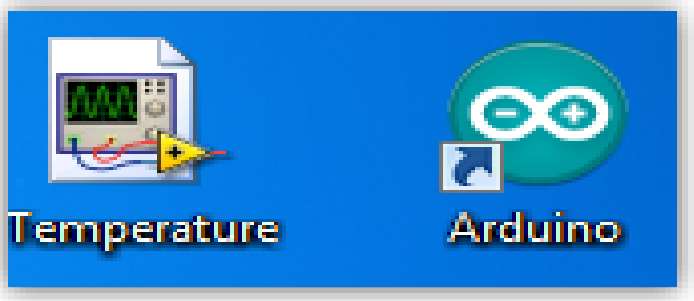
FACULTE DES SCINCES DE TUNIS 2015/2016

ARDUINO+LABVIEW

CONTROLE DE LA TEMPERATURE



Préparé par : DAHECH Hajer IE5

Table des matières

Liste des Figures

Liste des Tableaux

Introduction

Ce projet consiste à contrôler la température par l'intervention du plusieurs logiciels :

- Arduino.
- LabVIEW.
- ISIS.

Ce rapport comporte 5 étapes :

N°	Etape	Rôle
1	▪ Préparation de l'environnement du Labview+Arduino	C'est une étape d'initialisation qui nous permet d'accéder à la carte Arduino sous LabVIEW.
2	▪ Montage sous ISIS	Cette étape définie les composants utilisés et le schéma sous ISIS.
3	▪ Code du projet sous Arduino	Cette étape permet d'avoir comment coder ARDUINO sous LabVIEW.
4	▪ Création du projet sous LabVIEW	Block Diagram+Front Panel.
5	▪ Communication entre LabVIEW et ISIS	Comment réaliser une communication entre ISIS et LabVIEW.

Tableau 1 : Les étapes du projet

I. Préparation de l'environnement du Labview+Arduino

Pour installer Arduino sous LabVIEW, il faut d'abord avoir installé :

- VISA NI.
- Arduino IDE.
- JKI VI.

- Qu'elle est l'utilité de chaque logiciel et comment les utilisés pour avoir ARDUINO sous LabVIEW ?

  1 .Arduino IDE

  Arduino est une plateforme de prototypage open-source basé sur le matériel et un logiciel facile à utiliser. Cartes Arduino sont capables de lire les entrées - la lumière sur un capteur, un doigt sur un bouton, ou un message Twitter - et la transformer en une sortie - l'activation d'un moteur, d'allumer une LED, publier quelque chose en ligne. Vous pouvez dire à votre conseil quoi faire en envoyant un ensemble d'instructions au microcontrôleur sur la carte. Pour ce faire, vous utilisez le langage de programmation Arduino (basé sur le câblage), et le logiciel Arduino (IDE), basée sur le traitement.
  Ci-dessous vous trouvez la fenêtre du logiciel Arduino (IDE) sous lequel on va tapez notre code.

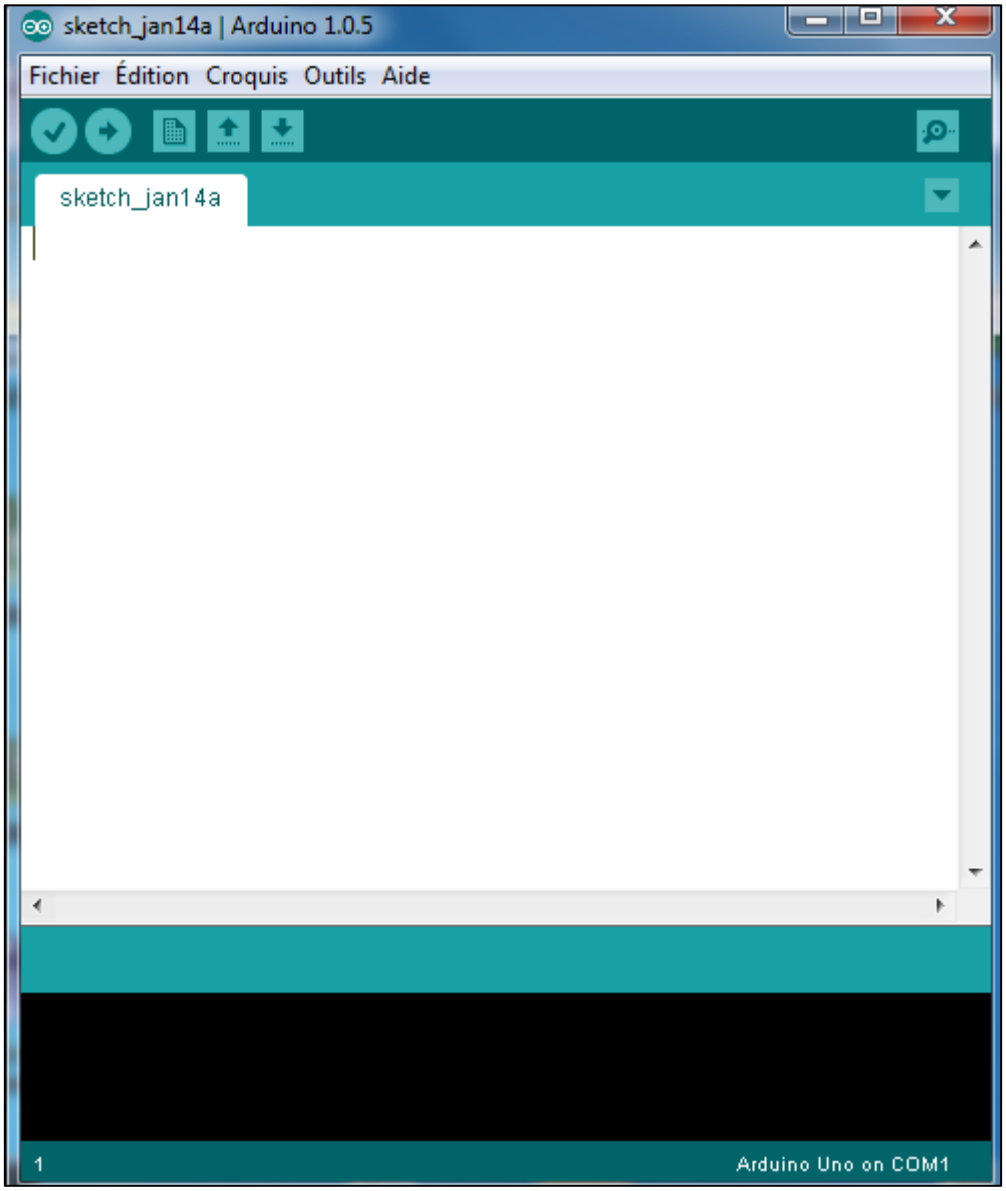


Figure 1 : Logiciel Arduino(IDE)

2. VISA NI

Le Salon Virtuel Instrument Software Architecture (VISA) est un standard pour la configuration, la programmation et le dépannage des systèmes d'instrumentation comprenant des interfaces GPIB, VXI, PXI, série, Ethernet et / ou USB. VISA fournit l'interface de programmation entre les environnements matériels et de développement tels que LabVIEW, LabWindows / CVI et Measurement Studio pour Microsoft Visual Studio. NI-VISA est la mise en œuvre de National Instruments de la norme / S VISA je. NI-VISA comprend des bibliothèques de logiciels, les services publics interactifs tels que NI I / O Trace et le VISA Interactive Control, et des programmes de configuration via Measurement & Automation

Explorer pour tous vos besoins de développement. NI-VISA est standard sur toute la gamme de produits de National Instruments.

IL faut mettre VISA sous le dossier National Instruments du LabVIEW.

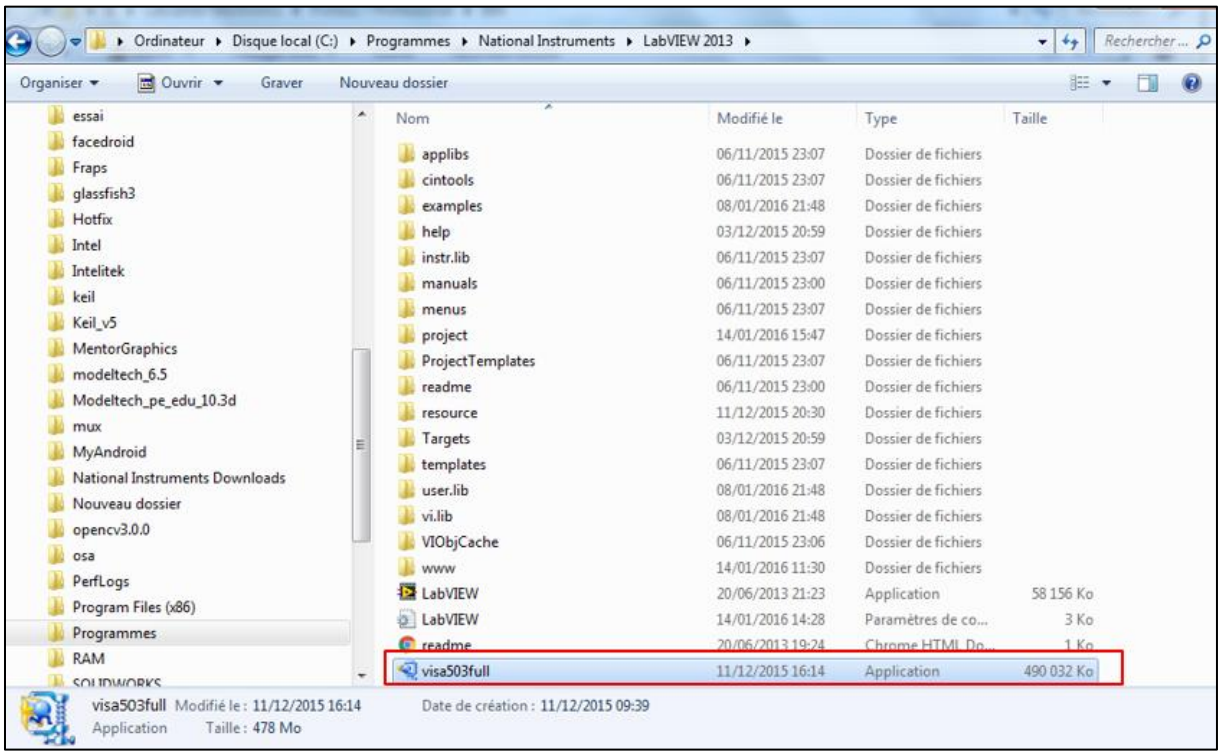


Figure 2 : Emplacement du VISA sous LabVIEW

3. JKI VI

VI Package Manager (de VIPM) est un outil de gestion des paquets qui organise et gère les paquets au sein de votre environnement LabVIEW. Il est l'outil pour l'obtention et de la configuration des bibliothèques et des outils de développement.

Après l'installation du VI, on tape Arduino dans l'icône de la recherche et on obtient la fenêtre suivante.

Voir la figure ci-dessous.

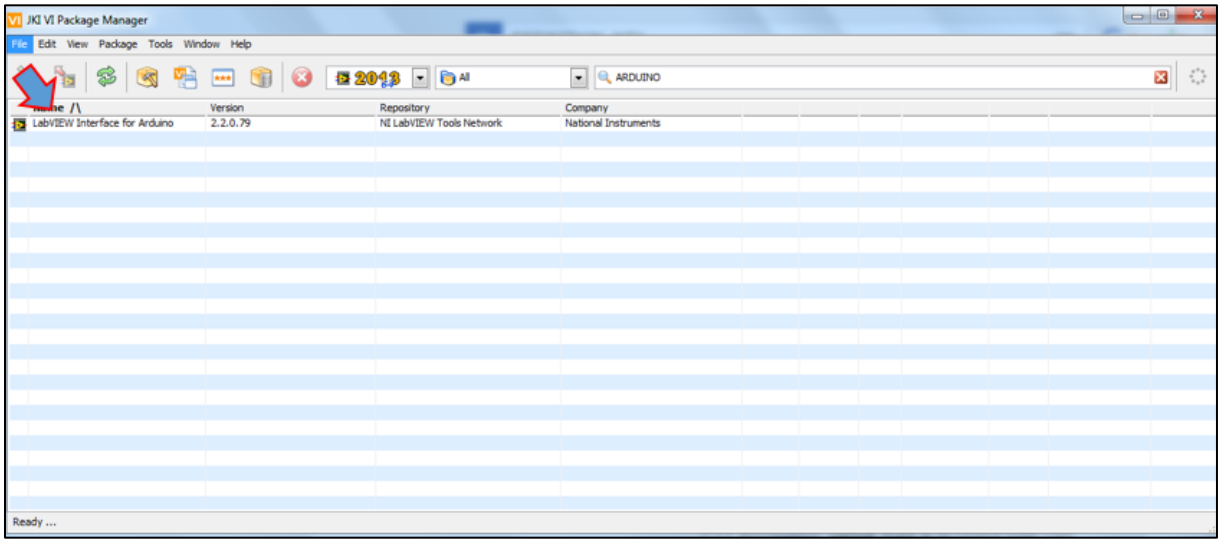


Figure 3 : Arduino sous JKI VI

Double cliques sur « LabVIEW Interface for Arduino », puis Install.

