Arduino et Simulink/Matlab® un outil innovant à coût réduit pour le prototypage

Mohamed Ali ZERZRI

zerzriiset@gmail.com

Institut Supérieur des Etudes Technologiques (ISET) de Bizerte, Route Menzel Abderrahmen, Zarzouna, 7021, Bizerte, Tunisie

Résumé : L'enseignant des travaux pratiques (TP) en génie électrique affronte toujours une situation problématique qui se traduit par le fait qu'il fait appel soit à des simulations, soit à des maquettes pédagogiques très spécifiques, intégrant des fonctions figées généralement non modifiables et peu extensibles.
Dans cet article, nous avons essayé d'apporter une solution à cette situation en présentant une nouvelle approche qui est l'apprentissage par prototypage. Le choix s'est fixé sur la combinaison de l'environnement Simulink/Matlab et de la plateforme Arduino. Cette approche est très pertinente, car elle permet d'associer un puissant logiciel de simulation et une carte de prototypage permettant d'envisager des applications complexes. Cette expérience a rendu les TP plus interactifs. Notre travail s'articule autour de trois parties : la première partie est consacrée à la présentation de la carte Arduino et son environnement de développement, la deuxième partie est réservée à l'utilisation de l'environnement Simulink/Matlab avec Arduino et enfin la troisième partie est consacrée à la mise en œuvre simultanée de l'environnement Simulink/Matlab et la plateforme Arduino, et ce pour la réalisation d'une chaine d'acquisition et d'asservissement de température.

Mots clés : Arduino, Matlab, Simulink, Prototypage.

Introduction

Un retour d'expérience pédagogique, au sein du département génie électrique de l'ISET de Bizerte, montre que l'enseignement des TP de génie électrique est assuré essentiellement avec du matériel pédagogique consacré seulement à des simulations.
En effet, l'étudiant est appelé à suivre des instructions et des séries de manipulations qui vont être exécutées d'une manière automatique pour aboutir à des résultats concrets, lesquels résultats sont difficiles à interpréter de la part de l'étudiant qui n'arrive pas à les associer avec les connaissances théoriques acquises dans les séances de cours.
Ce mode d'apprentissage instaure un climat de monotonie aux séances de TP et empêche l'étudiant de s'inscrire dans une démarche globale d'apprentissage qui englobe aussi bien les connaissances théoriques que les connaissances opérationnelles.

Ainsi, nous avons essayé d'apporter plus d'interactivité aux séances de TP et surtout de laisser le soin à l'étudiant de monter, modéliser et simuler son propre système en utilisant les moyens de bord disponibles dans les labos de notre département (composants électroniques,

Article publié par EDP Sciences et disponible sur le site http://www.j3ea.org ou http://dx.doi.org/10.1051/j3ea/2013020

actionneurs… etc.). Ainsi le prototype réalisé est approprié au contenu théorique de la matière enseignée.

Le choix s'est fixé sur la combinaison de l'environnement Simulink/Matlab et la plateforme Arduino. En effet, cette combinaison nous donne un outil à coût réduit, performant, innovant et extensible pour réaliser des prototypes de systèmes pluridisciplinaires.

1. La plateforme Arduino

C'est une plateforme open-source d'électronique [1] programmée qui est basée sur une simple carte à microcontrôleur (de la famille AVR.), et un logiciel, véritable environnement de développement intégré (IDE) pour écrire, compiler et transférer le programme vers la carte à microcontrôleur.

Arduino peut être utilisé pour construire des objets interactifs indépendants (prototypage rapide), ou bien peut être connecté à un ordinateur pour communiquer avec ses logiciels.

1.1. Partie matérielle

En effet, une carte Arduino est généralement construite autour d'un microcontrôleur Atmel AVR (ATmega328 ou ATmega2560 pour les versions récentes ATmega168 ou ATmega8 pour les plus anciennes), et de composants complémentaires qui facilitent la programmation et l'interfaçage avec d'autres circuits. Chaque carte possède au moins un régulateur linéaire 5V et un oscillateur à quartz 16 MHz (ou un résonateur céramique dans certains modèles). Le microcontrôleur est pré-programmé avec un « bootloader » de façon à ce qu'un programmateur dédié ne soit pas nécessaire.

