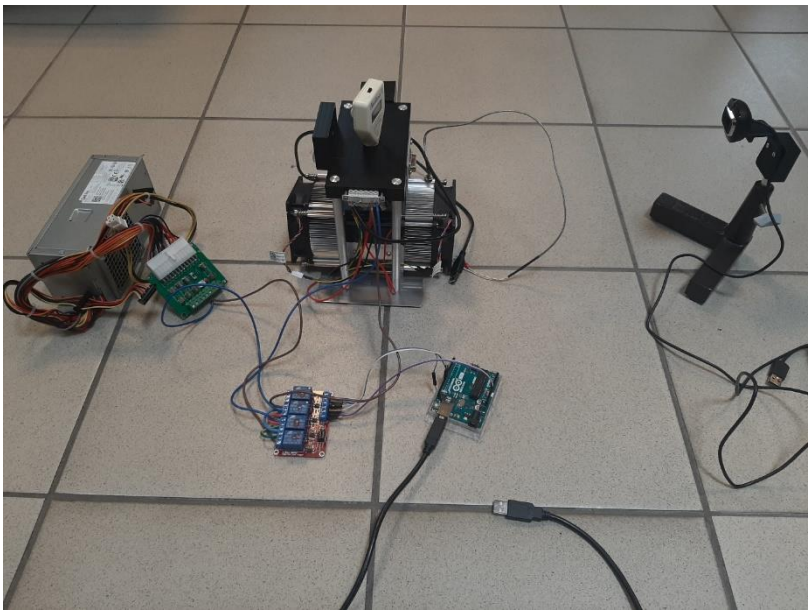
UNIVERSITÉ
Clermont Auvergne

PROJET N3 EEA

ANNÉE 2022-2023



PARTICIPANTS :
Eva DA COSTA
Thierno Oumar Sere DIALLO
Tuteur : M. Pierre BIGENWALD
Responsable UE : M. Jérôme BRUNET

I. Description du projet :

Notre projet consiste à automatiser certaines parties d'un TP proposé aux étudiants de master 1 appelé conduction pour lequel ils étudient les propriétés courant-tension I(V) en fonction de la température. Le TP non automatisé ne permet pas la gestion fine de la température car ils utilisent un chauffe-ballon ainsi qu'un autotransformateur.

1) Principe général du TP :

Le but de ce TP est d'étudier les caractéristiques I(V) des éléments suivants en fonction de l'évolution de la température :

- Résistances de Platine et de Carbone.
- Thermistances à C.T.N. et à C.T.P.
- Diode au Germanium en direct et en inverse.
- Diode à effet Zener en inverse

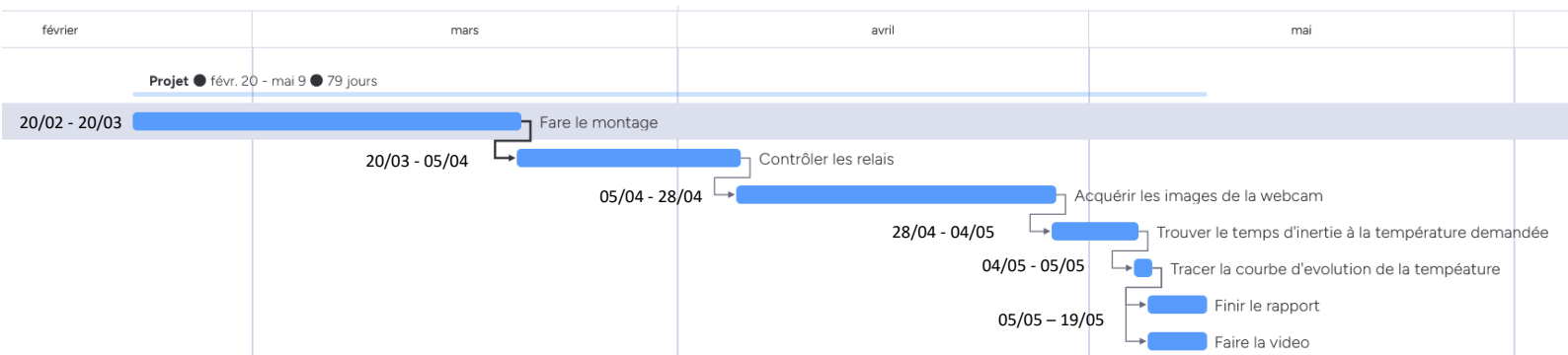
Pour cela, ils utilisent le matériel suivant :

- 5 sondes
- un boitier de commutation et d'alimentation des sondes
- Un générateur de courant continu
- Deux multimètres
- 2 modules PELTIER
- 2 ventilateurs

