Laboratoire 1
Oscilloscope et générateur de formes d'onde : Montage avec microcontrôleur

Michelle Janusz, Camille Morin, Gabriel Gagné et Benoit Gosselin

Université Laval - Hiver 2023

Préparation

- ✓ Procurez-vous le kit de laboratoire du cours.
- ✓ Télécharger l'IDE Arduino sur votre ordinateur et explorer pour comprendre son fonctionnement et être prêt à l'utiliser au laboratoire.
- ✓ Assurez-vous d'avoir une version de Matlab fonctionnelle sur votre ordinateur.

Répartition des notes (5% de la note finale)

- ✓ Démonstration des codes et des circuits fonctionnels aux assistants (50%)
- ✓ Rapport (50%)

Objectifs

- ✓ Familiarisation avec les concepts de base d'acquisition de signaux
- ✓ Familiarisation à l'environnement de développement de systèmes électroniques mixtes
- ✓ Réalisation et validation de montages d'oscilloscope et générateur de formes d'ondes maison avec le microcontrôleur Arduino Uno

Description

Ce travail pratique a pour but de vous introduire aux outils de développement de systèmes électroniques mixtes (analogiques et numériques). Dans un premier temps, vous réaliserez l'acquisition de signaux analogiques grâce aux convertisseurs analogiques-numériques de l'Arduino Uno. Par la suite, vous utiliserez ce même microcontrôleur pour générer des formes d'ondes carrées et sinusoïdales à diverses fréquences. Finalement, votre système d'oscilloscope maison sera complété par la capture et l'affichage des signaux en temps-réel sur MATLAB.

Partie 1 : Acquisition et génération de signaux avec l'Arduino Uno

Cette partie du travail s'effectue au PLT-3101 et consiste à réaliser l'acquisition de signaux à partir de l'Arduino. Rapportez les valeurs et figures demandées (celles **en gras** dans le protocole) dans votre rapport. Au besoin, référez-vous à la documentation en ligne de l'Arduino : https://www.arduino.cc/reference/en/.

1. La programmation de microcontrôleur

Le langage de programmation de l'Arduino emprunte des éléments de variables et structures des langages C/C++. Vous trouverez dans sa documentation en ligne tous les détails de son interface de programmation (API). Plusieurs fonctions sont prédéfinies, vous permettant d'utiliser ses périphériques, effectuer des opérations mathématiques, etc.

1.1. Démarrez l'environnement de développement (IDE) *Arduino* sur votre ordinateur et lancez un nouveau projet. Vous pouvez vous procurer l'IDE ici : https://www.arduino.cc/en/software

1.2. Remarquez que le projet est initialisé avec 2 fonctions *void* : *setup()* et *loop()* (Figure 1). Prenez note des commentaires dans le projet de base (lignes commençant par //) pour comprendre leur utilité. Au besoin, consultez la documentation de l'Arduino. **Dans votre rapport, pour chacune de ces fonctions, expliquez son rôle et donnez un exemple d'application ou de fonction à y écrire.**

2. Acquisition et transmission de signaux

L'Arduino Uno possède 6 entrées analogiques (pins A0 à A5) que vous pourrez utiliser pour convertir un signal analogique en numérique pour ensuite l'afficher à l'écran.

2.1. Dans un premier temps, configurez la communication sérielle avec l'ordinateur grâce à la fonction *Serial.begin()*. Celle-ci ne doit être appelée qu'une seule fois (voir question précédente et documentation au besoin) et prend en argument le débit (*baudrate*) de la communication numérique. Utilisez un *baudrate* de 115200 bits par seconde.

2.2. Configurez la *pin* A0 en tant qu'entrée. Pour ce faire, vous devrez utiliser la fonction *pinMode()*. Veuillez noter que la variable « A0 » est déjà définie dans l'environnement de développement pour désigner la pin correspondante. Cette fonction doit être appelée dans la fonction *void setup()* de votre programme. **Expliquez pourquoi.**

2.3. Une fois votre *pin* A0 configurée, vous pouvez utiliser la fonction *analogRead()* pour en lire son contenu. Sachant que cette fonction devra être appelée périodiquement, écrivez cette fonction à l'endroit approprié de votre code. La valeur de la *pin* A0 doit être enregistrée dans une variable de type *int* nommée *data*.