A la fin de l'installation, on a pour résultat la fenêtre suivante.

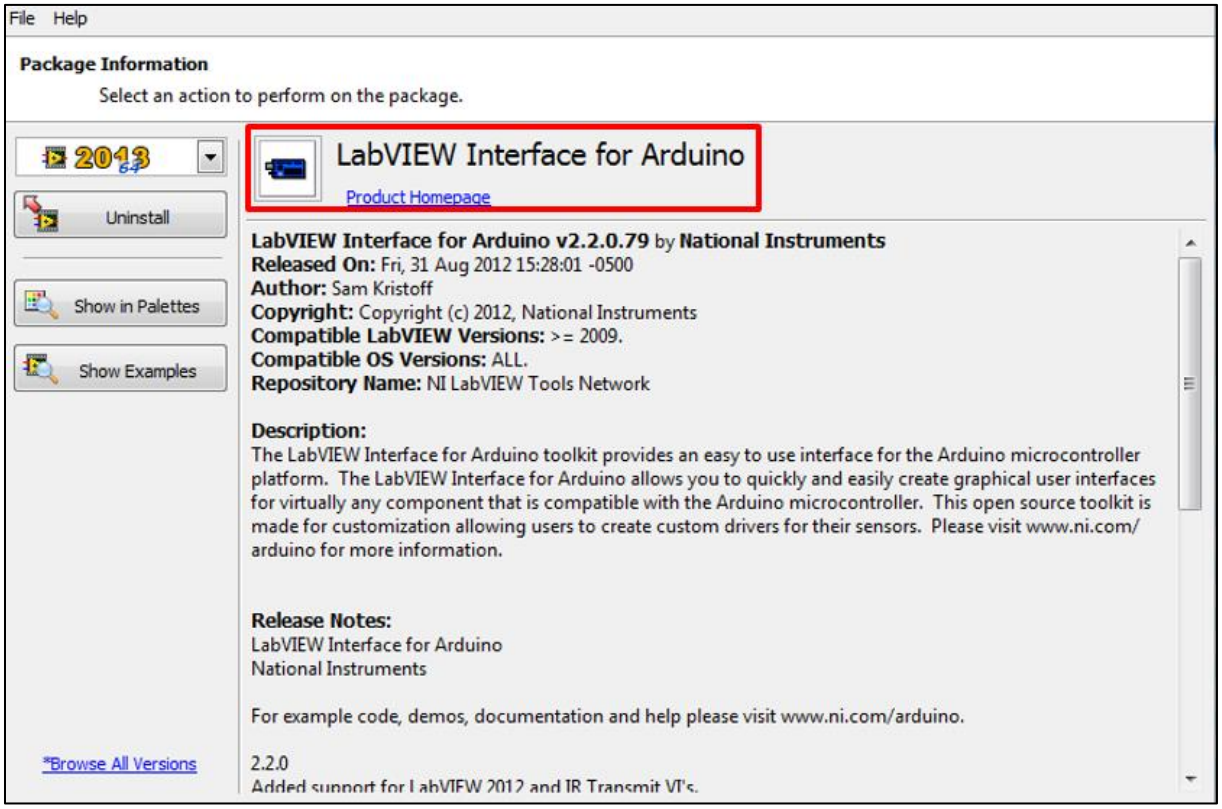


Figure 4 :L'installation de l'interface ARDUINO sous LabVIEW

Maintenant quand on lance le logiciel LabVIEW, on vérifie quand la carte Arduino est ajoutée on non.

Clique droit →on trouve les fonctions →on cherche le mot Arduino →Et voilà on trouve cette fenêtre qui contient les composants qui définissent Arduino sous LabVIEW.

Et par suite nous pouvons commencez notre projet qui permet de contrôler la température.

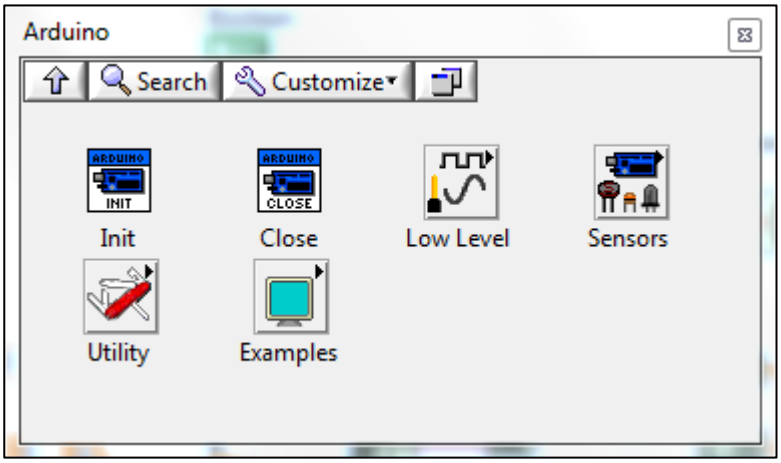


Figure 5 : Arduino sous LabVIEW

II. Montage sous ISIS

Pour contrôler la température on va utiliser les composantes suivantes :

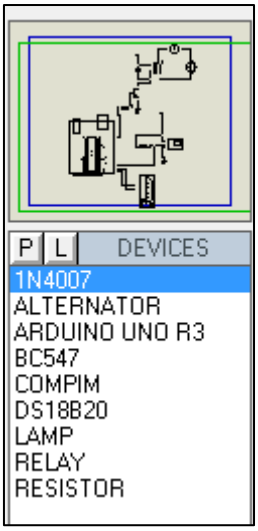
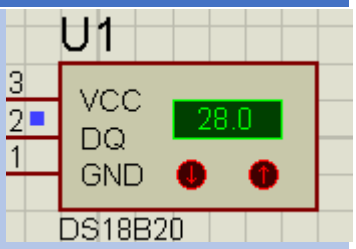
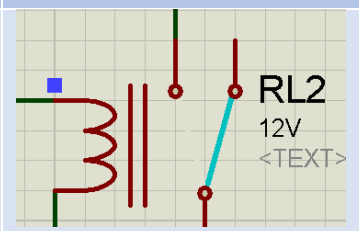
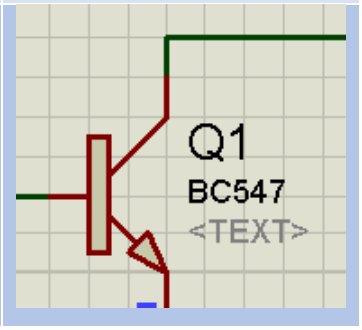


Figure 6 : Composants utilisés

Non du Composant	Rôle	Schéma sous ISIS
DS18B20	Le thermomètre numérique DS18B20 9 bits fournit à Celsius mesures de température de 12 bits et a une fonction d'alarme avec des points non volatiles programmables par l'utilisateur déclenchement supérieur et inférieur. Le DS18B20 communique sur un bus 1-Wire® que, par définition, ne nécessite qu'une seule ligne de données (et la terre) pour la communication avec un microprocesseur central. Il a une plage de température de fonctionnement de -55 ° C à + 125 ° C et une précision de ± 0,5 ° C sur la plage de -10 ° C à + 85 ° C. En outre, le DS18B20 peut tirer un pouvoir directement depuis la ligne de données («puissance parasite»), éliminant le besoin d'une alimentation externe.	
RELAY	Un relais est un commutateur à commande électrique. Les relais sont utilisés là où il est nécessaire de commander un circuit par un signal de faible puissance (par isolation électrique complète entre les circuits de commande et contrôle), ou où plusieurs circuits doivent être contrôlés par un seul signal.	
BC547	Le BC547est un transistor à usage général de jonction bipolaire trouvé couramment dans les équipements électroniques.	

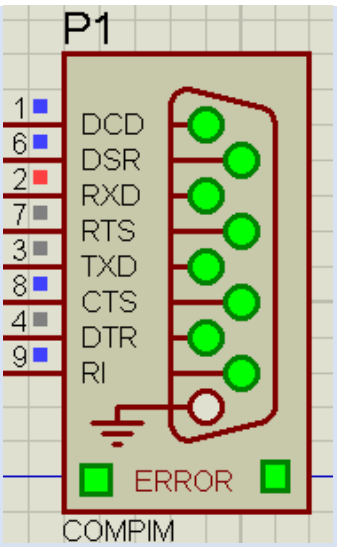
COMPIM	Communication entre ISIS et LabVIEW.	

Tableau 2 : Tableau des composants

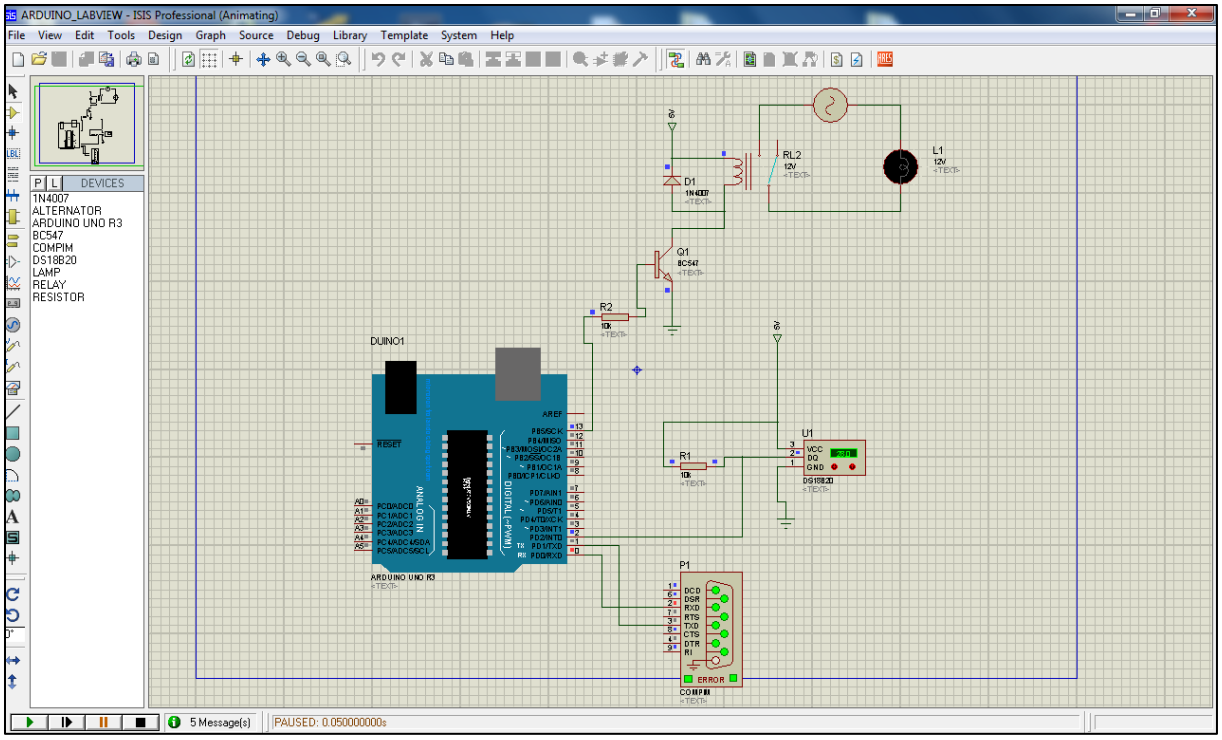


Figure 7 : Schéma du montage sous ISIS

III. Code du projet sous Arduino

Pour coder Arduino sous LabVIEW, il faut accéder au chemin suivant « C :\Program Files\National Instruments\LabVIEW 2013\vi.lib\LabVIEW Interface for Arduino\Firmware\LIFA_Base »

On obtient la fenêtre ci-dessous.