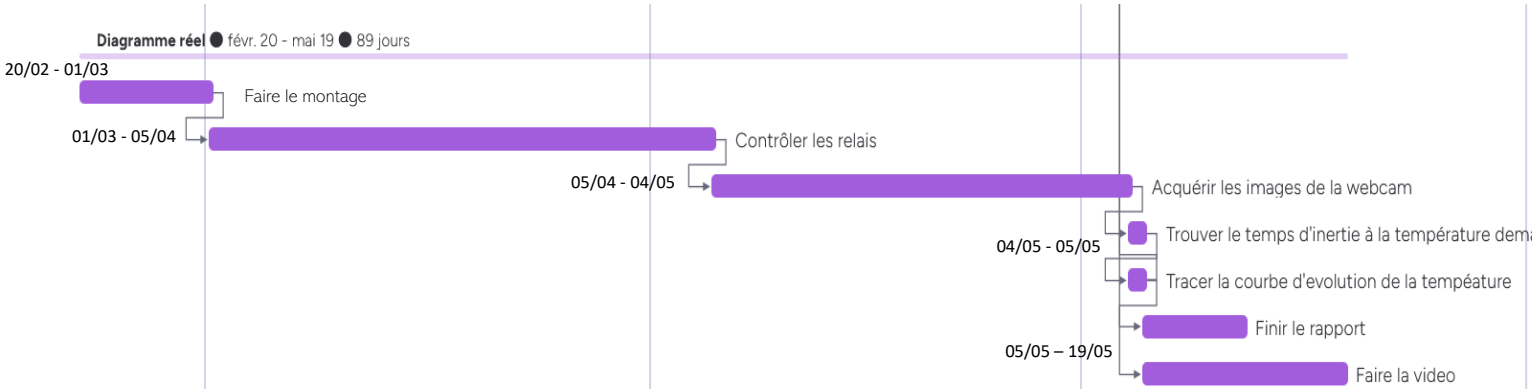
2) Objectifs du projet :

- Automatiser le chauffage et le refroidissement d'un peltier
- Tracer la courbe de température en fonction du temps
- Déterminer le temps pendant lequel la température ne change pas après l'arrêt de l'application d'une tension

- Diagramme de Gantt prévisionnel :



- Diagramme de Gantt réel :

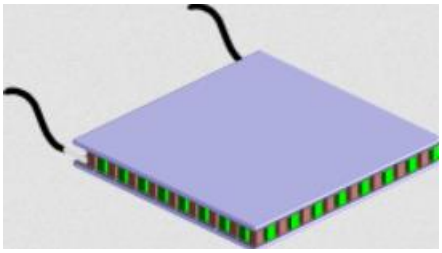


3) Matériel utilisé pour le projet :

Dans le cadre de notre projet, nous allons utiliser les deux modules peltier, les deux ventilateurs, une carte arduino, une webcam numérique, un système de relais.

a. Qu'est-ce qu'un peltier ?

Un peltier est un semi-conducteur constitué de deux plaques dans lequel on fait passer un courant continu. En fonction du sens du courant le peltier chauffe ou refroidit.

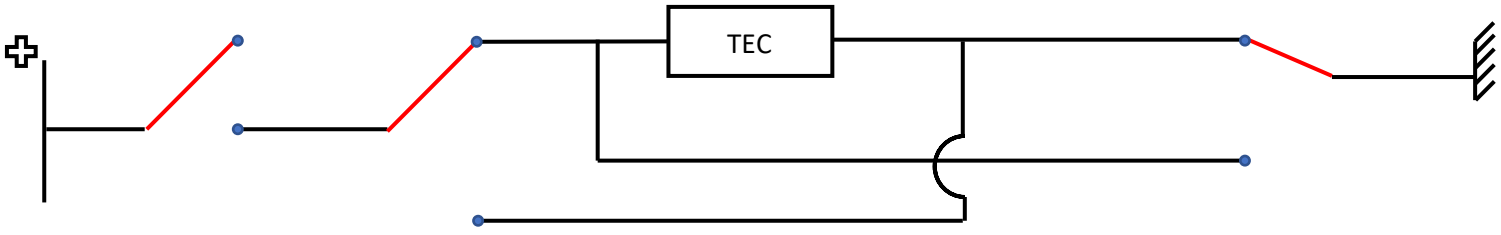


De plus, il est entouré de deux ventilateurs permettant de réguler la température des composants afin de ne pas les endommager.

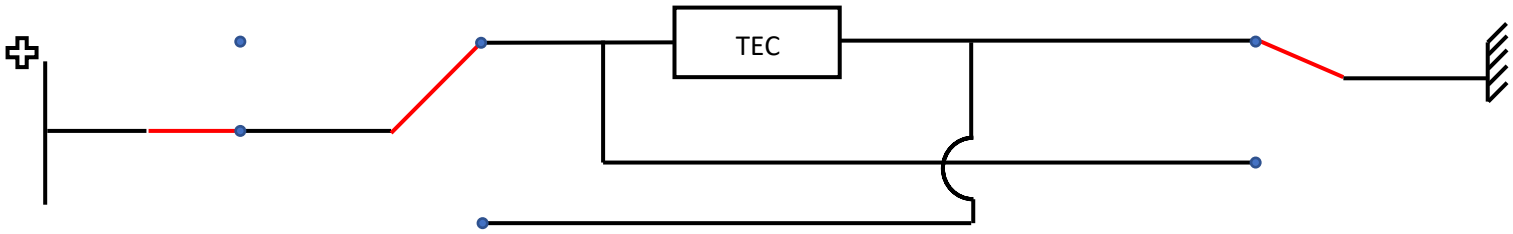
b. Les relais :

Les relais servent à contrôler si les peltiers refroidissent ou chauffent.

