Université Mohammed V de Rabat
L´Ecole Normale Supérieure de l'Enseignement Technique

COMPTE RENDU

SUPERVISION DE LA TEMPERATURE PAR ARDUINO SUR LABVIEW

REALISÉ PAR :
EL ASRI *YASSINE*
LAKNIZI *SAMI*
FILIERE:
GÉNIE
ELECTRIQUE

SOUS
LA SUPERVISION DE
Dr. EL YOUSFI ALAOUI

ANNÉE
UNIVERSITAIRE
2019/2020

Introduction :

L'acquisition de données implique la collecte de signaux provenant de sources de mesure et la numérisation des signaux pour le stockage, l'analyse et la présentation sur un PC. Les systèmes d'acquisition de données se présentent sous différentes formes de technologie PC pour offrir une flexibilité lors du choix de votre système. Vous pouvez choisir parmi PCI, PXI, PCI Express, PXI Express, PCMCIA, USB, sans fil et acquisition de données Ethernet pour les applications de test, de mesure et d'automatisation.

L'acquisition des données commence par le phénomène physique à mesurer. Ce phénomène physique pourrait être la température comme notre cas d'étude.
Un transducteur LM35 est un composant qui convertit le phénomène physique de la température en un signal électrique mesurable, tel que la tension. La capacité d'un système d'acquisition de données à mesurer différents phénomènes dépend des transducteurs pour convertir les phénomènes physiques en signaux mesurables par le matériel d'acquisition de données. Les transducteurs sont synonymes de capteurs dans les systèmes d'acquisition de données.

LABVIEW :

LabVIEW (abbreviation de Laboratory Virtual Instrumentation Engineering Workbench) est un plate-forme et environnement de développement pour un langage de programmation visuel de National Instruments. Le langage graphique est nommé "G".
Initialement publié pour Apple Macintosh en 1986, LabVIEW est couramment utilisé pour l'acquisition de données, le contrôle d'instruments et automatisation industrielle sur une variété de plates-formes, y compris Microsoft Windows.
LabVIEW lie la création d'interfaces utilisateur (appelées faces-avant) au cycle de développement.
Les programmes / sous-programmes LabVIEW sont appelés instruments virtuels (VI). Chaque VI a trois composants: un schéma fonctionnel, un panneau avant et un panneau de connecteurs. Le dernier sert à représenter le VI dans les schémas fonctionnels des autres, appelant des VIs. Commandes et indicateurs sur le panneau avant permet à un opérateur d'entrer des données dans un instrument virtuel. Cependant, le panneau avant peut également servir d'interface de programmation.

Un des avantages de LabVIEW par rapport aux autres environnements de développement est le support étendu d'accéder au matériel d'instrumentation. Pilotes et couches d'abstraction pour de nombreux types différents des instruments et des bus sont inclus ou sont disponibles pour inclusion. Ces présents eux-mêmes comme des nœuds graphiques. Les couches d'abstraction offrent des interfaces logicielles standard pour communiquer avec des périphériques matériels.

Carte Arduino :

Les E/S de la carte Arduino sont réparties comme suit :
Une série de 12 Pin E/S digitales numérotés de 2 à 13, ayant les caractéristiques suivantes :

- chaque Pin peut être déclaré comme une entrée ou une sortie ;
- les Pin dont le numéro est précédé de « ~ » sont pilotables en PWM.

Une série de Pin POWER, composée de :

- une sortie 5 V ;
- une sortie 3,3 V ;
- deux GND
- une entrée en tension Vin
- un Reset.

Une série de 6 Pin ANALOG INPUT, numérotés de A0 à A5 ;

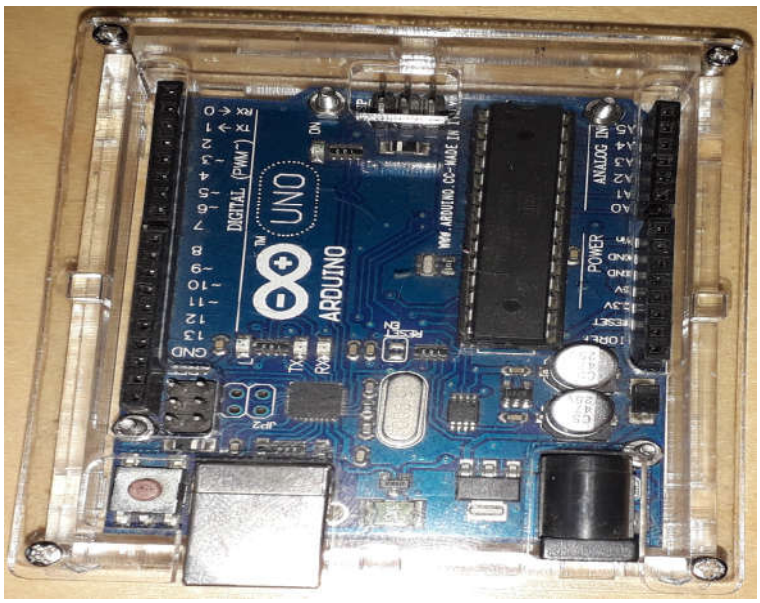


Figure : Carte Arduino UNO

Objectif su TP :

Réaliser une solution Hard et Soft pour acquérir en temps réel un signal d'un capteur de température, en utilisant la carte Arduino UNO, un capteur de température LM35 ou un potentiomètre et une interface de supervision et de traitement de signal sur le logiciel LABVIEW.
Il existe plusieurs solutions pour remédier à cette problématique:
La première c'est d'exploiter les bibliothèques internes de l'Arduino sur le logiciel LABVIEW.
Cette solution est rapide, fiable et plus pratique grâce aux différents objets de la bibliothèque Arduino, ce qui rend la programmation assez simple et facile.
La deuxième solution consiste a faire une communication UART direct entre l'Arduino et le PC, cette méthode permette la lecture du port série de façon de lire les trames et faire une mise a l'échelle pour déduire la valeur transmise avant la projeter sur l'interface homme machine du LABVIEW.
Cette dernière méthode est plus compliquer que la première, et donne parfois des retards de la détection de la carte, mais les résultats sont toujours les mêmes.

A-Méthode LIFA :

1-Chargement des codes LIFA Base sur Arduino

LIFA BASE CODE :

```
#include <Wire.h>
#include <SPI.h>
#include <Servo.h>
#include "LabVIEWInterface.h"

void setup()
{
 // Initialize Serial Port With The Default Baud Rate
 syncLV();

 // Place your custom setup code here
 }
void loop()
{
 // Check for commands from LabVIEW and process them.

 checkForCommand();
 // Place your custom loop code here (this may slow down communication with LabVIEW)

  if(acqMode==1)
 {
  sampleContinously();
 }
}
```

LABVIEW INTERFACE CODE :

```
#define FIRMWARE_MAJOR 02
#define FIRMWARE_MINOR 00
#if defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
#define DEFAULTBAUDRATE 9600    // Defines The Default Serial Baud Rate (This must match the baud rate specifid in LabVIEW)
#else
#define DEFAULTBAUDRATE 115200
#endif
#define MODE_DEFAULT 0          // Defines Arduino Modes (Currently Not Used)
#define COMMANDLENGTH 15        // Defines The Number Of Bytes In A Single LabVIEW Command (This must match the packet size specifid in LabVIEW)
#define STEPPER_SUPPORT 1       // Defines Whether The Stepper Library Is Included - Comment This Line To Exclude Stepper Support


// Declare Variables
```

```
unsigned char currentCommand[COMMANDLENGTH];    // The Current Command For The Arduino To Process
//Globals for continuous aquisition
unsigned char acqMode;
unsigned char contAcqPin;
float contAcqSpeed;
float acquisitionPeriod;
float iterationsFlt;
int iterations;
float delayTime;

void syncLV();
void setMode(int mode);
int checkForCommand(void);
void processCommand(unsigned char command[]);
void writeDigitalPort(unsigned char command[]);
void analogReadPort();
void sevenSegment_Config(unsigned char command[]);
void sevenSegment_Write(unsigned char command[]);
void spi_setClockDivider(unsigned char divider);
void spi_sendReceive(unsigned char command[]);
unsigned char checksum_Compute(unsigned char command[]);
int checksum_Test(unsigned char command[]);
void AccelStepper_Write(unsigned char command[]);
void sampleContinously(void);
void finiteAcquisition(int analogPin, float acquisitionSpeed, int numberOfSamples );
void lcd_print(unsigned char command[]);
```

Les deux codes ci-dessus sont des codes qui se trouve dans un repertoire du disque C ou se trouve l'Install du logiciel LABVIEW, ainsi d'autre code pour les moteurs, commandes infrarouge et interfaçage LABVIEW.