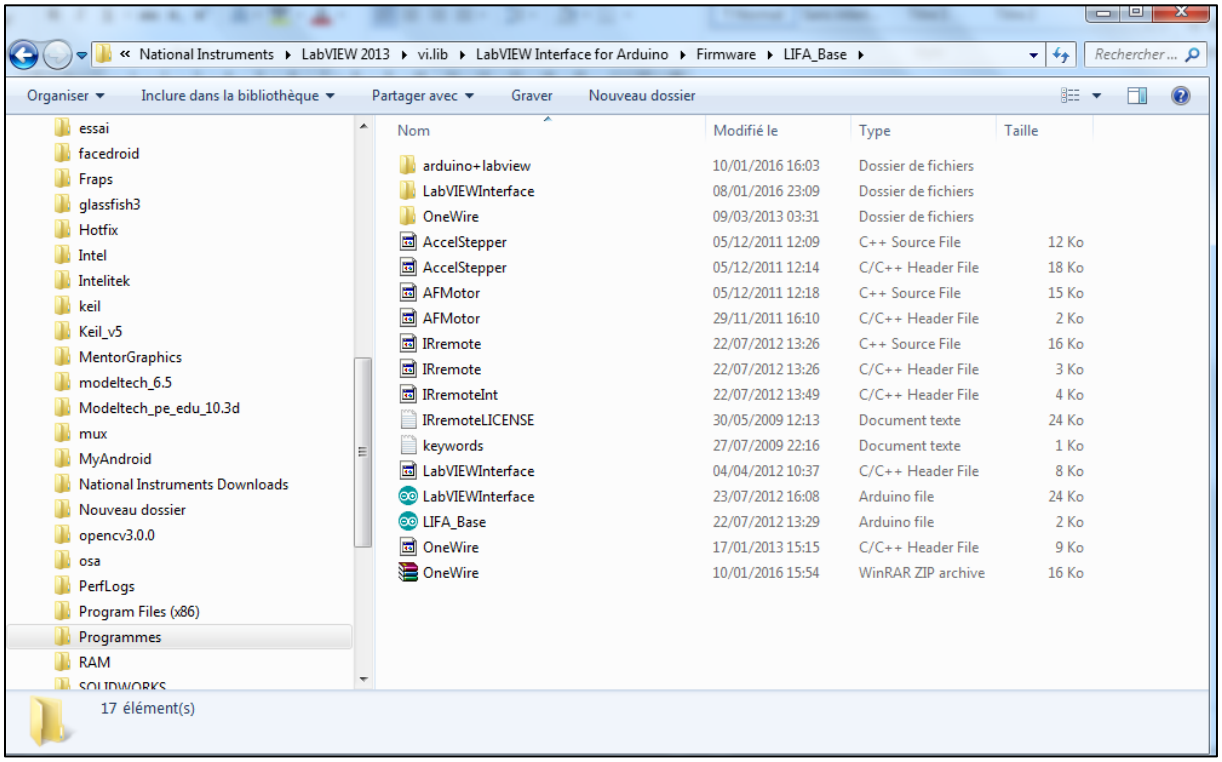


Figure 8:Dossier LIFA_Base

On copie les fichiers qui sont indiqués dans la figure qui suit et on les met dans un dossier, puis on lance le logiciel Arduino et on copie ces fichiers dans le sketch ouvrit.

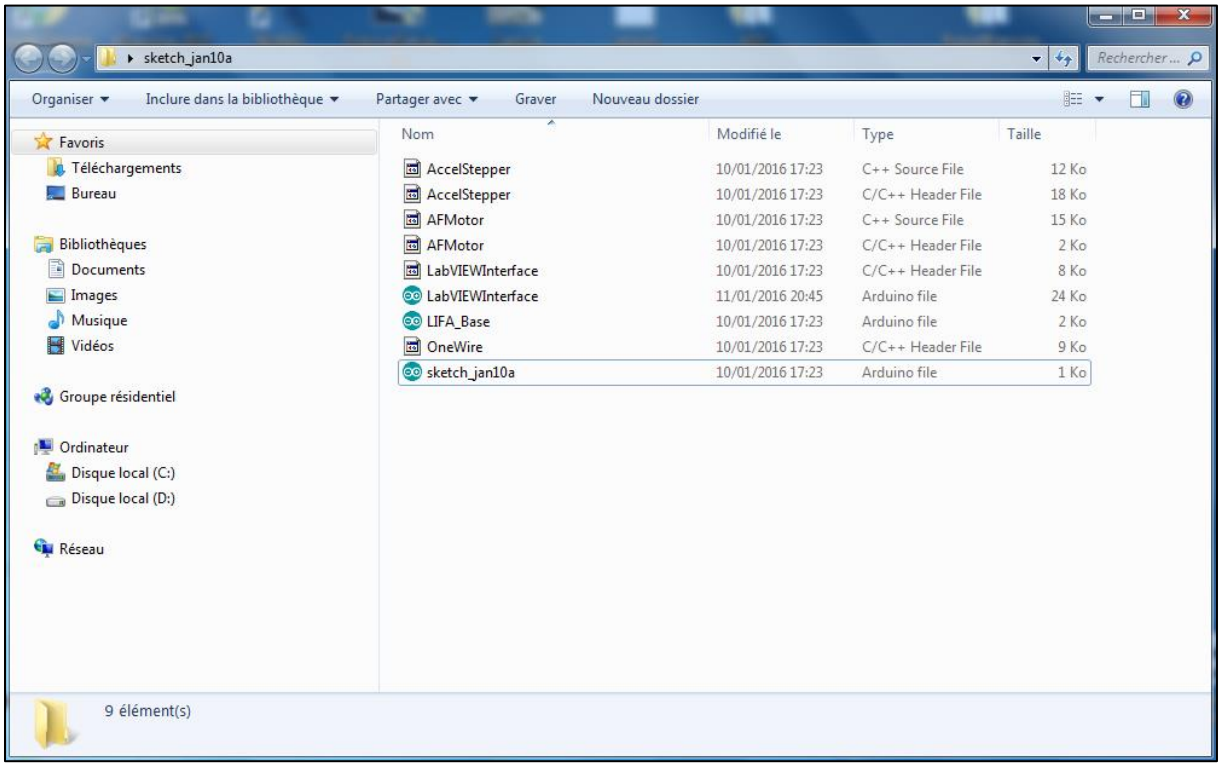


Figure 9 : Fichiers LIFA_Base nécessaires

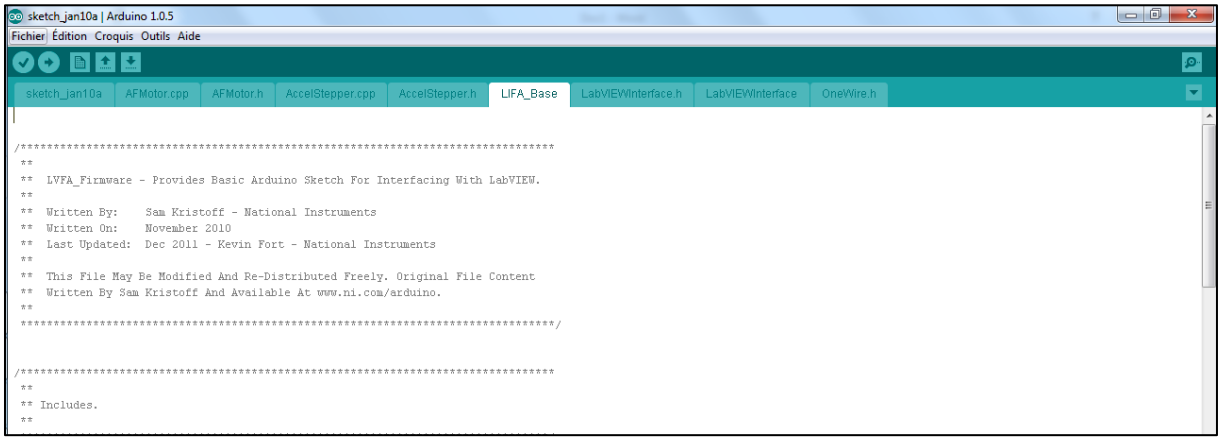


Figure 10 : Fichiers du LIFA_Base sous Arduino

Maintenant, on va ajouter le code qui nous permet de contrôler la température.

On va téléchargera le code DS18S20.

Voici le code.

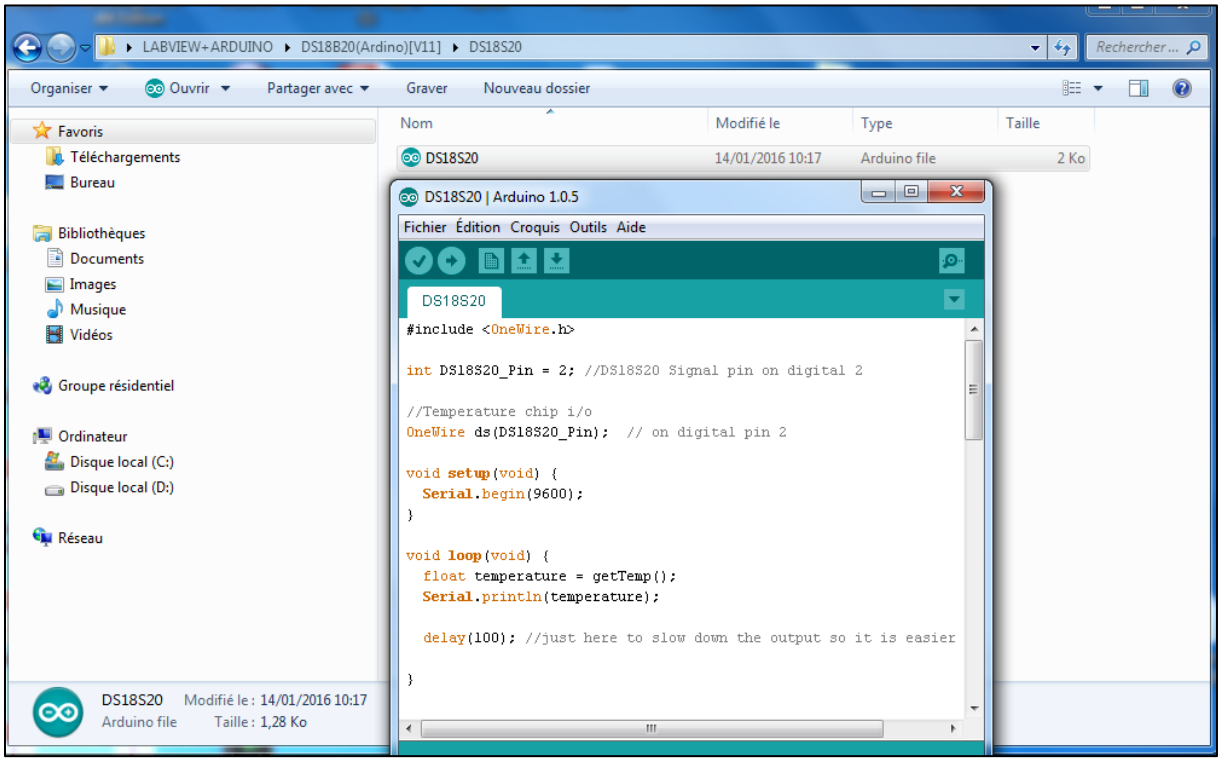


Figure 11:DS18S20

DS18S20

```
#include <OneWire.h>

int DS18S20_Pin = 2; //DS18S20 Signal pin on digital 2

//Temperature chip i/o
OneWire ds(DS18S20_Pin);  // on digital pin 2

void setup(void) {
  Serial.begin(9600);
}

void loop(void) {
  float temperature = getTemp();
  Serial.println(temperature);

  delay(100); //just here to slow down the output so it is easier to read

}


float getTemp(){
  //returns the temperature from one DS18S20 in DEG Celsius

  byte data[12];
  byte addr[8];

  if ( !ds.search(addr)) {
      //no more sensors on chain, reset search
      ds.reset_search();
      return -1000;
```

Figure 12 : Code du DS18S20

```
  if ( OneWire::crc8( addr, 7) != addr[7]) {
      Serial.println("CRC is not valid!");
      return -1000;
  }

  if ( addr[0] != 0x10 && addr[0] != 0x28) {
      Serial.print("Device is not recognized");
      return -1000;
  }

  ds.reset();
  ds.select(addr);
  ds.write(0x44,1); // start conversion, with parasite power on at the end

  byte present = ds.reset();
  ds.select(addr);
  ds.write(0xBE); // Read Scratchpad
  for (int i = 0; i < 9; i++) { // we need 9 bytes
    data[i] = ds.read();
  }

  ds.reset_search();

  byte MSB = data[1];
  byte LSB = data[0];
  float tempRead = ((MSB << 8) | LSB); //using two's compliment
  float TemperatureSum = tempRead / 16;
  return TemperatureSum;
}
```

Figure 13 : Suite du code DS18S20

On revient maintenant sketch qui contient les fichiers LIFA_Base, on s'intéresse au fichier « LabVIEW Interface »

On va suivre les étapes présentées par les figures suivantes.