Treize versions des cartes de type Arduino ont été développées jusqu'au nos jours, à titre indicatif nous citons Arduino Uno et Arduino Mega2560 qui sont les plus utilisées dans l'enseignement et la recherche. Le tableau 1 résume les principales caractéristiques de ces deux cartes.

Arduino	**Uno**	**Mega2560**
Processeur	ATmega328P	ATmega2560
Flash (KB)	32	256
EEPROM(KB)	1	4
SRAM(KB)	2	8
Broches d'E/S numériques	14 dont **6 PWM**	54 dont **14 PWM**
Broches entrées analogiques	6	16
Type d'interface USB	ATmega8U2	ATmega8U2
Dimensions (mm)	68,6 x 53,3	101,6 x 53,3

Tab. 1 : comparatif entre Arduino Uno et Mega2560

1.2 Partie logicielle

L'environnement de programmation Arduino est en fait un Intégré de Développement (EDI) dédié au langage Arduino[1], le logiciel Arduino permet d'écrire les programmes, appelés « Sketch » de les compiler et de les transférer dans la carte Arduino à travers une liaison USB, il intègre aussi un moniteur de port série.

L'avantage du langage Arduino est qu'il est basé sur les langages C/C++ et supporte toutes les syntaxes standards du langage C et quelques-uns des outils du C++. En plus de très nombreuses librairies sont disponibles, gratuitement, pour communiquer avec le matériel connecté à la carte (Afficheurs LCD, Afficheurs 7 segments, capteurs, servomoteurs… etc.).

Pour écrire un programme avec le langage Arduino, il faut respecter certaines règles. En effet, l'exécution d'un programme Arduino s'effectue de manière séquentielle, c'est-à-dire que les instructions sont exécutées les unes à la suite des autres, le compilateur vérifie l'existence de deux structures obligatoires à tout programme Arduino qui sont :

- la partie initialisation et configuration des entrées/sorties ➔ la fonction setup ()
- la partie principale qui s'exécute en boucle ➔ la fonction loop ()

Par contre, la partie déclaration des variables est optionnelle. La figure1 montre l'interface graphique de l'EDI ainsi que la structure d'un programme réalisé avec le langage Arduino.

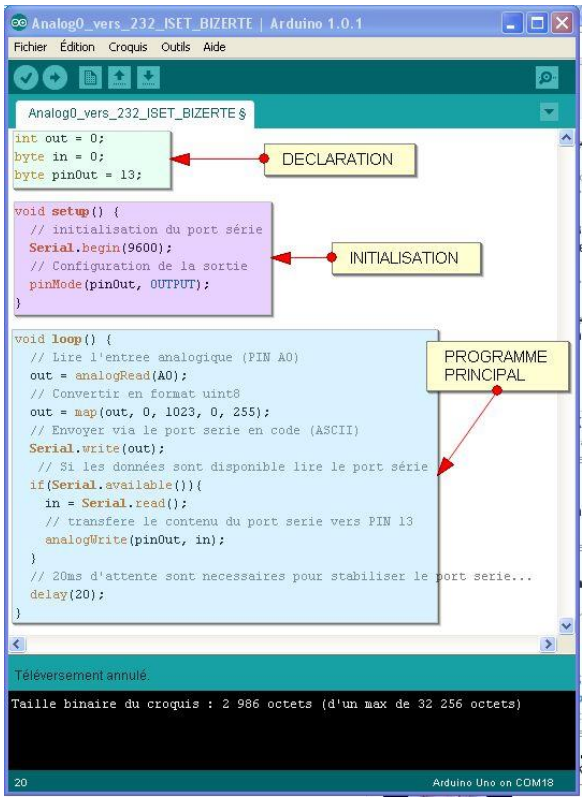

Fig. 1 : EDI Arduino et structure du programme

[1] Téléchargement de l'IDE Arduino : www.arduino.cc/en/main/software .

1.3. La carte Arduino Uno

Pour les essais pratiques et le prototypage, nous avons choisi d'utiliser la carte Arduino Uno pour trois raisons :

- Sa capacité de communication avec Simulink/Matlab
- Ses performances, 32kB de mémoire flash, 14 entrées/sorties dont 6 PWM, vitesse d'horloge 16Mhz et un port de communication USB. Ces performances sont largement suffisantes pour les applications que nous comptons réaliser.
- Son prix abordable, environ 25 euros pour la dernière version d'Arduino Uno (R3).

Une vue réelle de la carte est schématisée par la figure 2.