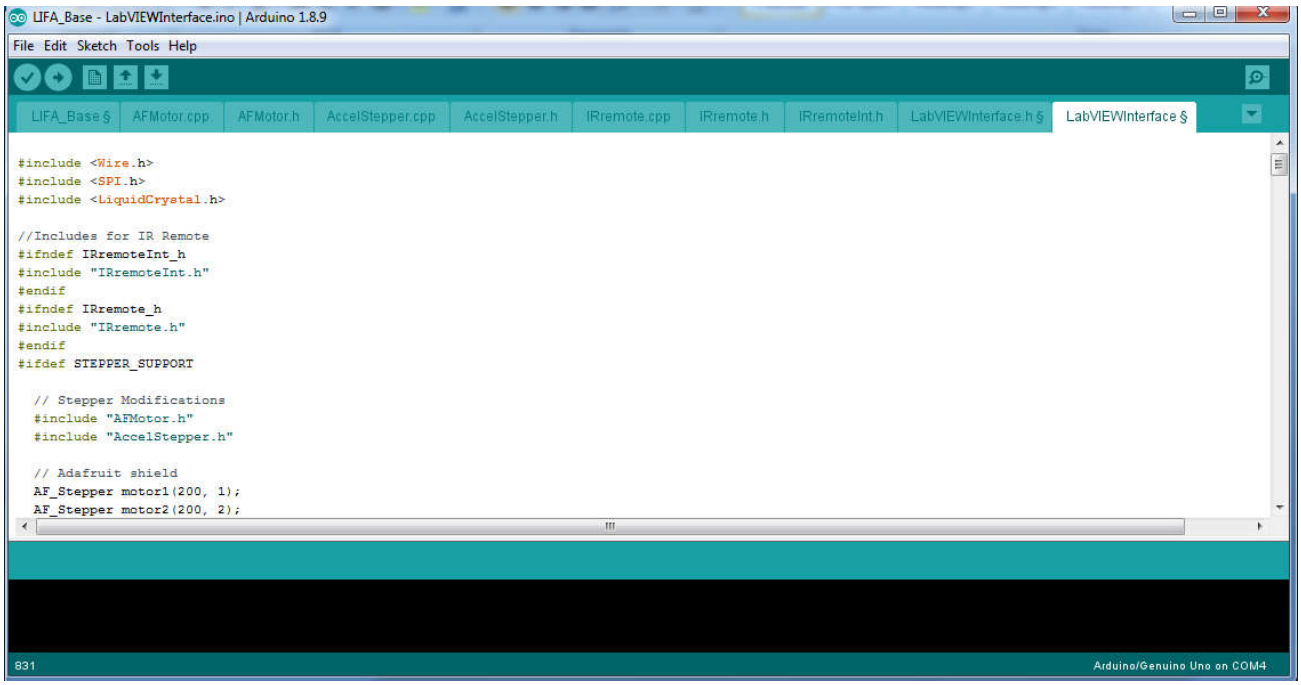


Figure : Upload du programme LIFA BASE sur la carte Arduino Uno

2-Réalisation de l'interface homme machine sur LABVIEW

L'interface Homme machine ou la face avant du logiciel est la seule façon pour communiquer, recevoir et envoyer des commandes sans besoin d'être connaisseur dans le domaine, ce qui implique qu'elle doit être plus détailler et significatif en terme d'informations.
Pour notre application, on a décidé de crée une interface de température avec des indicateurs analogique, et numérique ainsi l'ajout des entrées pour entrée des réglages qui supervise la dégrée de criticité de la température. Dans notre exemple, on a choisie deux seuils, le premier pour faire sortir une alarme qui signal le premier degré de danger, et la deuxième pour alarmer l'état critique du system.

