Ajouté par Alexandre DE CARVALHO

Polytech Projets Ge » Polytech Ge Sous Traitance

Temps estimé : 8h

WifiBot - Développement d'un soft de lecture d'un télémètre, et envoi de la mesure sur I²C

Objectifs:

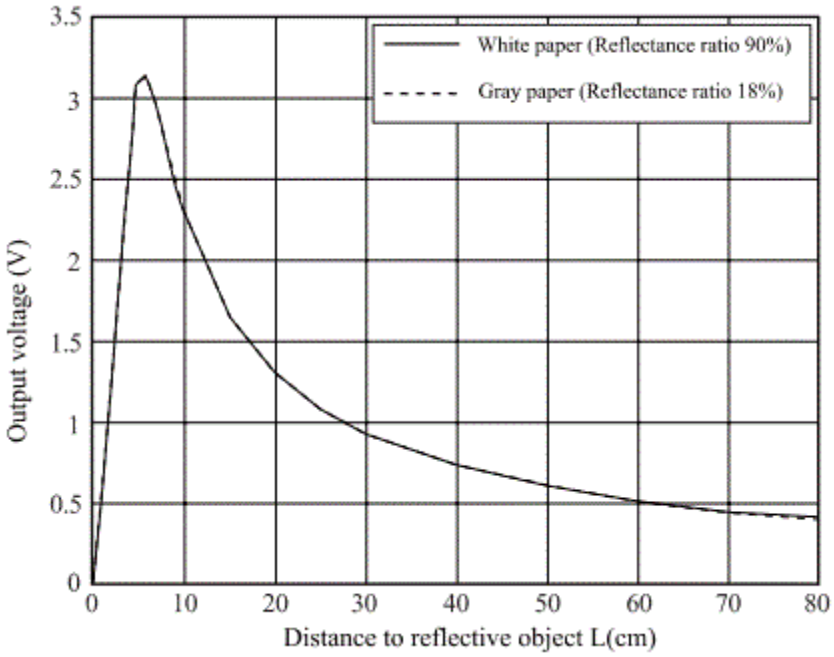
Dans le cadre du projet WifiBot, vous développerez un programme qui effectuera la lecture d'un télémètre SHARP et enverra la mesure sur le bus I²C.

Vous développerez en C avec MPLABX sur la carte de synthèse 2013 PIC18F45X50.

Délivrables :
- Projet MPLABX
- Maquette fonctionnelle

1. Etude du capteur sharp:

Le graphique suivant met en relation la distance et les tensions mesurées.



La fonction tension-distance n'est pas une bijection et par conséquent, son inverse n'existe pas. C'est pourquoi nous n'allons considérer que la partie de la courbe allant de 10 [cm] jusqu'à 80 [cm]. Dans cette plage, la fonction tension-distance peut être approchée par la fonction suivante :

$$d = \left(\frac{v}{\hat{k}_1}\right)^{\frac{1}{\hat{k}_2}}$$

Avec :

$\hat{k}_1 = 16.7647563$ et $\hat{k}_2 = -0.85803107$

Les calculs sont déjà faits voici le lien pour avoir plus de details :

https://aleksandarhaber.com/noise-reduction-and-calibration-of-distance-sensors-sharp-infrared-sensors/

1. implementation:

Pour réaliser le code du projet, je suis passé par le MPLAB Code Configurateur (ou MCC), qui permet de configurer le module ADC et I²C plus efficacement qu'à la main.

Fonctionnement du projet :

»Fichier main :

```
#include "mcc_generated_files/mcc.h"
#include <math.h>

float distance=1;
float resultat1;
float resultat2;
float k1=16.7647563;
float k2=-0.85803107;
char k=1;
uint8_t V;
```

On commence par initialiser le ficher mcc et la bibliothèque math.h pour utiliser la fonction pow afin d'implémenter l'équation.

```
void main(void)
{
    // initialisation
    SYSTEM_Initialize();
    INTERRUPT_GlobalInterruptEnable();
    INTERRUPT_PeripheralInterruptEnable();

    ADC_SelectChannel(0);
    I2C1_Open ();
    while (1)
    {
        resultat1=ADC_GetConversion(0);
        resultat2=(resultat1*5)/1024;
        distance = pow(resultat2*(1/k1), 1/k2);
        V=(uint8_t)(distance);
        __delay_ms(1000);

    }
}
```

Apres l'initialisation des interruptions, le module ADC lié avec le Port A0 et l'I2C qui envoyer les données avec - broche RB0 : SDA (data) - broche RB1 : SCL (clock)

La fonction « while » calcule la distance et le convertie en « uint8_t ».

>>i2c1_slave.c :

Techniquement on envoie les données avec la fonction << I2C1_SlaveDefWrInterruptHandler()>>

```
extern uint8_t V; // apporter le variable V du fichier main

static void I2C1_SlaveDefWrInterruptHandler() {
    I2C1_SlaveSendTxData(V); //Envoyer les données en I2C
}
```

Le deuxième code, nommé "Projet_wifibot_Prog_arduino_master", a pour but de communiquer via I²C avec le PIC, et voir les valeurs de la distance. En utilisant la fonction Wire.requestFrom(80, 6);qui s'adresse au pic avec l'adresse 80 et affiche les valeurs reçus avec Wire.read();

```
#include <Wire.h>

void setup()
{
  Wire.begin();        // join i2c bus (address optional for master)
  Serial.begin(9600);  // start serial for output
}

void loop()
{
  Wire.requestFrom(80, 6);    // request 6 bytes from slave device #80

  while(Wire.available())    // slave may send less than requested
  {
    char c = Wire.read();    // receive a byte as character
    Serial.println(int(c));         // print the character
  }

  delay(1000);
}
```

Vous pouvez visualiser les valeurs de la distance avec le serial monitor