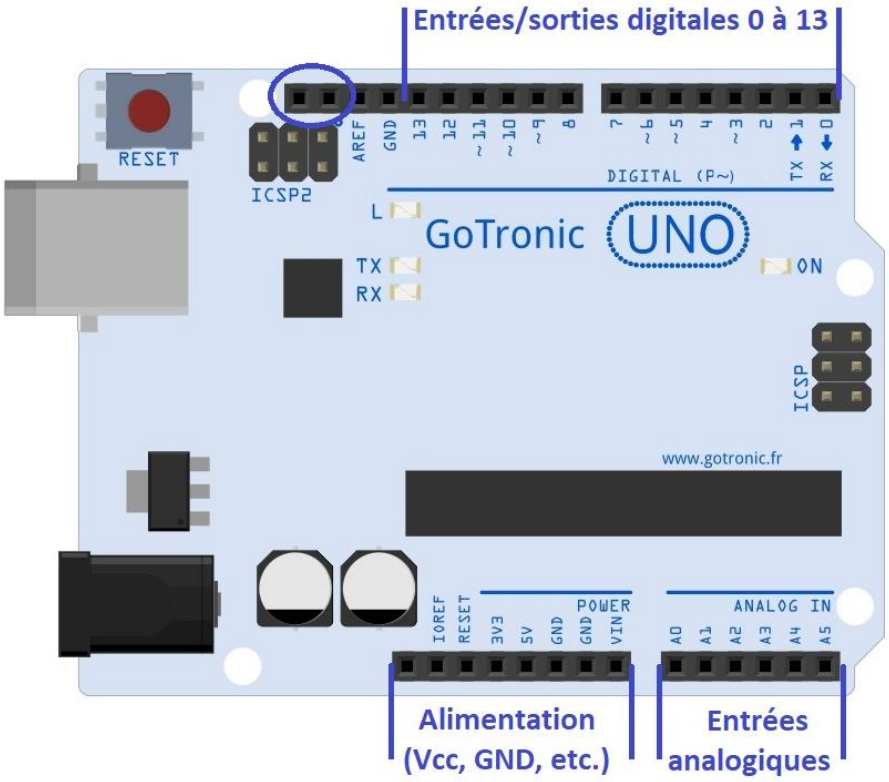
- 1 carte GoTronic UNO R3 avec câble USB
- 1 plaque d'essais 830 contacts
- 1 pack de 30 jumpers M/M
- 1 pack de 20 jumpers M/F
- 15 leds de différentes couleurs
- 1 module led RGB
- 8 résistances 220 Ω
- 5 résistances 1 kΩ
- 5 résistances 10 kΩ
- 4 boutons-poussoirs miniatures
- 1 buzzer actif
- 1 buzzer passif
- 2 afficheurs 7 segments
- 1 afficheur 4 digits
- 1 registre à décalage 74HC595
- 1 capteur de température LM35DZ
- 1 potentiomètre 50 kΩ
- 3 photodiodes LDR
- 1 récepteur IR
- 1 télécommande IR
- 1 détecteur de flamme
- 2 interrupteurs à bille
- 1 matrice 8x8 leds
- 1 coupleur 6 piles AA (piles non incluses)

Carte UNO :

La carte UNO est une carte à microcontrôleur programmable via USB.

Elle se programme via le logiciel IDE téléchargeable gratuitement.

Elle dispose de 14 broches d'entrées/sorties digitales et de 6 entrées analogiques.



Les entrées/sorties digitales :

- une entrée digitale permet le raccordement de différents capteurs. Elle ne peut prendre que deux valeurs, le capteur est soit à l'état bas ''0'' ou soit à l'état haut ''1''. Par exemple porte ouverte ou porte fermée, elle ne peut pas être en position intermédiaire.

- une sortie digitale permet le raccordement de modules en sortie. Elle peut prendre uniquement deux états ''0'' ou ''1''. Pour le cas d'une carte Uno ''0'' = 0V et ''1'' = 5 Vcc. Elle permet par exemple d'allumer une led à l'état ''1'' et de l'éteindre à l'état ''0''.

- chaque entrée/sortie de la carte Uno peut être configurée en entrée ou en sortie mais pas les deux en même temps.

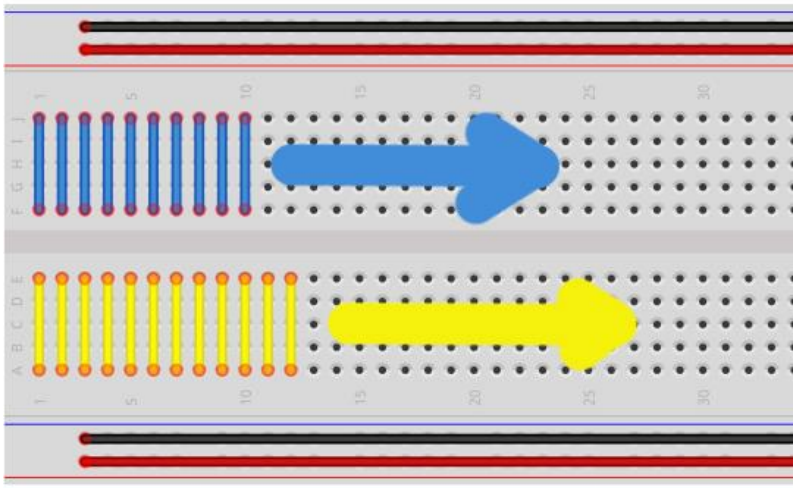
Les entrées analogiques :

A l'inverse d'une entrée digitale, une entrée analogique peut prendre une valeur comprise entre 0 et 1023, 0V correspond à la valeur 0 et 5 V correspond à la valeur 1023. Elles permettent donc le raccordement de capteurs analogiques : capteur de lumière, potentiomètre, etc.

Les autres raccordements :

La carte Uno possède également plusieurs connexions série, UART, I2C, etc. Elles permettent de communiquer en envoyant des données sur une seule broche ce qui permet de transmettre plusieurs informations avec une seule broche.

Plaque d'essais :

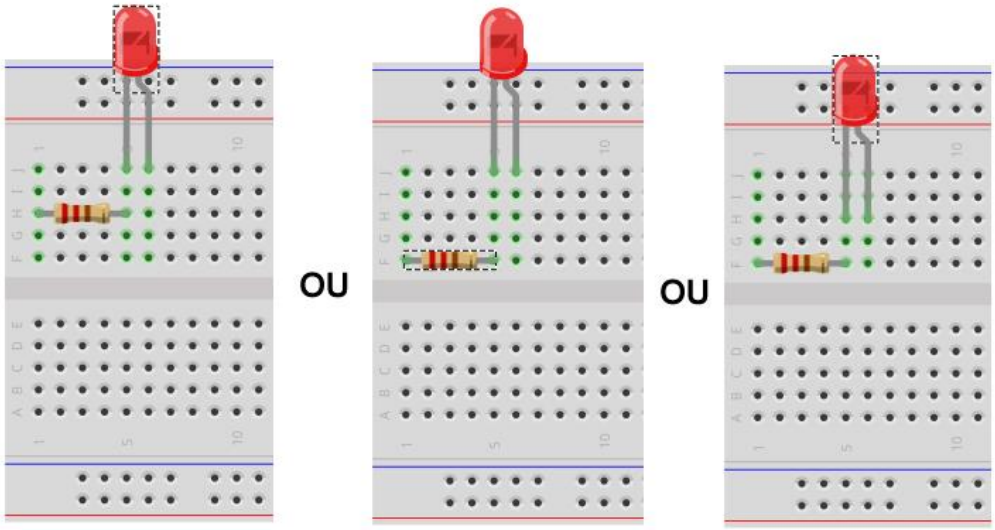


La plaque d'essais est raccordée **en interne** comme les fils bleus et jaunes sur toute la largeur.

Elle est également raccordée en interne sur toute la longueur comme les fils rouges et noirs.

Attention : sur les raccordements rouges et noirs, il y a une coupure au milieu. Il suffit de mettre un fil pour continuer la liaison sur toute la longueur.

Si vous souhaitez raccorder la broche d'une résistance sur la broche d'une led par exemple vous pouvez le faire de différentes manières :

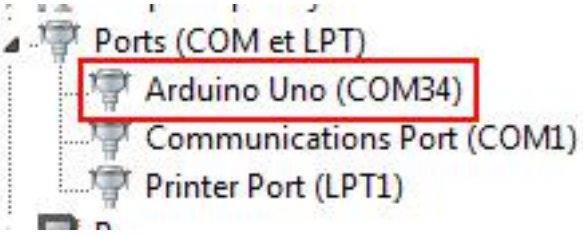


Quelle que soit la position de la résistance et/ou de la led, le résultat est identique. On voit une ligne verticale verte correspondant à toutes les liaisons connectées.

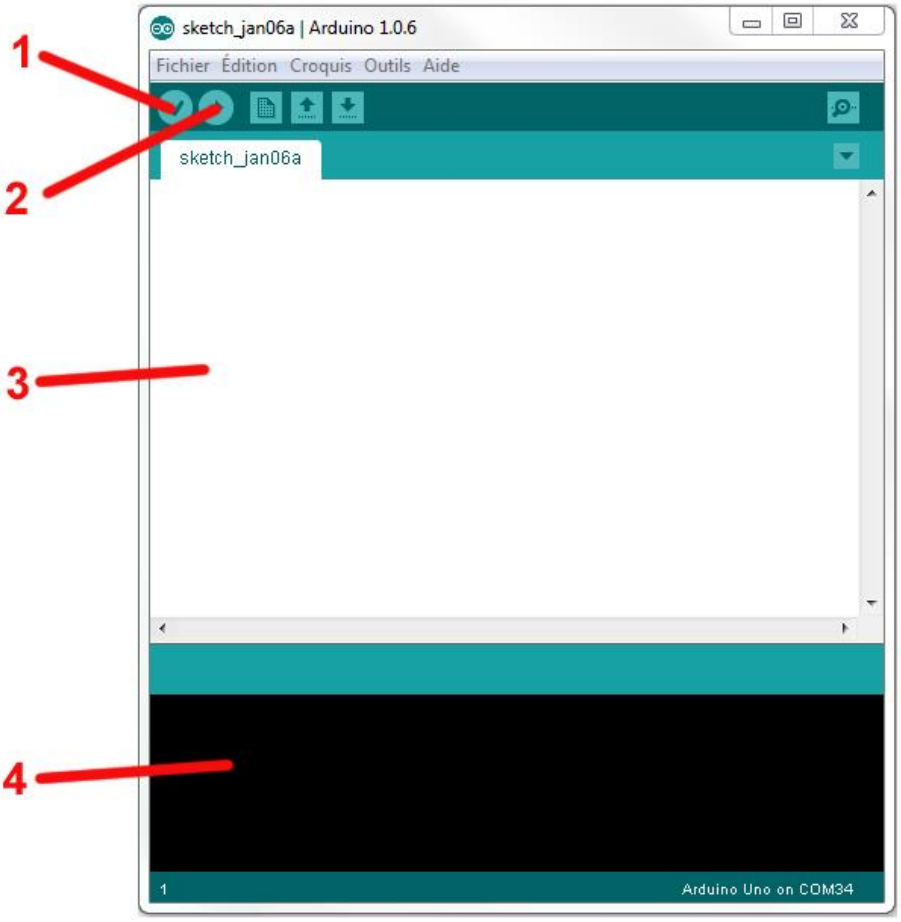
Installation du logiciel et des drivers :

Il est préférable d'installer le logiciel avant de raccorder votre carte Uno pour la première fois. Il est disponible en téléchargement sur le site officiel à l'adresse : https://www.arduino.cc/en/Main/Software. Vous pouvez prendre la dernière version ''Windows Installer''.

Une fois le logiciel installé, vous pouvez raccorder votre carte. Si le driver est correctement installé vous devez voir apparaitre la carte Uno dans votre gestionnaire de périphériques (faire une recherche du panneau de configuration) :



Le logiciel IDE se présente sous cette forme :



1- Permet de compiler votre programme pour vérifier les erreurs de syntaxe.
2- Permet de téléverser votre programme dans votre carte Uno. Le logiciel commence par compiler votre programme puis le transfère sur la carte Uno. Une erreur lors de la compilation annule la procédure.
3- Contenu du programme
4- Messages relatifs aux éventuelles erreurs de compilation ou de transfert et les messages lors d'un transfert réussi.

Dans l'onglet outils :

- Vous devez sélectionner le bon type de carte en fonction de votre utilisation, (Uno pour la plupart des cas).
- Port série : vous devez sélectionner le bon port COM en fonction de celui indiqué pour votre carte dans le gestionnaire de périphérique.

Installation et utilisation de la librairie GoTronic :

La librairie reprenant les programmes des différents montages de ce manuel est disponible sur www.gotronic.fr.