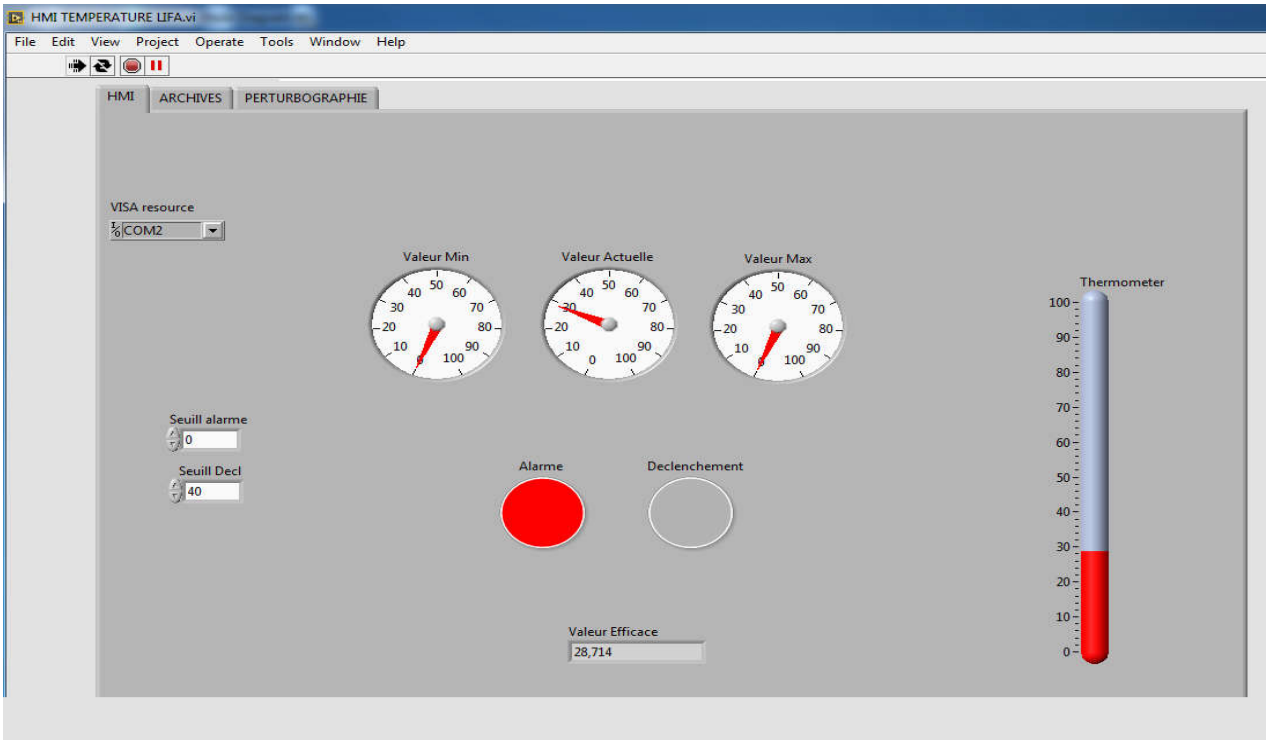


Figure : Interface pour affichage des indicateurs sur HMI

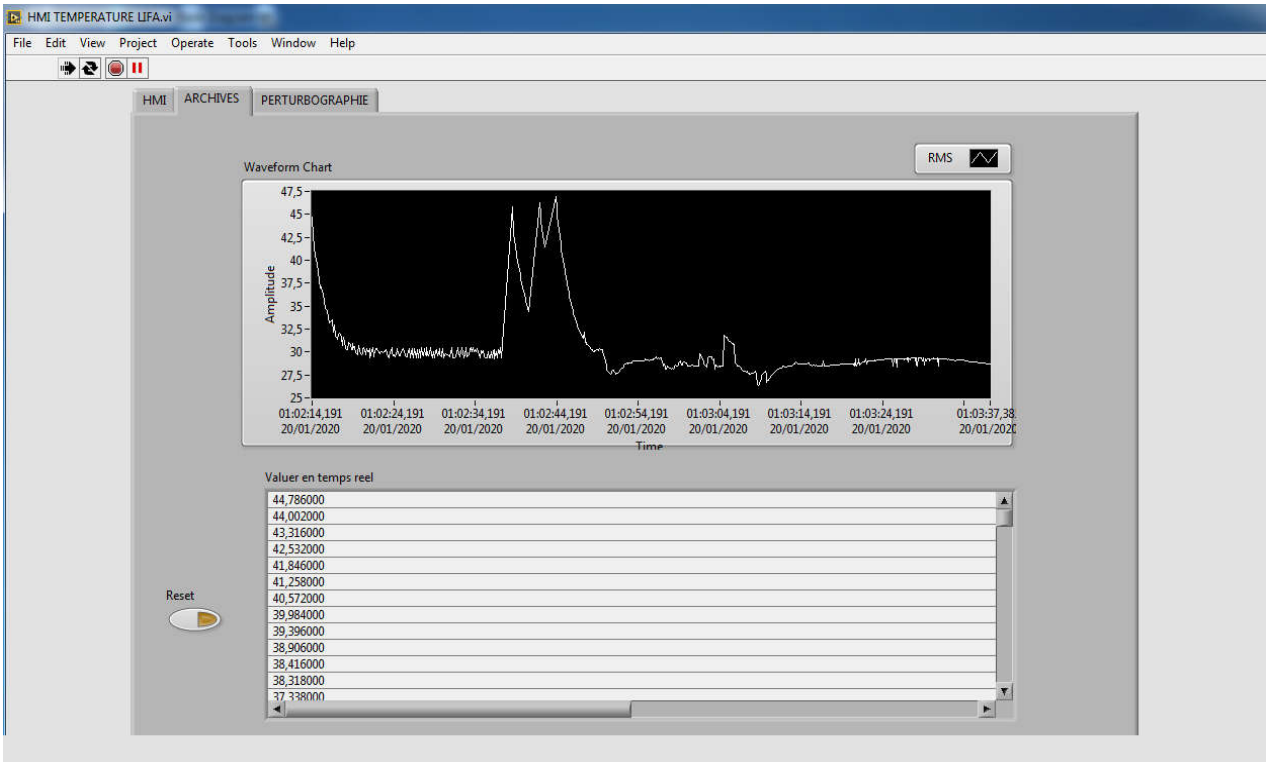


Figure : Interface pour tracer et afficher en temps réel les valeurs analogique

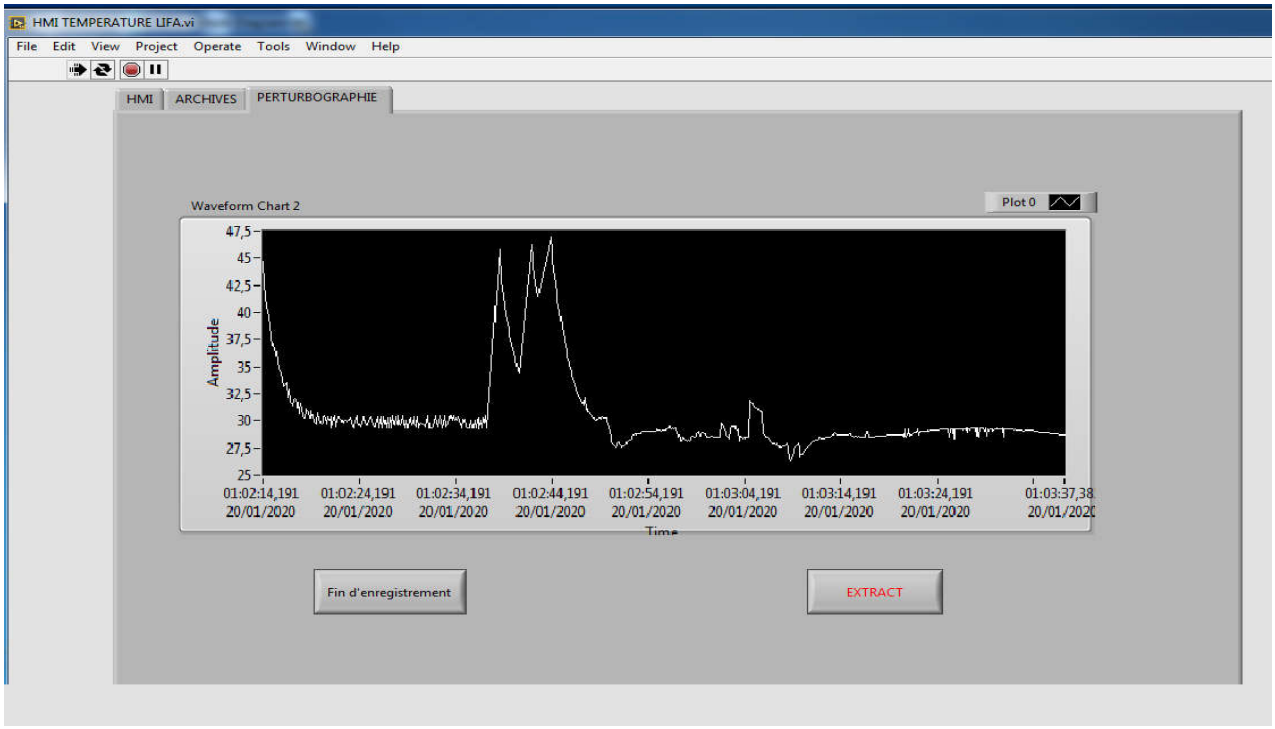


Figure : Interface pour extraire les anciennes valeurs enregistrées sur un fichier Exel

Les interfaces sont très basiques et elles répondent strictement au cahier des charges, seulement pour faire apparaitre le fonctionnement de l'acquisition des données. En faite on peux créer d'autres options tel que la sortie d'une alarme sonore du pc ou faire apparaitre des messages d'erreur sur l'interface après chaque dépassement de la température des seuils raisonnables ou bien envoyer des SMS sur mobile.

3-Réalisation du programme par les bloc des bibliothèques LABVIEW :