- On ajoute #include<OneWire.h>

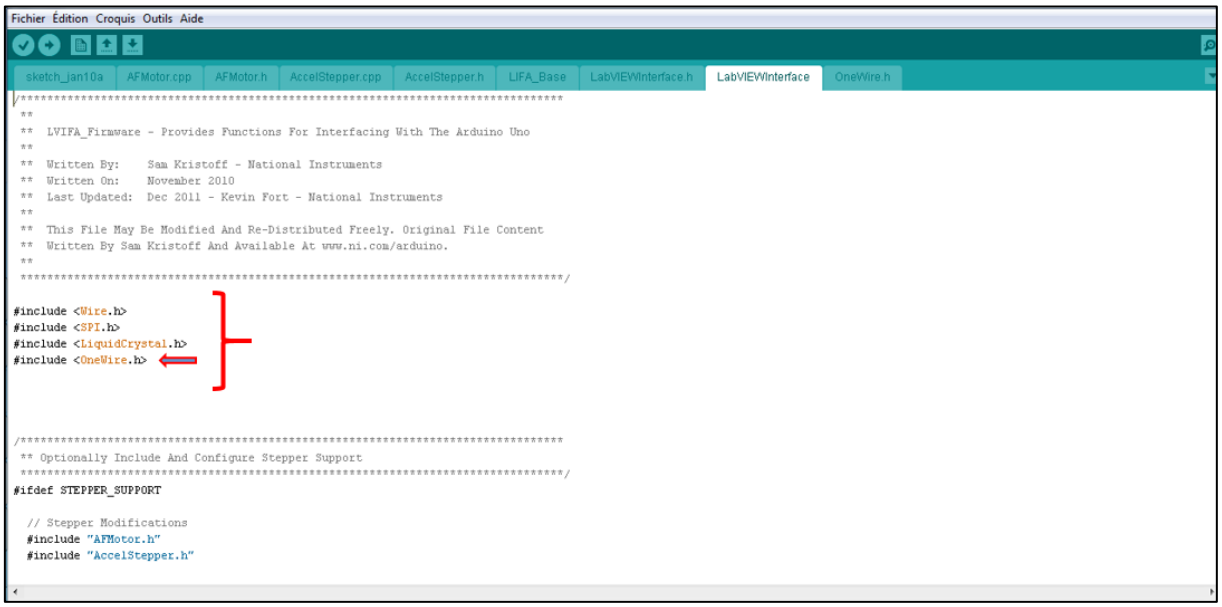


Figure 14 : Les librairies

- On ajoute les variables utilisées dans le code DS18S20.

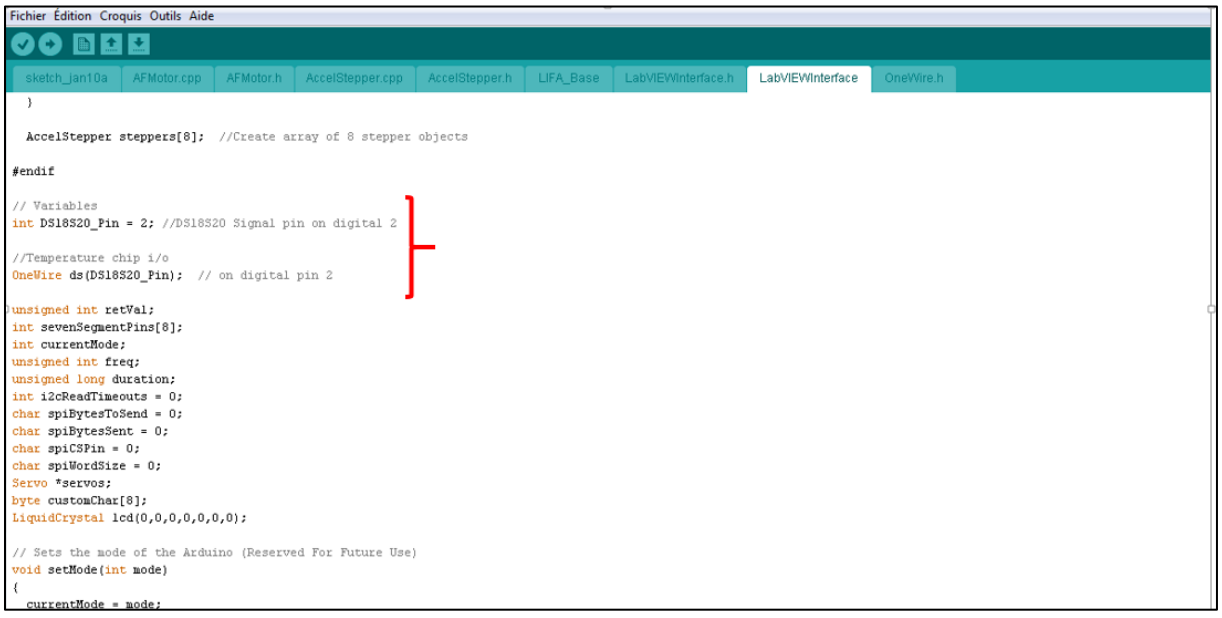


Figure 15 : Ajout des variables

Puis on ajoute une nouvelle case qui fait appel à la fonction qui donne la température.

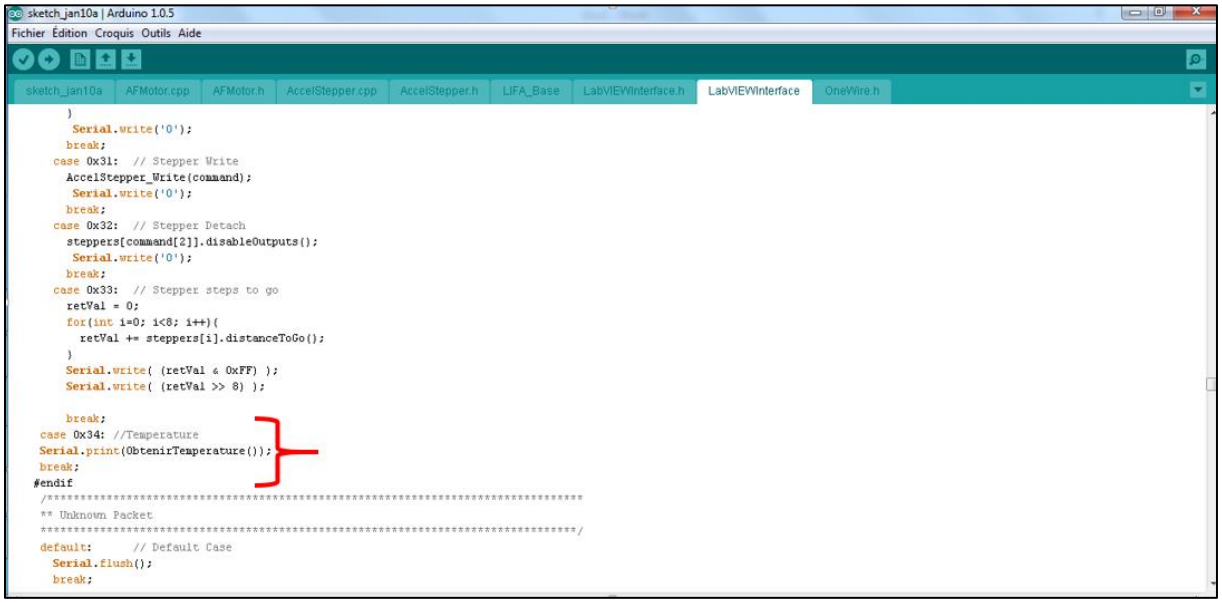


Figure 16 : Case 0x34

- Maintenant, on ajoute la fonction « ObtenirTemperature » qui constitue le reste du code DS18S20.

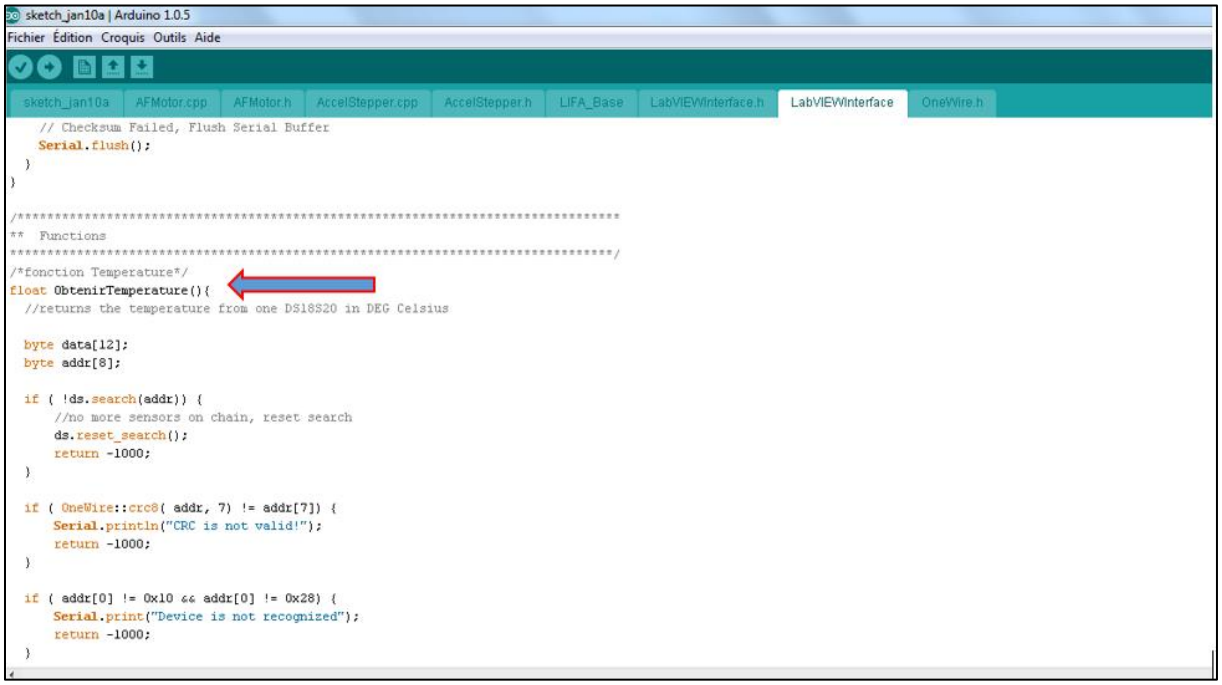


Figure 17 : Fonction « ObtenirTemperature »

- Maintenant on peut compiler le programme et corriger les erreurs.

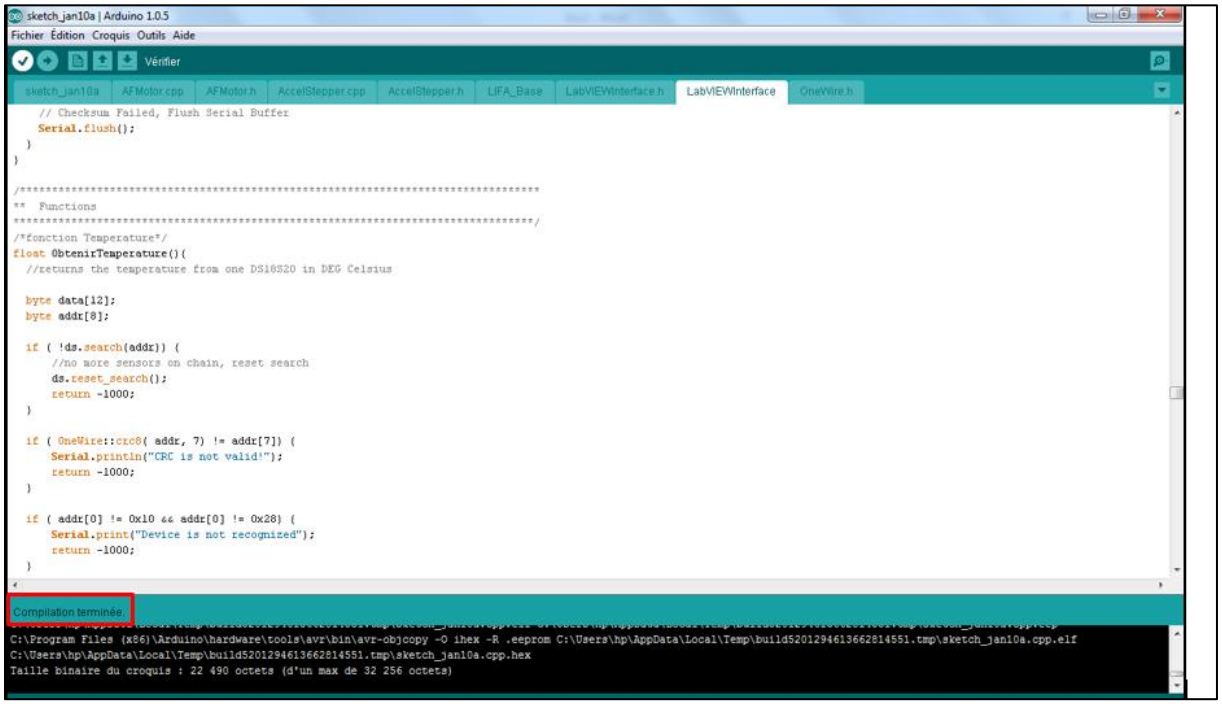


Figure 18 : Compilation terminée

IV. Création du projet sous LABVIEW

Dans cette étape on va créer deux blocs :

- ObtenirTemperature. (2).
- Température. (1).

Puis on va injecter le bloc (2) dans le bloc (1) qui forme le projet global.

