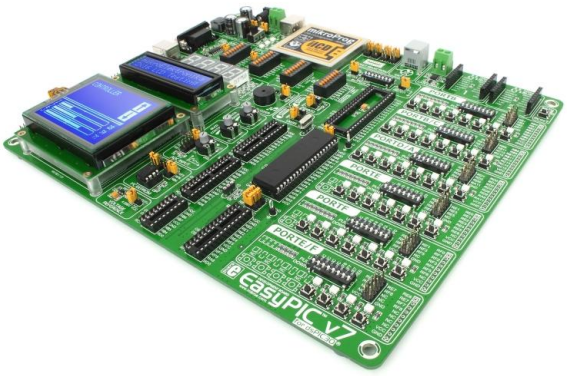
SONKO Robert-Junior **MUNOZ Vincent**

Programmation d'interprètes / Réseaux de capteurs, réseaux mobiles

Rapport de projet : Transmission d'informations entre une carte EasyPICV7 et Android



Encadré par : L. SEDDIKI et Y.TOUATI

Introduction

Durant ce cours, notre travail consistait à effectuer une transmission de données entre une carte spécifique et une application sur Android. Nous avions à notre disposition deux types de capteurs, que nous avons du programmer pour pouvoir récupérer les valeurs de ces derniers sur la carte premièrement, pour ensuite les transférer sur l'application Android.
Nous allons donc détailler ce projet au cours de ce rapport, nous traiterons du contexte et de la problématique, puis nous expliquerons un peu plus en détail le matériel et les différents outils utilisés, et enfin nous présenterons les réalisations effectuées ainsi que les principales difficultés rencontrées au durant ce projet.

1. Etat de l'art / Contexte

N'étant pas trop habitué à programmer sur des cartes telles que Raspberry ou Arduino par exemple, on nous a proposé de travailler sur un modèle de carte intitulé EasyPIC v7, modèle 18f45K22. C'est une excellente carte lorsque l'on débute dans ce domaine, qui est utilisée dans l'éducation mais aussi dans le développement.

2. Problématique

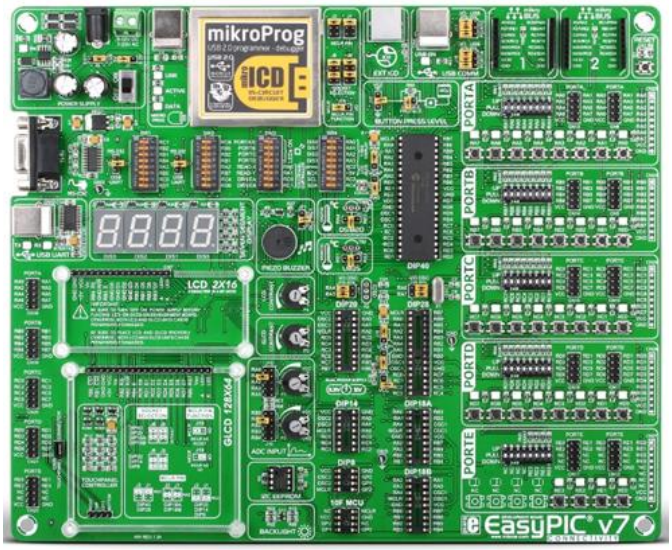
Notre problématique principale est donc la transmission de données entre la carte EasyPIC et une application Android. On peut découper notre problématique en plusieurs petites étapes.
Dans un premier temps, il a fallu apprendre comment la carte fonctionnait, grâce à la documentation de la carte ainsi qu'en analysant la carte, des informations écrites pour indiquer chaque port et chaque bouton.
Ensuite, lorsque l'on a reçu les capteurs, il a fallu programmer la carte afin qu'elle affiche la valeur de ces derniers sur l'afficheur de la carte.
Après cela, il fallait, via un module Bluetooth et en utilisant l'Uart de la carte, envoyer les valeurs de ces capteurs dans une base de données SQLite sur une application Android.

3. Présentation du matériel utilisé

Voici donc la liste des différents composants que nous avons utilisé au cours de ce projet.