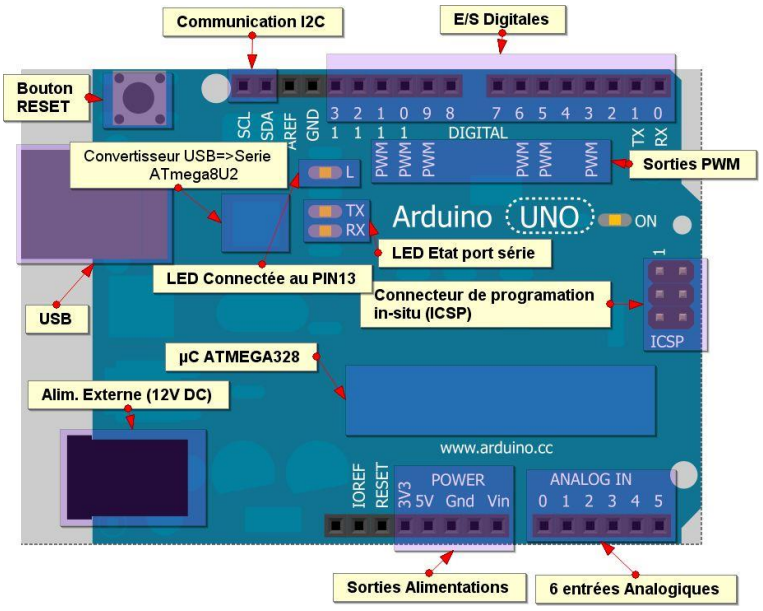

Fig. 2 : synoptique d'une carte Arduino Uno

On note bien la présence des bornes suivantes (PIN) :

- Entrée alimentation externe 12V DC.
- IOREF est une réserve, non connectée, prévue pour être utilisée dans les prochaines versions.
- Reset est une entrée pour un reset externe.
- Gnd, 3.3V et 5V sont des sorties d'alimentations régulées.
- Analog In (0..5) appelées aussi A0..5, sont des entrées analogiques 0-5V, la résolution du convertisseur ADC est 10bits, la référence de tension pour le convertisseur ADC est 5V par défaut.
- ICSP (In-Circuit Serial Programming) permet la programmation du microcontrôleur in-situ, utile dans le cas d'une mise à jour du firmware du microcontrôleur.
- 14 entrées/sorties digitales, dont 6 PWM

- AREF entrée pour une référence de tension externe pour le convertisseur analogique numérique (ADC)
- SDA, SCL lignes de communication I2C

2. Application pratique

2.1. Contexte

Dans ce cadre, nous proposons une application pratique réalisée avec les étudiants de la deuxième année licence génie électrique. Cette application consiste à réaliser un prototype pédagogique d'une chaine de mesure, traitement et d'asservissement de température, elle sera le point de départ pour développer des séries de TP pluridisciplinaires.

La réalisation du prototype est faite en deux étapes :

- Configurer la carte Arduino Uno comme une carte d'acquisition. Le traitement des données est réalisé sous Simulink.
- Le programme peut être ensuite compilé directement [2] de l'environnement Simulink vers la carte Arduino Uno, on dit que la carte Arduino est devenue une cible (Target) et elle peut fonctionner d'une façon autonome (sans avoir recours à MATLAB/Simulink), mais peut conserver sa liaison série.

2.2. Le matériel

Nous avons utilisé une thermistance à CTN 10k à 25°C de référence TDC05C310 [3]. La température est convertie en tension grâce au pont diviseur formé par la CTN et une résistance 10k est alimenté par une tension +5V issue de la carte. La sortie du pont diviseur est reçue par l'entrée analogique A0 (PIN 0). La communication de la carte et le PC se fait à travers une liaison USB. La figure 3 schématise l'implantation du matériel.

