C5-160551-INFO
Objet Connectés : Programmation microcontrôleur
TD2

©B. Besserer

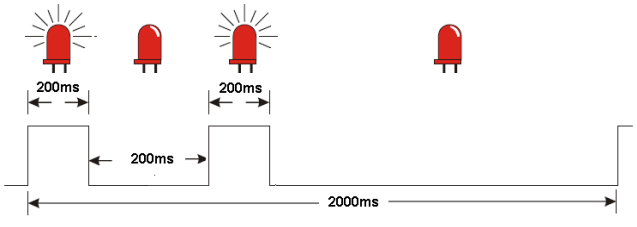
année universitaire 2021-2022

1 Chaîne de production de code

1. Du coté de la cible (microcontrôleur), dans quel type de mémoire le **code** est il téléchargé ?
   ☐ RAM ☒ ROM (en général mémoire flash) ☐ registres
2. Le programmeur peut-il spécifier le type de mémoire à utiliser pour les données (constantes,variables) ?, comment ?
   ☐ Non, les données sont toujours dans la RAM
   ☐ Oui, le programmeur peut utiliser des classes de stockage et des modificateurs dans le code source
   ☒ Oui, mais il faut intervenir sur le code assembleur (pas possible avec un langage de haut niveau)
3. (Recherche sur internet) Le framework Arduino peut-être utilisé :
   ☒ Avec la plupart des processeurs ATMEL à l'exclusion de ceux avec un noyau ARM
   ☒ Avec la plupart des processeurs ATMEL y compris ceux avec un noyau ARM
   ☐ Avec certains processeurs de la famille MSP430 de Texas Instrument
   ☒ Avec les processeurs PIC de microchip
   ☒ Avec les processeurs core i3 et i5 de Intel

2 Entrées / Sorties

1. Quelles fonctions (en utilisant le framework Arduino) permettent généralement de récupérer des informations en provenance d'un capteur connecté au microcontrôleur
   ☐ digitalRead et digitalWrite ☐ digitalWrite et AnalogWrite
   ☒ digitalRead et AnalogRead ☐ digitalWrite et AnalogRead
2. Complétez le programme (uniquement la fonction `loop()`) qui doit fait continuellement clignoter la LED selon la séquence suivante. La LED utilisée sera sur une GPIO nommée LED_BUILTIN, ce qui correspond à la LED intégrée aux platines microcontrôleur. Celle-ci s'allume lorsqu'on écrit un état HIGH sur ce GPIO. Utilisez la fonction delay(). Max 4 lignes de codes, mais cela peut-être plus court.



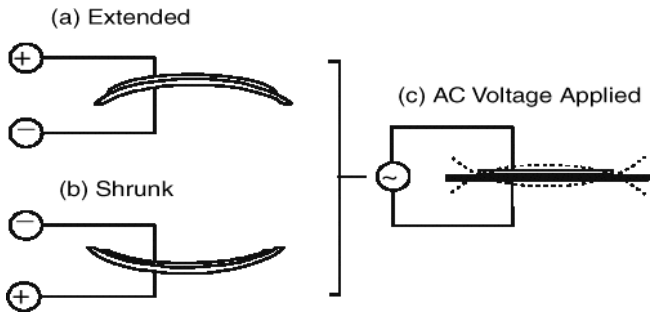

```
setup()
{
pinMode(LED_BUILTIN, OUTPUT);
}

loop()
{




}
```

3. Un buzzer piézoélectrique est constitué d'une lamelle d'un matériau qui est déformé lorsqu'on applique une tension, comme illustré ci-contre. A ce moment, on entend un "clic". En appliquant la polarité inverse, on entend à nouveau un "clic". Pour générer un son continu, il faut donc faire osciller la lamelle en inversant la polarité à la fréquence de son choix (dans le domaine de l'audible, disons de 500 à 5000 Hz).



Écrivez en syntaxe Arduino/Wiring le code nécessaire pour obtenir un son continu (**avec le maximum de puissance**) d'une fréquence d'approximativement 500 Hz. On suppose que le buzzer est relié entre les sorties 14 et 15 d'un microcontrôleur, soit les définitions suivantes :

```
#define BUZZER1 14    // Borne de sortie
#define BUZZER2 15    // Borne de sortie
```

L'utilisation de la fonction `delay()` est autorisée.