Vous pouvez l'installer en suivant ces étapes :

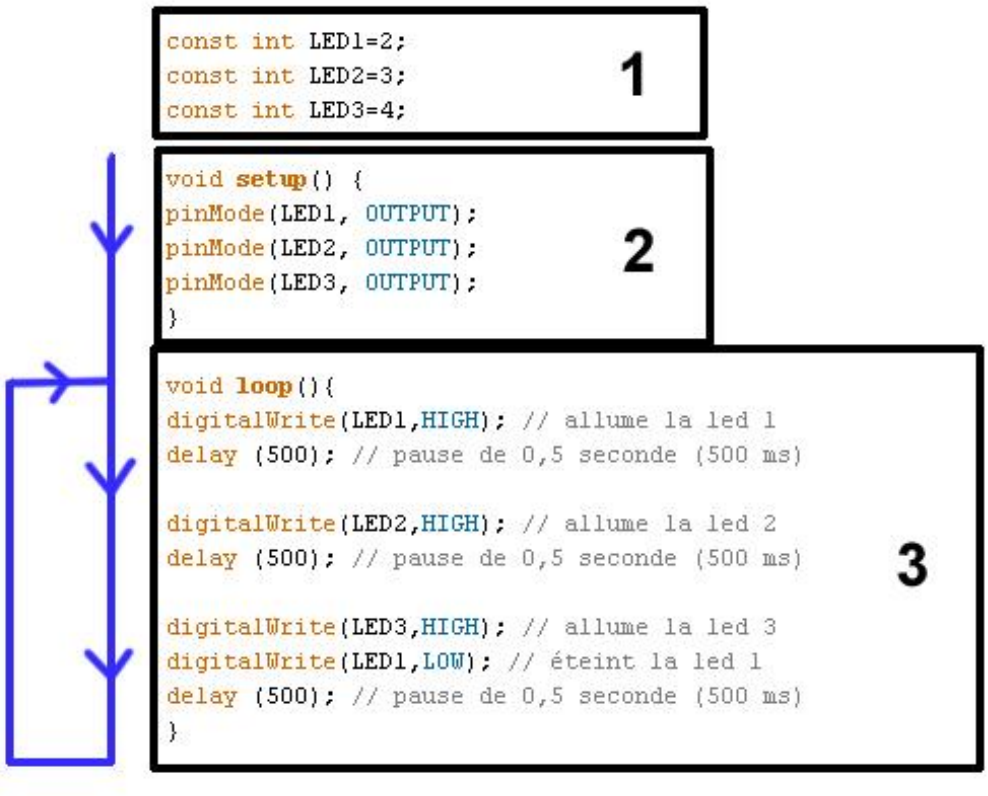
- Fermer le programme
- Télécharger le dossier sur www.gotronic.fr
- Extraire le dossier GoTronic_35110 et le coller dans votre répertoire : C:\Program Files\Arduino\examples\
- Relancer le logiciel IDE
- Pour ouvrir un programme correspondant à un montage, aller dans : Fichier > Exemples > GoTronic_35110 > Montage-(x)

Remarque : il s'agit du chemin d'installation par défaut, il est nécessaire de le changer en fonction de l'endroit choisi lors de l'installation du logiciel IDE

Fonctionnement d'un programme pour carte Uno :

Un programme pour carte Uno est constitué au minimum de la fonction *setup()* et de la boucle *loop()*. La fonction *setup()* est exécutée une fois lors de l'alimentation de la carte ou après un reset. La boucle *loop()* est ensuite exécutée à l'infini jusqu'à la coupure de l'alimentation de la carte.

Prenons un exemple de programme (le déroulement suit les flèches en bleu) :



1 - Déclaration des variables et/ou constantes : cette partie n'a pas d'impact direct avec le déroulement de votre programme. Elle n'est pas exécutée par la carte et sert uniquement à créer des appellations pour les réutiliser plus tard dans votre programme.

2 - Début de la fonction *setup()* : la carte commence par exécuter toutes les instructions présentes au démarrage. Généralement cette fonction sert d'initialisation : mise des broches en entrée ou en sortie, initialisation d'un écran LCD, démarrage du moniteur série, etc.

3 - Ensuite la carte exécute le programme principal *loop()*.

Que ce soit pour *setup()*, *loop()* ou une autre boucle, ces fonctions commencent toujours par une accolade { et finissent par une autre }, ce qui permet à votre programme de savoir quand commencent et finissent vos boucles.

Il est possible d'ajouter des commentaires qui ne seront pas interprétés par le programme en ajoutant ‘‘//’’ comme dans l'encadré 3. Ils permettent une meilleure lecture de votre programme et facilitent les modifications ultérieures.

Il est également possible de mettre un paragraphe en commentaire en le plaçant entre /* et */, exemple : /* *votre paragraphe de commentaire* */. Les commentaires apparaissent en gris sur le logiciel IDE Arduino® (en vert dans les montages suivant pour faciliter la lecture).

Montage 0 : Utiliser le moniteur série

Le moniteur série est une fenêtre dans laquelle s'affichent des informations envoyées depuis la carte UNO et où l'utilisateur peut envoyer des informations vers la carte.

Pour pouvoir l'utiliser, la carte UNO doit être raccordée en USB à l'ordinateur et la communication doit être initialisée dans le programme :

Dans le programme d'initialisation, le commande *Serial.begin(9600);* définit le débit de transmission des données. La valeur doit être la même que dans le moniteur série (choix en bas à droite).

Pour envoyer des informations vers le moniteur série, il suffit d'ajouter dans le programme la commande *Serial.print(LaValeurAAfficher);*. La valeur ou le texte à envoyer est à mettre entre les parenthèses.
La commande *Serial.println(LaValeurAAfficher);* envoie également la valeur et fait un retour à la ligne.

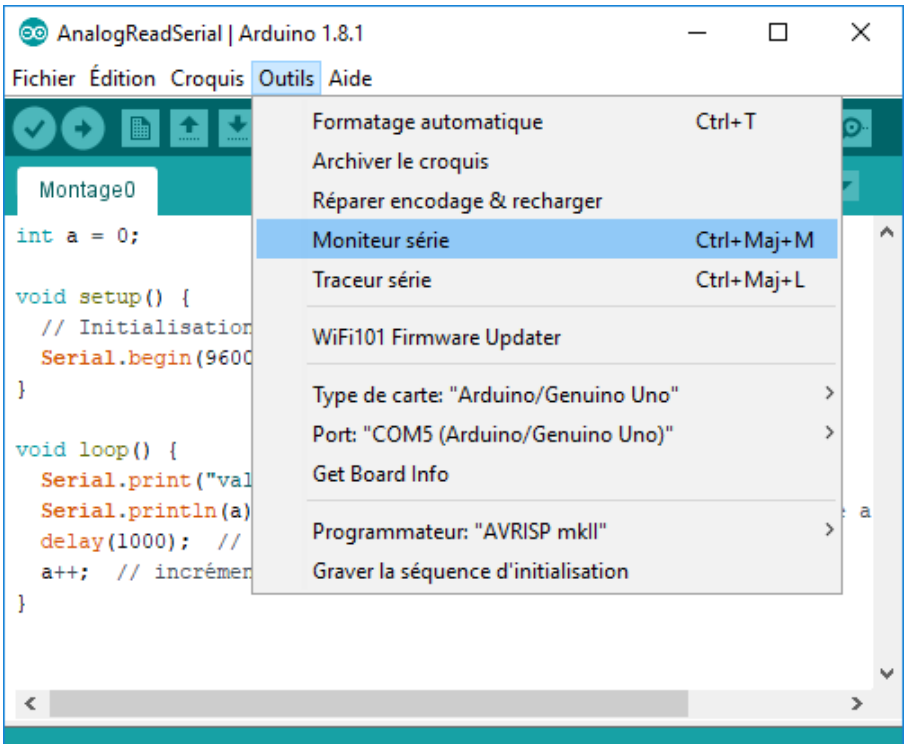
L'exemple de programme ci-dessous incrémente une variable « a » et affiche sa valeur dans le moniteur série toutes les secondes.

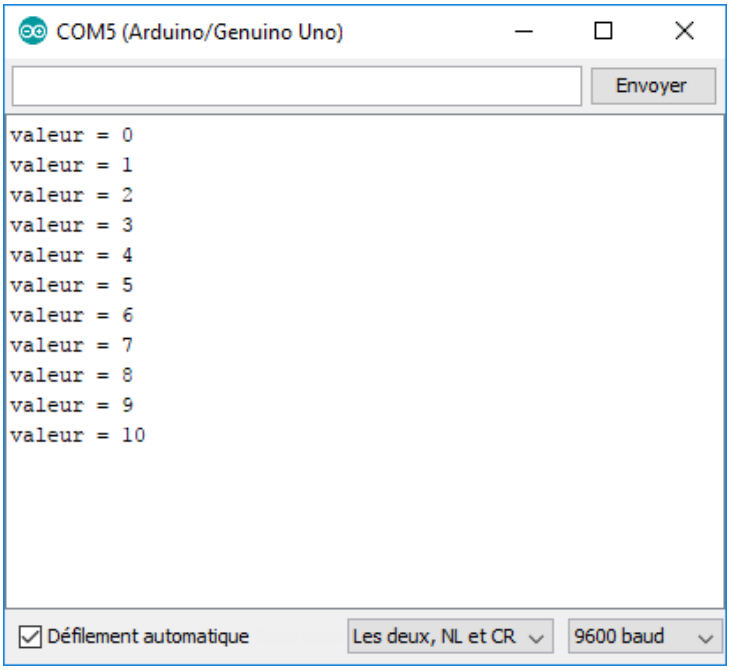
Le code

```
int a = 0;

void setup() {
  Serial.begin(9600);            // Initialisation de la communication série
}

void loop() {
  Serial.print("valeur = ");      // envoie le texte « valeur = »
  Serial.println(a);             // envoie le nombre contenu dans la variable « a »
                                 // et passe à la ligne suivante
  delay(1000);                   // fait une pause de 1 sec (1000 ms)
  a++;                           // incrémente la variable a de 1
}
```

Ouvrir le moniteur série

Pour ouvrir le moniteur série, allez dans le menu *Outils* > *Moniteur série*. La fenêtre du moniteur s'ouvre et vous pouvez l'utiliser.





Remarques :

- La carte UNO doit être connectée en USB à l'ordinateur avant d'ouvrir la fenêtre
- Le débit de transmission des données doit être le même dans le moniteur série que dans l'initialisation du programme (ici 9600 baud).

Montage 1 : Faire clignoter une led

Présentation

Ce premier montage permet de faire clignoter une led raccordée sur la sortie D13 de votre carte Uno.

Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-01. Téléverser le programme dans la carte Uno. La led clignote.

Liste des composants

Une led, une résistance de 220 Ω, deux cordons (1 bleu et 1 noir).

Nouveau composant utilisé

La led : il suffit d'alimenter la led pour qu'elle s'allume. L'utilisation d'une led nécessite l'ajout d'une résistance (ici 220 Ω => anneaux rouge/rouge/marron/or) en fonction de la tension d'alimentation et de la couleur de la led. La led possède un sens de raccordement : le - est à raccorder sur la patte la plus courte (du côté du méplat) et le + sur la broche la plus longue.

La résistance peut se raccorder dans les deux sens.

La formule permettant de calculer la valeur de la résistance est :

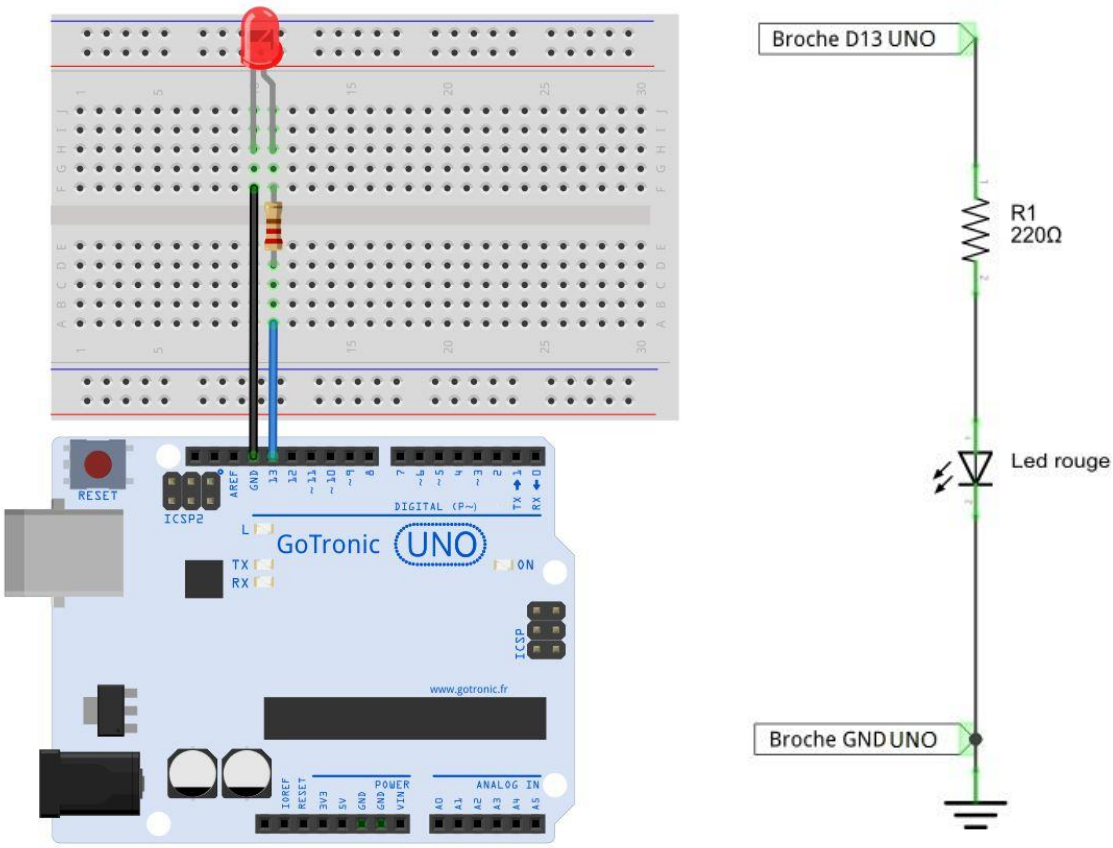
R = (Vcc – Vf) /I

Avec :
R : la valeur de la résistance à mettre en série [Ω]
Vcc : la tension d'alimentation [V]
Vf : la tension de seuil de la led (en fonction de la couleur) [V]
I : l'intensité de courant circulant dans la led [A]

Dans notre cas, R = (5 – 1,6) / 0,015 = 226 Ω

➔ On prend donc une résistance de 220 Ω (valeur inférieure la plus proche)

Montage et schéma



Le code

```
void setup() {                  // Instructions « setup » exécutées 1 fois au démarrage
    pinMode(13, OUTPUT);        // On configure la broche D13 en sortie
}                               // Fin de la boucle « setup »

void loop() {                   // début de la boucle «loop»
                                // s'exécute en boucle jusqu'à la mise hors tension de la carte
    digitalWrite(13, HIGH);     // allume la led (HIGH = état «haut»)
    delay(1000);                // attendre 1 seconde (1000 ms)
    digitalWrite(13, LOW);      // éteindre la led (LOW = état «bas»)
    delay(1000);                // attendre 1 seconde (1000 ms)
}                               // fin de la boucle « loop »
// Le programme revient au début de la boucle « loop » sans fin
```

Fonctionnement

- déclaration de la broche D13 en sortie pour commander la led.