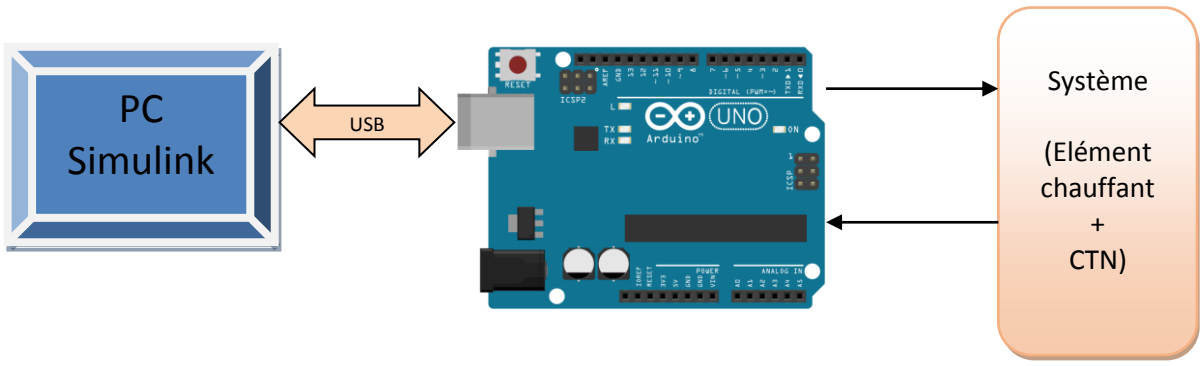

Fig. 3 : configuration du matériel utilisé

2.3. Les étapes de développement

En premier lieu, la carte Arduino est programmée pour être utilisée comme carte d'acquisition, la tension issue du pont diviseur formé par la CTN et la résistance 10k est appliquée à l'entrée (A0) du convertisseur analogique numérique (résolution 10bits). Ainsi, elle est convertie en une

valeur numérique de 0 à 255 pour être transmise au PC à travers le convertisseur série (RS232)/USB intégré à la carte. La figure 4 détaille le programme, écrit avec l'EDI Arduino, qui sera chargé au microcontrôleur de la carte.

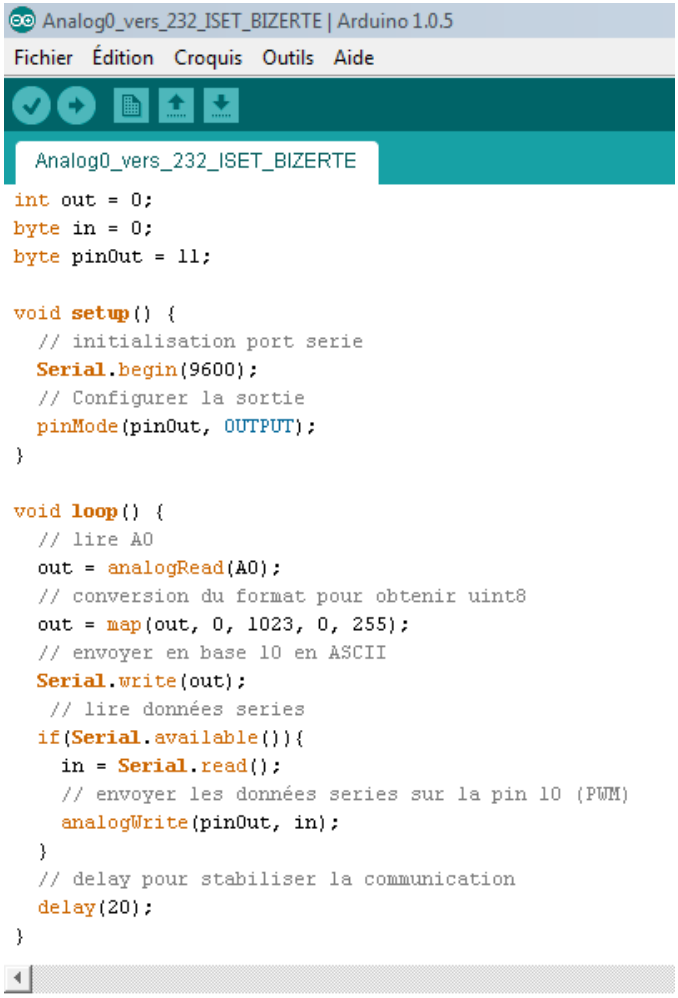

Fig. 4 : le programme assurant l'acquisition et le transfert des données vers le PC