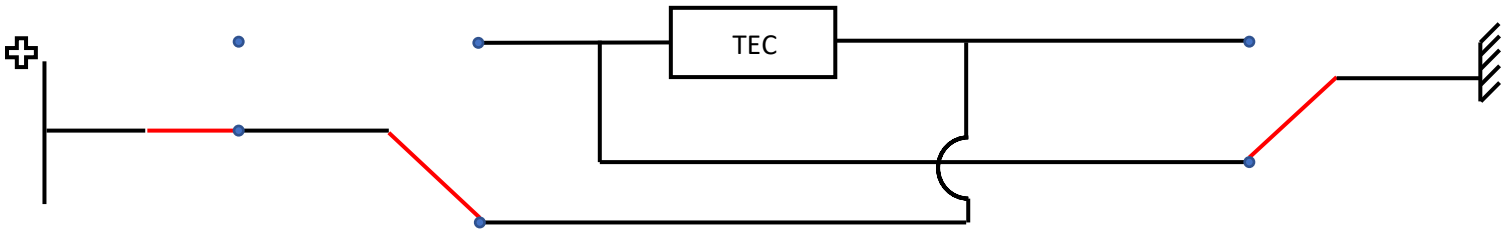
Après vérification, nous avons pu déterminer que nos relais se comportent suivant une logique positive. Par défaut, les relais sont dans cette position.



Si l'on active seulement le premier interrupteur, le peltier refroidit et le circuit devient alors :



Enfin, pour que le peltier chauffe les trois relais doivent être activés, le circuit devient donc :

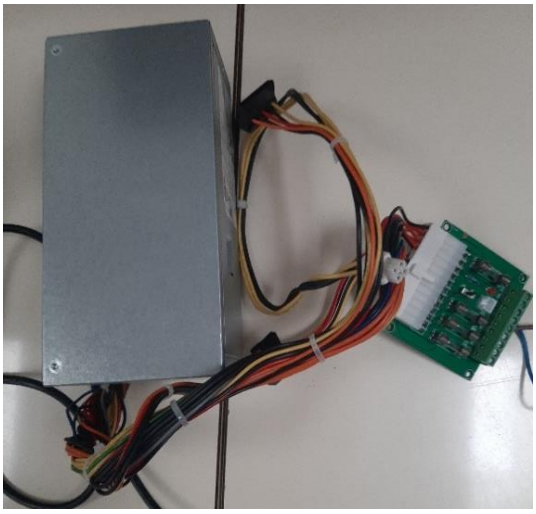


c. Carte Arduino UNO :

La carte Adruino nous permet de contrôler les relais en fonction des informations envoyées par le programme LABVIEW sur 3 de ses pins. Pour cela, nous utilisons l'extension LINX qui permet à LABVIEW d'interagir avec la carte Arduino. Une fois installée, il suffit de se rendre sur LabVIEW markerhub et de suivre les étapes jusqu'à établir la connexion entre un PORT de l'ordinateur et la carte.

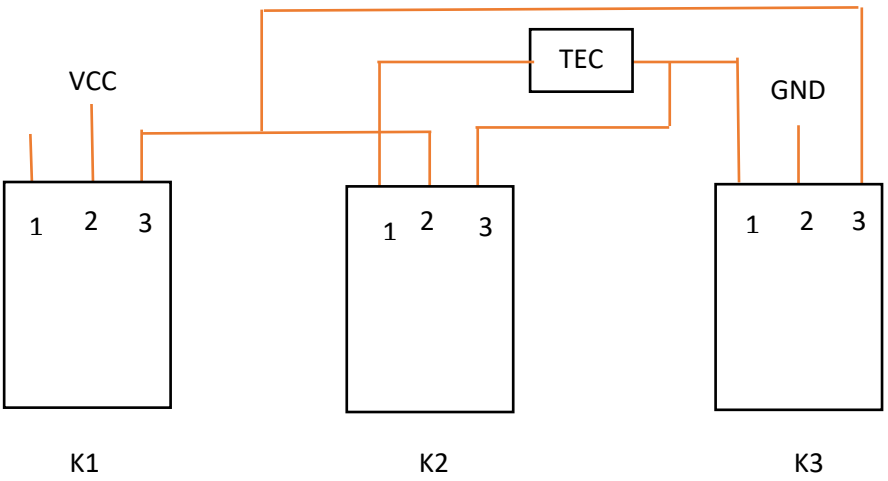
d. Boite d'alimentation :

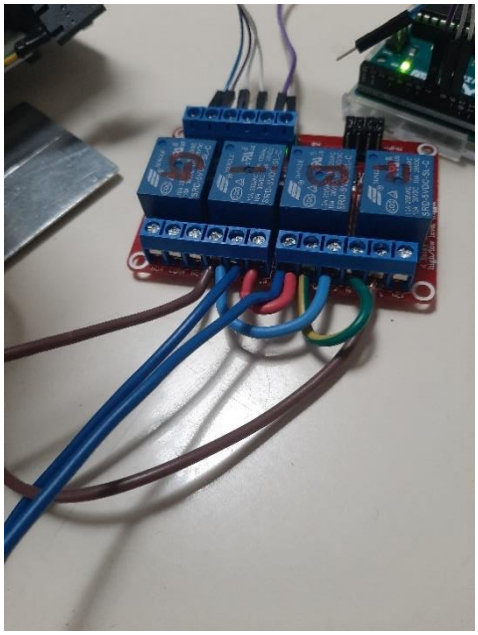
Bien que les relais soient alimentés par la carte Arduino, celle-ci ne peut délivrer qu'une faible tension. En effet, elle n'est pas assez puissante pour alimenter le peltier, il est donc nécessaire d'utiliser une boîte d'alimentation. Notre boiter délivre une tension alternative de 220V que l'on convertie en tension continue de 5V.