- si on met la sortie D13 à l'état haut (HIGH) la led s'allume (voir schéma)

- dans notre programme on allume la led pendant 1 seconde puis on l'éteint pendant 1 seconde, etc.

Un problème ?

- vérifier le sens de la led, le GND doit être raccordé sur la broche la plus courte (ou le côté avec le méplat)

- vérifier que votre programme a bien été transféré dans votre carte Uno

Pour aller plus loin

- vous pouvez essayer de raccorder votre led sur une autre sortie, par exemple D9 et modifier le programme en fonction : int ledPin = 9;

- vous pouvez essayer de faire varier la temporisation du clignotement en modifiant la ligne delay (1000);

- il est possible de modifier l'intensité de la led en modifiant la ligne: digitalWrite(ledPin, HIGH); par analogWrite(ledPin, xx); (xx est à remplacer par l'intensité désirée, un nombre compris entre 0 et 255).

Montage 2 : Chenillard à leds

Présentation

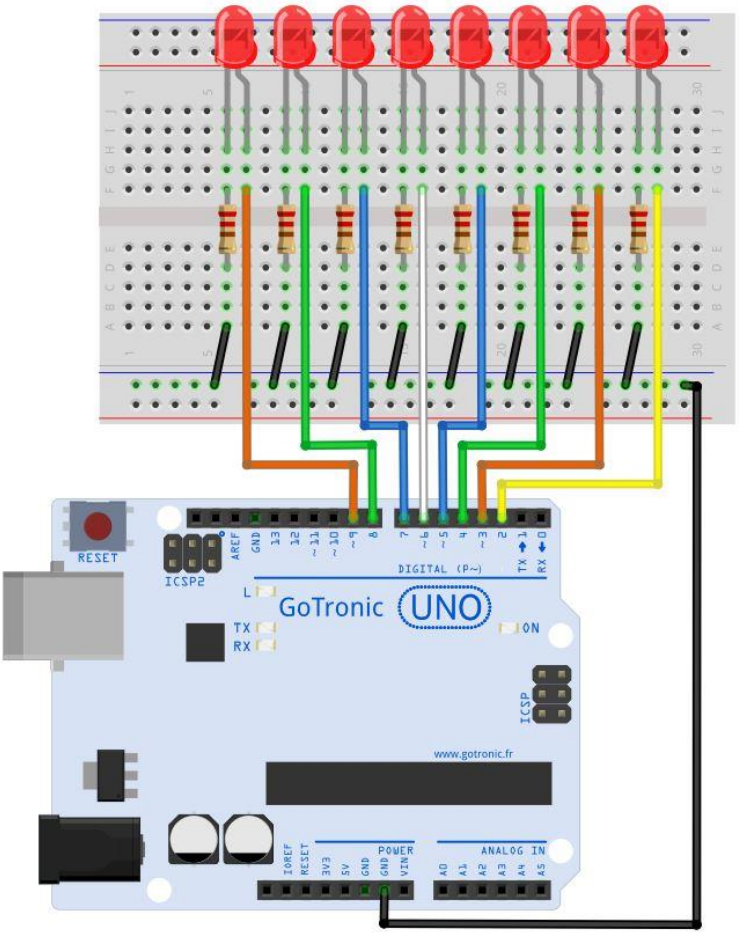
Ce montage permet de faire un chenillard à 8 leds.

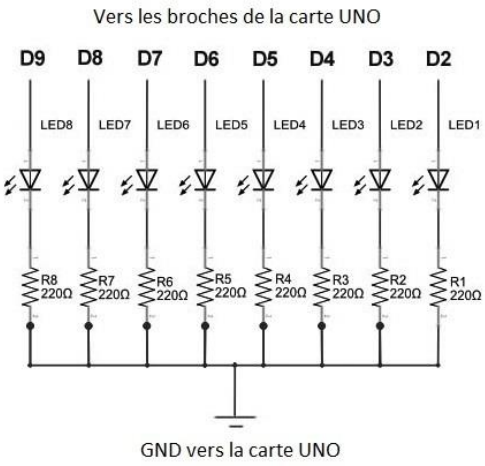
Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-02. Téléverser le programme dans la carte Uno. Les leds doivent s'allumer les unes après les autres.

Liste des composants

8 leds rouges, 8 résistances de 220 ohms et cordons de raccordement.

Montage et schéma





Le code

```
const int LED1=2;
const int LED2=3;
const int LED3=4;
const int LED4=5;
const int LED5=6;              // Déclaration des broches: LED1 sur broche D2,
const int LED6=7;              // LED sur broche D3, etc
const int LED7=8;
const int LED8=9;
int del=500;                   // La variable del contient la temporisation du clignotement (en ms)

void setup() { // Instructions «setup» exécutées 1 fois au démarrage
    pinMode(LED1, OUTPUT);
    pinMode(LED2, OUTPUT);
    pinMode(LED3, OUTPUT);
    pinMode(LED4, OUTPUT);
    pinMode(LED5, OUTPUT);     // On configure les broches en sortie
    pinMode(LED6, OUTPUT);
    pinMode(LED7, OUTPUT);
    pinMode(LED8, OUTPUT);
} // Fin de la boucle «Setup»

void loop() { // début de la boucle «loop». S'exécute sans fin jusqu'à la mise hors tension de la carte
    digitalWrite(LED1, HIGH);  // allume la led 1
    delay (del);               // pause de 0,5 seconde (500 ms)
    digitalWrite(LED2, HIGH);  // allume la led 2
    delay (del);               // pause de 0,5 seconde (500 ms)
    digitalWrite(LED3, HIGH);  // allume la led 3
    digitalWrite(LED1, LOW);   // éteint la led 1
    delay (del);               // pause de 0,5 seconde (500 ms)
    digitalWrite(LED4, HIGH);  // allume la led 3
    digitalWrite(LED2, LOW);   // éteint la led 2
    delay (del);               // pause de 0,5 seconde (500 ms)
    digitalWrite(LED5, HIGH);  // allume la led 5
    digitalWrite(LED3, LOW);   // éteint la led 3
    delay (del);                // pause de 0,5 seconde (500 ms)
    digitalWrite(LED6, HIGH);  // allume la led 6
    digitalWrite(LED4, LOW);   // éteint la led 4
    delay (del);               // pause de 0,5 seconde (500 ms)
    digitalWrite(LED7, HIGH);  // allume la led 7
    digitalWrite(LED5, LOW);   // éteint la led 5
    delay (del);               // pause de 0,5 seconde (500 ms)
    digitalWrite(LED8, HIGH);  // allume la led 8
    digitalWrite(LED6, LOW);   // éteint la led 6
    delay (del);               // pause de 0,5 seconde (500 ms)
    digitalWrite(LED7, LOW);   // éteint la led 7
    delay (del);               // pause de 0,5 seconde (500 ms)
    digitalWrite(LED8, LOW);   // éteint la led 8
    delay (del);               // pause de 0,5 seconde (500 ms)
}                               // fin de la boucle «loop»
```

Fonctionnement

- déclaration de toutes les broches en fonction du raccordement et déclaration des broches en sorties.

- le programme exécute la séquence complète (allumage successif de chaque led)

Un problème ?

- vérifier le sens de chaque led. Le GND doit être raccordé sur la broche la plus courte (ou le côté avec le méplat)

- vérifier que votre programme a bien été transféré dans votre carte Uno

- vérifier que vous avez bien raccordé le GND de la carte Uno sur votre plaque d'essai.

Pour aller plus loin

- essayer de faire varier la temporisation du clignotement en modifiant la variable « del ».

- essayer de modifier la séquence pour obtenir différents effets lumineux.

Montage 3 : Buzzer

Présentation

Ce montage permet de déclencher le buzzer lorsque l'on appuie sur un bouton-poussoir, chaque bouton-poussoir active une tonalité différente.

Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-03. Téléverser le programme dans la carte Uno.

Appuyer sur un bouton-poussoir, le buzzer retentit. Appuyer sur l'autre bouton-poussoir pour changer de tonalité.

Liste des composants

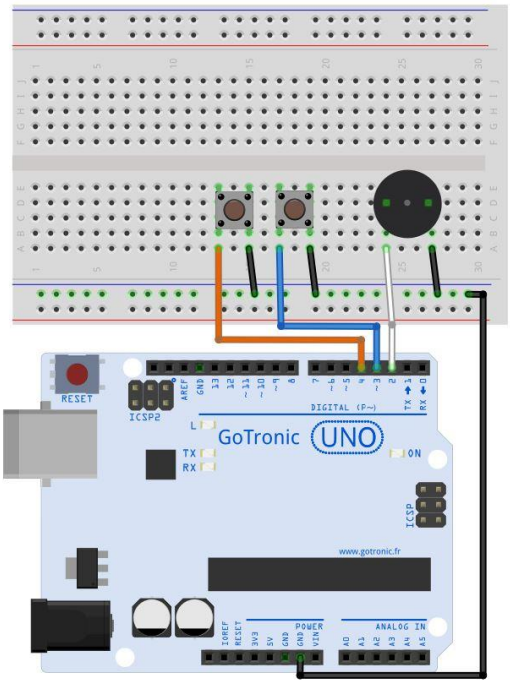
2 boutons-poussoirs, 1 buzzer et cordons de raccordement.

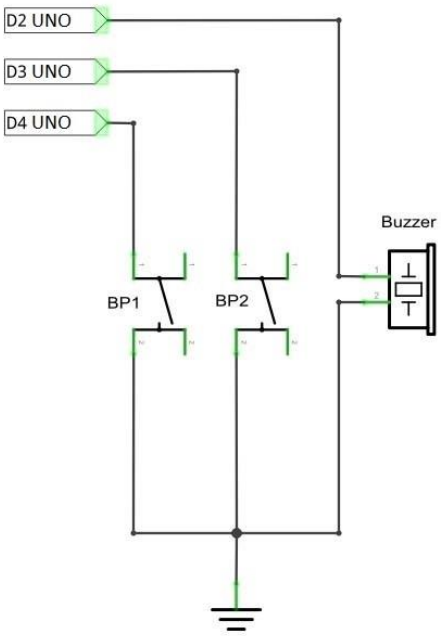
Nouveau composant utilisé

Le bouton-poussoir : un contact s'établit entre les deux broches lorsque l'on appuie sur le bouton (le contact se ferme). Il revient dans sa position initiale une fois qu'il est relâché (le contact s'ouvre). Les broches sont doublées pour une meilleure stabilité sur la plaque d'essai (voir schéma). Attention au sens de montage, il peut être tourné de 180° mais ne peut pas fonctionner si on le tourne de 90°.

Le buzzer : il permet de réaliser différents effets sonores en fonction de la fréquence appliquée. Le GND se raccorde sur le - du buzzer. Le + est commandé par une sortie de la carte Uno.

Montage et schéma





Le code :

```
int buzzer = 2;
int BP1 = 3;
int BP2 = 4;
bool ETATBP1;
bool ETATBP2;

void setup() {
    pinMode(buzzer, OUTPUT);  //broche en sorties
    pinMode(BP1, INPUT);
    pinMode(BP2, INPUT);        //broche en entrées + activation pullup pour les BPs
    digitalWrite(BP1, HIGH) ;
    digitalWrite(BP2, HIGH) ;
}

void loop() {
    ETATBP1=digitalRead(BP1); // lit l'état du BP+ et stocke la valeur
    ETATBP2=digitalRead(BP2); // lit l'état du BP+ et stocke la valeur
    if (ETATBP1==0){
          digitalWrite(buzzer,HIGH);
          delay(1);
          digitalWrite(buzzer,LOW);
          delay(1);
    }
    if (ETATBP2==0){
          digitalWrite(buzzer,HIGH);
          delay(3);
          digitalWrite(buzzer,LOW);
          delay(3);
    }
}
```

Fonctionnement

- déclaration des différentes entrées/sorties

- activation de la fonction ‘‘pullup’’ sur les deux boutons-poussoirs, la carte Uno nécessite un 5V pour l'état ‘‘haut’’ et 0V pour son état ‘bas’’. Une résistance interne raccordée sur le 5V est activée par cette fonction, au repos l'entrée est à l'état ‘‘haut’’ lorsque l'on appuie sur le bouton-poussoir, elle passe à l'état bas.

- la fonction digitalRead () permet de lire l'état d'une entrée, ici l'état de l'entrée pour chaque bouton-poussoir

- la boucle if (si) : elle commence par { et se finit par }, le programme n'exécute cette boucle que si la condition () est remplie, ici la boucle est vraie uniquement lorsque l'on appuie sur un bouton-poussoir.

- les délais étant différents dans les deux boucles if (), le son du buzzer est différent en fonction du bouton-poussoir appuyé.

Un problème ?

- vérifier que votre programme a bien été transféré dans votre carte Uno

- vérifier que vous avez bien raccordé le GND de la carte Uno sur votre plaque d'essai.

- vérifier le sens des deux boutons-poussoirs et du buzzer

Pour aller plus loin

- essayer de faire varier la temporisation en modifiant la ligne delay (); pour jouer sur la tonalité

- vous pouvez essayer de modifier la séquence pour obtenir différents effets sonores.