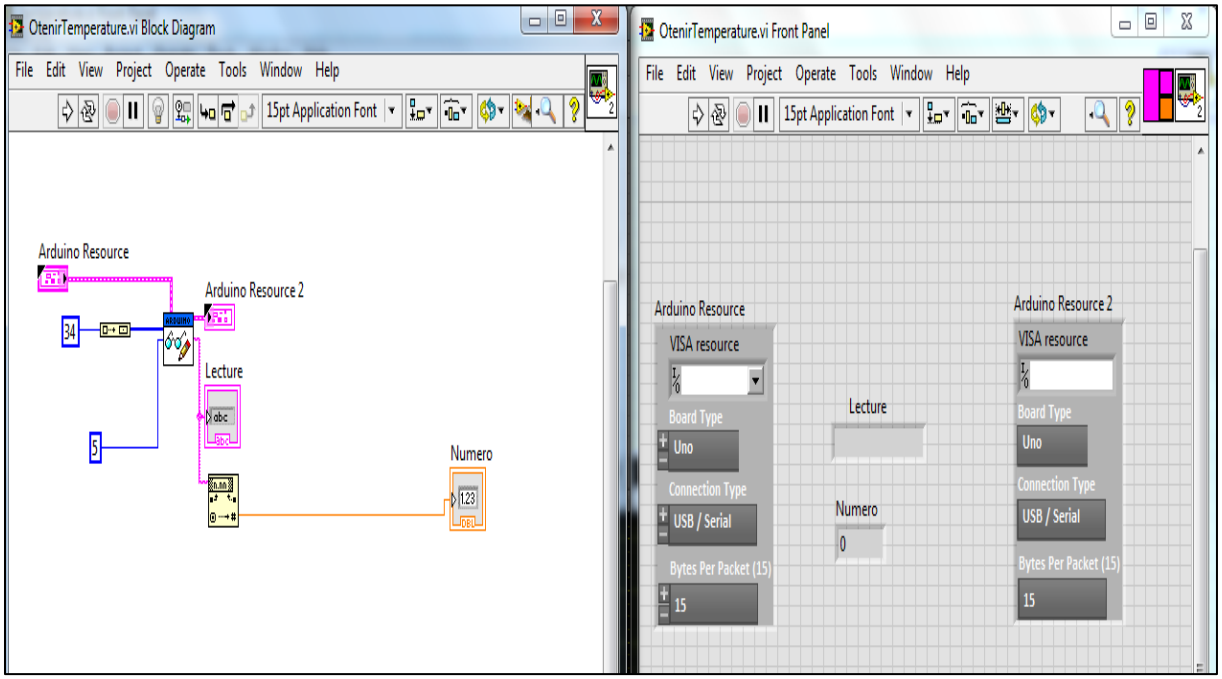


Figure 19 : ObtenirTemperature (Block Diagram+Front Panel)

Pour ce bloc on va créer 3 entrées pour qu'on quand puisse l'utiliser par suite comme de fonction sous le projet global.

Voici comment se manipuler ça par figure.

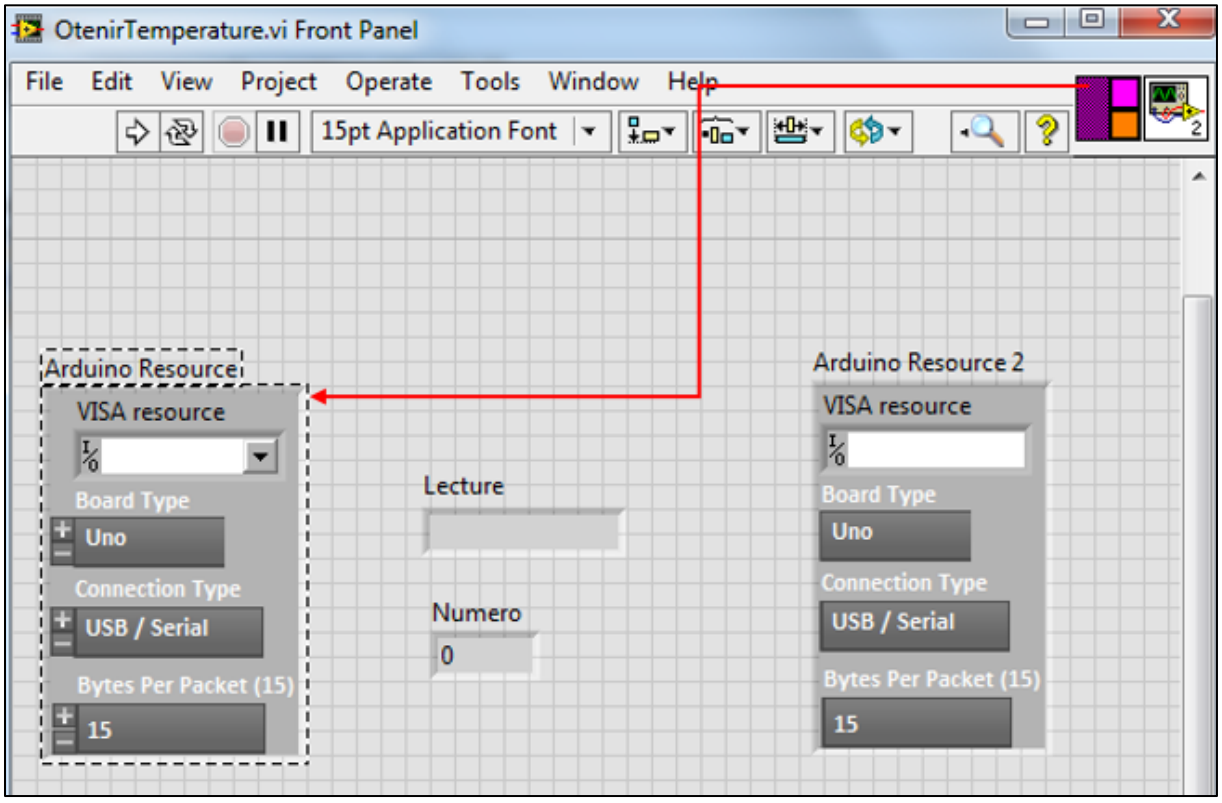


Figure 20 : Arduino Resource

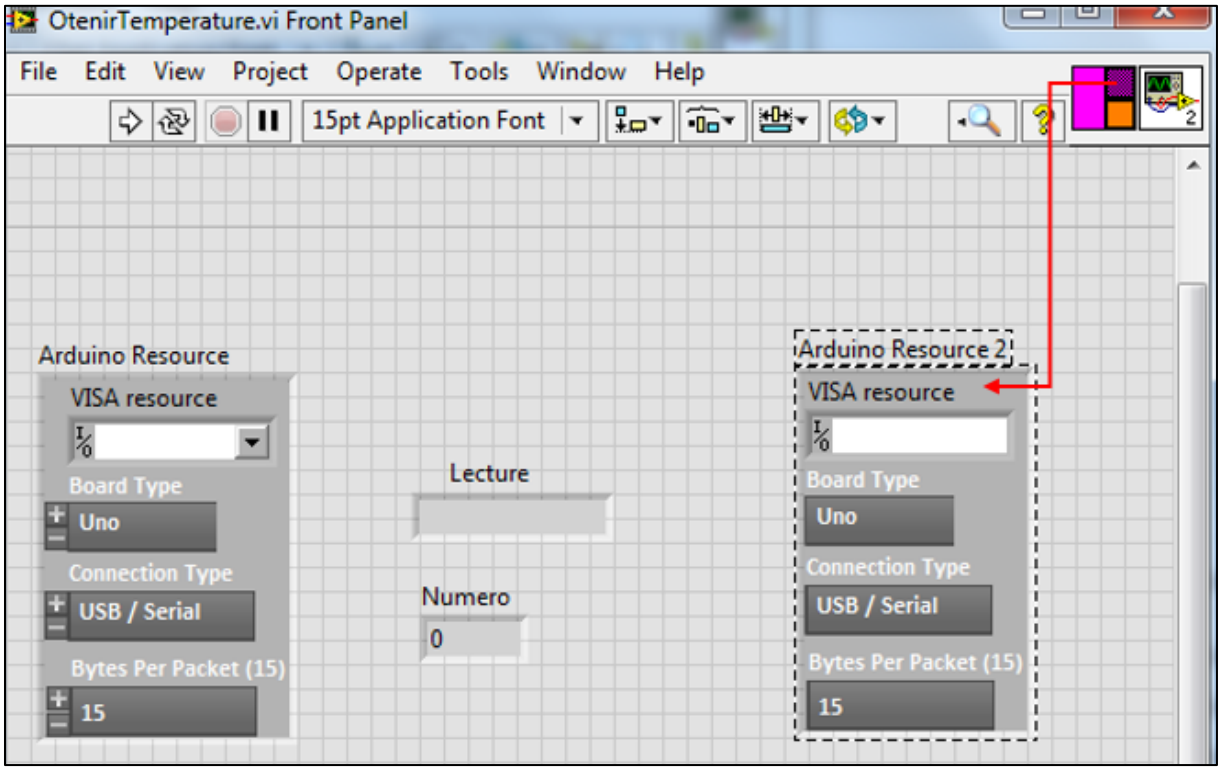


Figure 21 : Arduino Resource 2

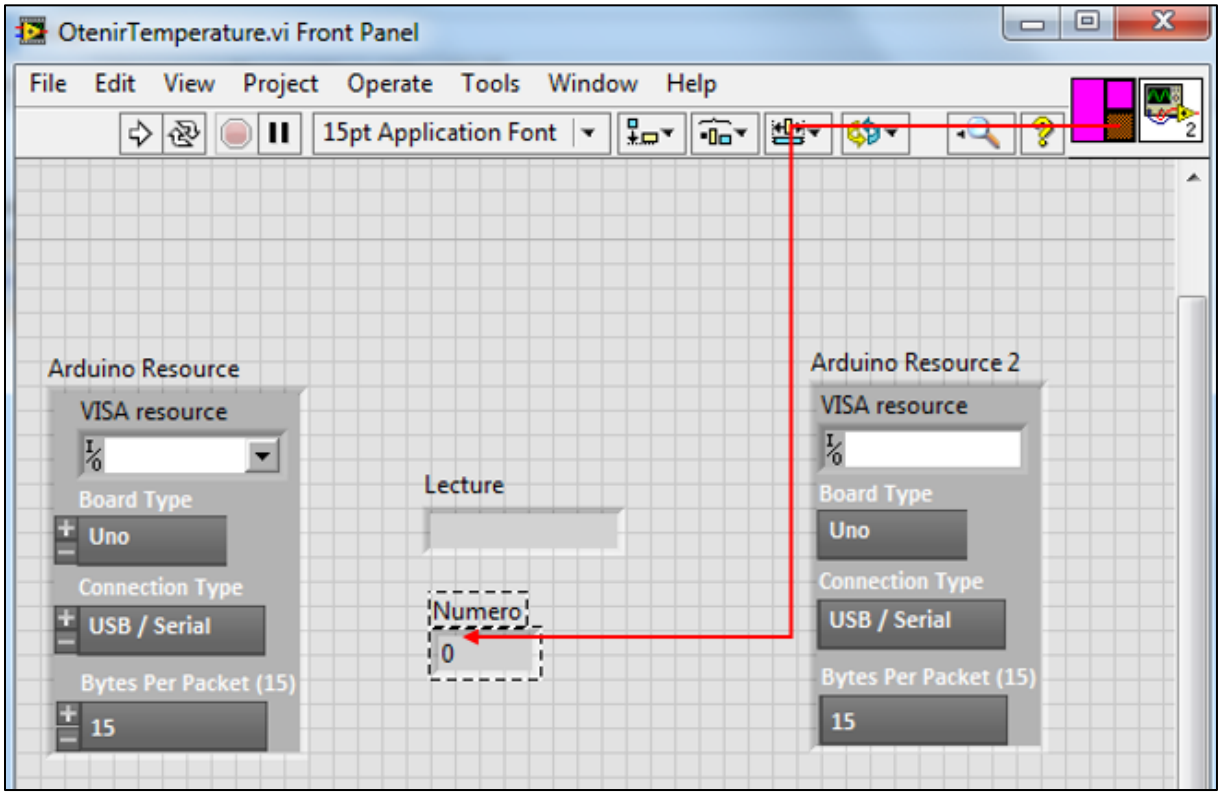


Figure 22 : Numéro

Voici par la suite le projet global sous LabVIEW.

Pour éliminer les erreurs il est utile de vérifier que les fonctions de l'arduino sont bien liées en entrée et en sortie.