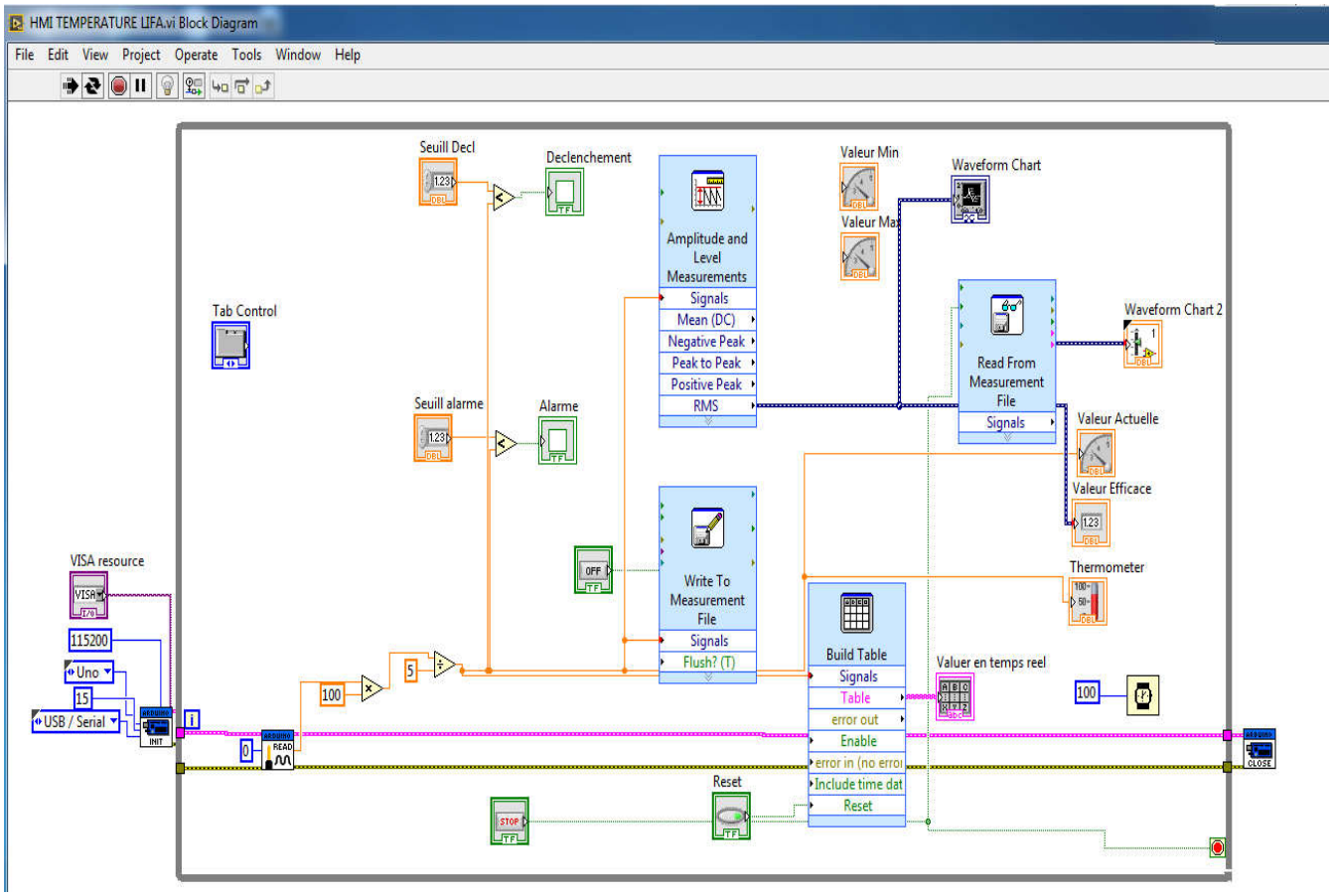


Figure : Interface pour extraire les anciennes valeurs enregistrées sur un fichier Exel

A-Méthode UART :

1-Chargement du code d'interface labview/Arduino

ARDUINO CODE :

```
char command;
String string;
#define led 13
#define lm A1

 void setup()
 {
  Serial.begin(9600);
  pinMode(led, OUTPUT);
 }

 void loop()
 {
  if (Serial.available() > 0)
  {string = "";}
  while(Serial.available() > 0)
  {
   command = ((byte)Serial.read());
   if(command == ':')
   {
    break;
   }
   else
   {
    string += command;
   }
   delay(1);
  }
   if(string == "TO")
  {
   TempOn();
  }
  if(string =="TF")
  {
   TempOff();
  }
}
  void TempOn()
  {
      int x = analogRead(lm);
      float temp = (5.0*x*100.0)/1024.0;
      Serial.println(temp);
      digitalWrite(led, HIGH);
      delay(500);
  }
     void TempOff()
  {
   digitalWrite(led, LOW);
   delay(500);
  }
```