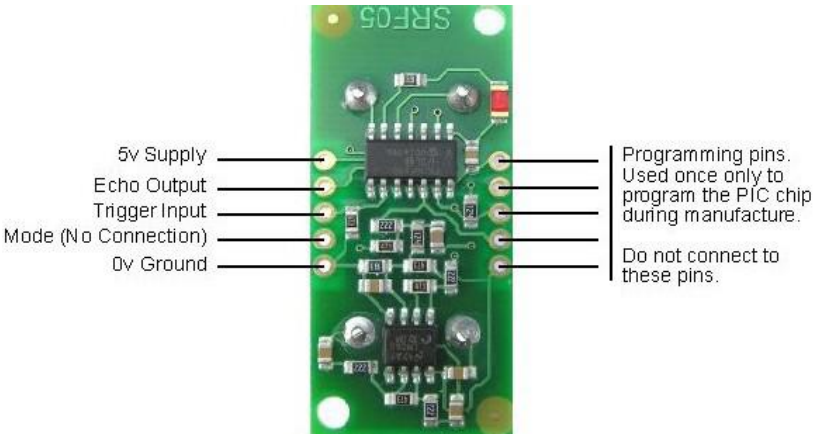
- La carte easyPicV7 18F45K22, possédant de nombreuses fonctionnalités, telles que :
    - Le connecteur ICD2 qui permet aux utilisateurs de la carte de travailler avec d'autres développeurs.
    - L'Uart RS-232 pour l'envoi de données sur des applications tiers.
    - La possibilité de brancher deux écran LCD

- Un afficheur à 4 digits et 7 segments
- 5 ports disponibles pour brancher divers modules
- 2 ports Uart pour les modules bluetooth, wi-fi…



https://www.mikroe.com/easypic

- Le capteur-émetteur SRF05, qui a une portée de 2cm à 4m.
  Voici une image des câblages du capteur :

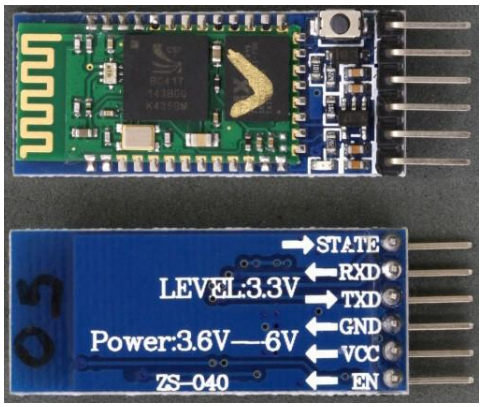


www.robot-electronics.co.uk

Dans notre code, nous avons branché le capteur de la façon suivante :

- 5v Supply sur VCC
- Echo Output sur RA0
- Trigger Input sur RA1
- 0v Ground sur GND

- Le capteur-émetteur Bluetooth HC-05 BC417, qui a une portée de 10m.



http://www.martyncurrey.com

  - RXD sur TX
  - TXD sur RX
  - GND sur GND
  - VCC sur +5V

- Capteur température DS1820, qui a un emplacement spécifique sur la carte

os.mbed.com

4. Présentation des outils utilisés

Pour la partie programmation de la carte, on a utilisé l'IDE propre à cette dernière qui se nomme « MikroC PRO for PIC », on l'a utilisé pour programmer les deux capteurs, l'affichage et la gestion de l'Uart. Pour pouvoir utiliser et compiler le code réalisé dans cet IDE avec la carte, il faut évidemment le connecter à la machine, et aussi installer au préalable le driver pour supporter la carte.