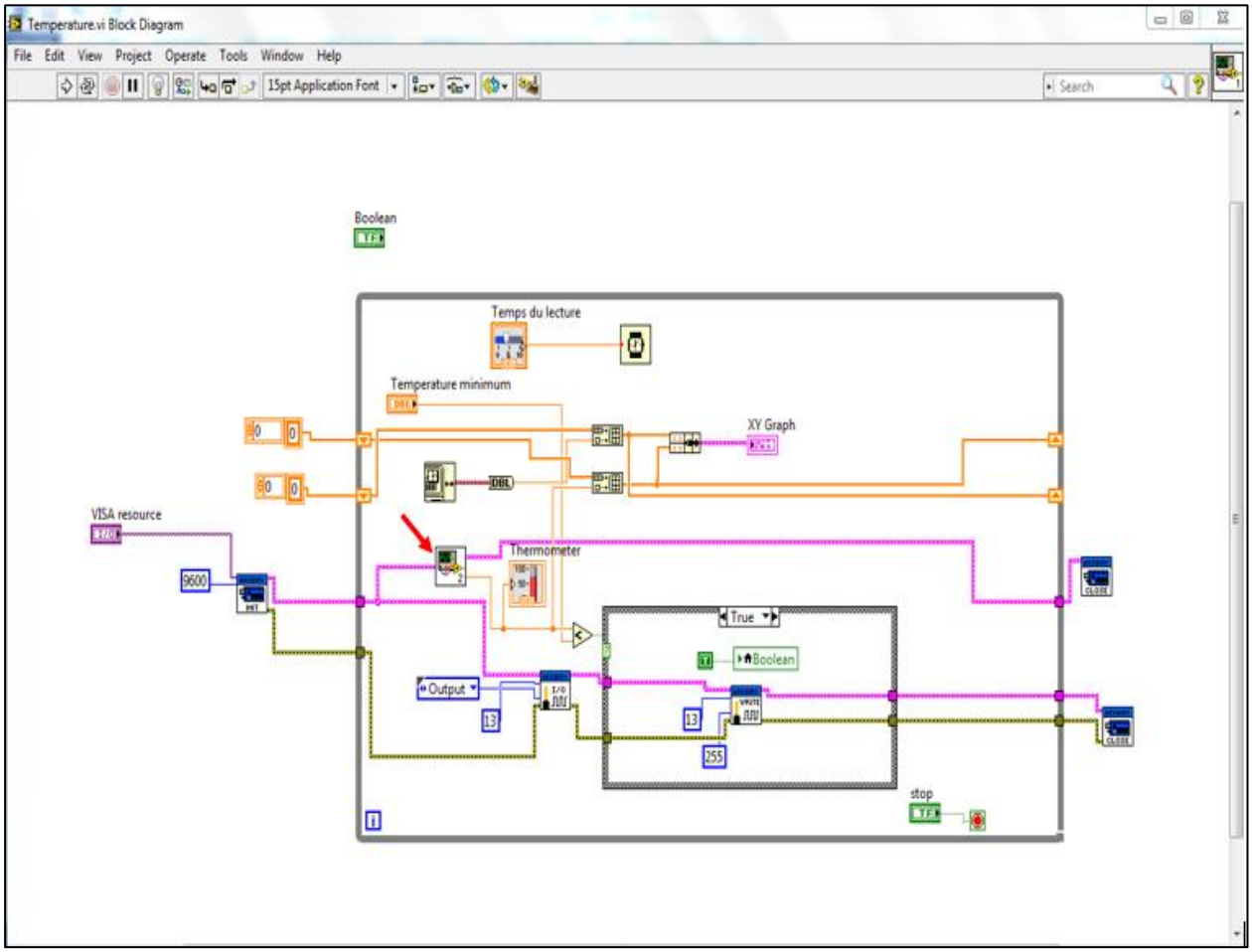


Figure 23 : Block Diagram « Température »

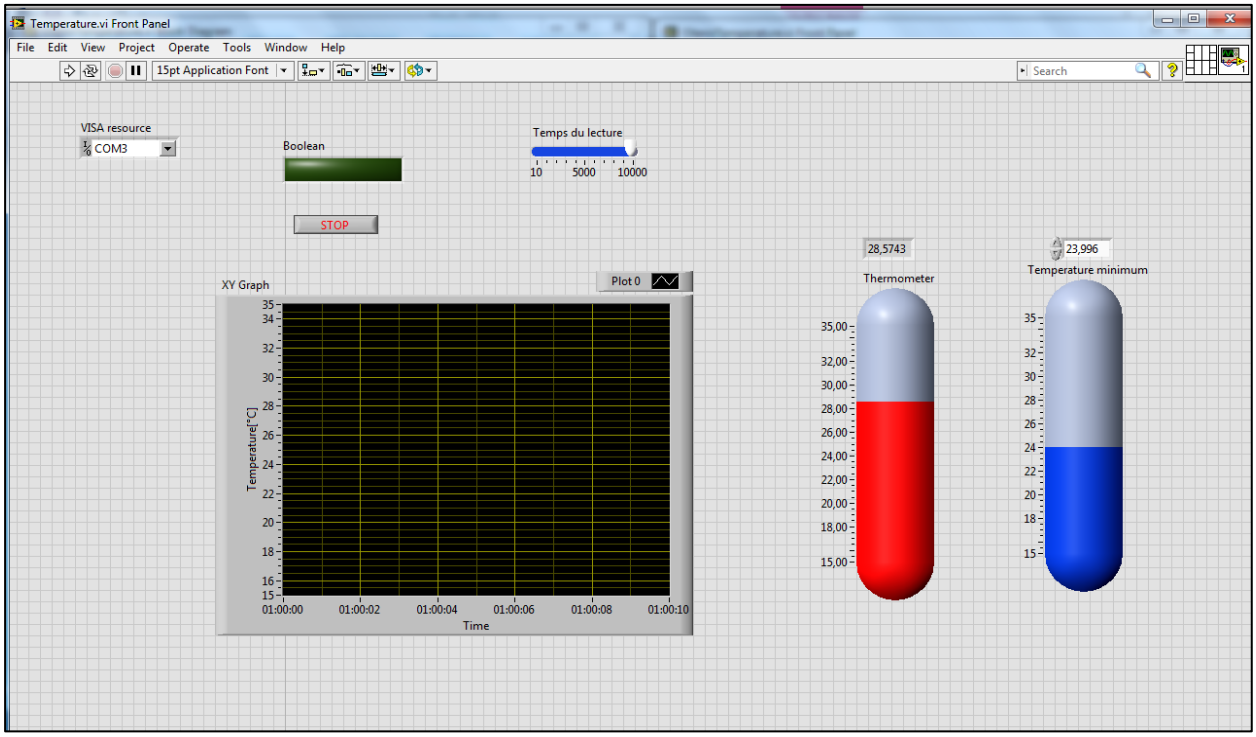


Figure 24 : Front Panel « Température »

V. Communication entre LABVIEW et ISIS

Enfin c'est l'étape de simulation, on va d'abord charger le code sous la carte Arduino.

Après la compilation du code, un fichier (.hex) se produit, on copie l'emplacement du fichier (comme c'est montré par figure) et on le met dans le programme file de la carte Arduino sous ISIS.

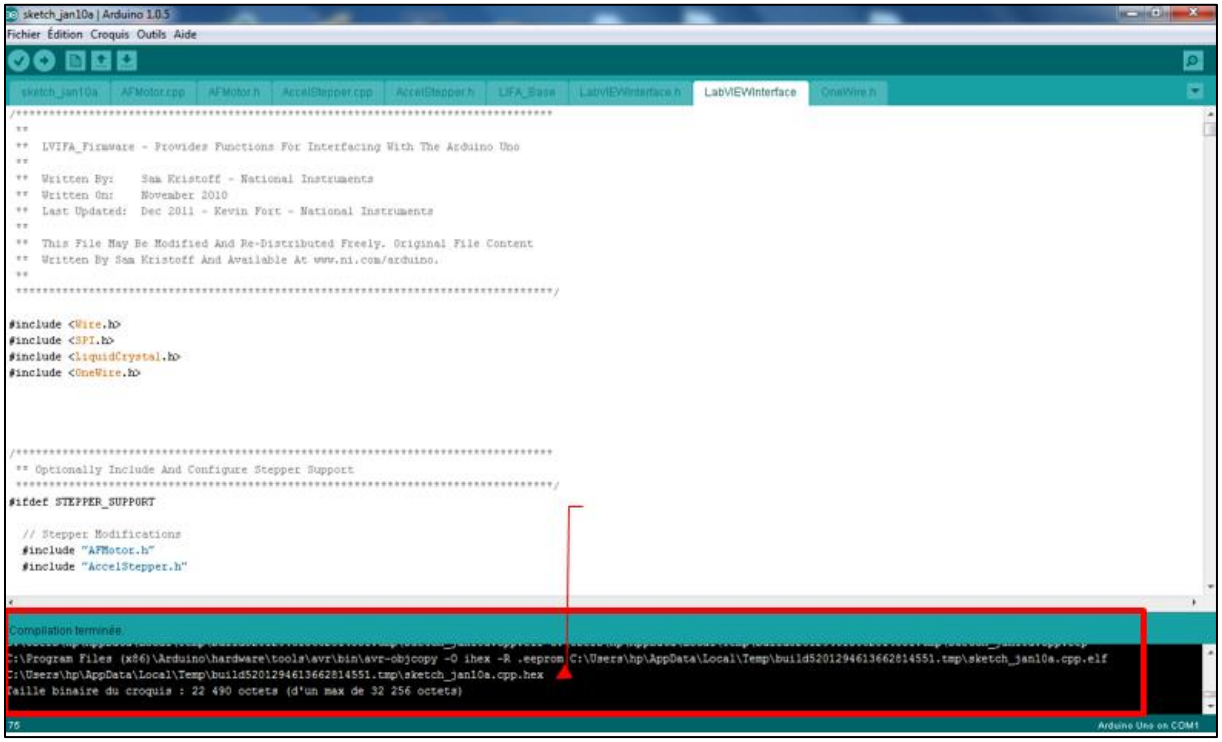


Figure 25 : Fichier (.hex)

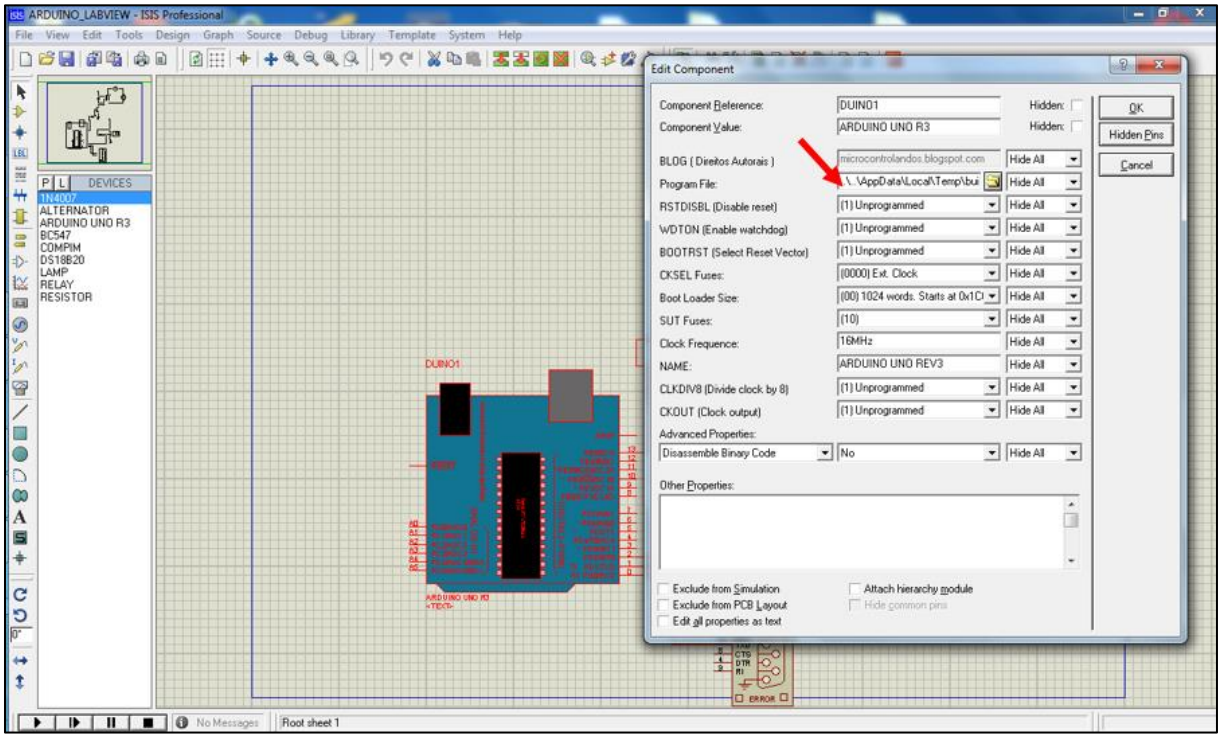


Figure 26 : Chargement du code

Pour faire la communication entre LabVIEW et ISIS, on a utilisé le logiciel « VSPE » (Virtual Serial Ports Emulator).Comme le montre son nom il permet d'avoir une commmunication virtuelle.

- Clique sous l'icône suivante (dans la figure).