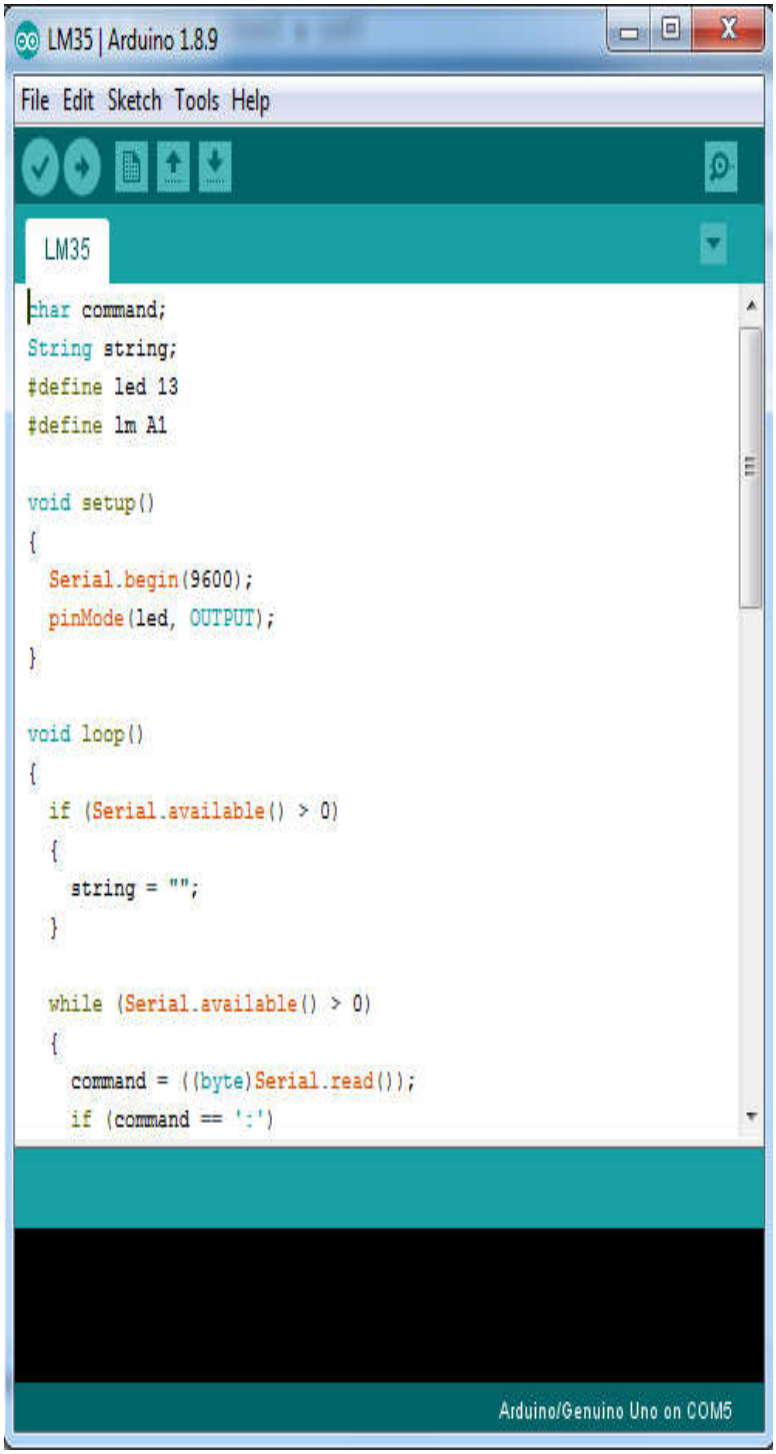


Figure : Upload du programme Uart sur la carte Arduino Uno

2-Réalisation de l'interface homme machine sur LABVIEW

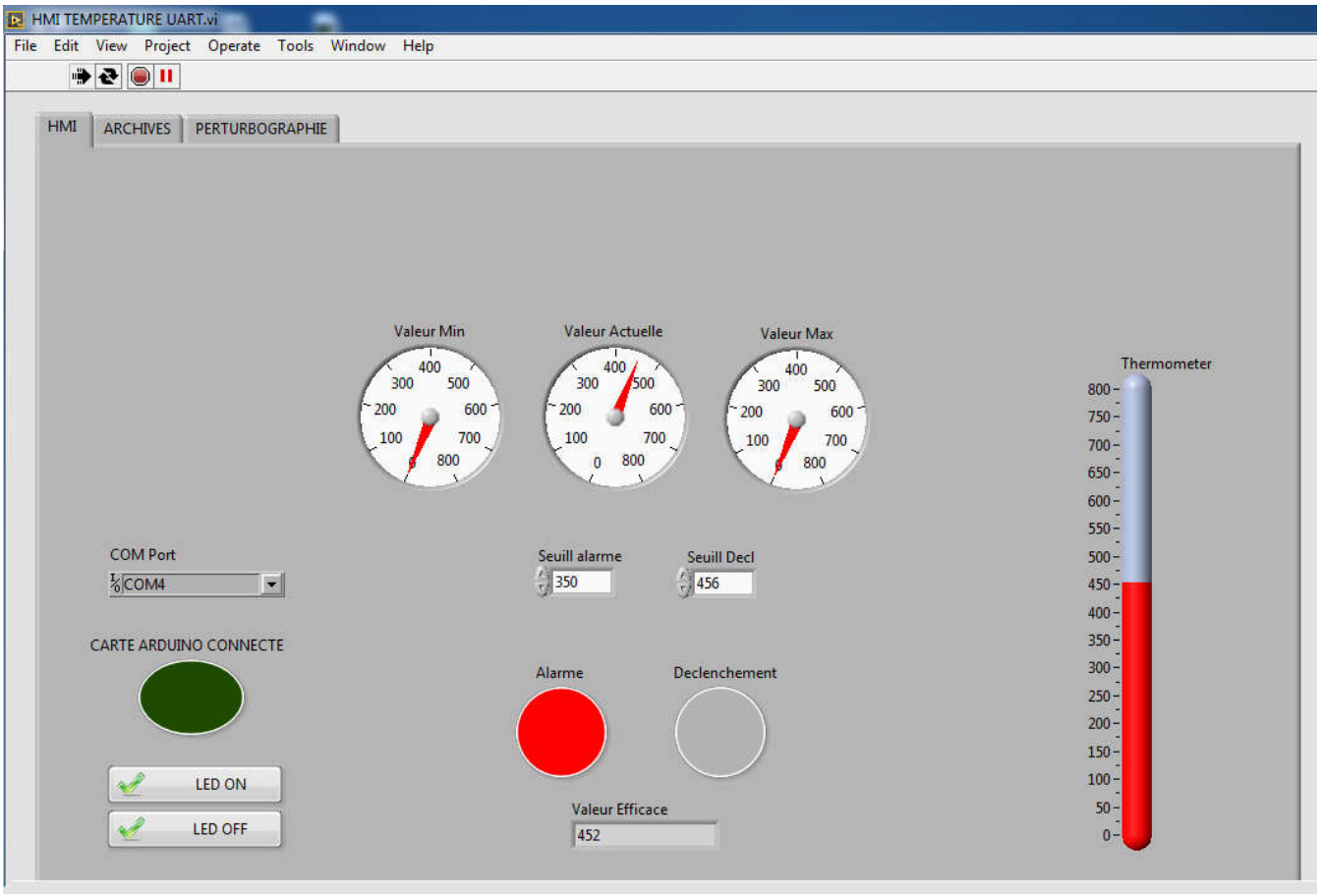
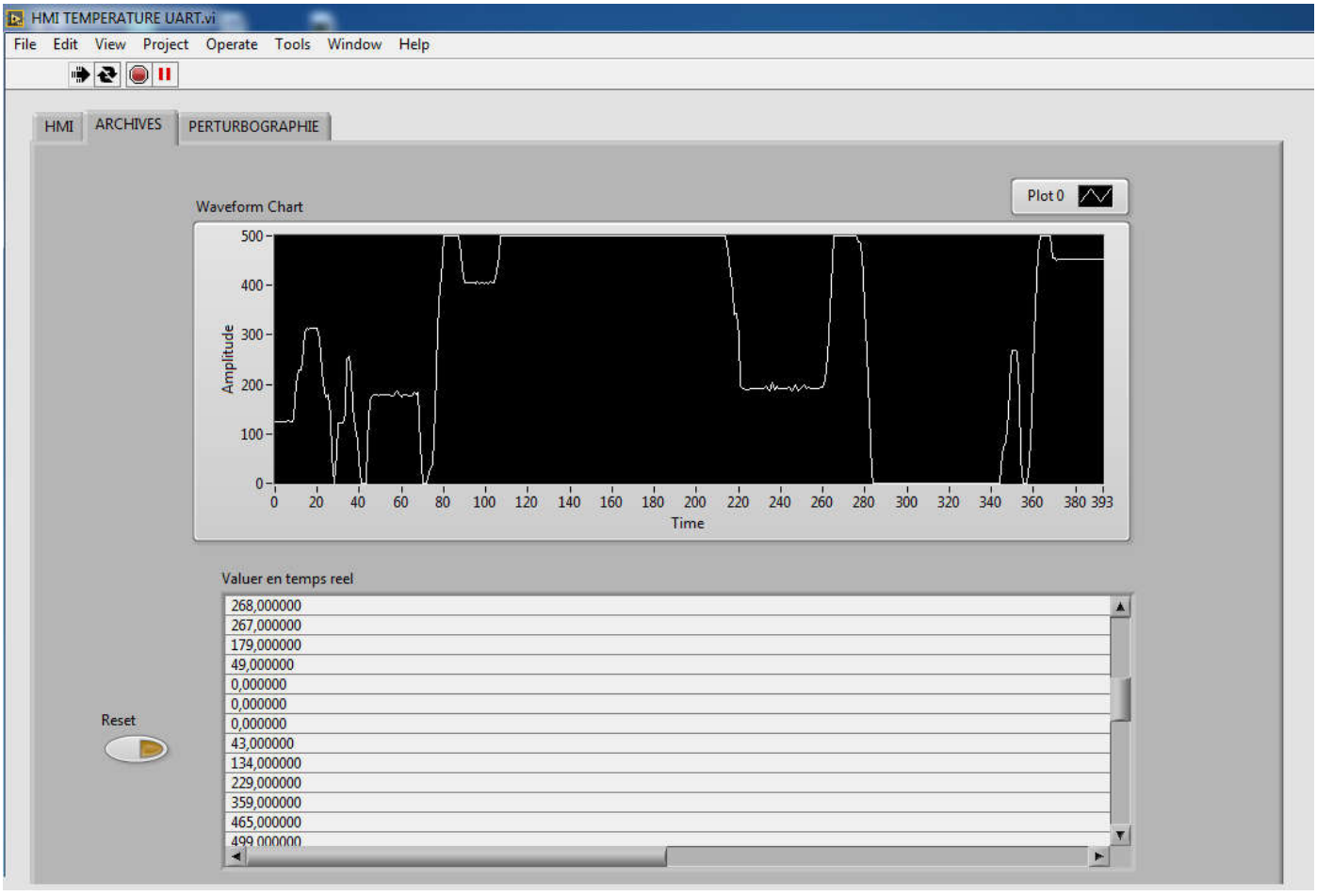