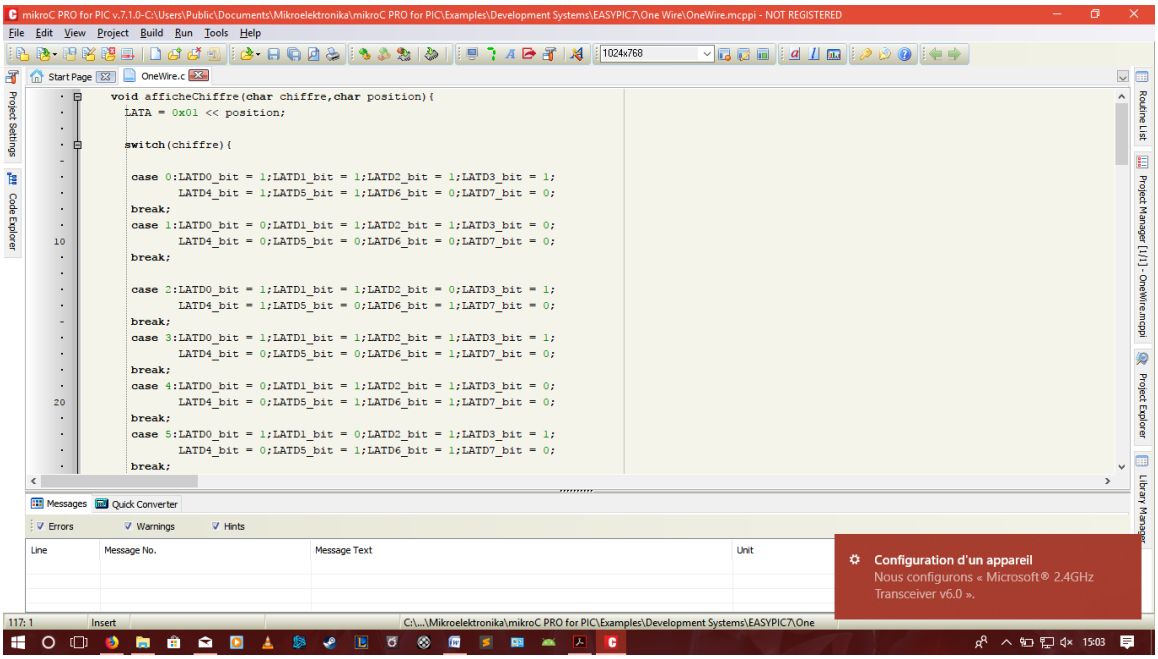


Image représentant l'IDE MikroC PRO for PIC

Concernant la partie Android, nous avons utilisé « Android Studio », pour le développement de l'application et la gestion de la base de données.