En second lieu, nous développons des blocs Simulink qui assurent les fonctions suivantes : acquérir, traiter et afficher les données depuis la carte Arduino Uno.
En effet, à partir de la version 2012a de Matlab, la carte Arduino UNO ainsi qu'Arduino MEGA peuvent être installées automatiquement on peut accéder à plusieurs ressources à ce sujet [1] et [4]. L'installation se fait en exécutant tout simplement la commande >>*targetinstaller* et suivre les étapes, de préférence il faut choisir l'installation en ligne. À partir de cette étape, la carte Arduino Uno est reconnue sous Matlab/Simulink et une librairie contenant des blocs spécifiques est installée automatiquement sous Simulink comme l'indique la figure 5. Les blocs Simulink installés peuvent être utilisés avec les autres pour réaliser une multitude d'applications telles que :

- Exploitation des entrées/sorties digitales ou analogiques
- Commande PWM
- Commander directement des servomoteurs
- Communiquer en utilisant la liaison série (via le convertisseur USB/Série)
- Commande de systèmes (Asservissement, automatisation, etc.)

Une fois que le programme de l'acquisition et du traitement de la température est testé sous Simulink, nous entamons la dernière phase qui est la compilation du programme directement de Simulink vers la carte Arduino UNO, cette dernière peut fonctionner, maintenant, d'une façon autonome.

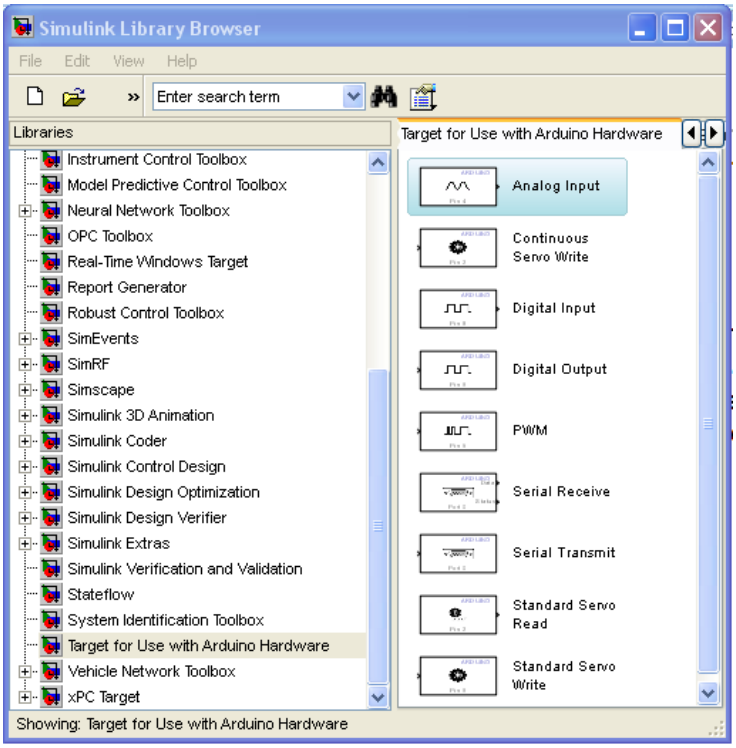

Fig. 5 : liste des blocs après l'installation de la carte Arduino Uno

2.4. Développement et test du programme sous Simulink

Dans cette partie, nous allons développer et tester, sous Simulink, le programme qui va traiter les données transmises à partir de la carte Arduino UNO.

La figure 6 montre la chaine d'acquisition qui regroupe le capteur de température, la carte Arduino UNO configurée en carte d'acquisition.

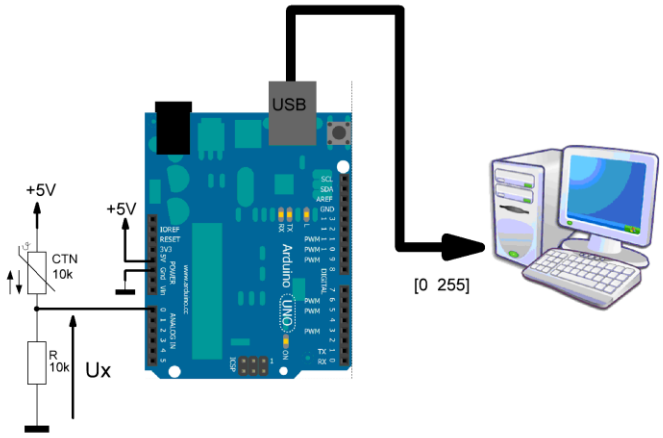

Fig. 6 : configuration de la chaine d'acquisition

La thermistance utilisée est du type CTN (Coefficient de Température Négatif) ses caractéristiques sont données par le document constructeur [2]. L'aspect compliqué de l'utilisation des thermistances réside dans le fait que la résistance en fonction de la température, $R_{CTN} = f(T°)$, n'est pas linéaire.