Montage 4 : Compteur / décompteur

Présentation

Ce montage compteur/décompteur à afficheur à leds compte lorsque l'on appuie sur un bouton-poussoir et décompte lorsque l'on appuie sur le deuxième bouton-poussoir.

Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-04. Téléverser le programme dans la carte Uno.

Appuyer sur un bouton-poussoir, l'afficheur à leds compte, appuyer sur l'autre bouton-poussoir pour décompter.

Liste des composants

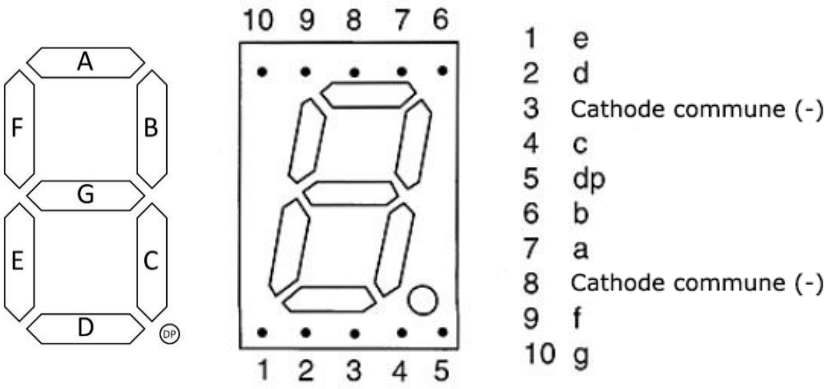
2 boutons-poussoirs, 1 afficheur à leds, 1 circuit 74HC595, 7 résistance de 220 ohms.

Nouveau composant utilisé

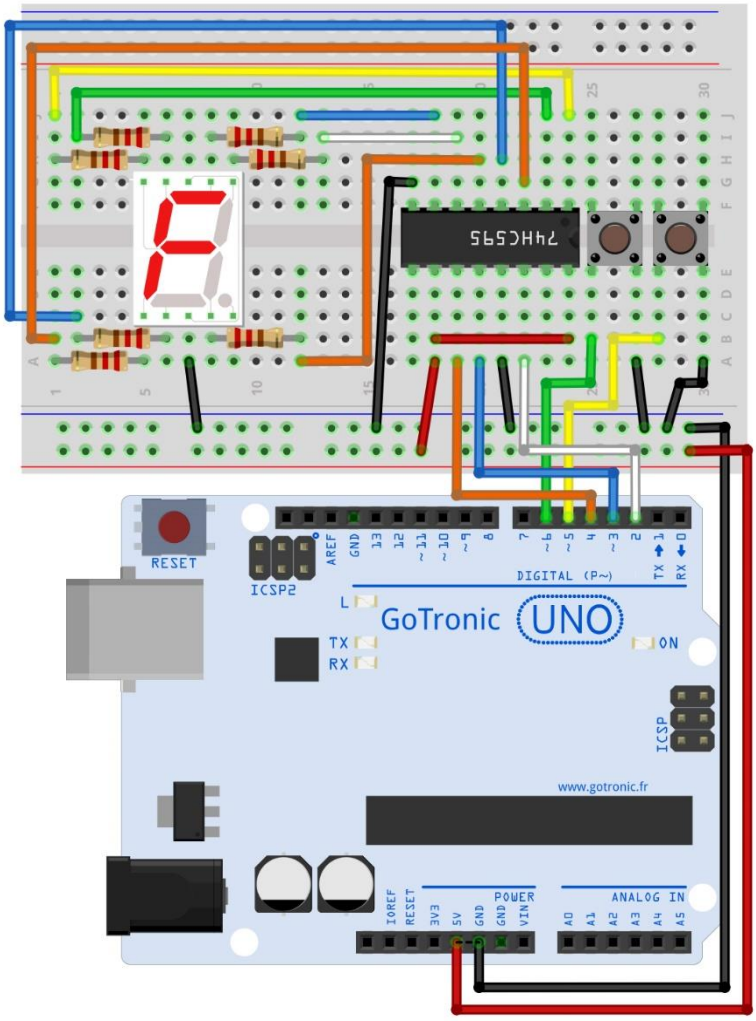
Le 74HC595 : c'est un composant permettant de raccorder 8 leds. Il suffit de lui envoyer les informations en série sur une seule broche pour le faire fonctionner en laissant libre les autres entrées/sorties de la carte Uno. Il est également équipé d'une sortie pour le raccordement de plusieurs 74HC595 en cascade (non utilisé ici).

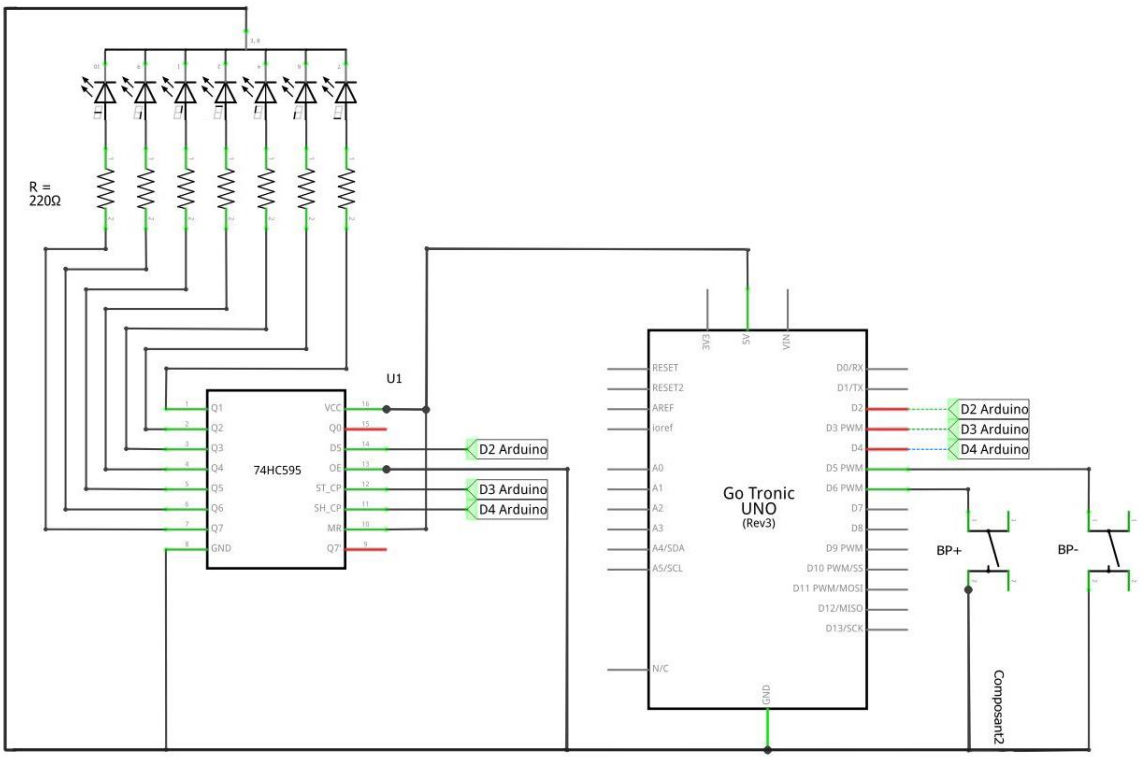
Afficheur 7 segments : afficheur constitué en interne de 8 leds (7 pour les segments et 1 pour le point). Il fonctionne de la même manière que des leds standards.

Le point n'est pas utilisé dans ce montage. Chaque segment possède une lettre d'identification :



Montage et schéma





Le code

```
int donnee = 2;
int latch = 3;
int clock = 4;
int BPMOINS = 5;
int BPPLUS = 6;
int chiffre;
int ETATBPP;
int ETATBPM;
byte dec_digits[] = {0b11111100, 0b01100000, 0b11011010, 0b11110010, 0b01100110,
0b00110110, 0b10111110, 0b11100000, 0b11111111, 0b11110110};    /* On configure ici les chiffres
que l'on veut afficher, les 8 chiffres après 0b correspondent à 1 segment ''0'' segment allumé et ''1''
segment éteint. Le premier représente le chiffre 0, le deuxième le chiffre 1, etc */

void setup() {                    //broches en sorties
    pinMode(latch, OUTPUT);
    pinMode(clock, OUTPUT);
    pinMode(donnee, OUTPUT);      //broches en entrées + activation pullup pour les BPs
    pinMode(BPMOINS, INPUT);
    pinMode(BPPLUS, INPUT);
    digitalWrite(BPMOINS, HIGH) ;
    digitalWrite(BPPLUS, HIGH) ;
    chiffre=5;                    // On configure la valeur 5 dans la variable chiffre dans l'initialisation
}

void loop() {
    ETATBPP=digitalRead(BPPLUS);      // lit l'état du BP+ et stocke la valeur
    ETATBPM=digitalRead(BPMOINS);     // lit l'état du BP+ et stocke la valeur
    if (ETATBPP==0){                  // On rentre dans la boucle si le 1er bouton-poussoir est appuyé
        chiffre = chiffre + 1;        // On ajoute +1 à la variable chiffre
        if (chiffre==10){
          chiffre = 9;                // Si chiffre = 10 on le remet à 9 pour éviter les valeurs > 9
        }
        digitalWrite(latch, LOW);
        shiftOut(donnee, clock, MSBFIRST, dec_digits[chiffre]);   // On envoie le chiffre sur l'afficheur
        digitalWrite(latch, HIGH);
        delay(300);
    }
    if (ETATBPM==0){                  // On rentre dans la boucle si le 2è bouton-poussoir est appuyé
        chiffre = chiffre - 1;        // On retire -1 à la variable chiffre
        if (chiffre==0){
          chiffre = 1;                // Si chiffre = 0 on le remet à 1 pour éviter les valeurs < 1
        }
        digitalWrite(latch, LOW);
        shiftOut(donnee, clock, MSBFIRST, dec_digits[chiffre]);   // On envoie le chiffre sur l'afficheur
        digitalWrite(latch, HIGH);
        delay(300);
    }
}
```

Fonctionnement

- la ligne byte dec_digits permet de stocker les données à envoyer pour afficher les digits, par exemple pour le chiffre 0 il faut afficher tous les segments sauf le segment G (voir schéma de l'afficheur) donc il faut envoyer : 0000 0011 (0 = led allumée et 1 = led, éteinte). Le dernier 1 correspond au point (non utilisé) et n'a donc pas d'importance.

- on crée une variable ''chiffre'' que l'on initialise à 5 dans la boucle setup()

- il suffit ensuite d'ajouter 1 à ''chiffre'' lorsque l'on appuie sur un bouton-poussoir et de lui retirer 1 lorsque l'on appuie sur le deuxième bouton-poussoir.

- il reste ensuite à afficher la valeur de chiffre sur l'afficheur via la ligne shiftOut.

Cette ligne envoie en série (sur le 74HC595) les données stockées dans dec_digits en fonction de la valeur de chiffre, si chiffre = 0, on envoie la première valeur de dec_digits (0b0000011), si chiffre = 1, on envoie la deuxième valeur de dec_digits (b10011111), etc

Un problème ?

- vérifier que votre programme a bien été transféré dans votre carte Uno

- vérifier le sens du composant 74HC595 et le sens de l'afficheur à leds

- vérifier que vous avez bien raccordé le GND de la carte Uno sur votre plaque d'essai.

Pour aller plus loin

- essayer de modifier les valeurs derrière byte dec_digits[] => ''0'' led allumée et ''1'' led éteinte

- vous pouvez essayer de faire un ''serpent'' tournant autour de l'afficheur avec deux segments continus.

Montage 5 : Capteur de température

Présentation

Montage permettant d'afficher la température sur l'afficheur 4 digits à leds.

Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-05. Téléverser le programme dans la carte Uno. La température est affichée, vous pouvez souffler sur le capteur pour faire monter la température en temps réel.

Liste des composants

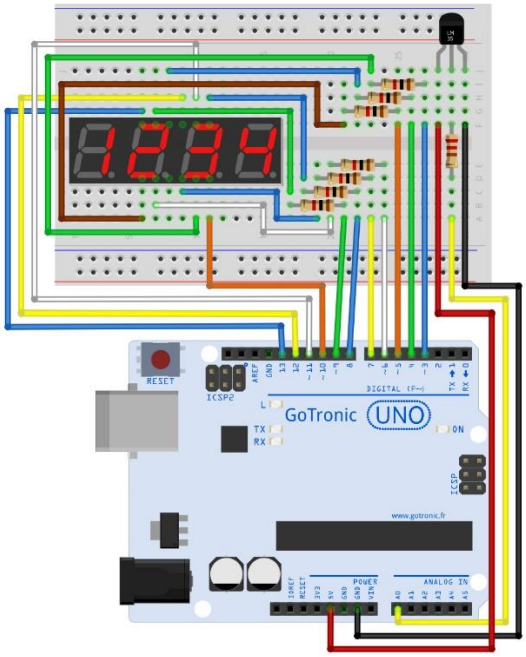
1 afficheur 4 digits à leds, 8 résistances de 220 ohms et 1 capteur LM35.