4) Montage :

Les relais sont reliés au peltier et à l'alimentation de la façon suivante :



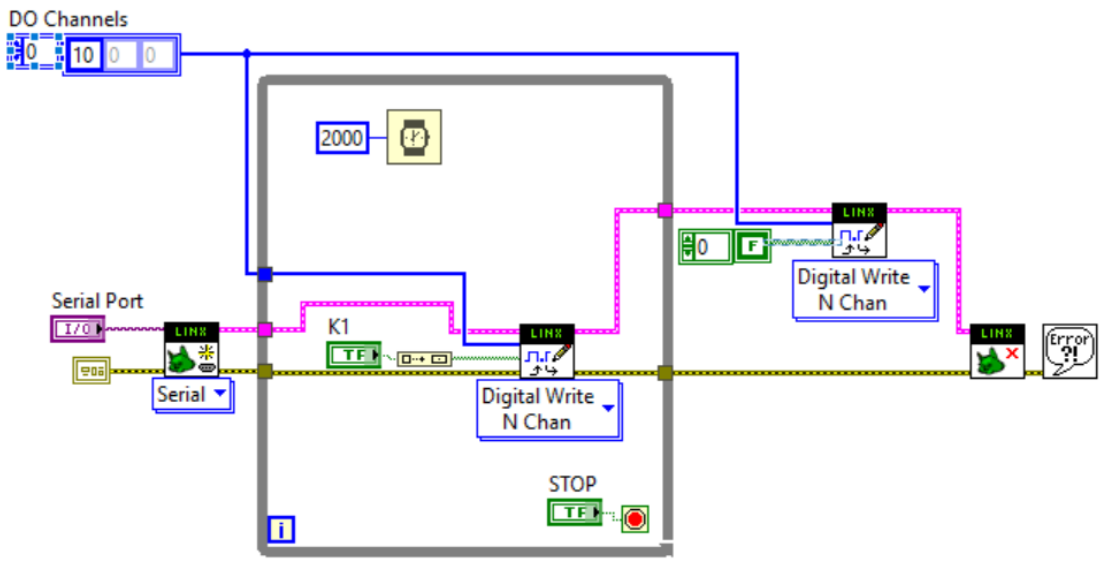


II. Prise en main des fonctionnalités de LABVIEW :

1) Comment contrôler les relais grâce à LabVIEW ?

Grâce à un module de LabVIEW appelé LINX il est possible d'écrire une donnée sur des pins d'une carte Arduino. Nous avons donc utilisé certaines de ses fonctionnalités pour activer ou désactiver les relais.

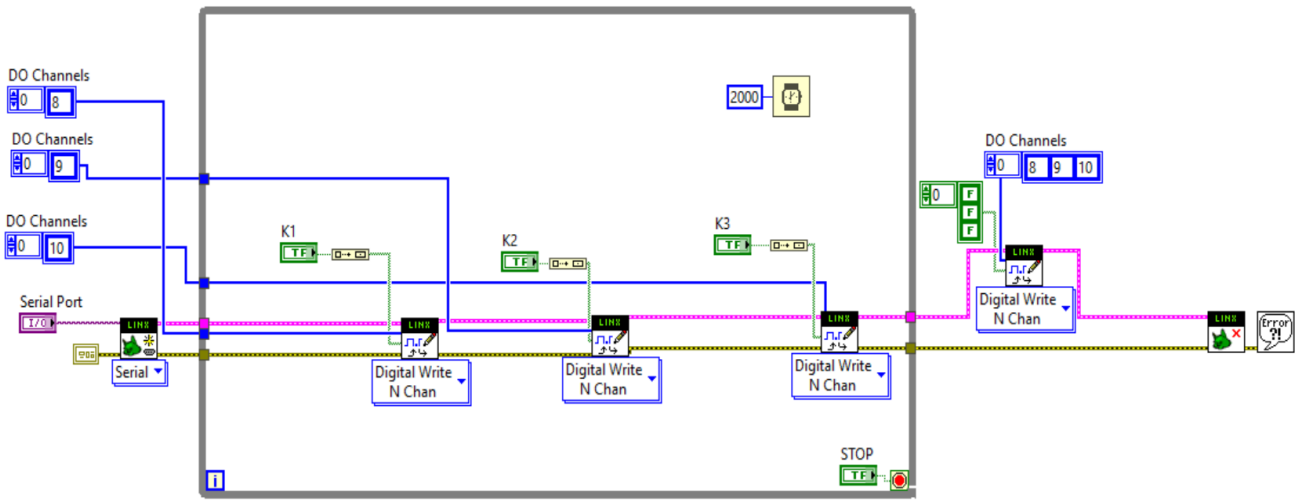
- Contrôle d'un relais grâce à un module Arduino piloté par LabVIEW + LINX :



- <u>Explication du VI :</u>

- On ouvre la connexion entre le port série de l'Arduino et LINX
- On écrit une valeur sur un pin de l'Arduino (dans ce cas le pin 10)
- Une fois la boucle while terminée, on remet la valeur du pin à 0, on ferme la connexion avec le port série et on indique s'il y a eu une erreur