Figure : Interface pour affichage des indicateurs sur HMI



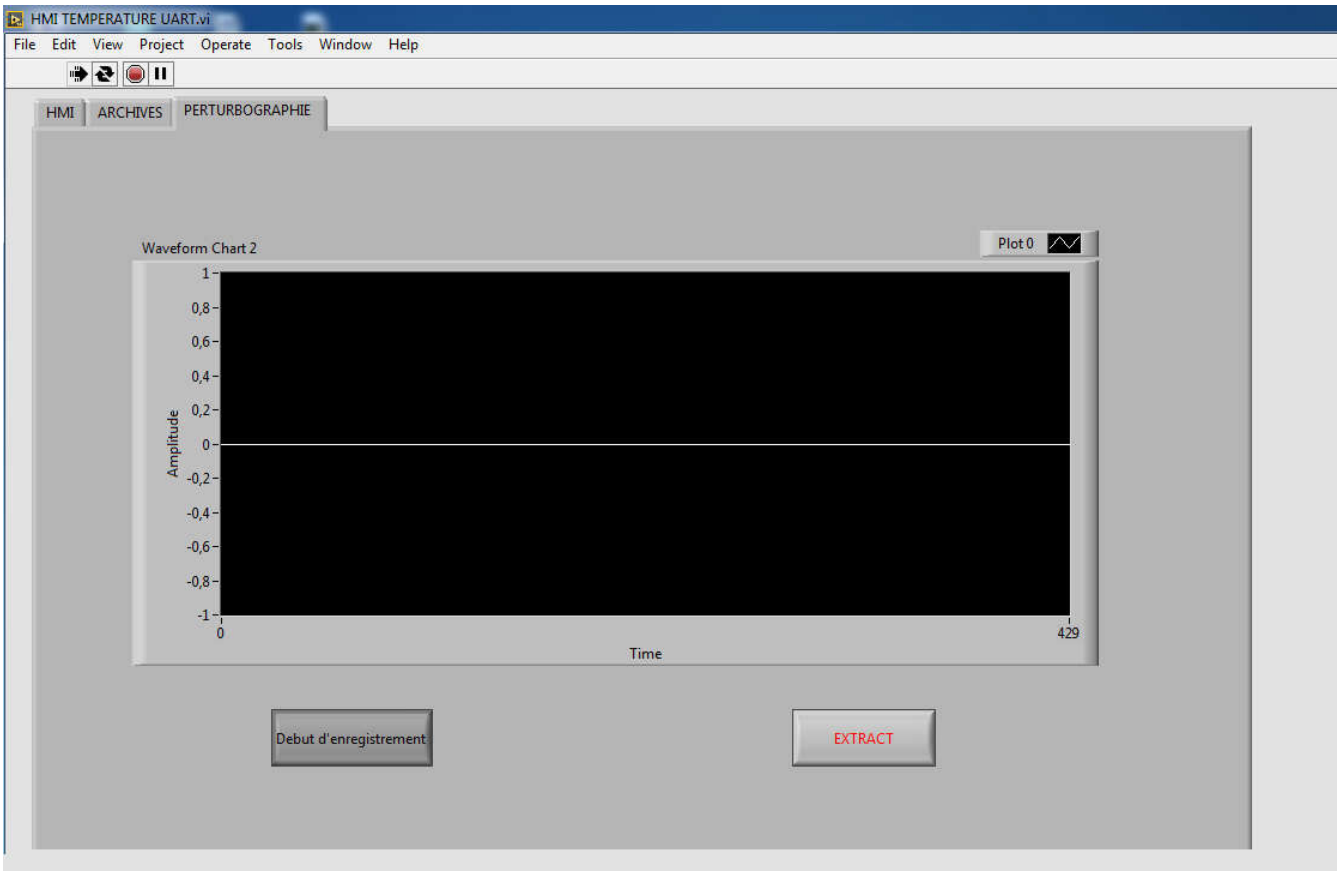


Figure : Interface pour extraire les anciennes valeurs enregistrées sur un fichier Exel

3-Réalisation du programme par les bloc des bibliothèques LABVIEW :

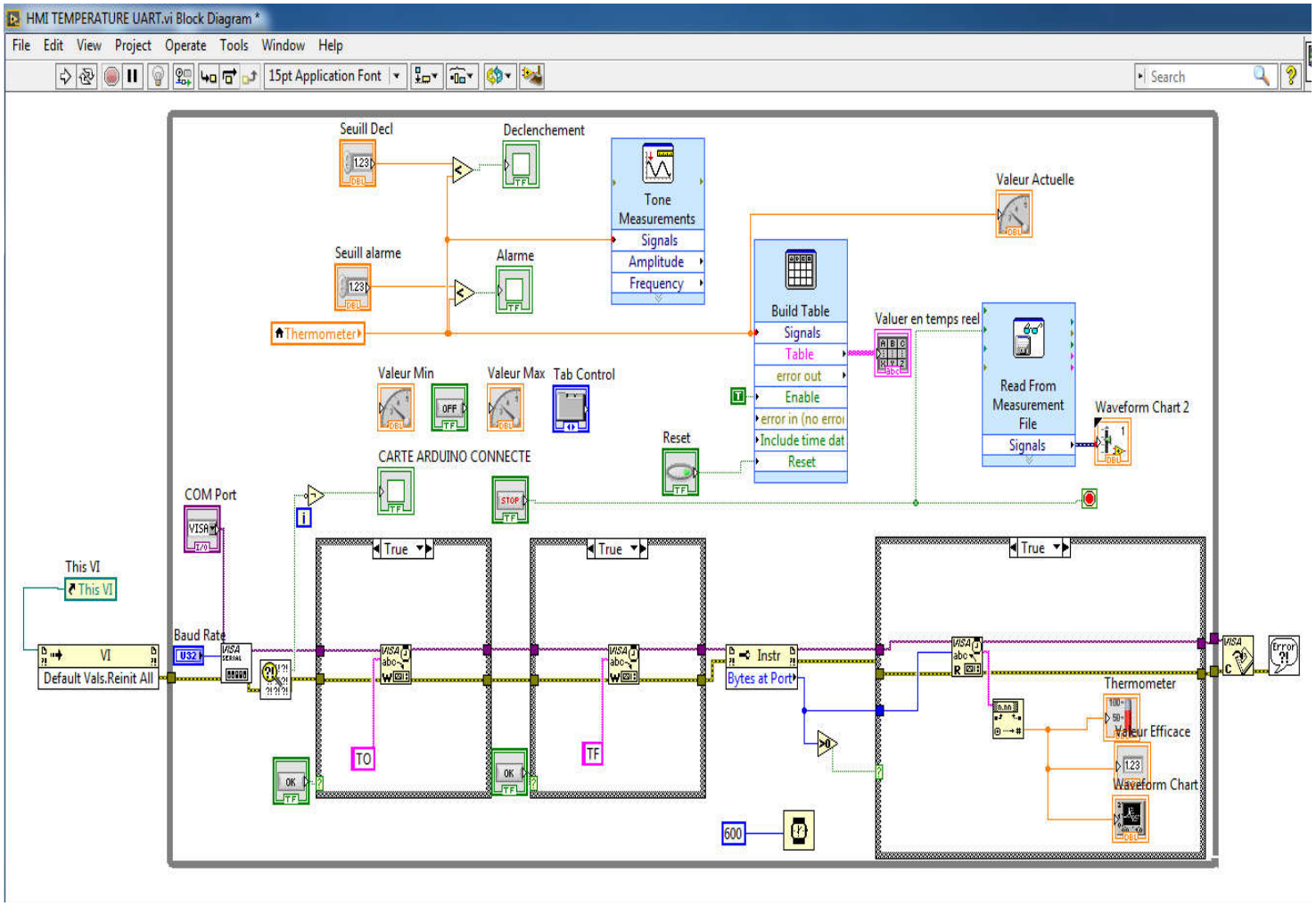


Figure : Interface pour extraire les anciennes valeurs enregistrées sur un fichier Exel

NB :

Les paramètres de communication entre la carte et le logiciel doivent être synchronisé (les mêmes paramètres doivent être identiques).