2.4. À la ligne suivant immédiatement la lecture de A0, envoyez le contenu de votre variable *data* dans le port de communication série grâce à la fonction *Serial.println()*.

2.5. Afin d'imposer la fréquence de lecture de l'ADC, ajoutez un délai dans la boucle d'acquisition grâce à la fonction *delay()*. En argument à la fonction, entrez la valeur permettant une fréquence d'échantillonnage de 10 Hz.

2.6. Pour valider le fonctionnement de l'acquisition, réalisez un diviseur de tension avec un potentiomètre et les *pins* d'alimentation *GND* et *5V* de l'Arduino. Connectez sa sortie à l'entrée A0.

2.7. Dans l'environnement de développement Arduino, assurez-vous que la connexion est faite avec votre carte Arduino Uno et utilisez l'outil *Tools > Serial Monitor* pour

afficher les valeurs lues à l'entrée A0. Faites varier le potentiomètre du diviseur de tension. **Quelle est la plage des valeurs numériques lues? Quelle information pouvez-vous déduire sur le convertisseur analogique-numérique de l'Arduino?**

2.8. Pour afficher temporellement la lecture de l'ADC, utilisez l'outil *Tools > Serial Plotter*. Faites varier le potentiomètre et **fournissez une capture d'écran du graphique *Serial Plotter*.**

2.9. Faites valider le fonctionnement de votre système d'acquisition par un des assistants du cours.

3. Générateur d'onde carrée

L'Arduino Uno possède des entrées/sorties numériques permettant de lire/écrire des niveaux logiques haut et bas (*HIGH* et *LOW* sont des variables prédéfinies dans l'API/interface de programmation Arduino). Deux types de sorties numériques sont disponibles sur l'Arduino : usage général et *PWM* (*pulse width modulation*). Nous utiliserons ici une sortie à usage général pour générer une forme d'onde carrée à une fréquence programmable.

3.1. Configurez l'une des *pins* numériques à usage général en tant que sortie.

3.2. La fonction *digitalWrite()* permet d'envoyer un niveau *HIGH* ou *LOW* à une *pin* de sortie. En passant périodiquement d'un niveau haut à bas, et vice-versa, on peut ainsi générer une onde carrée à une fréquence voulue.

3.2.1. Changez le paramètre de la fonction *delay()* de votre boucle principale pour lui imposer une fréquence de 1 kHz.

3.2.2. Sur la sortie numérique configurée, générez une onde carrée à 6 fois la période de la boucle principale. Attention : vous devez conserver le code d'acquisition analogique-numérique. (Indice : vous pouvez y arriver de plusieurs manières, entre autres avec l'utilisation d'un compteur incrémental et d'énoncés *if/else*)

3.2.3. **Quelle est la fréquence théorique attendue de l'onde carrée?**

3.2.4. **Mesurez la fréquence de votre onde carrée grâce à l'oscilloscope de votre poste de travail et rapportez sa valeur dans votre rapport.**

3.2.5. **Est-ce que la fréquence mesurée correspond à la fréquence théorique? Sinon, expliquez pourquoi et BONUS : proposez une solution pour régler ce problème!**

3.2.6. **Connectez la sortie numérique à l'entrée analogique A0 et montrez une capture d'écran du Serial Plotter dans votre rapport.**

3.3. **Faites valider votre code et le fonctionnement de votre générateur d'onde carrée par un des assistants du cours.**

4. Générateur d'onde sinusoïdale

Il existe plusieurs méthodes pour générer une onde sinusoïdale. Ces méthodes varient en complexité d'implémentation et en qualité du signal généré. L'une des méthodes les plus simples est de passer une onde carrée à travers un réseau de 3 filtres passe bas RC.

4.1. Le réseau de filtres RC suggéré ici consiste en une série de 3 filtres passe-bas passifs, chacun isolé des autres par un circuit suiveur (buffer). Les filtres partagent la même fréquence de coupure, qui doit correspondre à la fréquence fondamentale de votre onde carrée. La Figure 1 présente les connexions pour un suiveur de tension utilisant votre amplificateur LM358. **Tracez le circuit au complète et donnez la valeur des composantes RC choisies. Expliquez pourquoi il est recommandé d'avoir un suiveur qui sépare les trois étages. Indice : considérer les diagrammes de Bode après chaque filtre avec et sans le circuit buffer.**