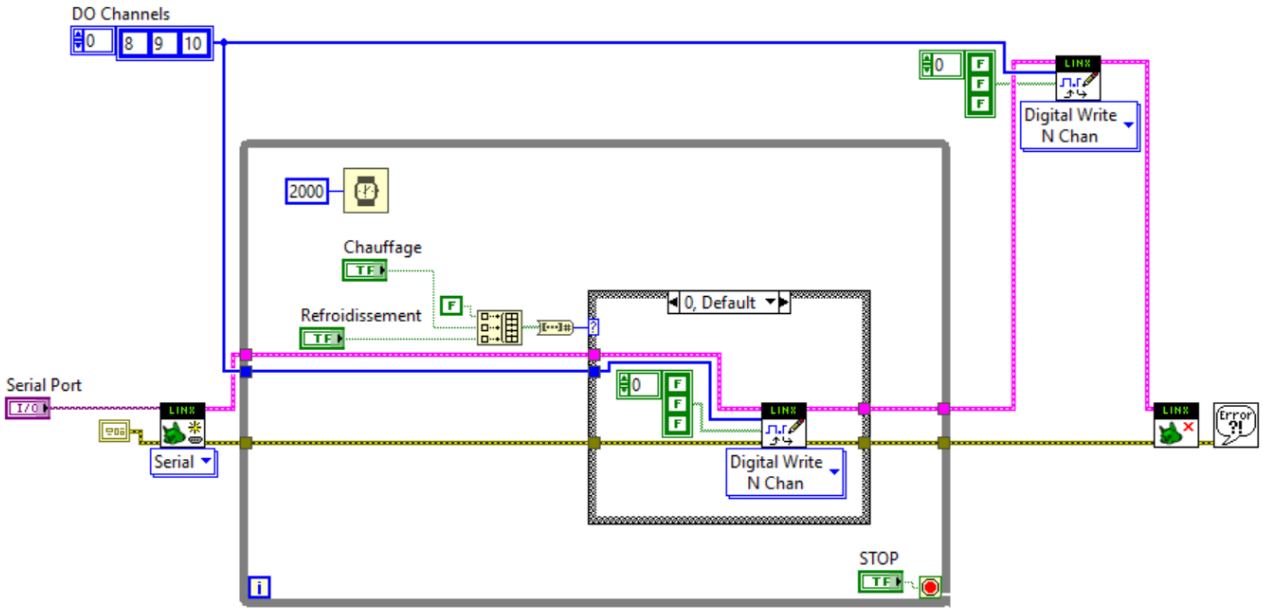
- <u>Contrôle de trois relais grâce à un module Arduino piloté par LabVIEW + LINX :</u>



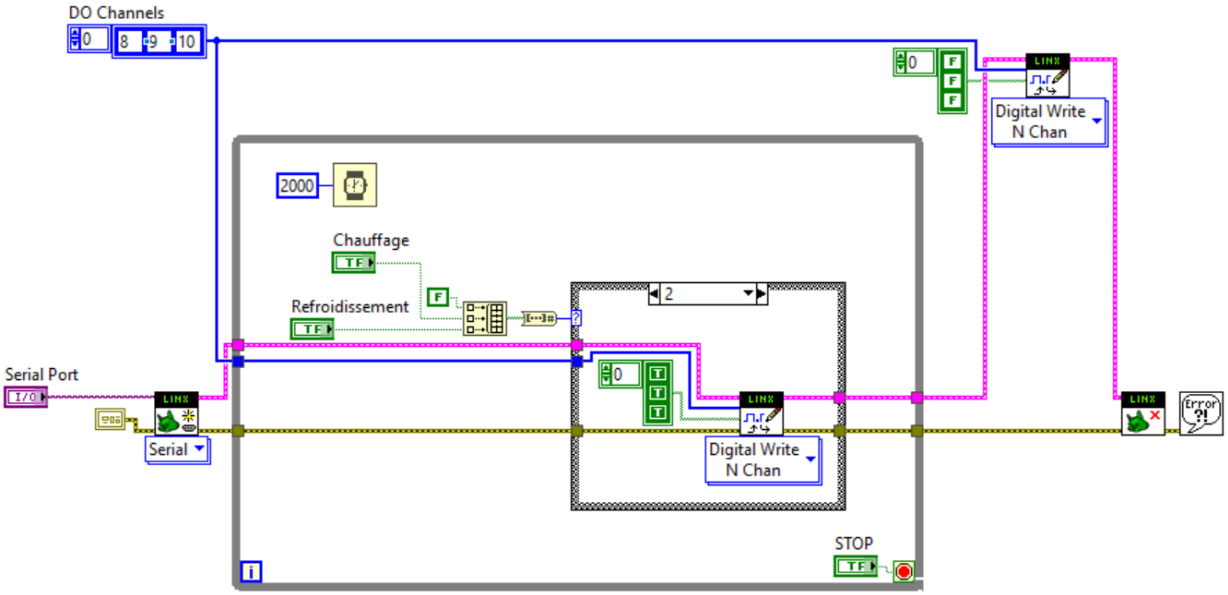
Cependant pour pouvoir chauffer ou refroidir le peltier, il ne faut pas seulement activer les relais un par un mais il faut activer une combinaisons des trois relais (correspondant aux circuits précédant).

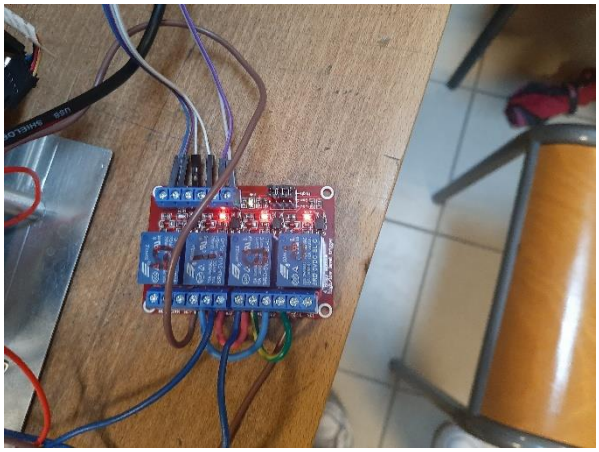
On introduit alors une boucle conditions :