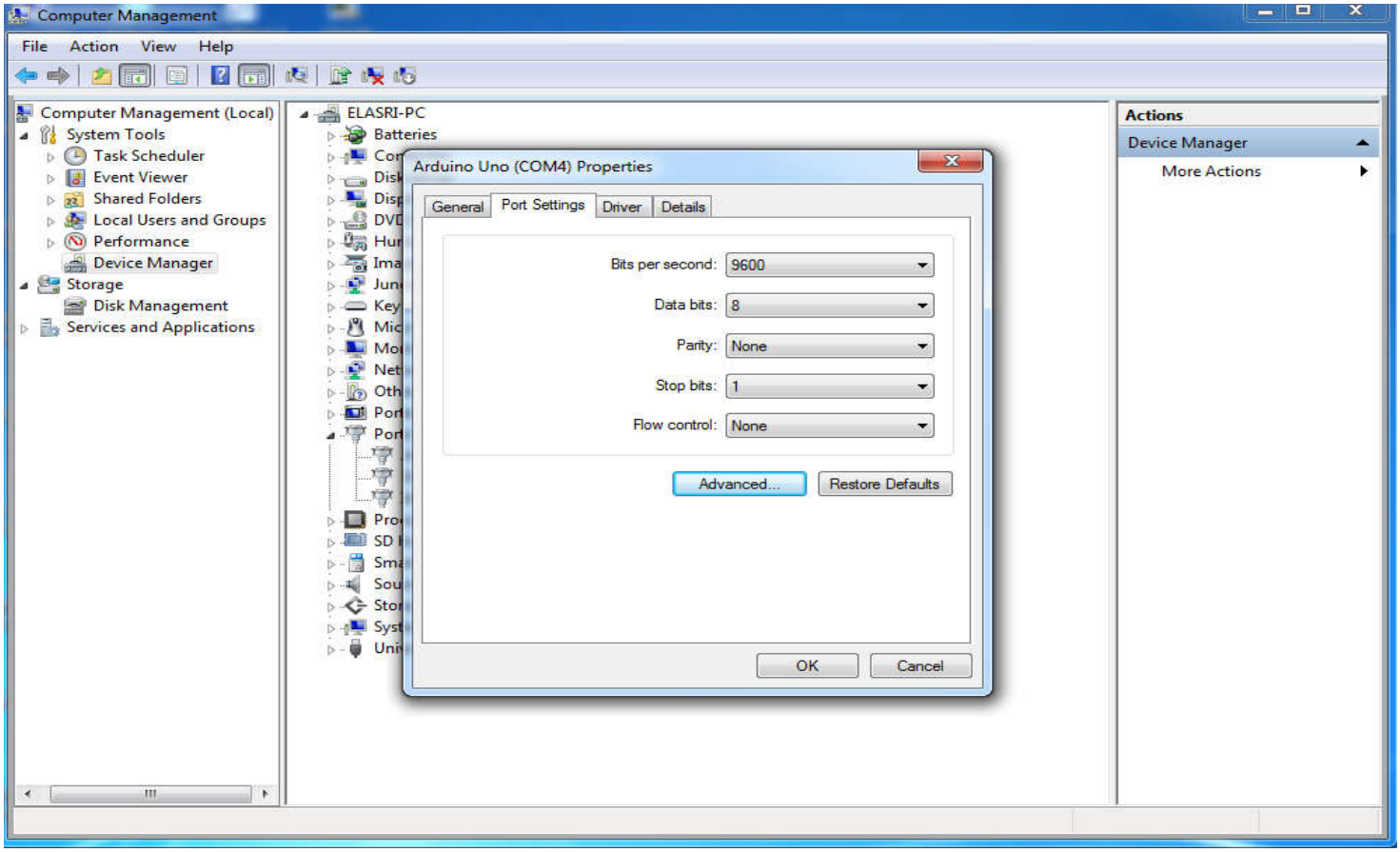


Figure : Configuration du port série sur Pc

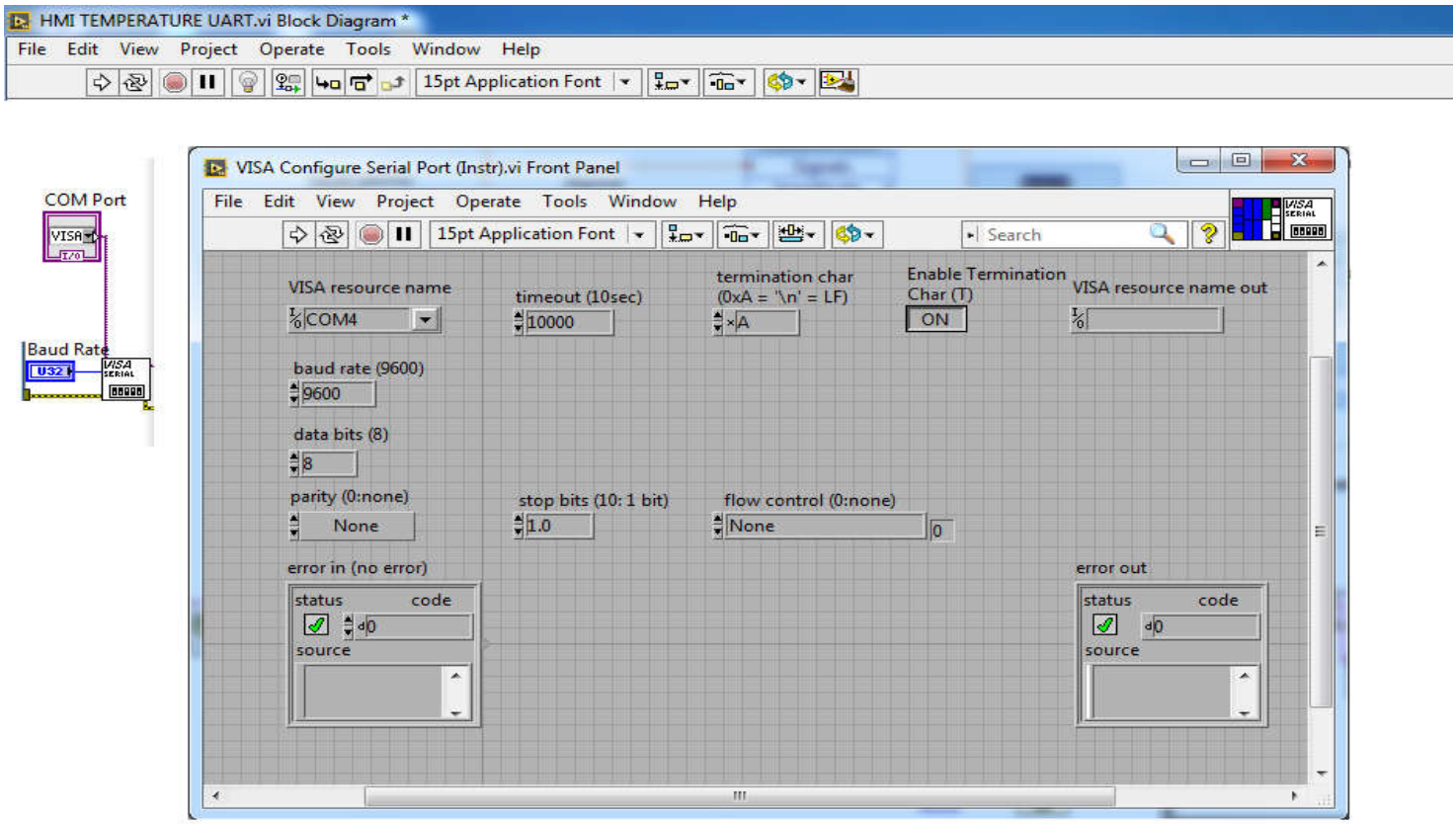


Figure : Configuration des caractéristiques du port série sur LABVIEW

Les résultats du TP sont démontrer au dessus dans les images, ainsi la manipulation physique est présenter au dessous comme suit :