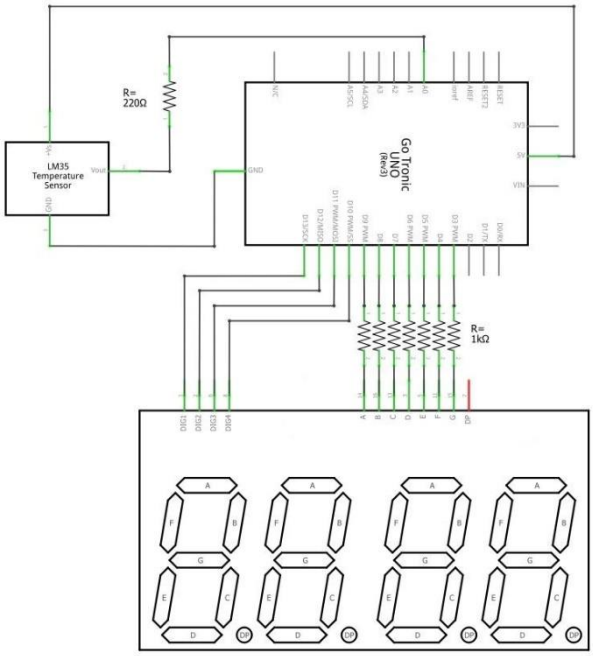
Nouveau composant utilisé

Le LM35 : c'est un capteur de température, il délivre une tension analogique en fonction de la température mesurée.

L'afficheur 4 digits : même fonctionnement que l'afficheur précédent (montage 4), les 8 segments sont raccordés ensemble par l'anode : le ''+'' (voir schéma ci-dessous). Pour faire fonctionner cet afficheur, il faut utiliser le multiplexage : les 4 chiffres ne sont pas allumés en même temps, on allume d'abord le 1 puis le 2, etc. La vitesse d'affichage donne l'impression que les 4 sont allumés en même temps.

Montage et schéma





Le code

```
int x;
int del = 2;
int val;
int temp;
int diz;
int uni;
int boucle;

void setup() {
    pinMode(13, OUTPUT);
    pinMode(12, OUTPUT);
    pinMode(11, OUTPUT);
    pinMode(10, OUTPUT);
    pinMode(9, OUTPUT);
    pinMode(8, OUTPUT);
    pinMode(7, OUTPUT);
    pinMode(6, OUTPUT);
    pinMode(5, OUTPUT);
    pinMode(4, OUTPUT);
    pinMode(3, OUTPUT);
}

void loop() {
    val = analogRead(0); // On lit la valeur du capteur
    temp = (500 * val) / 1024;; // On convertit la valeur en degrés (formule en fonction du capteur)
    diz = temp / 10; // On extrait la dizaine de la température
    uni = temp % 10; // On extrait l'unité de la température
    boucle = 0;
    while (boucle < 200) { // On rentre dans cette boucle 200 fois avant d'en sortir, cela permet d'ajouter
    une temporisation
        digitalWrite(13, LOW); // On allume le digit 1
        affdigit(diz); // On affiche la dizaine
        delay(del); //
        digitalWrite(13, HIGH); // On éteint le digit 1
        digitalWrite(12, LOW); // On allume le digit 2
        affdigit(uni); // On affiche l'unité
        delay(del);
        digitalWrite(12, HIGH); // On éteint le digit 2
        digitalWrite(11, LOW); // On allume le digit 3
        degres(); // On affiche degrés (symbole "°")
        delay(del);
        digitalWrite(11, HIGH); // On éteint le digit 3
        digitalWrite(10, LOW); // On allume le digit 4
        C(); // On affiche "C"
        delay(del);
        digitalWrite(10, HIGH);  // On éteint le digit 4
        boucle++; // On incrémente la valeur de boucle (On revient à l'instruction While jusqu'à ce que
boucle = 200)
  }
}
```

```
void affdigit(int x) {
  switch (x) {
    case 1: un(); break;
    case 2: deux(); break;
    case 3: trois(); break;
    case 4: quatre(); break;
    case 5: cinq(); break;   // Ici on ajoute une fonction pour l'afficheur d'un chiffre,
    case 6: six(); break;    // on réalise case 1 si x = 1, case 2 si x=2, etc
    case 7: sept(); break;
    case 8: huit(); break;
    case 9: neuf(); break;
    default: zero(); break;
  }
}


void un() {
  digitalWrite(9, LOW); // A
  digitalWrite(8, HIGH); // B
  digitalWrite(7, HIGH); // C
  digitalWrite(6, LOW); // D     Chiffre 1: "HIGH" = led allumée et "LOW" = led éteinte
  digitalWrite(5, LOW); // E
  digitalWrite(4, LOW); // F
  digitalWrite(3, LOW); // G
}

void deux() {
  digitalWrite(9, HIGH); // A
  digitalWrite(8, HIGH); // B
  digitalWrite(7, LOW); // C
  digitalWrite(6, HIGH); // D    Chiffre 2: "HIGH" = led allumée et "LOW" = led éteinte
  digitalWrite(5, HIGH); // E
  digitalWrite(4, LOW); // F
  digitalWrite(3, HIGH); // G
}

void trois() { ..... idem pour chiffre 3 à 0
******* voir librairie Montage-05 pour le code complet (chiffre 3 à 0) **************

void degres() {
  digitalWrite(9, HIGH); // A
  digitalWrite(8, HIGH); // B
  digitalWrite(7, LOW); // C
  digitalWrite(6, LOW); // D   Symbole °: "HIGH" = led allumée et "LOW" = led éteinte
  digitalWrite(5, LOW); // E
  digitalWrite(4, HIGH); // F
  digitalWrite(3, HIGH); // G
}
```

```
void C() {
  digitalWrite(9, HIGH); // A
  digitalWrite(8, LOW); // B
  digitalWrite(7, LOW); // C
  digitalWrite(6, HIGH); // D  Lettre C: "HIGH" = led allumée et "LOW" = led éteinte
  digitalWrite(5, HIGH); // E
  digitalWrite(4, HIGH); // F
  digitalWrite(3, LOW); // G
}
```

Fonctionnement

- boucle loop(): on récupère la valeur du capteur de température, on allume le 1er digit pour afficher la dizaine, on allume le 2ème digit pour afficher l'unité, on allume le digit 3 puis le digit 4 pour afficher ''°C''.

- la valeur del permet de configurer le temps d'allumage, vous pouvez mettre del=500 pour voir au ralenti le fonctionnement de l'ensemble.

- la boucle while se répète tant que boucle est < 200, ce qui permet d'afficher la température 200 fois avant de refaire une mesure (pour éviter une instabilité).

Un problème ?

- vérifier que votre programme a bien été transféré dans votre carte Uno

- si la valeur affichée ne correspond pas, vérifier le câblage du capteur

- si l'afficheur ne fonctionne pas ou mal, vérifier le câblage complet de l'afficheur.

Pour aller plus loin

- sur les boucles un(), deux(), etc vous pouvez modifier l'aspect de chaque chiffre.

- vous pouvez essayer d'ajouter les boucles A(), B(), C() et D() et d'afficher les lettres sur l'afficheur en modifiant la boucle loop().

Montage 6 : Détecteur de flamme

Présentation

Montage permettant de détecter une flamme.

Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-06. Téléverser le programme dans la carte Uno.

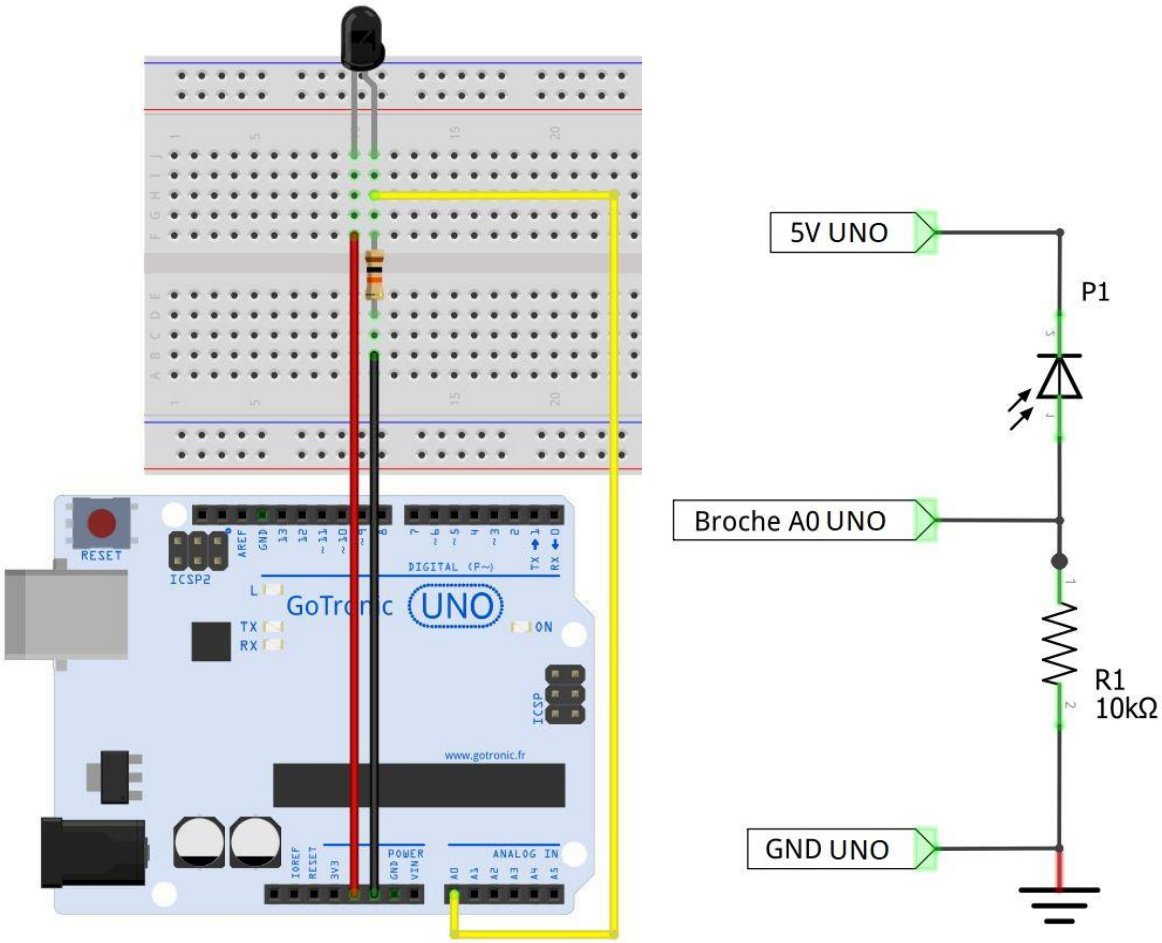
Liste des composants

1 capteur de flamme, 1 résistance de 10k.

Nouveau composant utilisé

Le capteur de flamme est une photodiode qui réagit à la longueur d'onde des infrarouges émis par une flamme. La diode est passante lorsqu'il y a une flamme à proximité, et bloquante quand il n'y en a pas.

Montage et schéma



Le code

```
int capteur = 0;
int valeur;

void setup() {
  pinMode(capteur, INPUT);
  Serial.begin(9600);
}

void loop() {
  valeur=analogRead(capteur);
  Serial.print("valeur : ");
  Serial.println(valeur);
  if (valeur>500) {
    Serial.println("ALERTE INCENDIE !!");
  }
  delay(500);
}
```

Fonctionnement

- Déclaration du capteur et de la variable

- Initialisation de l'entrée et du moniteur série

- Toutes les 1/2 secondes, la valeur du capteur est analysée et affichée sur le moniteur série. Si cette valeur est supérieure à 500, on affiche la texte « ALERTE INCENDIE » dans le moniteur série.

Un problème ?

- vérifier que votre programme a bien été transféré dans votre carte Uno.

- Vérifier le sens de la diode, la broche la plus longue vers l'entrée analogique.

Pour aller plus loin

- Vous pouvez ajouter des actions à faire en cas de détection de flamme, par exemple un buzzer, une led rouge, etc.

Montage 7 : Capteur de lumière

Présentation

Ce montage permet d'allumer plusieurs leds en fonction de la quantité de lumière reçue par le capteur LDR.

Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-07. Téléverser le programme dans la carte Uno. Les leds s'allument en fonction de la lumière.

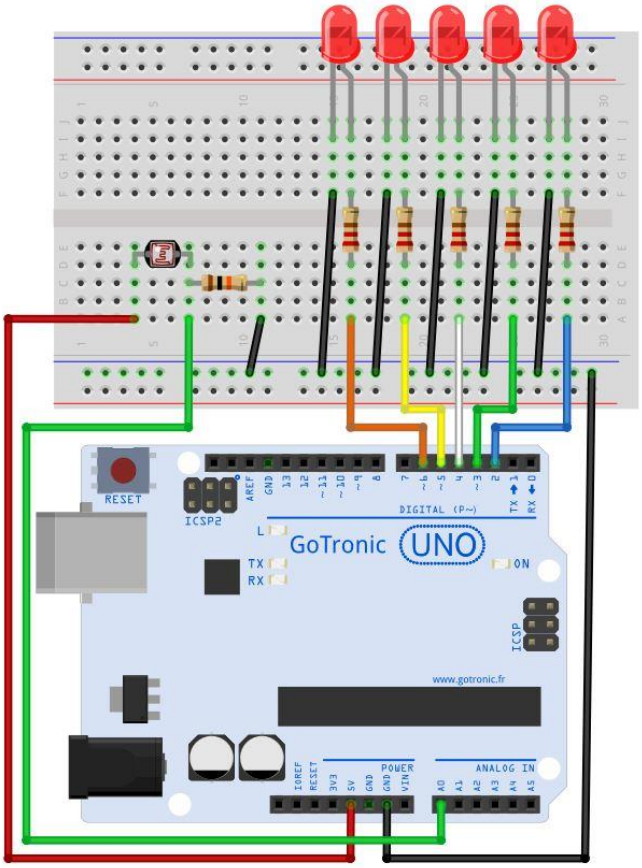
Liste des composants

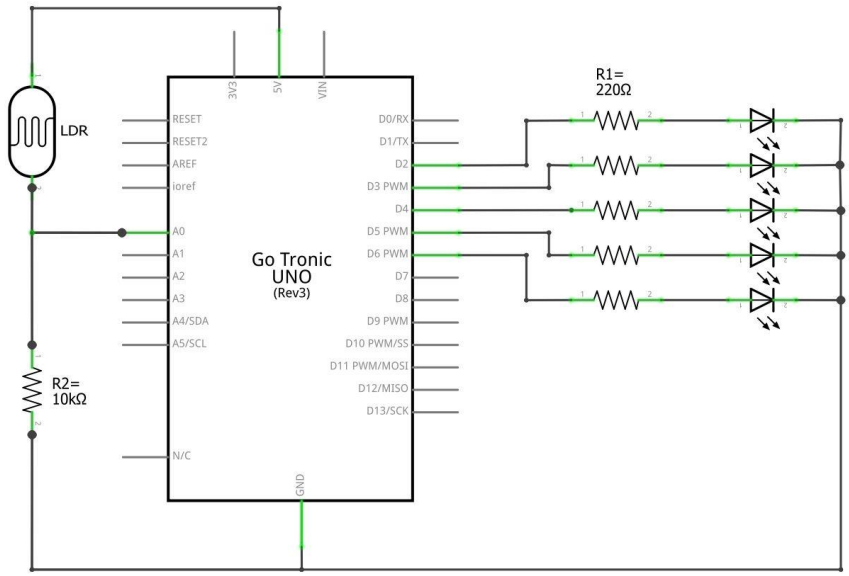
5 leds, 5 résistances de 220 ohms, 1 résistance de 10 kohms et un capteur de température LDR.