Cette dépendance n'est linéaire que dans des gammes de températures très faibles. Pour les mesures de températures exactes dans les diverses émissions de température, on peut utiliser l'équation exponentielle de troisième ordre de Steinhart-Hart [3]. Pour les thermistances CTN, il existe une équation de Steinhart-Hart simplifiée (1) en paramètre β [5] :

$$R_{CTN} = R_{CTN0} e^{\beta\left(\frac{1}{T} - \frac{1}{T_0}\right)} \qquad (1)$$

Avec R_{CTN0} la résistance de la CTN à $T_0 = 298.15$ K (25°C), dans notre cas $R_{CTN0} = 10k$ et $\beta = 4100$°K.

La température (T) est alors calculée à partir de l'équation (2)

$$T = \left[\left(\left(\frac{1}{\beta}\right)\ln\left(\frac{R_{CTN}}{R_{CTN_0}}\right)\right) + \frac{1}{T_0}\right]^{-1} \qquad (2)$$

Il faut donc convertir la tension issue du pont diviseur en résistance ce qui nous amène à l'expression suivante :

$$R_{CTN} = R\left(\frac{5}{U_x} - 1\right) \qquad (3)$$

La tension U_x est mesurée par la carte Arduino est convertie en une valeur numérique comprise entre 0 et 255 pour être par la suite envoyée au PC grâce au convertisseur série/USB intégré à la carte.

Le schéma bloc, correspondant au fonctionnement de la chaine d'acquisition et de traitement de la température, développé sous Simulink est représenté par la figure 7.

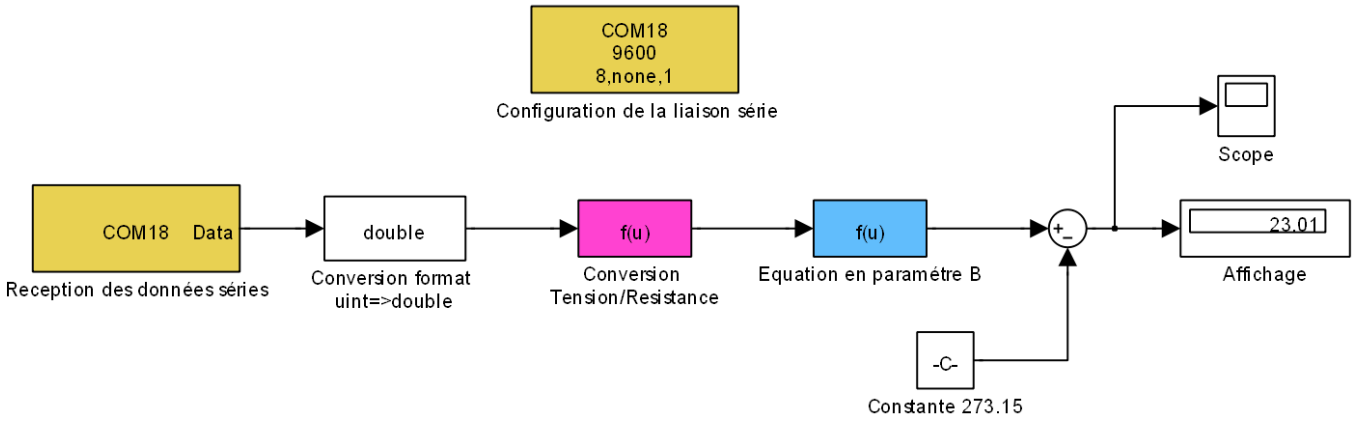

Fig. 7 : les blocs Simulink assurant l'acquisition et le traitement de la température

3. Extension de l'application

L'application de la chaine d'acquisition de température est considérée comme une base pour réaliser un asservissement de température à échelonner sur une série de TP. Dans cet ordre d'idées, nous proposons de compléter l'application précédente pour présenter une mise en œuvre simplifiée d'un TP d'asservissement de température d'un système de chauffage.

3.1. Maquette

Le matériel utilisé est une CTN 10k (Ref : TDC05C310) comme capteur de température et une résistance 15Ω de puissance 17W comme élément chauffant. Cette dernière est installée sur une plaque en aluminium au-dessus de laquelle est fixé le capteur constituant le système de chauffage. La figure 8 schématise les connexions avec la carte Arduino et la figure 9 montre une vue réelle de la maquette réalisée.