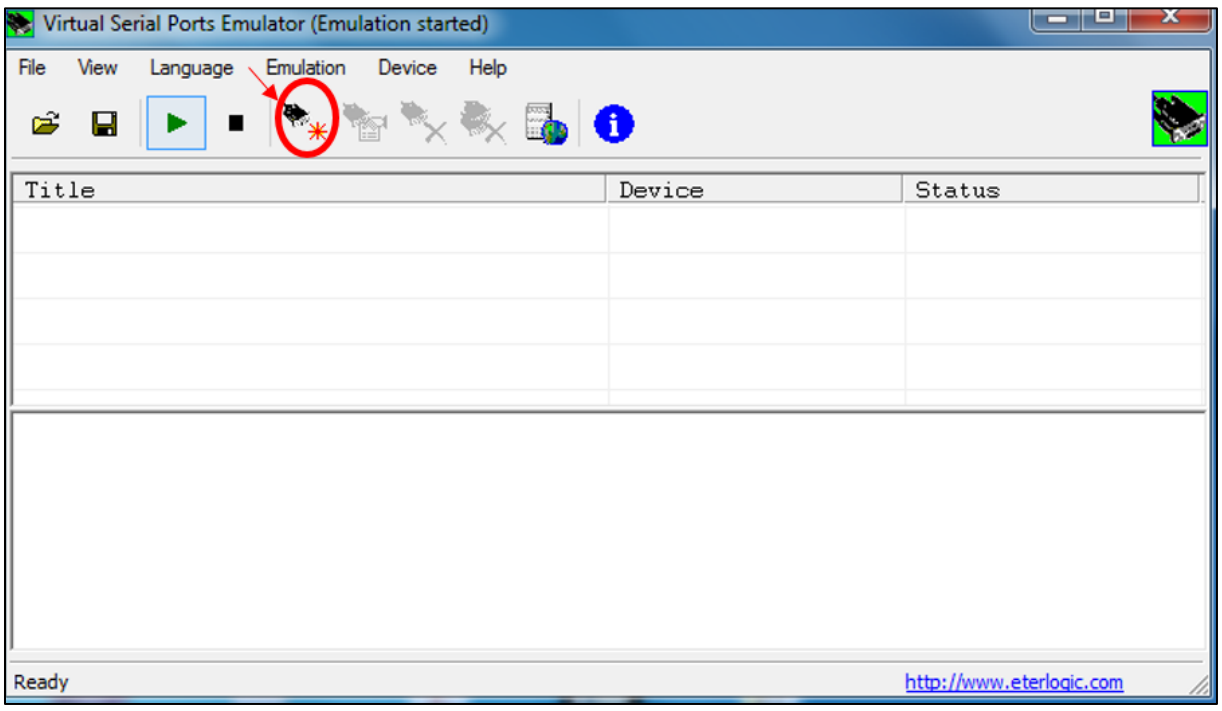


Figure 27 : VSPE

- Choisir Pair.
- Suivant.

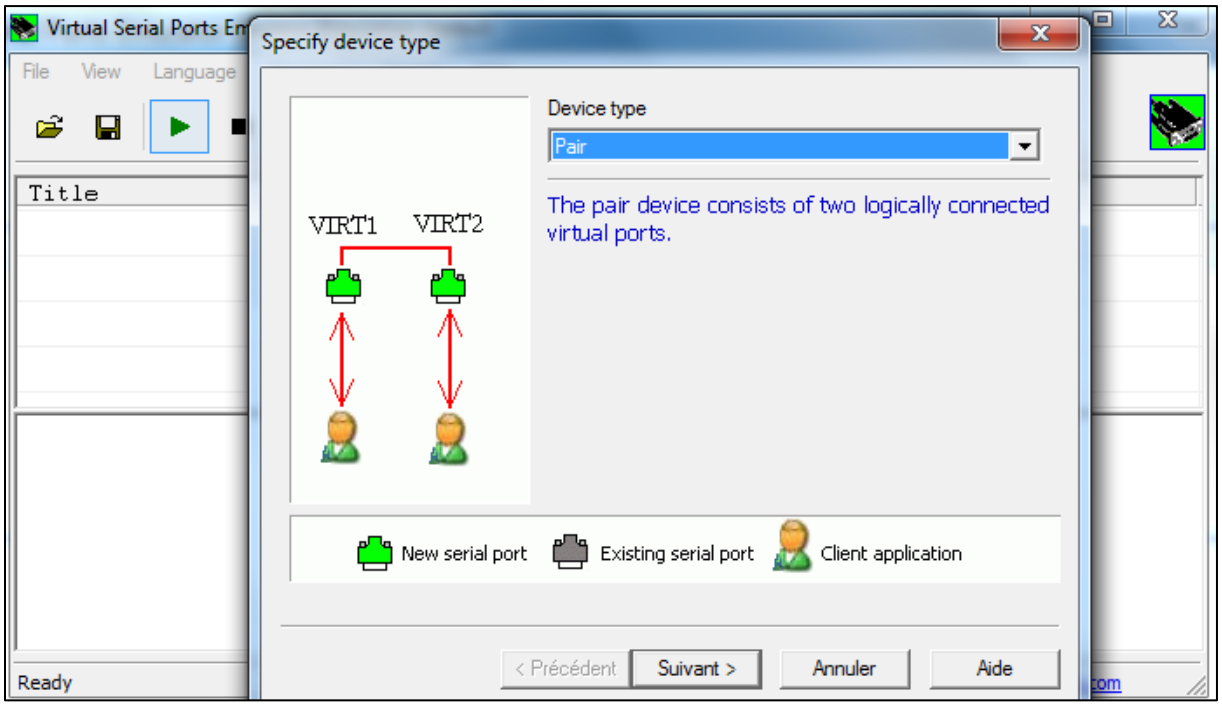


Figure 28: Ports pour ISIS et Arduino(IDE)

- Terminer.

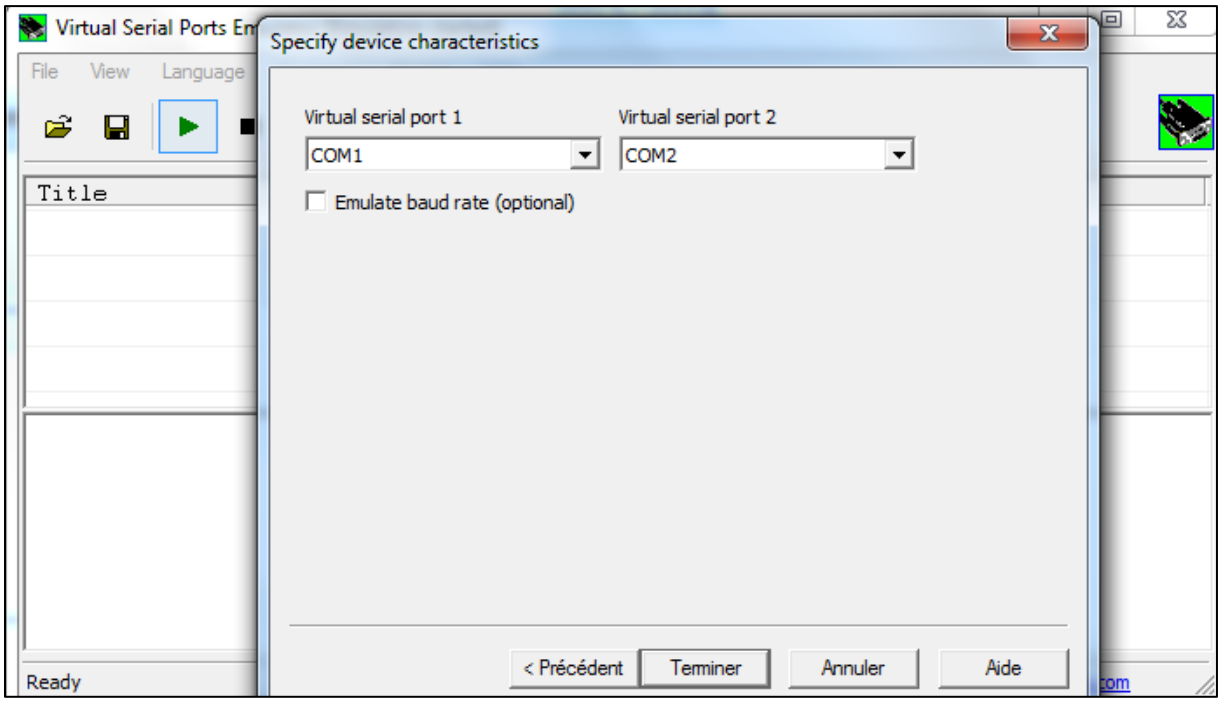


Figure 29 : Ports pour ISIS et Arduino(IDE)

COM1→Arduino(IDE).

COM2→ISIS.

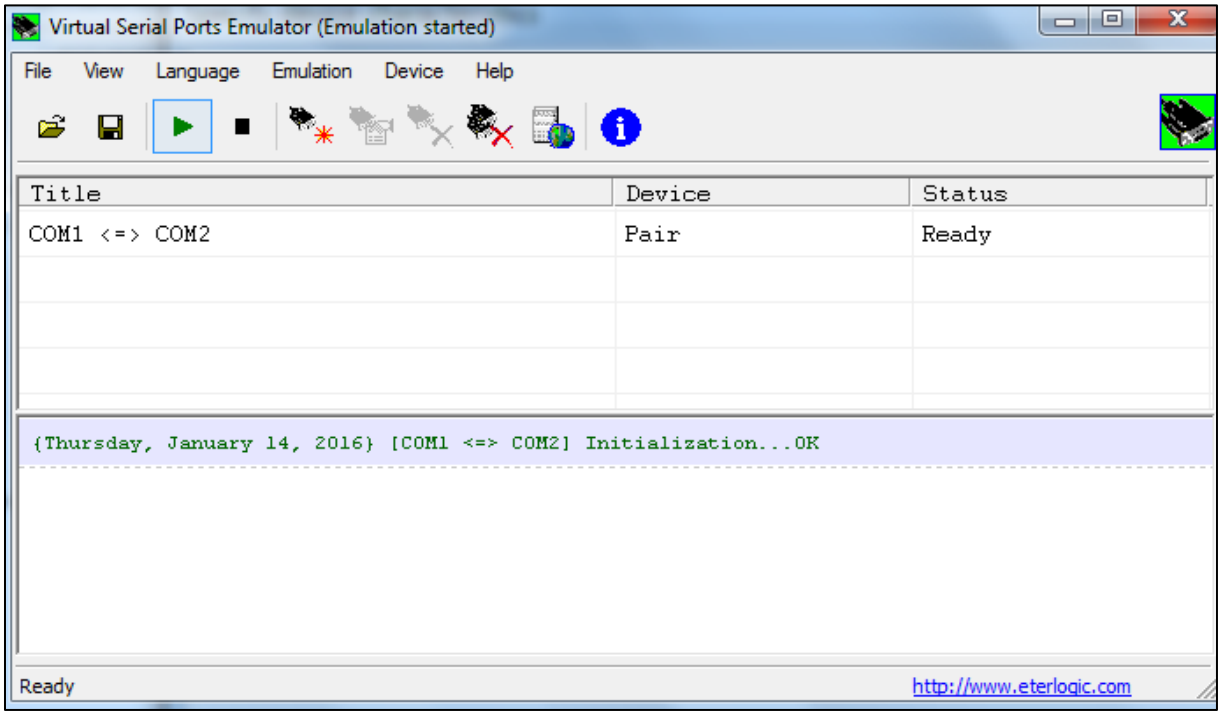


Figure 30 : Ports pour ISIS et Arduino(IDE)

- Connecter.
- Suivant.

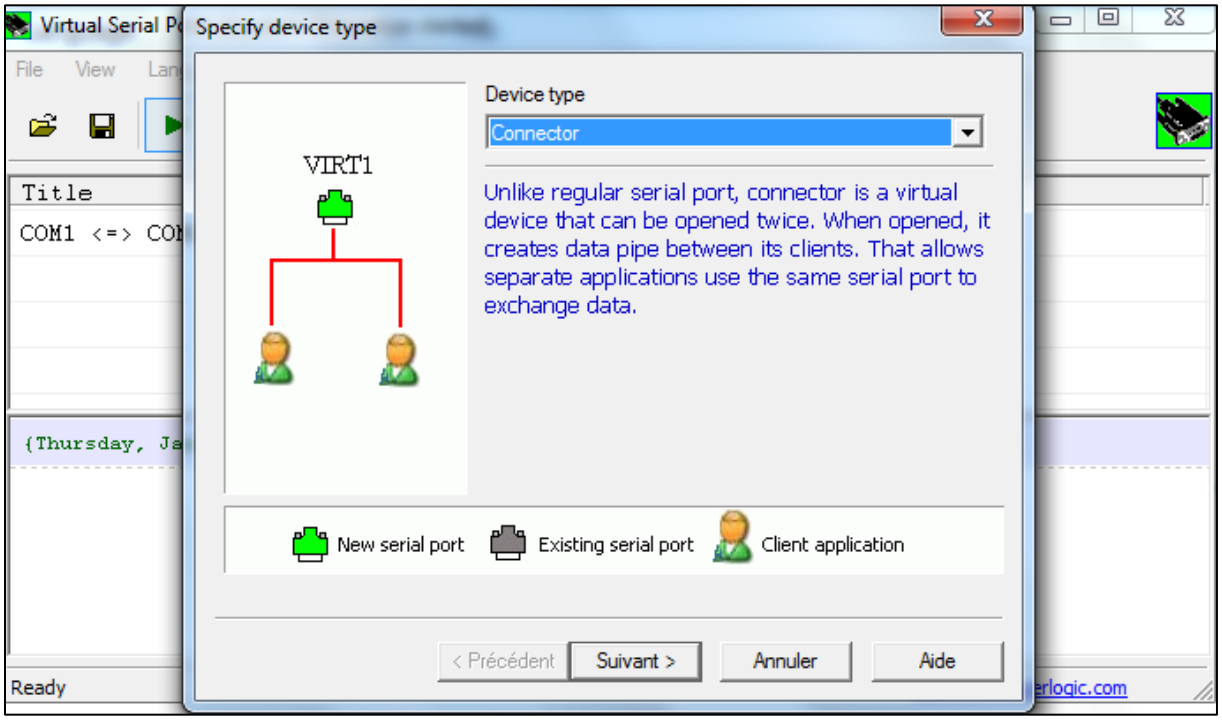


Figure 31 : Port pour LabVIEW

- COM3.
- Terminer.