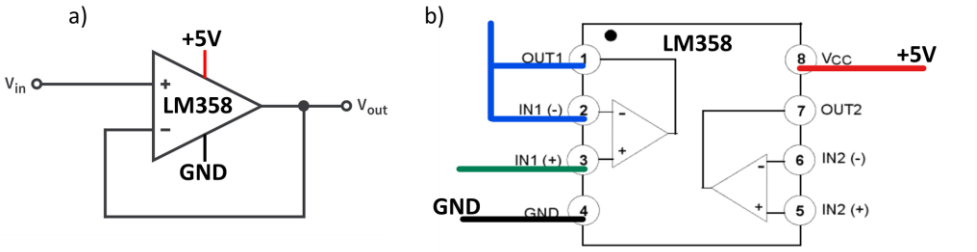

Figure 1 : (a) Schéma de circuit d'un suiveur de tension. (b) Suiveur de tension implémenté en utilisant le boîtier amplificateur LM358.

4.2. Implémentez le circuit sur votre *breadboard*, un filtre RC à la fois, chacun séparé par un suiveur (utilisez en tout 2 suiveurs). Assurez-vous que l'alimentation de votre amplificateur est égale ou supérieure à l'amplitude de votre onde carrée.

4.2.1. La première résistance et le premier condensateur créent la forme d'onde classique de charge et de décharge du condensateur. **Observez l'effet du filtre sur votre onde carrée à l'oscilloscope de votre poste de travail et montrez une capture d'écran dans votre rapport. Commentez vos observations**.

4.2.2. Le deuxième filtre RC prend la forme d'onde exponentielle et la convertit en une forme d'onde triangulaire. Il redresse les formes d'onde exponentielles paraboliques afin qu'elles apparaissent comme une pente ascendante droite du côté de la charge et une pente descendante droite du côté de la décharge. **Observez l'effet du deuxième étage à l'oscilloscope et montrez une capture d'écran dans votre rapport. Commentez vos observations.**

4.2.3. Le troisième filtre RC remodèle ensuite les ondes triangulaires, en atténuant leur rectitude et en les rendant plus courbes. Après le troisième réseau RC, la forme d'onde de sortie ressemble plus ou moins à une onde sinusoïdale. **Observez l'effet du troisième étage à l'oscilloscope et montrez une capture d'écran dans votre rapport. Commentez vos observations.**

4.3. Connectez la sortie de votre filtre RC à l'entrée analogique A0 et observez l'acquisition de l'Arduino via *Serial Plotter*. **Observez-vous des différences par rapport à la mesure sur l'oscilloscope? Discutez.**

4.4. **Faites valider le fonctionnement de votre générateur d'onde sinusoïdale par un des assistants du cours**.

5. Distorsion harmonique

L'onde sinusoïdale créée par le filtrage passe-bas de l'onde carrée n'est pas parfaite. Afin d'évaluer la qualité du signal généré, il est intéressant d'observer son contenu spectral.

5.1. À l'aide du générateur de forme d'onde de votre poste de travail, générez une onde sinusoïdale à la même fréquence (mesuré) que l'onde carrée de l'Arduino. Afin d'observer le contenu spectral: Envoyez votre signal dans l'oscilloscope, et sélectionnez la fonction MATH. Sous l'onglet opération, sélectionnez FFT. Assurez-vous que la résolution temporelle de l'oscilloscope est réglée pour afficher plusieurs périodes de l'onde. **Observez son contenu spectral à l'oscilloscope et rapportez une capture d'écran dans votre rapport.**

5.2. Mesurez maintenant votre onde sinusoïdale (onde carrée filtrée) avec l'oscilloscope et affichez son contenu spectral. **Montrez la capture d'écran dans votre rapport. Commentez la différence entre les contenus spectraux de votre onde sinusoïdale et de celle du générateur de votre poste de travail**.

5.3. Le taux de distorsion harmonique (THD ou *Total Harmonic Distortion*) donne le ratio du voltage RMS équivalent des fréquences harmoniques sur le voltage RMS de la fréquence fondamentale (voir Équation 1 ci-dessous). Cette mesure est typiquement utilisée pour caractériser la linéarité des systèmes en évaluant la déformation induite sur une entrée sinusoïdale. On utilise ici le *THD* pour évaluer la qualité du signal sinusoïdal généré par votre Arduino et filtre RC. **Calculez le taux de distorsion harmonique pour les deux ondes, soit celle générée par votre système, et celle du générateur de forme d'onde de votre poste de travail. Comparez et commentez les résultats dans votre rapport.**

$$\boldsymbol{THD} = \frac{\sqrt{V_{1\_RMS}{}^2 + V_{2\_RMS}{}^2 + V_{3\_RMS}{}^2 + \cdots + V_{n\_RMS}{}^2}}{V_{0\_RMS}} \quad (1)$$

Où 0 correspond à la fréquence fondamentale et [1,2,3,…n] aux fréquences harmoniques subséquentes.