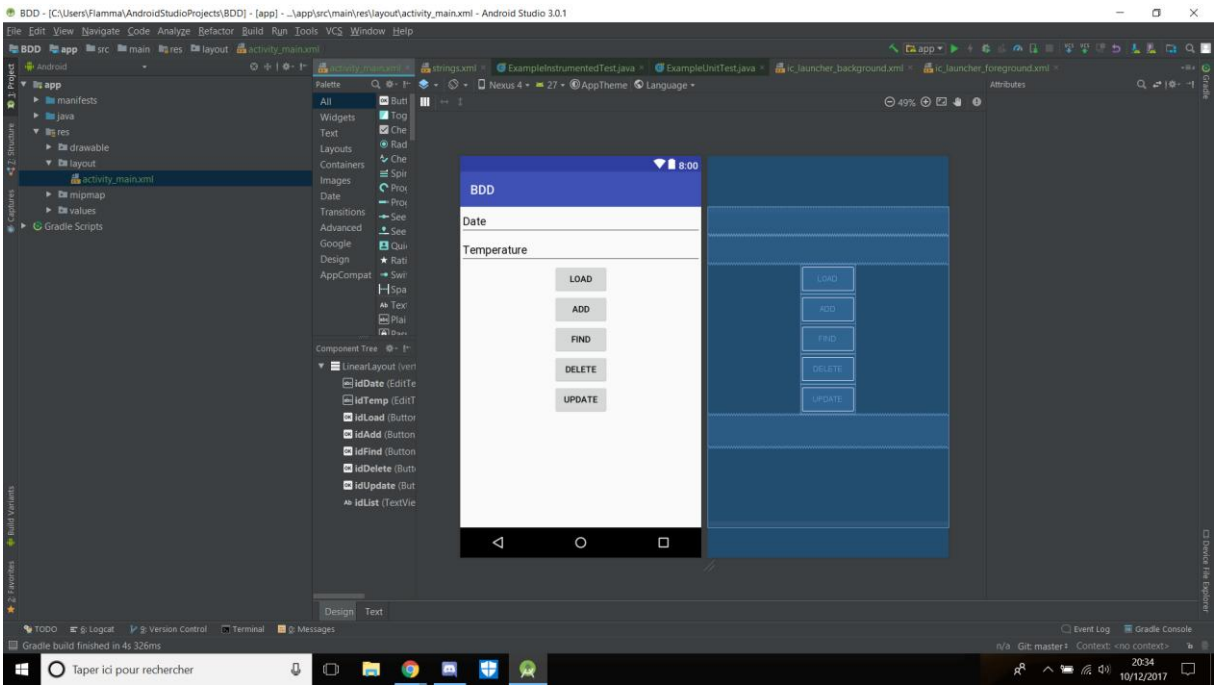


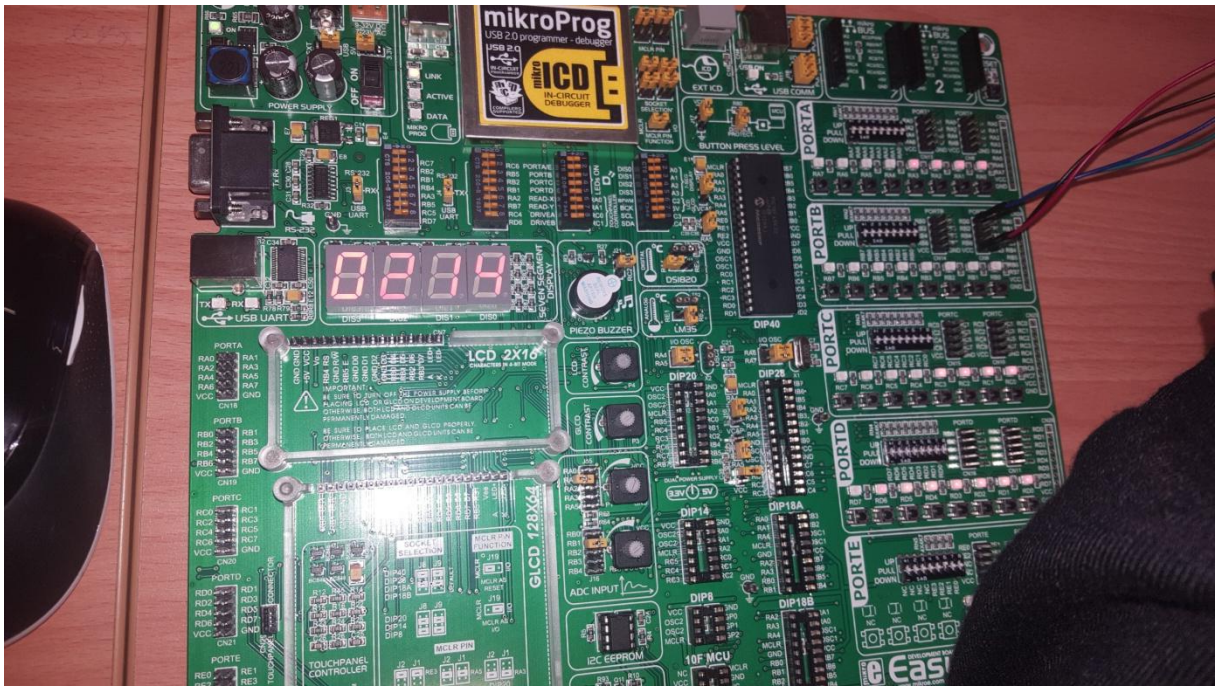
Image représentant Android Studio et l'interface de notre application

5. Réalisation

Pour afficher les valeurs récupérées par nos capteurs, nous avons implémenté une fonction permettant de séparer un nombre entier en 4 chiffres. Chaque segment d'un afficheur est codé sur 1 bit (pour un total de 8). Notre fonction d'affichage d'un chiffre synchronise donc les bits du port D (port qui gère l'afficheur) avec l'afficheur.

Pour l'ultrason, on fait tourner une boucle de 4 étapes:

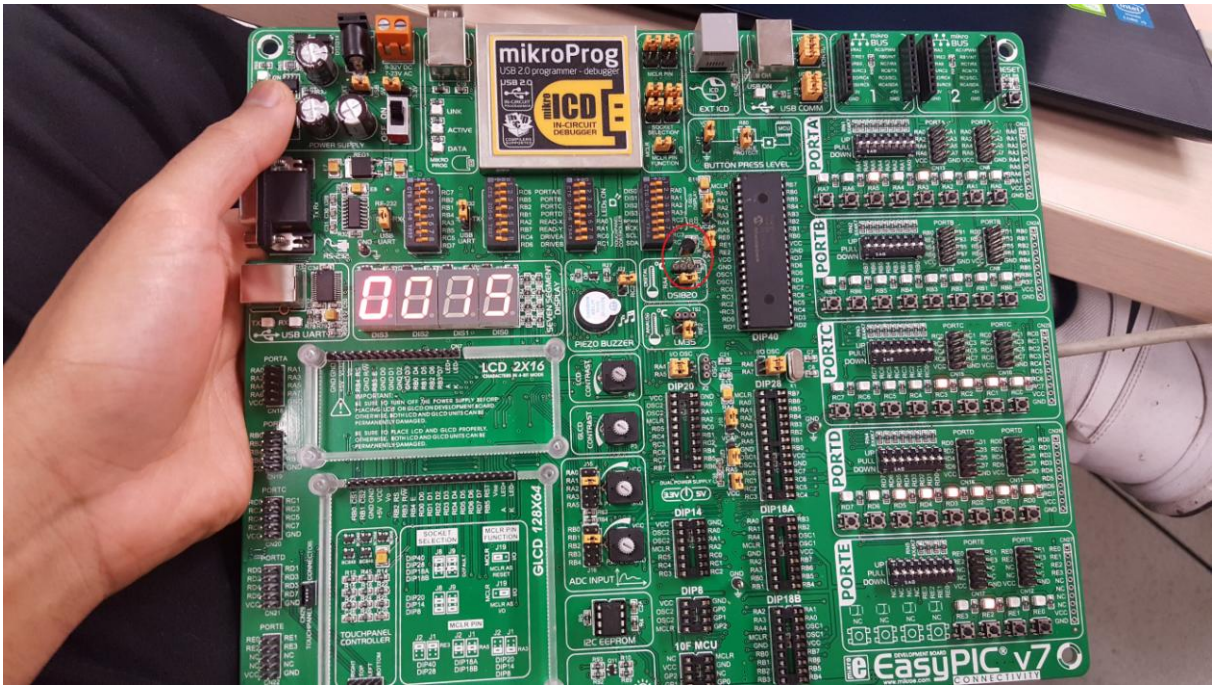
- Etape 1: Permet de lancer l'envoi de l'onde.
- Etape 2: Permet d'attendre le temps d'envoi de l'ultrason (1 ms) avant la récupération.
- Etape 3: On compte le temps que met l'onde à revenir.
- Etape 4: On affiche le nombre représentant cette distance.



Exemple d'affichage de la distance avec l'ultrason

Pour la température, on fait tourner une boucle qui:

- Actualise le capteur.
- Récupère la température.
- Enregistre cette valeur.
- Affiche la valeur enregistrée.



Exemple d'affichage de la température avec le capteur

Pour l'envoi de données des capteurs via le module Bluetooth, nous avons utilisé l'Uart1 de la carte EasyPIC. Au lieu d'afficher la valeur du capteur sur l'afficheur de la carte, on envoie la donnée sur l'Uart avec la commande « Uart_write », puis on la récupère du côté de l'application Android. Nous n'avons cependant pas pu vérifier si ce que l'on a fait marchait correctement, comme on ne pouvait pas connecter nos téléphones au module Bluetooth de la carte.

Concernant notre application Android, on a une application avec une base de données SQLite, qui est censée récupérer les valeurs envoyées par la carte et la stocker dans la base de données avec une date, et potentiellement l'actualiser en temps réel.

6. Difficultés rencontrées

Les principales difficultés que nous avons rencontrées venaient principalement du matériel.
Au début, lorsque l'on cherchait à afficher la distance de l'ultrason, nous avons perdu pas mal de temps à cause du port A de la carte EasyPIC qui était défectueux, ce qui faisait que le signal de l'ultrason n'était pas envoyé en continu. En passant du port A au port B, nous avons résolu ce problème.
Concernant les valeurs du capteur de température, elles sont relativement approximatives malgré le calcul pour passer des degrés Fahrenheit aux degrés Celsius.
De plus, nos appareils téléphoniques n'arrivaient pas à se connecter au capteur Bluetooth, ce qui nous a empêchés de finaliser la dernière partie de notre projet, à savoir récupérer les valeurs des capteurs sur l'application Android.

Conclusion

Durant ce projet, nous avons su utiliser la carte EasyPic avec différents matériels. Nous ne sommes cependant pas arrivés à récupérer les données de la carte sur une application Android. Ce cours de programmation d'interprètes, réseau de capteurs et réseaux mobiles nous à permis d'acquérir une base sur l'utilisation de microcontrôleur et du mikroC ainsi que sur la manipulation de différents capteurs.