3 Mesures analogiques

Considérez le code ci-dessous.

```
void setup()
{
  Serial.begin(9600);
}
void loop()
{
  min = 4095;
  for(int count = 0 ; count < 10 ; count++)  // on effectue 10 mesures
  {
     value = analogRead(A0);
     delay(5);                 // pause très courte, mais nécéssaire si le capteur reagit lentement
     if (value < min)
       min = value;
  }
  Serial.print(value);
  Serial.print("  ");
  Serial.println(min);
  delay(950);
}
```

1. Concernant les variables `min` et `value`, ajouter les déclarations au bon endroit (variable globale ou locale) et en précisant le type.

2. Est-ce que la broche A0 est utilisée en entrée ou en sortie ? ☒ Entrée ☐ Sortie

3. Pour quelle raison la "broche" A0 n'est-elle pas initialisée dans le setup() à l'aide de la fonction pinMode() ?
   ☒ La broche est toujours une entrée
   ☐ La broche est toujours une sortie
   ☐ La broche n'est pas utilisé comme un GPIO en mode entrée/sortie digital, c'est une fonction secondaire de la broche qui est utilisée, l'initialisation avec pinMode() n'est pas nécessaire.

4. Quelle est la fréquence à laquelle s'affiche les relevés de mesure sur le terminal qui est à l'écoute sur le port série et quelles sont les informations qui sont affichées ?

5. Quelle est le comportement du programme :
   ☐ Le programme effectue 10 mesures, détermine le minimum, affiche le resultat puis sarrête
   ☒ Le programme effectue des salves de 10 mesures, détermine le minimum de ces 10 mesures, et répète l'opération toutes les secondes environ.
   ☐ Le programme effectue des mesures de façon continue, et indique le minimum des 10 dernières mesures.
   ☐ Le programme effectue des mesures tant qu'une mesure ne soit pas supérieure à 4095.

6. Pour quelle raison la variable `min` est elle initialisée à la valeur 4095 ?
   La lecture analogique sur la broche. Bien qu'elle soit limitée à la résolution du convertisseur analogique-numérique (0-1023 pour 10 bits ou 0-4095 pour 12 bits). Type de données : int.

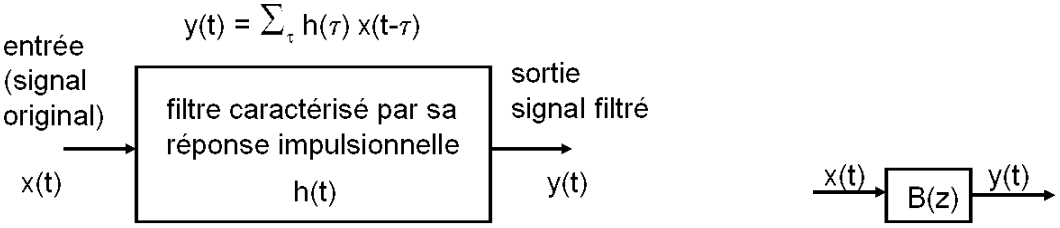
7. Les sketchs Arduino sont plus ou moins portables d'une plateforme à une autre, mais pas toujours. Est-ce que la valeur d'initialisation (ici 4095) sera la même si la plate-forme cible est un **Arduino uno** ou un **ESP32** ? ☐ Oui ☒ Non

8. Modifier maintenant ce code afin de garder constamment dans 2 variables `min1` et `min2` les 2 valeurs les plus faibles de chaque séquence de mesures.

4 Preparation au TP : Filtrage et moyenne glissante (ou moyenne mobile)

Pas de réponses à donner sur moodle. Réaliser ces exercices vous donnera un avantage considérable lors des travaux pratiques.

Le filtrage numérique est une traitement appliqué à un signal échantillonné ou signal discret (formé d'une suite d'échantillons). Une fonction classique d'un filtre est par exemple de lisser un signal (filtre passe-bas). Un signal d'entrée $x(t)$ est donc modifié par le filtre pour fournir un signal de sortie $y(t)$. La modification effectué par le filtre est une convolution discrète (pour les filtres dit filtre FIR — Finite Impulse Response = Réponse Impulsionelle Finie — ou filtre non récursif). Pour ces filtres, la fonction de transfert $B(z)$ est un polynôme. Beaucoup de traitements numériques des signaux font appel à des produits de convolution discrète se traduisant par des **sommes de produits**. Par exemple, interpoler des valeurs entre 2 instants d'échantillonnage génère un nouvel échantillon de sortie par addition pondérée d'échantillons du signal d'entrée.