- <u>0, Par défaut :</u>

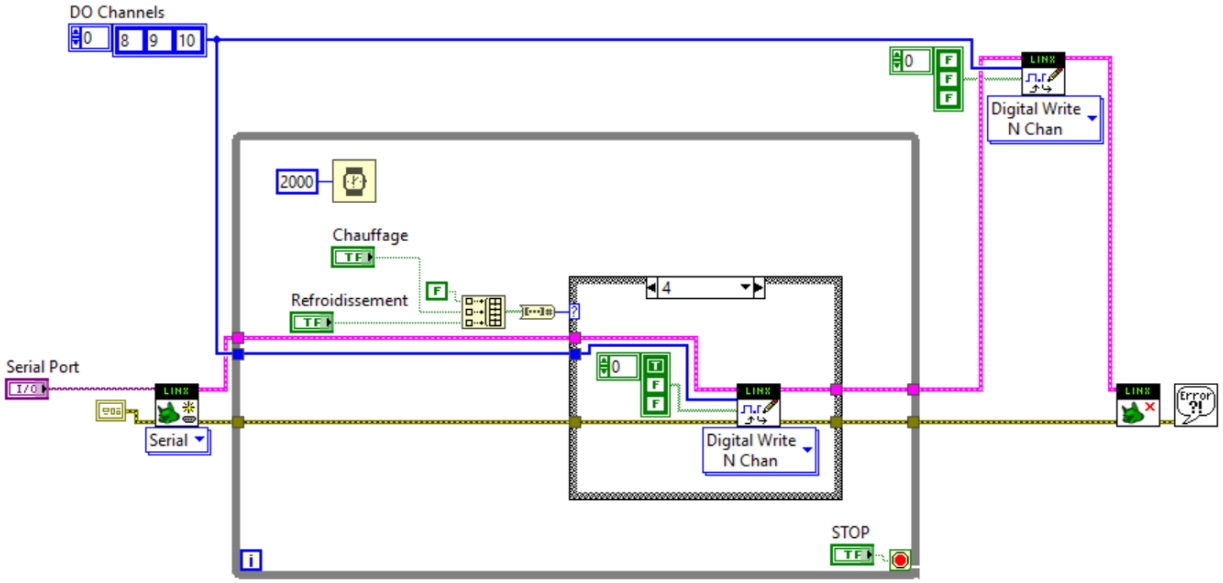


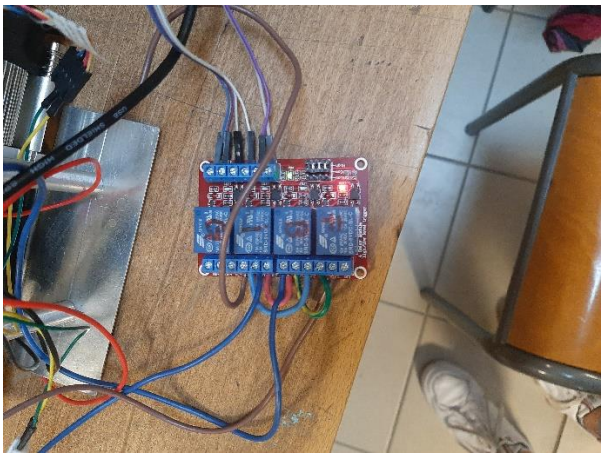
- 2, Chauffage :





- 4, Refroidissement :





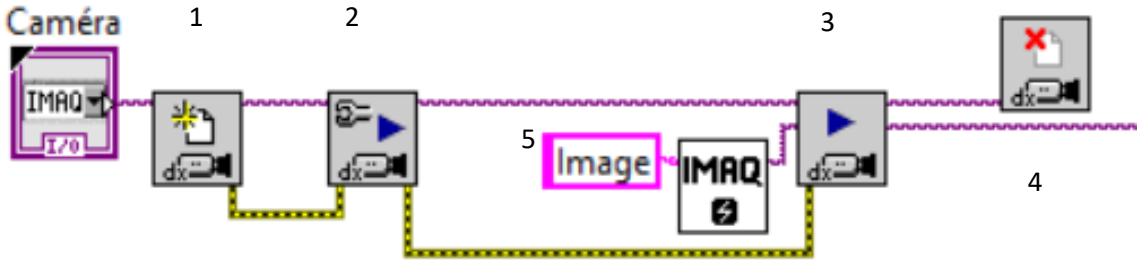
Il faut maintenant réussir à récupérer la température affichée sur le thermomètre pour :

- Arrêter le chauffage ou le refroidissement quand on atteint la température souhaitée
- Tracer l'évolution de la température en fonction du temps
- Déterminer le temps pendant lequel la température ne change pas après l'arrêt du chauffage ou du refroidissement

2) Comment acquérir la température :

Pour cela nous allons nous servir d'une webcam USB fonctionnant dans le visible pour la lecture des chiffres affichés par le thermomètre numérique en utilisant le module OCR par LABVIEW Vision Acquisition :

a. Acquisition d'une image :