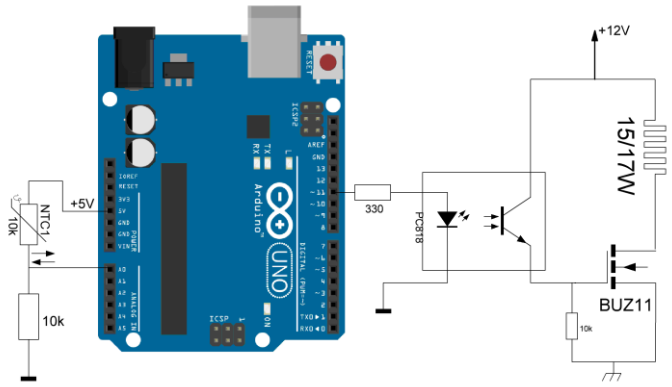

Fig. 8 : schéma électronique de la maquette

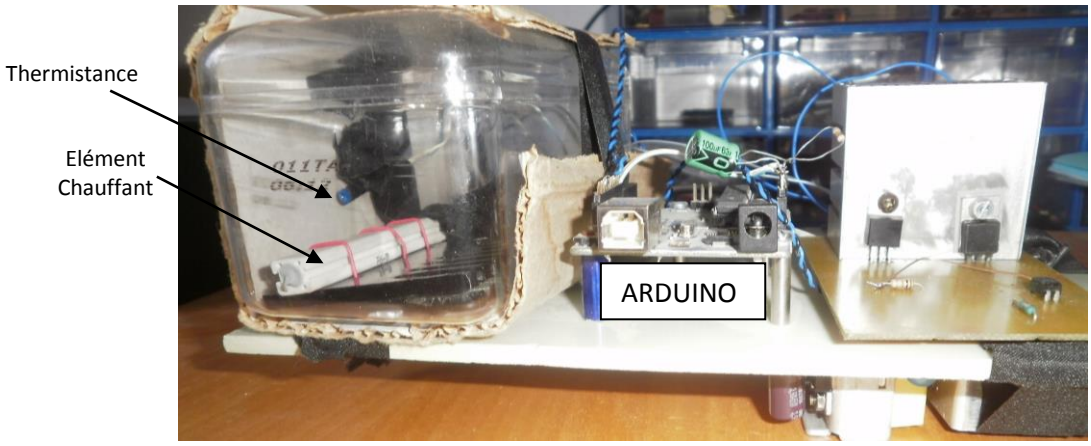

Fig. 9 : une vue de la maquette

3.2. Réalisation de l'asservissement avec Simulink

La carte Arduino se charge de la communication avec le processus réel et Simulink. Le programme permettant de gérer les entrées- sorties est celui de la figure 4. Dans Simulink, nous avons implémenté la boucle d'asservissement regroupant la consigne, le comparateur, le correcteur et le traitement de la température issue du capteur. Un exemple de mise en œuvre comportant un correcteur PID est illustré par la figure 10.