Nouveau composant utilisé

Le capteur de lumière LDR, c'est un capteur résistif, son impédance (résistance) varie en fonction de la quantité de lumière reçue. Il suffit de l'associer à une résistance de 10 kohms pour obtenir une tension analogique proportionnelle à la quantité de lumière reçue.

Montage et schéma





Le code

```
const int LED1=2;
const int LED2=3;
const int LED3=4;
const int LED4=5;
const int LED5=6;
const int LDR=A0;
int valeur = 0;
int valeurcal = 0;

void setup() {
    pinMode(LED1, OUTPUT);
    pinMode(LED2, OUTPUT);
    pinMode(LED3, OUTPUT);
    pinMode(LED4, OUTPUT);
    pinMode(LED5, OUTPUT);
    pinMode(LDR, INPUT);
}

void loop() {
    valeur = analogRead(LDR); // On lit la valeur du capteur LDR
    valeurcal = map(valeur, 250, 900, 0, 1000); // adapter ici l'échelle si besoin
    digitalWrite(LED1,HIGH);      // On allume uniquement la led 1 et éteint les autres
    digitalWrite(LED2,LOW);
    digitalWrite(LED3,LOW);
    digitalWrite(LED4,LOW);
    digitalWrite(LED5,LOW);
    // Plus la valeur du capteur est basse et plus on allume de leds
    if (valeurcal < 800) {digitalWrite(LED2,HIGH);}
    if (valeurcal < 600) {digitalWrite(LED3,HIGH);}
    if (valeurcal < 400) {digitalWrite(LED4,HIGH);}
    if (valeurcal < 200) {digitalWrite(LED5,HIGH);}
    delay (100);
}
```

Fonctionnement

- déclaration de toutes les broches en fonction du raccordement et déclaration des broches en sorties.

- on adapte l'échelle via l'instruction map() pour conserver le même programme quelles que soitent nos valeurs obtenues par la LDR.

- plus la valeur de notre capteur est petite plus on allume de leds avec la fonction if ().

Un problème ?

- vérifier le sens de chaque led. Le GND doit être raccordé sur la broche la plus courte (ou le côté avec le méplat)

- vérifier que votre programme a bien été transféré dans votre carte Uno

- vérifier que vous avez bien raccordé le GND de la carte Uno sur votre plaque d'essai.

Montage 8 : Potentiomètre

Présentation

Montage permettant de l'allumer des leds suivant la position d'un potentiomètre.

Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-08. Téléverser le programme dans la carte Uno. Une des leds est allumée suivant la position du potentiomètre.

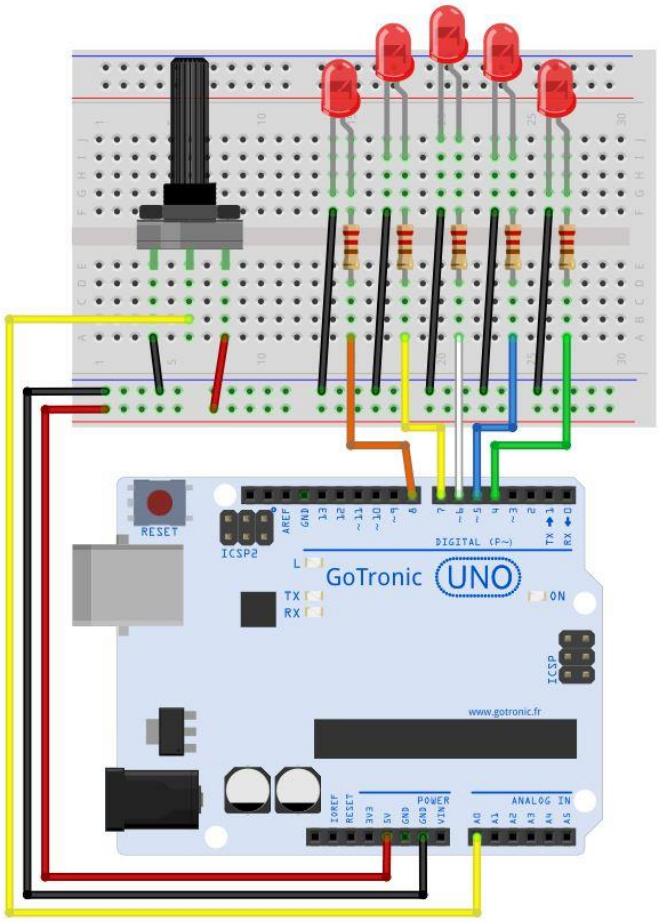
Liste des composants

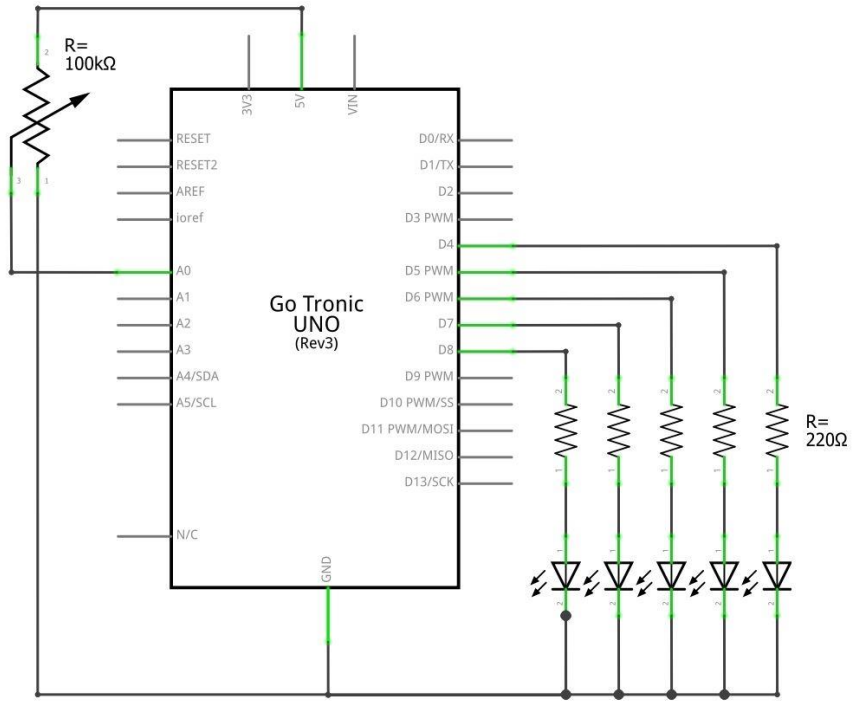
Un potentiomètre, 5 leds et 5 résistances de 220 ohms.

Nouveau composant utilisé

Le potentiomètre se comporte comme une simple résistance le long de laquelle se déplace un curseur. La résistance entre une borne du potentiomètre et le curseur varie de 0 ohms à la valeur maximale (ici 50 kohms) en fonction de la position de l'axe. Il suffit de brancher les deux bornes du potentiomètre à Vcc et GND et le curseur à une entrée analogique pour obtenir une tension analogique qui variera en en fonction de la position du curseur.

Montage et schéma





Le code

```
const int LED1=8;
const int LED2=7;
const int LED3=6;
const int LED4=5;
const int LED5=4;
int brochepot = A0;
int valeur;

void setup() {
    pinMode(LED1, OUTPUT);
    pinMode(LED2, OUTPUT);
    pinMode(LED3, OUTPUT);
    pinMode(LED4, OUTPUT);
    pinMode(LED5, OUTPUT);
}

void loop() {
// On éteint toutes les leds
    digitalWrite(LED1, LOW);
    digitalWrite(LED2, LOW);
    digitalWrite(LED3, LOW);
    digitalWrite(LED4, LOW);
    digitalWrite(LED5, LOW);
// On récupère la valeur du potentiomètre
    valeur = analogRead(brochepot);
```

```
// On allume les leds en fonction de la valeur récupérée
    if (valeur > 1) {digitalWrite(LED1, HIGH);}
    if (valeur > 205) {digitalWrite(LED2, HIGH);}
    if (valeur > 410) {digitalWrite(LED3, HIGH);}
    if (valeur > 615) {digitalWrite(LED4, HIGH);}
    if (valeur > 820) {digitalWrite(LED5, HIGH);}
    delay(15);
}
```

Fonctionnement

- déclaration de toutes les broches en fonction du raccordement et déclaration des broches en sorties.

- on va lire la valeur du potentiomètre et on la place dans une variable.

- on allume les leds en fonction de la valeur du potentiomètre.

Un problème ?

- vérifier le sens de chaque led. Le GND doit être raccordé sur la broche la plus courte (ou le côté avec le méplat)

- vérifier que votre programme a bien été transféré dans votre carte Uno

Montage 9 : Interrupteur à bille

Présentation

Ce montage permet d'allumer une led quand l'interrupteur à bille est incliné.

Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-09. Téléverser le programme dans la carte Uno. La led s'allume quand l'interrupteur à bille est incliné.

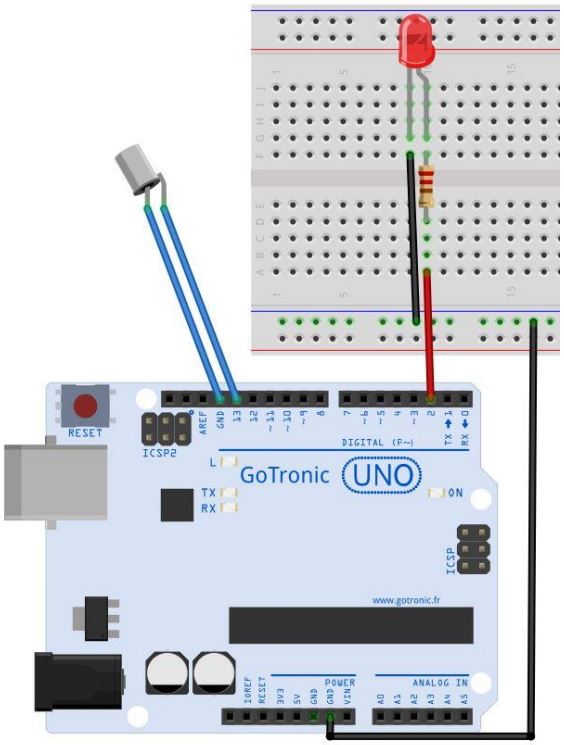
Liste des composants

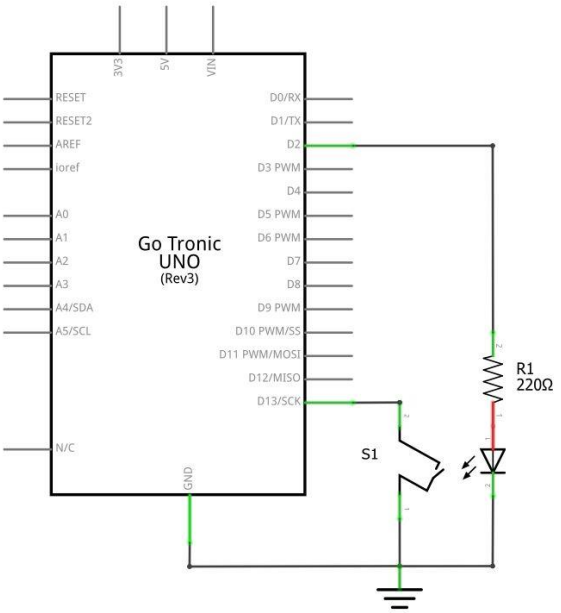
1 interrupteur à bille, 1 led et 1 résistance 220 ohms.

Nouveau composant utilisé

L'interrupteur à bille se comporte comme un bouton-poussoir : le contact est fermé lorsqu'il est à la verticale et le contacte s'ouvre lorsqu'il est penché, retourné ou simplement secoué.

Montage et schéma





Le code

```
int INTERR = 13;
int LED = 2;
int ETAT;

void setup() {
  pinMode(LED, OUTPUT);        // broche en sorties
  pinMode(INTERR, INPUT);      // broche en entrées + activation pullup
  digitalWrite(INTERR, HIGH) ;
}

void loop() {
  ETAT = digitalRead(INTERR); // lit l'état du BP+ et stocke la valeur
  digitalWrite(LED, ETAT);        // on donne l'état de la lecture à la sortie led
  delay(10);
}
```

Fonctionnement

- déclaration des différentes entrées/sorties

- activation de la fonction ‘‘pullup’’ sur l’interrupteur

- la fonction digitalRead () permet de lire l’état de l’entrée (l’interrupteur)

- La valeur de la sortie est l’image de l’entrée pour que la led s'allume lorsque le contact s’ouvre

Un problème ?