Figure : Carte Arduino UNO utilisée dans le TP

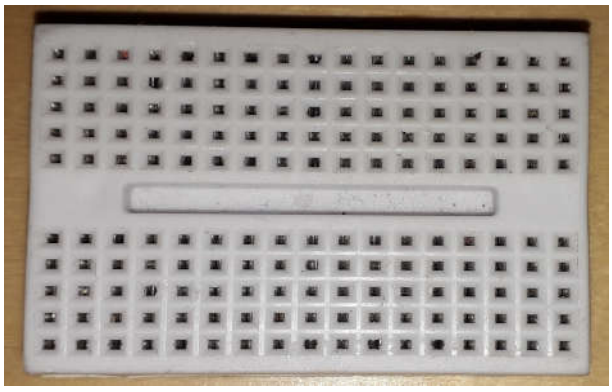


Figure : Labdec pour faire les connections entres divers composants

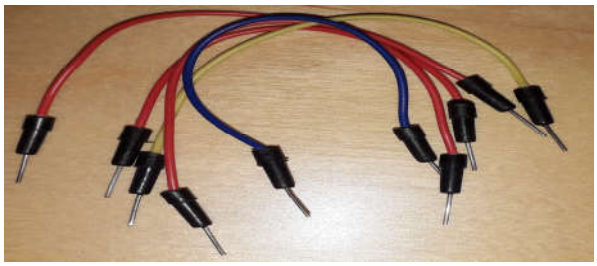


Figure : Connecteurs

Figure : Capteur LM35

Figure : Potentiomètre 100k

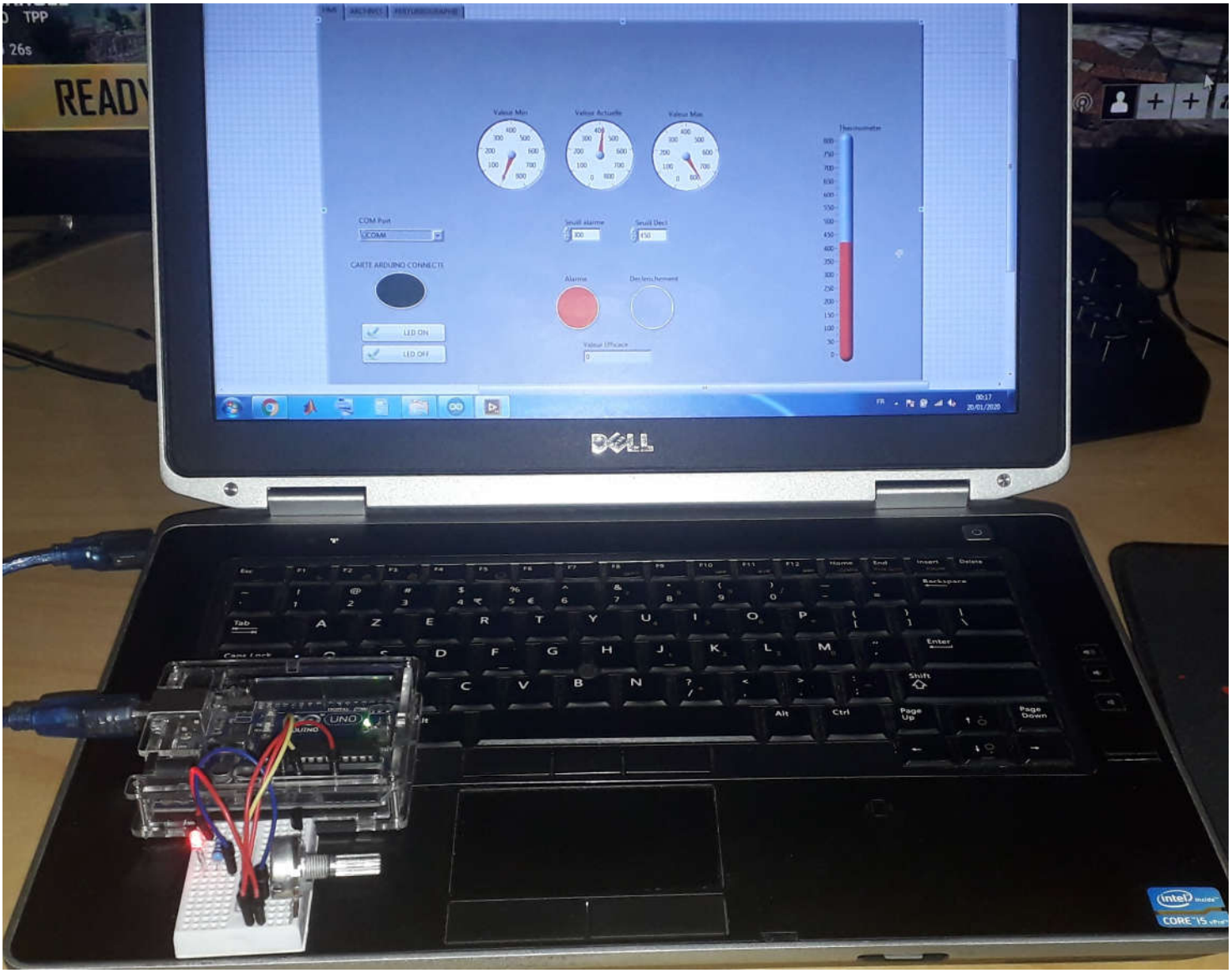


Figure : Manipulation (Acquisition en temps réel de la température)

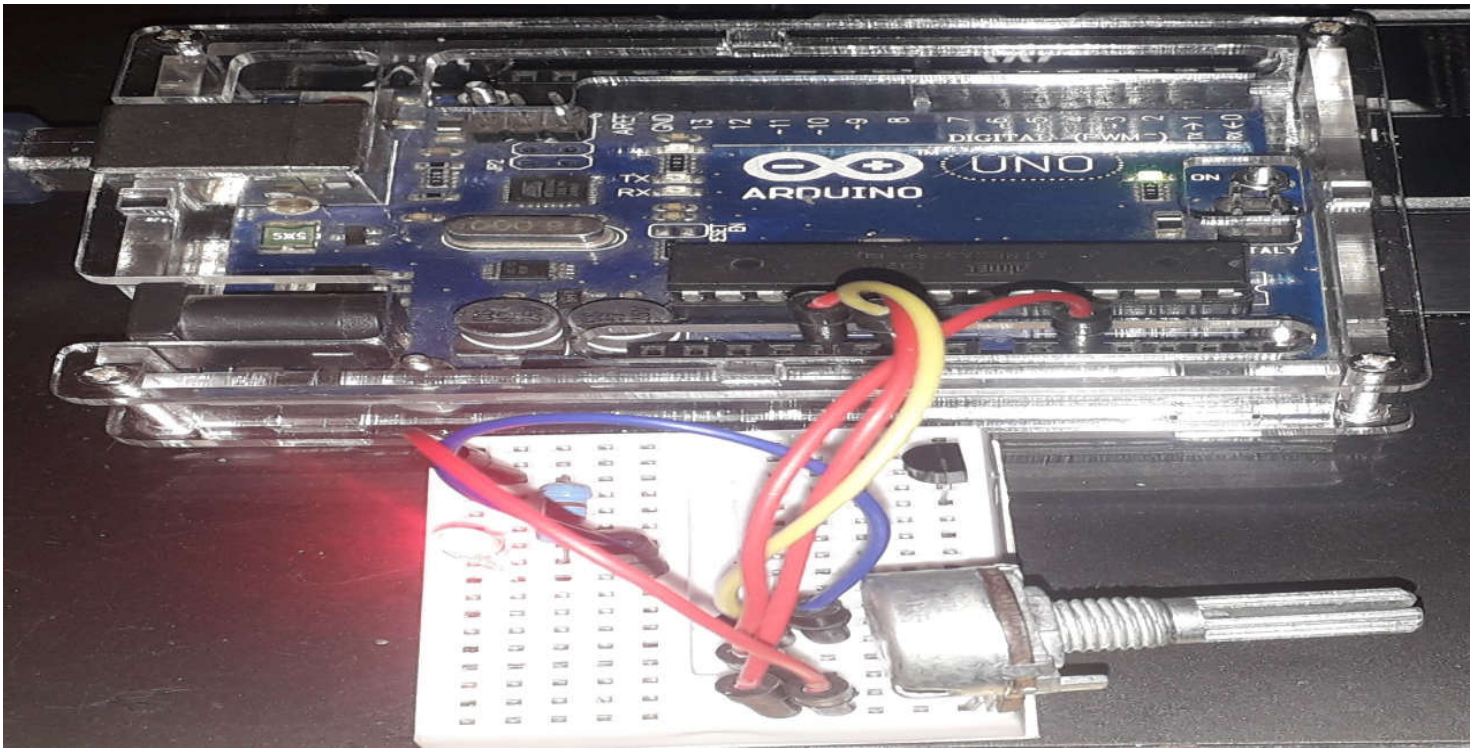


Figure : Connexion entre les composants