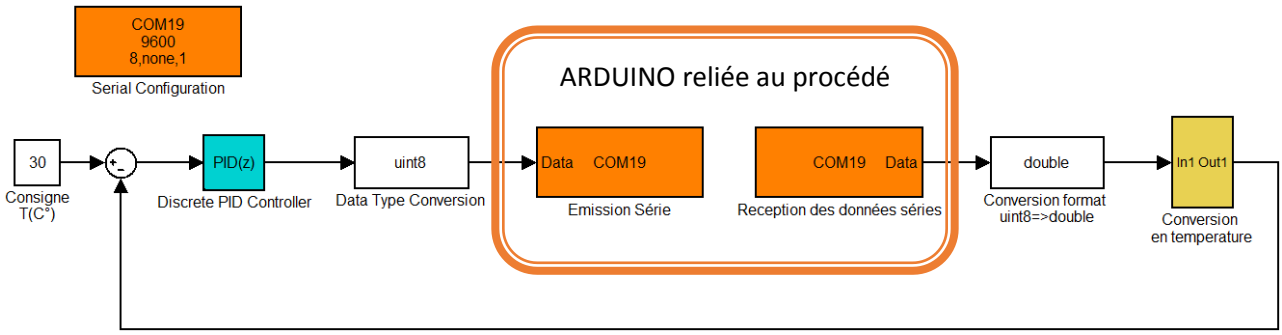

Fig. 10 : modèle Simulink de l'asservissement de température en temps réel

Grâce à cette maquette, à coût réduit, l'étudiant peut réaliser des séries de TP d'asservissement. En effet, l'outil Matlab/Simulink est assez puissant pour identifier, modéliser le système en sujet ensuite, faire la synthèse du correcteur ou même implanter d'autres lois de commandes plus avancées. À cet effet, Matlab propose une série d'outils qui permettent d'étudier et de synthétiser un système linéaire, on cite, à titre d'exemple : *LTI Viewer, Sisotool, PID Tuner*…

Un exemple de réponse du système à une consigne de température de 30 C° est donné par la figure 11.

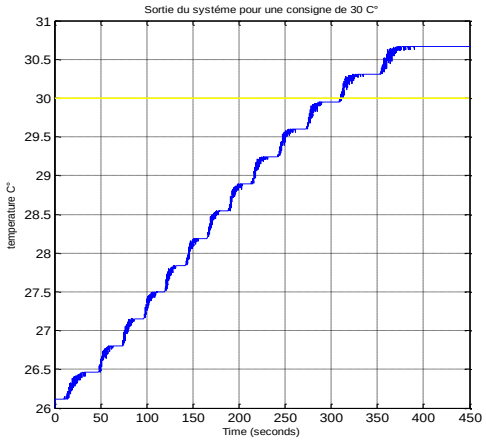

Fig. 11 : un exemple de régulation de température

Conclusion

Dans cet article, nous avons présenté une nouvelle approche d'enseignement de travaux pratiques de génie électrique. Nous avons fait appel à Arduino une plateforme matérielle très répandue et Matlab/Simulink un programme de modélisation/simulation professionnel bien connu. Cette approche permet aux étudiants d'appréhender des problèmes réels qui leurs permettent de développer leurs esprits de synthèse et d'analyse.

Avec une carte Arduino, il est intéressant d'exploiter les performances de l'environnement Matlab/Simulink pour attirer l'attention des étudiants. Pour ce faire, il suffit de faire appel à des exemples réels basés sur la simulation, expliquer les contraintes sur le plan pratique à travers la connectivité au matériel didactique à bas coût et exécuter des lois de commande sur un microcontrôleur.

L'exemple pratique qui a été présenté dans cet article, est considéré d'une part, comme étant un didacticiel permettant la mise en œuvre de cette nouvelle technologie et, d'autre part, un point de départ d'une série de prototypes pédagogiques pluridisciplinaires à échelonner sur une série de TP.

Références

[1] Site officiel de la plateforme arduino : www.aduino.cc .

[2] Jérôme PIETRE. « Matlab et Arduino en S-SI ? ». http://sti.tice.ac-orleans-tours.fr/spip2/IMG/pdf/Tutoriel_Matlab_Arduino.pdf

[3] Document constructeur de la CTN TDC05C310 : www.uei.com.tw/ptdc.pdf.

[4] Ressources officielles Matlab/Simulik et Arduino : http://www.mathworks.com/academia/arduino-software/arduino-simulink.html.

[5] John S. Steinhart and R. Hart, Stanley. « Calibration curves for thermistors ». Deep-Sea Research,. Vol.15, pp.497–503.

[6] Thermistor Constant Conversions - Beta to Steinhart-Hart: http://assets.newport.com/webDocuments-EN/images/TNSTEIN-1_Thermistor_Conversions_IX.pdf

Mohamed Ali ZERZRI, né le 19 mai 1977 à Tunis, Tunisie, est enseignant Technologue aux Instituts Supérieurs des Études Technologiques (ISET) depuis 2002. Actuellement Technologue à l'ISET de Bizerte, Tunisie. Il a obtenu un mastère en électronique de l'École d'Ingénieurs de Sfax (ENIS) en 2005 et le certificat des études supérieures spécialisées en génie électrique de l'École Supérieure des Sciences et Techniques de Tunis (ESSTT) en 2001 et de la même école, une maitrise en électronique et instrumentation.

Il assure des cours et des travaux pratiques d'électronique et de distribution électrique aux étudiants de licence appliquée en génie électrique.