- vérifier que votre programme a bien été transféré dans votre carte Uno

- vérifier que vous avez bien raccordé le GND de la carte Uno sur votre plaque d’essai

- vérifier le sens de la led (le sens de l’interrupteur à bille n’a aucune importance)

Pour aller plus loin

- vous pouvez essayer d’ajouter des composants pour obtenir un projet plus complexe, par exemple un buzzer qui émet un son quand on secoue la table.

Montage 10 : Led RVB

Présentation

Montage permettant d'allumer une led RVB (rouge, verte, bleue).

Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-10. Téléverser le programme dans la carte Uno. La led s'allume avec les 3 couleurs au ¼ de leur luminosité. A chaque pression sur un bouton, la couleur associée éclairera à 0%, 25%, 50%, 75% et 100% de sa luminosité maximale.

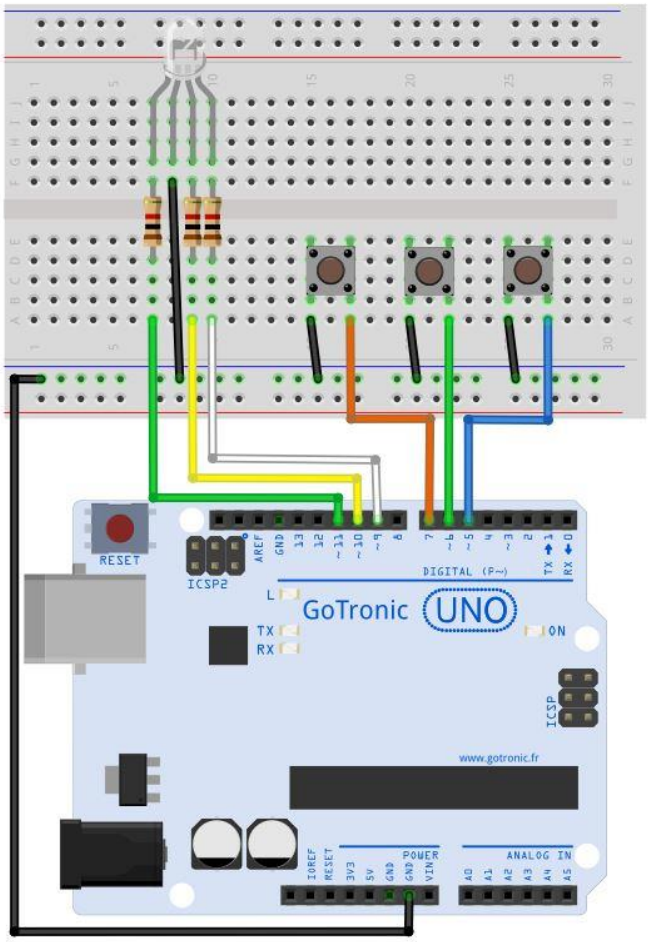
Liste des composants

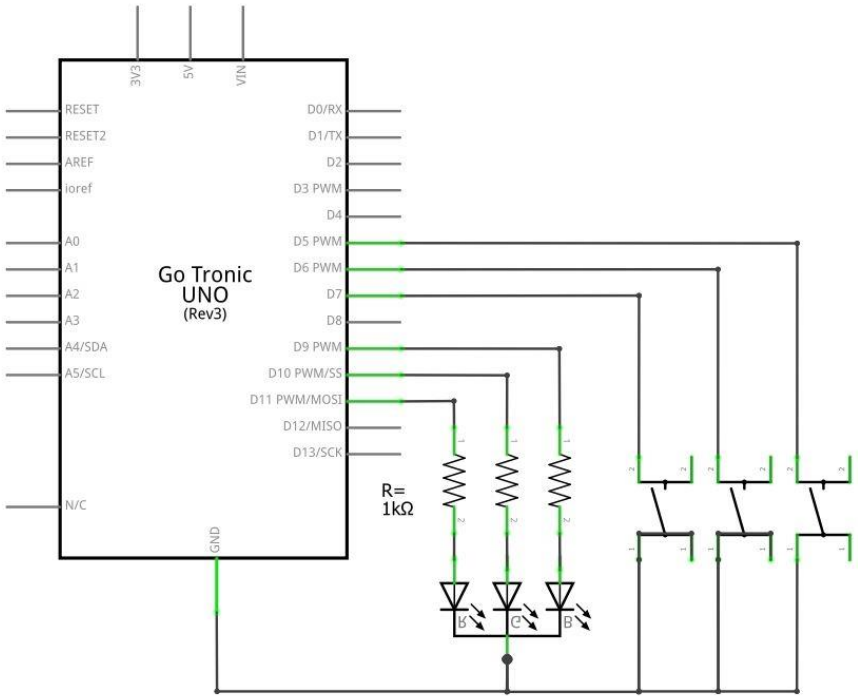
1 led RVB, 3 boutons-poussoirs et 3 résistances de 220 ohms.

Nouveau composant utilisé

Une led RVB : c'est une led constituée de 3 leds avec cathode (-) commune. Le pilotage de chaque couleur en PWM permet d'obtenir une multitude de couleurs (en fonction de la quantité de chaque couleur).

Montage et schéma





Le code

```
int BPR = 7;
int BPV = 6;
int BPB = 5;
int LEDR = 11;
int LEDV = 10;
int LEDB = 9;
int INTR = 64;
int INTV = 64;
int INTB = 64;
int ETATBPR;
int ETATBPV;
int ETATBPB;

void setup() {
// Déclaration des entrées et sorties
  pinMode(LEDR, OUTPUT);
  pinMode(LEDV, OUTPUT);
  pinMode(LEDB, OUTPUT);
  pinMode(BPR, INPUT);
  pinMode(BPV, INPUT);
  pinMode(BPB, INPUT);
  digitalWrite(BPR, HIGH);
  digitalWrite(BPV, HIGH);
  digitalWrite(BPB, HIGH);
}

void loop() {
// On lit l'état des 3 BP
  ETATBPR = digitalRead(BPR);
  ETATBPV = digitalRead(BPV);
  ETATBPB = digitalRead(BPB);
// On augmente la luminosité de la couleur d'1/4 à chaque appui sur un BP
  if (ETATBPR == 0) {INTR = INTR + 64;}
  if (ETATBPV == 0) {INTV = INTV + 64;}
  if (ETATBPB == 0) {INTB = INTB + 64;}
// Si on arrive à 256, on envoie la valeur max 255
  if (INTR == 256) {INTR = 255;}
  if (INTV == 256) {INTV = 255;}
  if (INTB == 256) {INTB = 255;}
// Si on dépasse 256, on retourne à 0
  if (INTR > 256) {INTR = 0;}
  if (INTV > 256) {INTV = 0;}
  if (INTB > 256) {INTB = 0;}
// On envoie en PWM l'intensité pour chaque couleur
  analogWrite(LEDR, INTR);
  analogWrite(LEDV, INTV);
  analogWrite(LEDB, INTB);
  delay (150);
}
```

Fonctionnement

- on crée 3 variables INTR, INTV et INTB correspondant à l'intensité de chaque couleur que l'on initialise à ‘‘64’’ en les déclarant.

- on incrémente de 64 (25% de 256) la variable associée à chaque appui d'un bouton-poussoir, et on la met à 0 si on dépasse 256 (100% de la luminosité).

- on pilote ensuite chaque couleur en PWM en fonction de la valeur des variables INTR, INTV et INTB.

Un problème ?

- vérifier que votre programme a bien été transféré dans votre carte Uno.

- vérifier le sens de la led RGB et des boutons-poussoirs.

Montage 11 : Télécommande et récepteur IR

Présentation

Ce montage permet d'utiliser la télécommande infrarouge pour envoyer des données à la carte et les afficher sur le moniteur série.

Installer la bibliothèque (voir plus bas). Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-11. Téléverser le programme dans la carte Uno. Ouvrir le moniteur série : Outils > Moniteur série et configurer la vitesse sur 9600 bauds (en bas à droite de la fenêtre).

Liste des composants

Télécommande et récepteur infrarouge

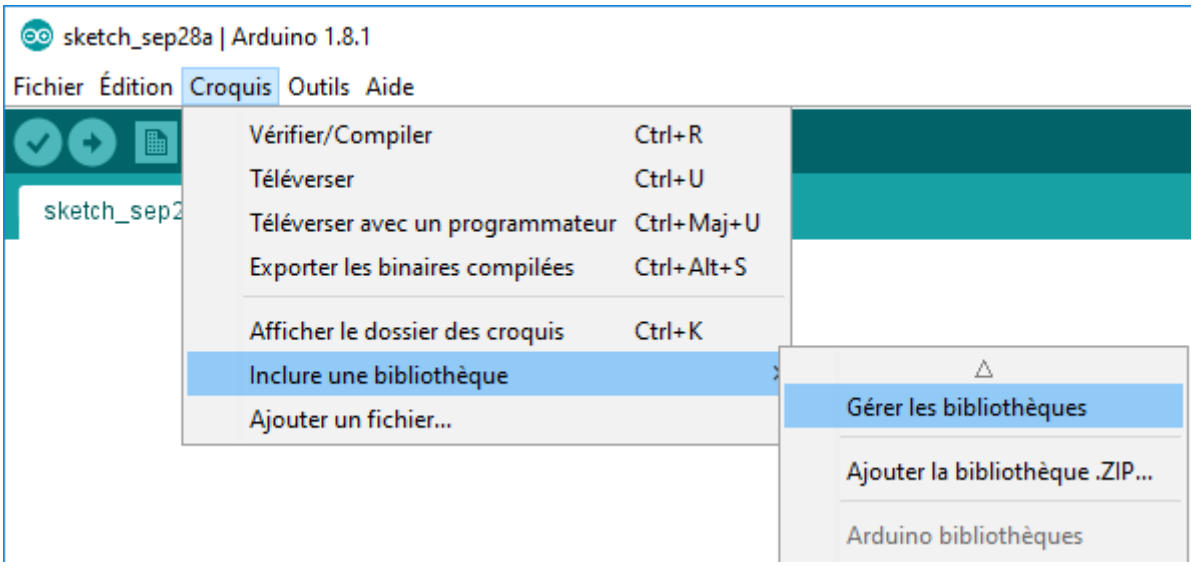
Nouveau composant utilisé

Le récepteur infrarouge est un capteur à trois broches : Vcc, GND et DATA. La lumière infrarouge captée est transformée en signal et envoyé par la broche DATA.

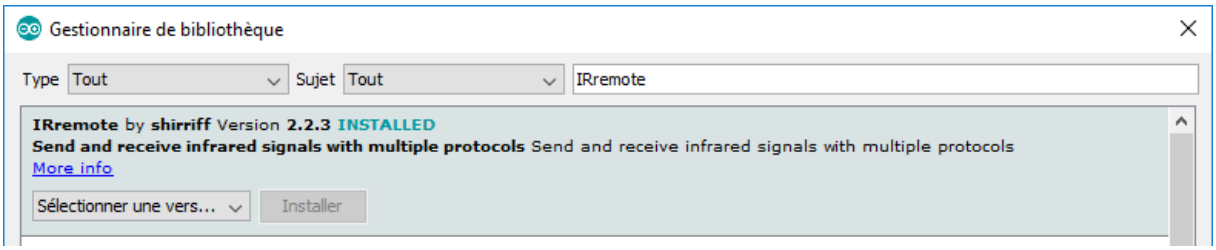
Installation de la bibliothèque

Pour que la carte puisse interpréter le signal reçu sur le capteur, il faut installer la bibliothèque « IRremote ».

Pour cela, allez dans : *Croquis* > *Inclure une bibliothèque* > *Gérer les bibliothèques*

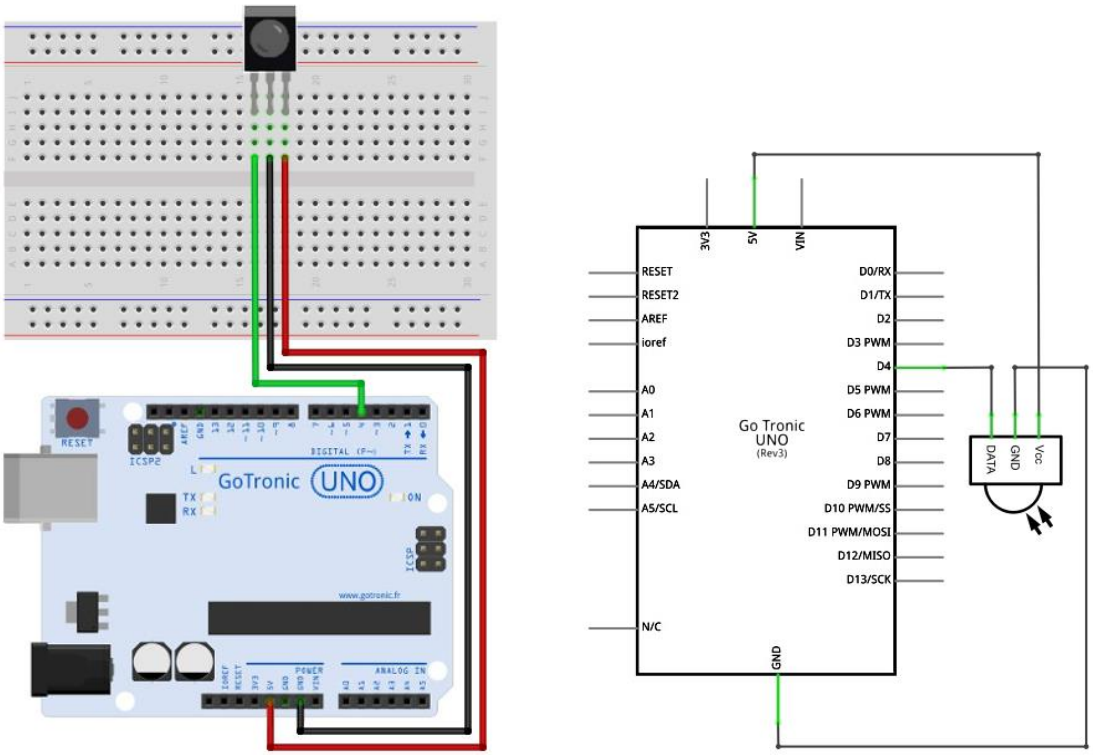


Recherchez ensuite « IRremote » dans le champ de recherche et cliquez sur le bouton pour l'installer :



La bibliothèque est installée.