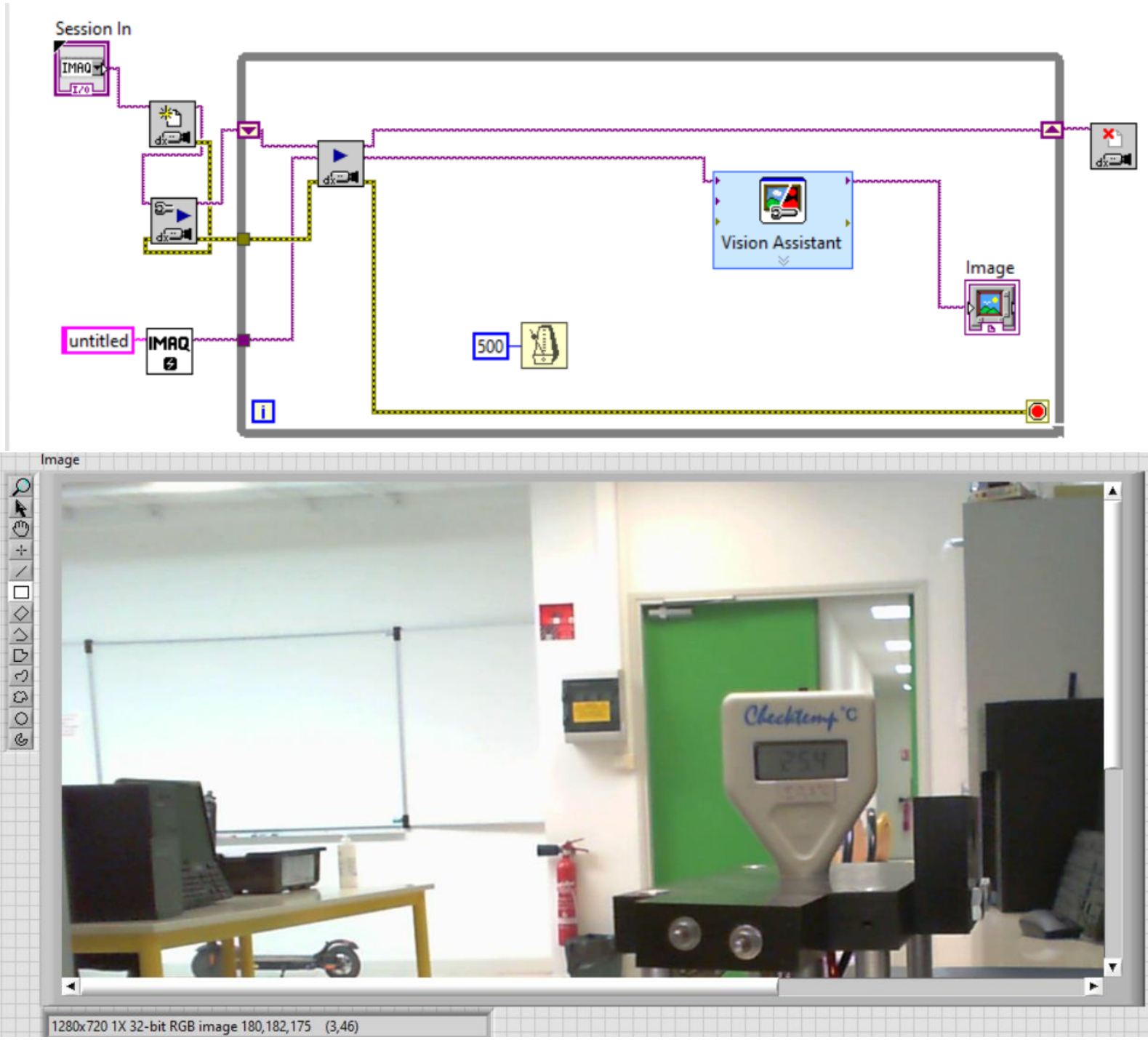
On utilise les fonctionnalités de NI-IMAQdx pour :

- Activer et créer une référence unique correspondant à la caméra (1).
- Commencer l'acquisition d'image (2).
- Acquérir l'image la plus récente (3).
- Arrêter l'acquisition d'image (4).

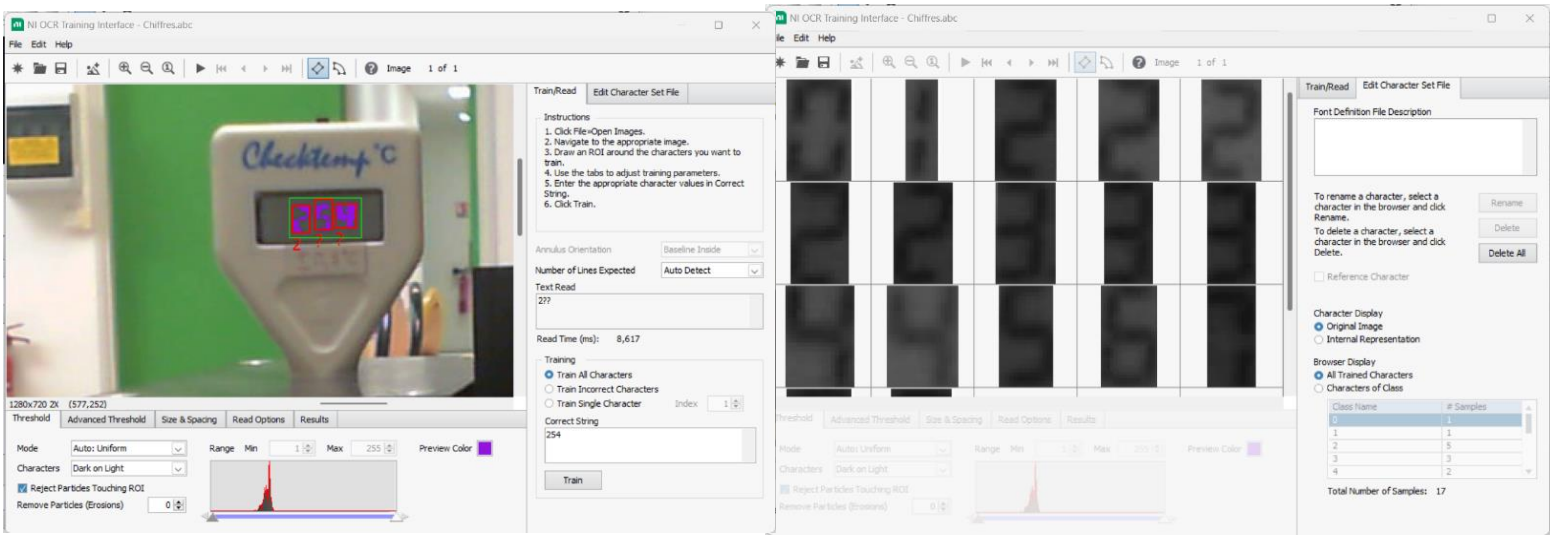
Il faut aussi penser à créer un emplacement de mémoire temporaire pour l'image que l'on va obtenir (5).