Prenons un exemple simple :

$$s_n = a_0 \times e_n + a_1 \times e_{n-1} + a_2 \times e_{n-2}$$

Avec $a_0 = 0.25, a_1 = 0.5, a_2 = 0.25$, cette opération réalise une moyenne pondérée de 3 échantillons consécutifs du signal d'entrée. Vous devez écrire le code qui effectue ce traitement.
Pour tester l'implémentation de l'algorithme on place 100 échantillons d'un signal dans un tableau statique (`data_in[]`). Chaque échantillon est stocké sur 16 bits (10 bits significatifs) :

```
const unsigned int data_in[] =
{512,531,443,390,858,351,656,363,778,409,287,713,172,611,298,683,539,855,313,565,420,759,294,725,462,...};
```

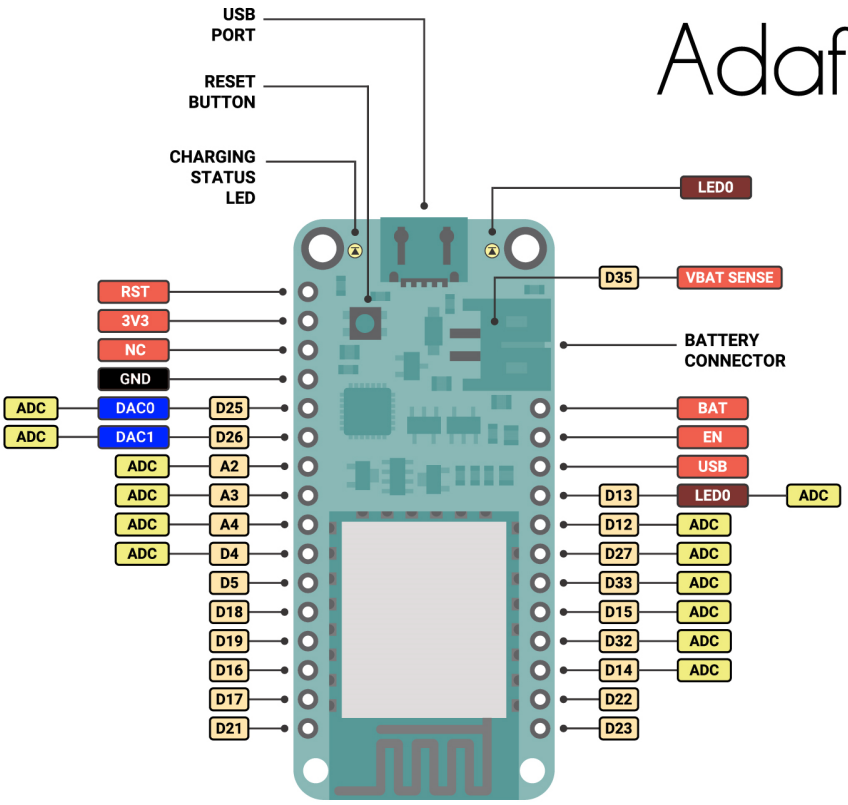
Les données traités par le logiciel doivent être stockées dans un tableau de sortie (`data_out`).

1. Écrivez la boucle de traitement sans tenir compte des "effets de bords" au début et à la fin de tableau. Prêtez particulièrement attention :
   - à la déclaration des différentes variables que vous utilisez
   - aux opérateurs que vous utilisez pour les calculs.

2. Concernant les "effets de bords", proposez une solution pour traiter ces cas particuliers.

3. Pourrait-on modifier le code pour supprimer toute opération de type multiplication et division ?

Adafruit Huzzah32



DO NOT USE D6 TO D11

PWM IS ENABLED ON EVERY DIGITAL PIN

ADC ON PINS D4, D12, D13, D14, D15, D25, D26, D27
CAN BE READ ONLY WITH WI-FI NOT STARTED