Montage et schéma



Le code

```
#include <IRremote.h> // importation de la bibliothèque infrarouge

int pinIR = 4;
IRrecv irrecv(pinIR);
decode_results results;

void setup() {
  Serial.begin(9600);                    // Démarrer la communication série
  irrecv.enableIRIn();                   // Démarrer le récepteur IR
  Serial.println("Pret a demarrer");     // Afficher un message sur le moniteur série
}

void loop() {
// Si une valeur est reçue, on l'affiche sur le moniteur série (en hexadécimale)
  if (irrecv.decode(&results)) {
    Serial.print("code IR = ");
    Serial.println(results.value, HEX);
    irrecv.resume();
  }
}
```

Fonctionnement

- On importe la bibliothèque IRremote via l'instruction *#include <IRremote.h>* et on indique la broche sur laquelle la broche DATA du capteur est connectée (ici la broche 4).

- On initialise le moniteur série à 9600 bauds pour permettre la communication série et on affiche « Prêt à démarrer ».

- La boucle principale teste les données reçues par le capteur. Lorsqu'une nouvelle donnée est reçue, elle est affichée sur le moniteur série.

Un problème ?

- Vérifier que votre programme a bien été transféré dans la carte Uno.

- Vérifier que la protection de la pile de la télécommande a bien été enlevée et qu'elle n'est pas vide.

- Vérifier le sens et les broches du récepteur IR.

Pour aller plus loin

- La variable results.value contient le code infrarouge reçu. Vous pouvez donc exécuter des actions en fonction du code reçu : Allumer/Eteindre des leds, émettre un bruit, etc.

Montage 12 : Matrice à leds

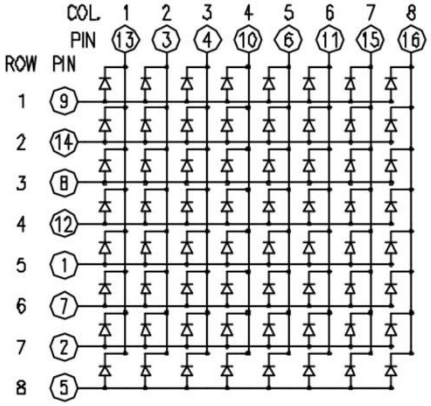
Présentation

Ce montage permet d'utiliser une matrice à leds.

Réaliser le montage à l'aide du schéma et de la liste ci-dessous. Brancher votre carte Uno en USB, ouvrir le logiciel Arduino® et ouvrir le programme : Fichier > Exemples > Gotronic > Montage-12. Téléverser le programme dans la carte Uno. Une des leds s'allume. La led allumée change à chaque appui sur le bouton-poussoir.

Liste des composants

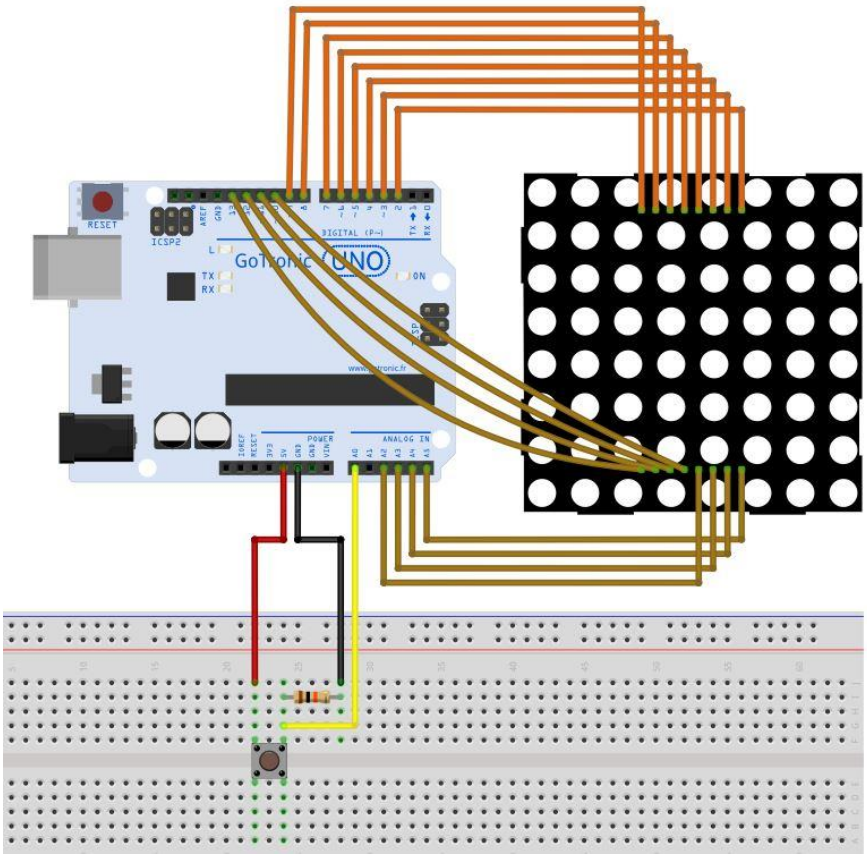
1 matrice à leds, 1 bouton-poussoir et une résistance de 10 kohms.

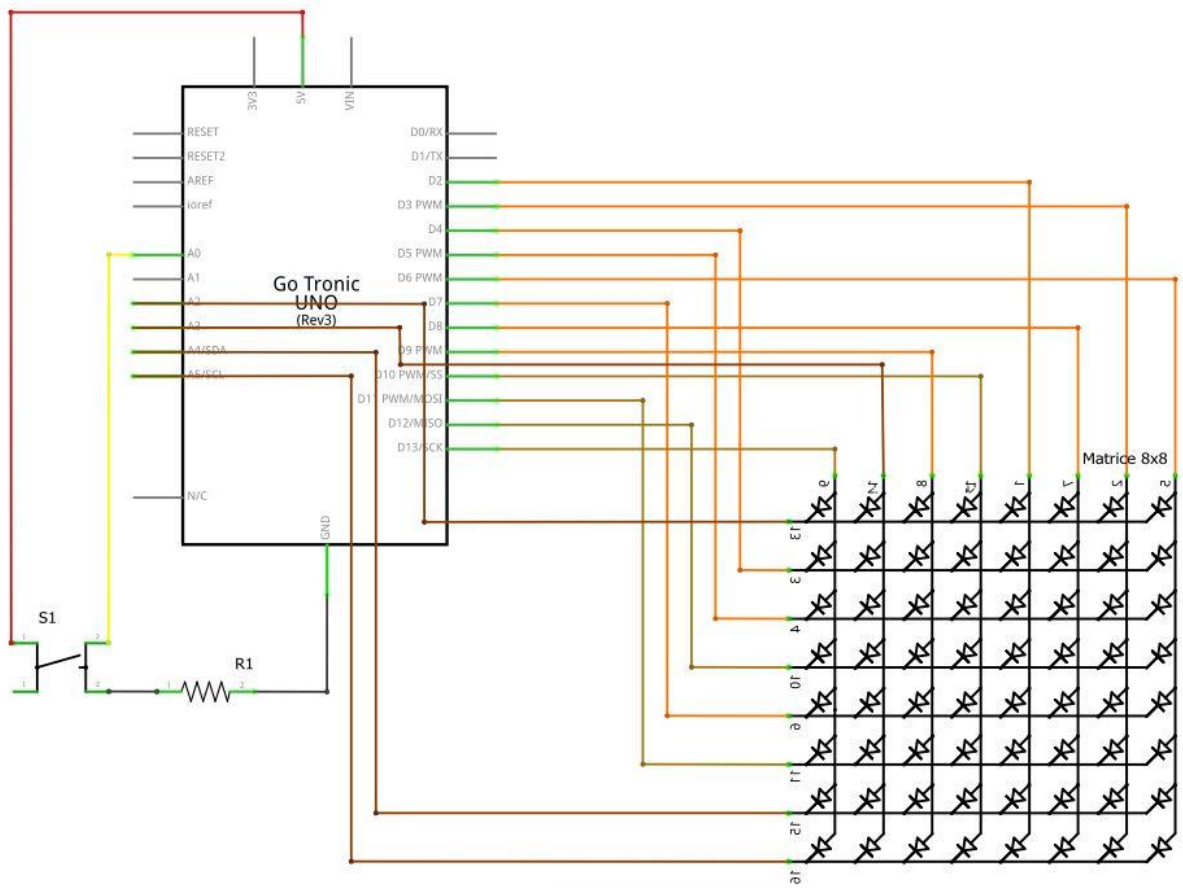


Nouveau composant utilisé

La matrice led est un ensemble de leds disposées de façon à ce que les anodes des leds d'une même rangée soient communes, de même que les cathodes des leds d'une même colonne. De cette manière, il est possible de jouer sur l'alimentation des rangées et des colonnes pour choisir la led à allumer.

Montage et schéma





Le code

```
// tableaux contenant les broches des auxquelles sont raccordées les lignes et la colonne
const int row[8] = { 2, 7, 19, 5, 13, 18, 12, 16 };
const int col[8] = { 6, 11, 10, 3, 17, 4, 8, 9 };
int pixels[8][8];                    // Tableau de pixels 2 dimensions

int x = 5;                           // Variables pour la position du curseur
int y = 5;

int buttonPin = 14;                  // Déclaration du bouton
bool buttonPousse = 0;

void setup() {
  // Initialisation des entrées et sorties
  for (int thisPin = 0; thisPin < 8; thisPin++) {
    pinMode(col[thisPin], OUTPUT);
    pinMode(row[thisPin], OUTPUT);
    pinMode(buttonPin, INPUT);
    digitalWrite(col[thisPin], HIGH);
  }
```

```
  // Initialisation des pixels de la matrice :
  for (int x = 0; x < 8; x++) {
    for (int y = 0; y < 8; y++) {
      pixels[x][y] = HIGH;
    }
  }
}

void loop() {
// Si le bouton est poussé, appel de la fonction pour donner une nouvelle position aléatoire au pixel
  if (buttonPousse == 0) {
    if (digitalRead(buttonPin) == HIGH) {
      Aleatoire();
      buttonPousse = 1;
    }
  }

  if (digitalRead(buttonPin) == LOW) {
    buttonPousse = 0;
  }

  // Appel de la fonction pour afficher le pixel à sa nouvelle position
  refreshScreen();
}

void Aleatoire() {
  pixels[x][y] = HIGH;
  x = 7 - random(0, 7);
  y = random(0, 7);
  pixels[x][y] = LOW;
}

void refreshScreen() {
  for (int thisRow = 0; thisRow < 8; thisRow++) {
    digitalWrite(row[thisRow], HIGH);
    for (int thisCol = 0; thisCol < 8; thisCol++) {
      int thisPixel = pixels[thisRow][thisCol];
      digitalWrite(col[thisCol], thisPixel);
      if (thisPixel == LOW) {
        digitalWrite(col[thisCol], HIGH);
      }
    }
    digitalWrite(row[thisRow], LOW);
  }
}
```

Fonctionnement

- Déclaration des variables et des broches sur lesquelles sont branchées les rangées et colonnes de la matrice.

- Initialisation des entrées (bouton) et sorties (matrice).

- Lorsque le bouton-poussoir est enfoncé, le programme appelle une fonction qui attribue des coordonnées (X et Y) aléatoires.

- La deuxième fonction est appelée en continu et allume le point de la matrice avec les coordonnées X et Y

Un problème ?

- Vérifier que votre programme a bien été transféré dans la carte Uno.

- Vérifier les connexions de la matrice et du bouton-poussoir.

Pour aller plus loin

- Vous pouvez utiliser le circuit 74HC595 pour réduire le nombre de broches requises sur la carte.

Aide et dépannage

Le montage ne fonctionne pas :

- Vérifier dans un premier temps le sens des composants liés au problème. Par exemple, vérifier le sens de la led si elle ne s'allume pas.

- Vérifier si les composants sont insérés au bon endroit. Un décalage d'une rangée empêche le fonctionnement du montage.

- Vérifier que votre programme est téléversé correctement et que vous avez sélectionné la bonne carte en fonction du montage (Menu Outils > Type de carte). Vous pouvez téléverser le programme une seconde fois en cas de doute.

- Vérifier l'alimentation de votre montage (5V et GND de la carte Uno)

- Essayer de changer le ou les jumpers sur la partie du montage qui ne fonctionne pas.

- En cas de doute, vous pouvez démonter le montage complet et recommencer étape par étape.

Le montage ne fonctionne toujours pas :

Vous pouvez envoyer un e-mail à l'adresse sav@gotronic.fr en indiquant les éléments suivants :

- Code article du kit acheté
- Numéro du montage concerné
- Type de problème rencontré (montage fonctionne partiellement, pas du tout, problème de téléversement, etc.)
- Joindre une photo du montage si possible

Nous vous répondrons dans les plus brefs délais afin de résoudre votre problème.

Les illustrations et schémas de cet ouvrage ont été réalisés à l'aide du logiciel Fritzing.