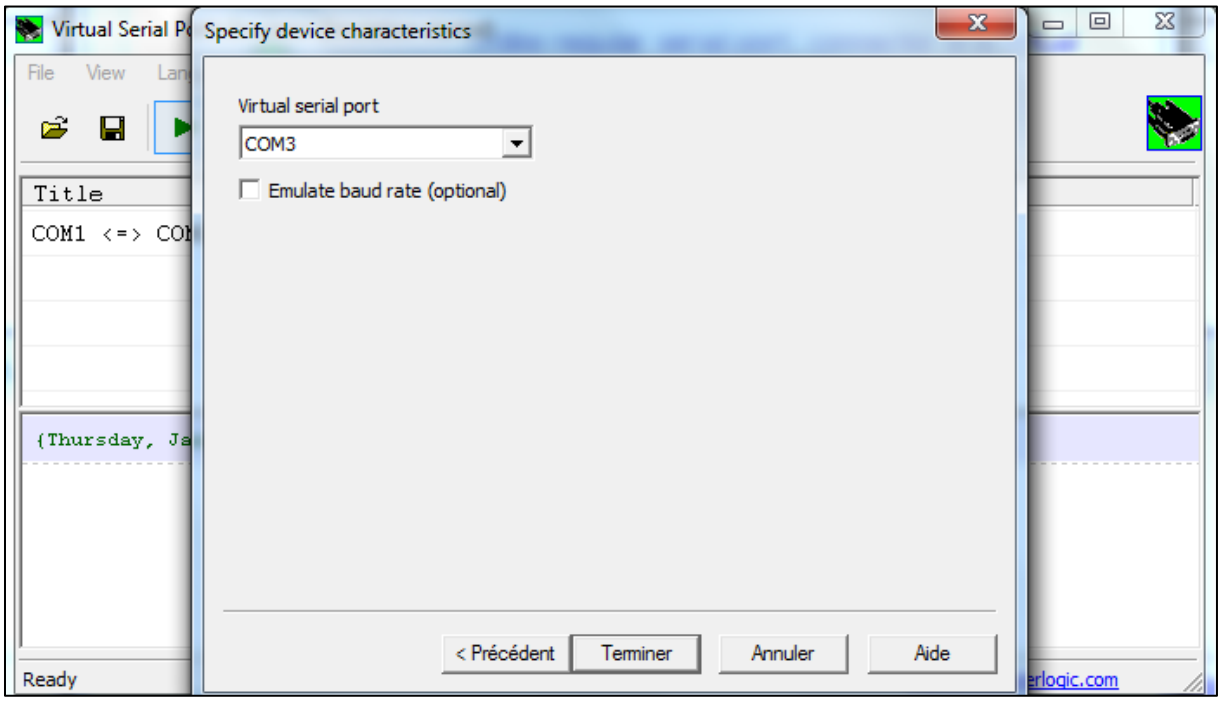


Figure 32 : Port pour LabVIEW

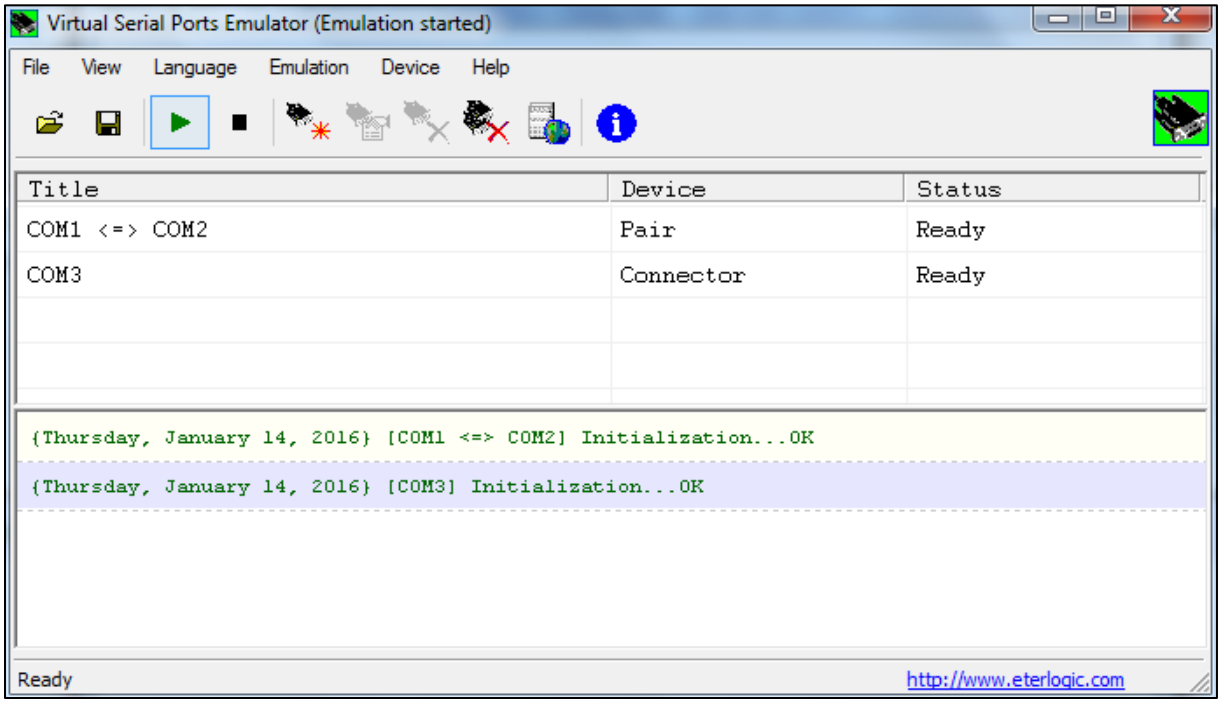


Figure 33 : VSPE(les virtuels ports)

- Double clique COMPIM (schéma ISIS).
- Faire la configuration comme indiqué dans la figure.

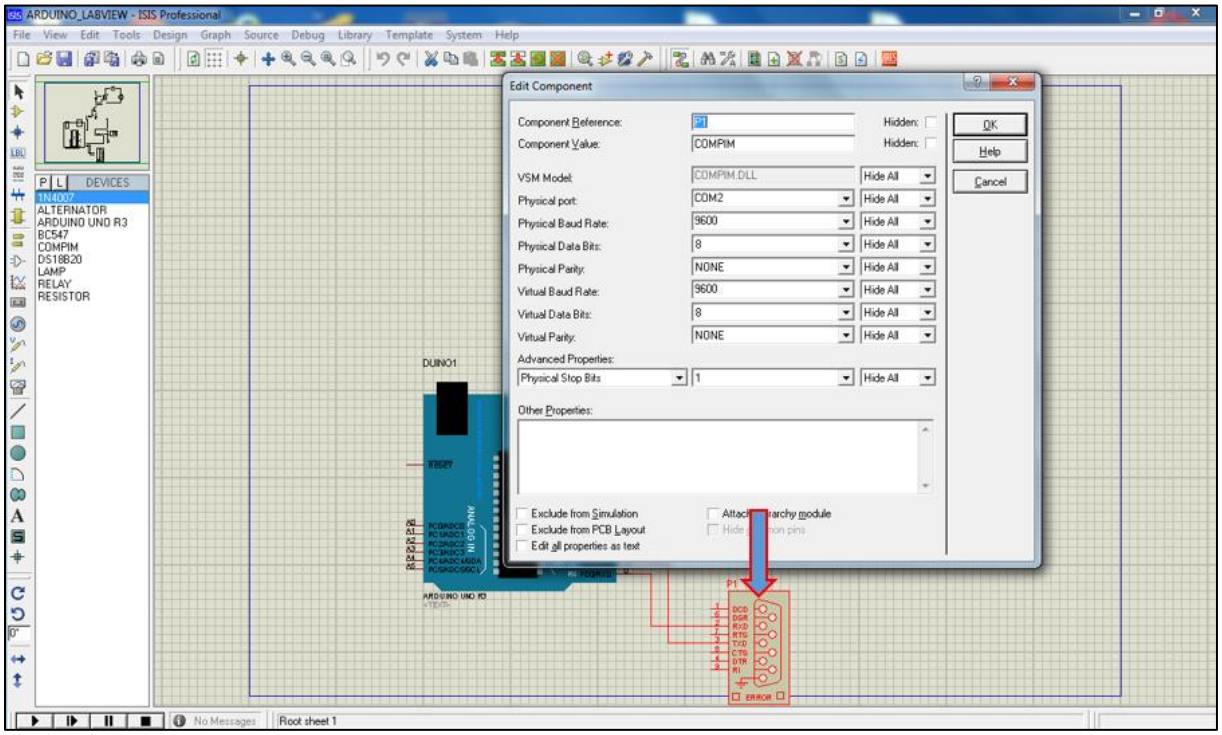


Figure 34 : Configuration du COMPIM

- Sous LabVIEW, choisir COM3.

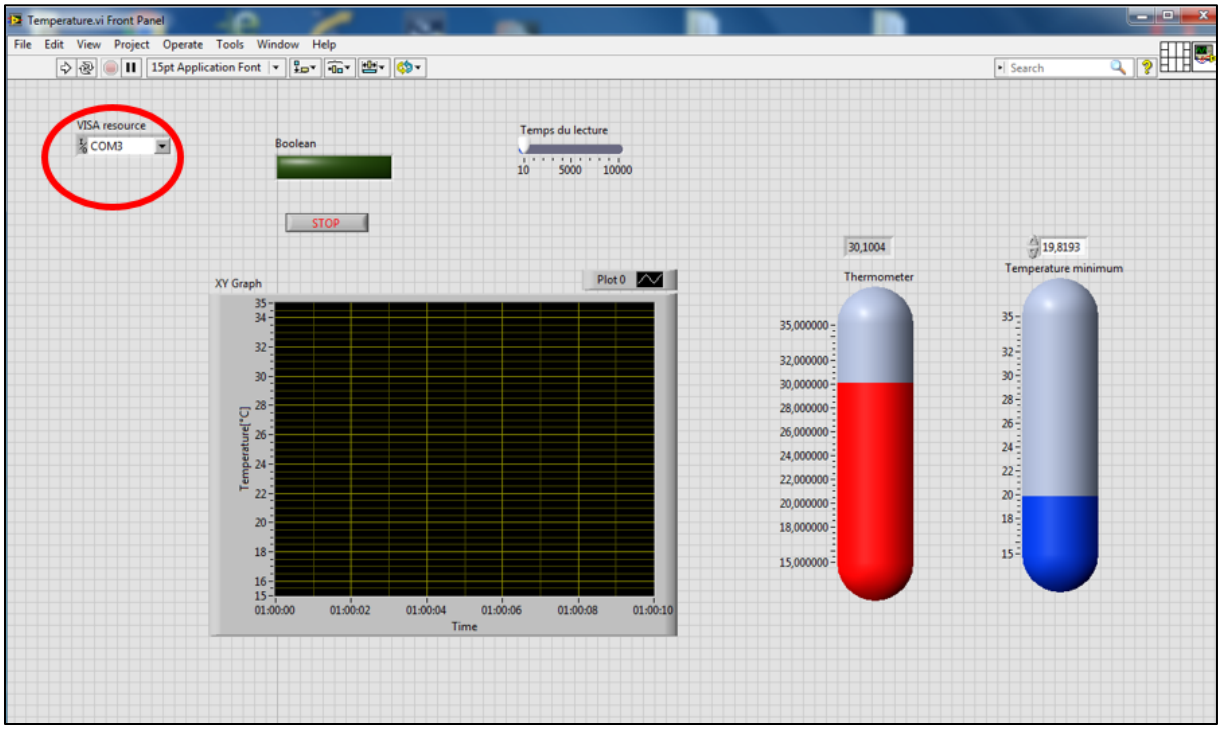


Figure 35 : VISA (COM3)

Conclusion

Ce projet m'a permis de découvrir un nouvel outil de programmation « LabVIEW » qui me semble différent en sa méthode qui se base sur des fonctions en forme des blocs. De plus, l'intégration de la carte Arduino sous LabVIEW m'a permis de manipuler autres logiciels savoir faire la communication entre eux.