Le niveau dB des harmoniques peut être converti en V_{RMS} en utilisent l'équation 2.

$$\boldsymbol{V_{RMS}} = 10^{\frac{-A_{dB}}{20}} \quad (2)$$

Partie 2 : Prise de mesures via MATLAB

6. Capture de l'Arduino via MATLAB

L'outil *Serial Plotter* de l'IDE Arduino est pratique pour vérifier rapidement le fonctionnement de vos projets. Toutefois, il est très limité et ne permet pas la sauvegarde et la prise de mesure précise. À cette fin, nous pouvons utiliser MATLAB pour recevoir les données transmises par l'Arduino.

6.1. La fonction utilisée précédemment *println()* affiche les données en ASCII. Ce format n'est pas pratique pour la communication de bas niveau de Matlab. Il faut donc utiliser la fonction *write* qui envoie un octet (8 bits). La valeur fournie par l'ADC est sur 10 bits dans un *int*. Il faut donc envoyer la valeur en deux instructions et des balises pour reconnaitre le début et la fin de la transmission.

6.1.1. Envoyer la balise de début (0x01)

6.1.2. Envoyer les 8 LSB de la quantification

6.1.3. Envoyer les 8 MSB de la quantification. Note: Voir comment faire un décalage de bit ou bitshift.

6.1.4. Envoyer la balise de fin (0xFE)

6.2. Dans Matlab, reconstruisez les valeurs reçues à l'aide du code fourni.

6.3. Convertissez la valeur quantifiée en Volts.

6.4. Afficher la courbe en temps réel sur Matlab. N'oubliez pas d'ajuster les échelles et les noms des axes aux unités appropriées. Une vitesse d'actualisation de 500ms est acceptable si vous affichez plusieurs données à la fois. **Faites valider par un dépanneur. Donnez votre code Arduino et Matlab avec des commentaires dans votre rapport.**

6.5. **Donnez une capture d'écran d'un signal sinusoïdale d'amplitude de 2 volts, DC de 2.5 volts de fréquence de 100 Hz. Donnez une capture du même signal, mais cette fois-ci à l'oscilloscope. Commentez**.

Questions Post- laboratoire

Répondez aux questions suivantes:

1. Quelles sont la résolution et la précision d'un ADC 10-bits avec une plage de 0-5 volts comme dans l'Arduino? Quel serait l'avantage d'utiliser un ADC 16-bits?
2. Donnez la représentation d'une valeur de 2.7 volts à chaque étape de la chaine d'acquisition (du voltage mesuré par l'ADC jusqu'à l'affichage sur le graphique Matlab). Indice : N'oubliez pas la conversion en binaire; il y a en gros 6 étapes.
3. Expliquez pourquoi le réseau de 3 filtres RC utilisé pour convertir une onde carrée en onde sinusoïdale dans l'étape 4 n'est pas une solution idéale et proposez une solution alternative pour générer une onde sinusoïdale à l'aide de l'Arduino.

Rapport

1. Dans votre rapport, répondez aux questions posées de façon succincte. Il n'est pas nécessaire de rédiger une introduction et une conclusion. Les courbes et valeurs demandées (les informations en gras dans le protocole) devront être présentées

en spécifiant les numéros de question correspondants. De plus, placez la dernière page de cet énoncé comme 1re page de votre rapport et remplissez la. Fournissez le code de votre Arduino en annexe à votre rapport.

2. Votre rapport doit être remis en format électronique (PDF) sur le site du cours via monPortail avant la date limite [3 février 2023 à 17h00].
3. Nom du fichier pdf à remettre: nomdefamille#1_nomdefamille#2_Lab1.pdf

Laboratoire 1

Oscilloscope et générateur de formes d'onde : Montage avec microcontrôleur

	Nom	Matricule
1.	_____	_____
2.	_____	_____

Signature de l'assistant : _____

Date : _____