b. Création de la base de données :

Afin de reconnaitre les chiffres affichés sur le thermomètre numérique, il faut tout d'abord créer une base de données grâce au module OCR. Pour cela, il faut capturer plusieurs images du thermomètre grâce à ce programme :



Une fois que l'on a capturé l'image, il faut se rendre sur vision acquisition pour entrainer le logiciel à reconnaitre les chiffres un à un en créant la base de données suivante :



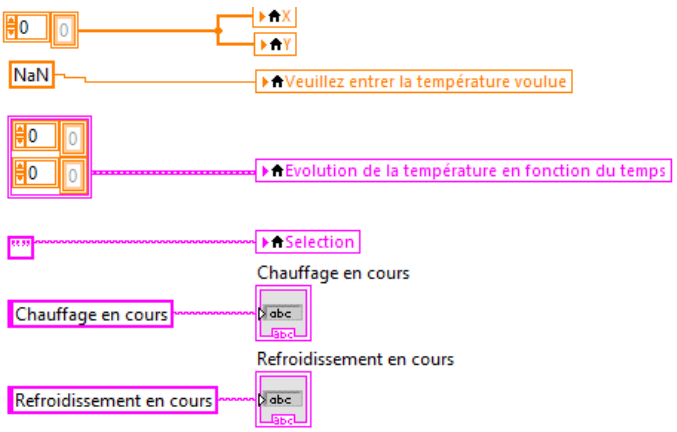
Nous avons fait le choix de créer plusieurs références pour chaque chiffre afin d'augmenter la précision de la lecture.

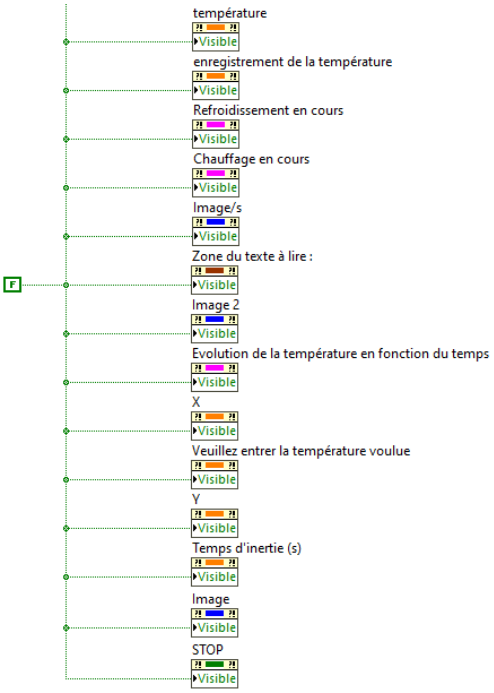
III. Description du VI :

Notre VI se décompose en 4 séquences :

- Une séquence d'initialisation
- Une séquence de sélection de la caméra et du port pour la carte Arduino
- Une séquence pour définir la zone de lecture de la température
- Une séquence qui se décompose en deux parties :
  - Le contrôle des peltiers et le tracé de l'évolution de la température en fonction du temps
  - La mesure du temps d'inertie

1) Initialisation du VI :

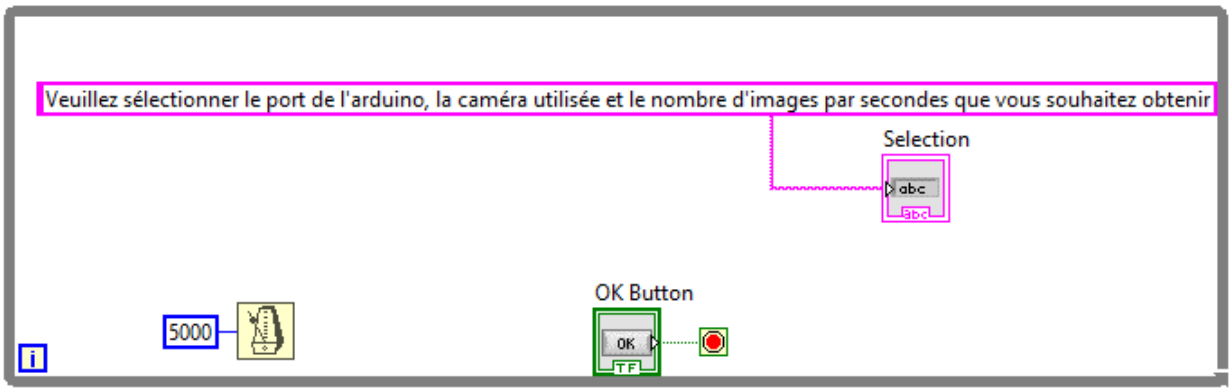




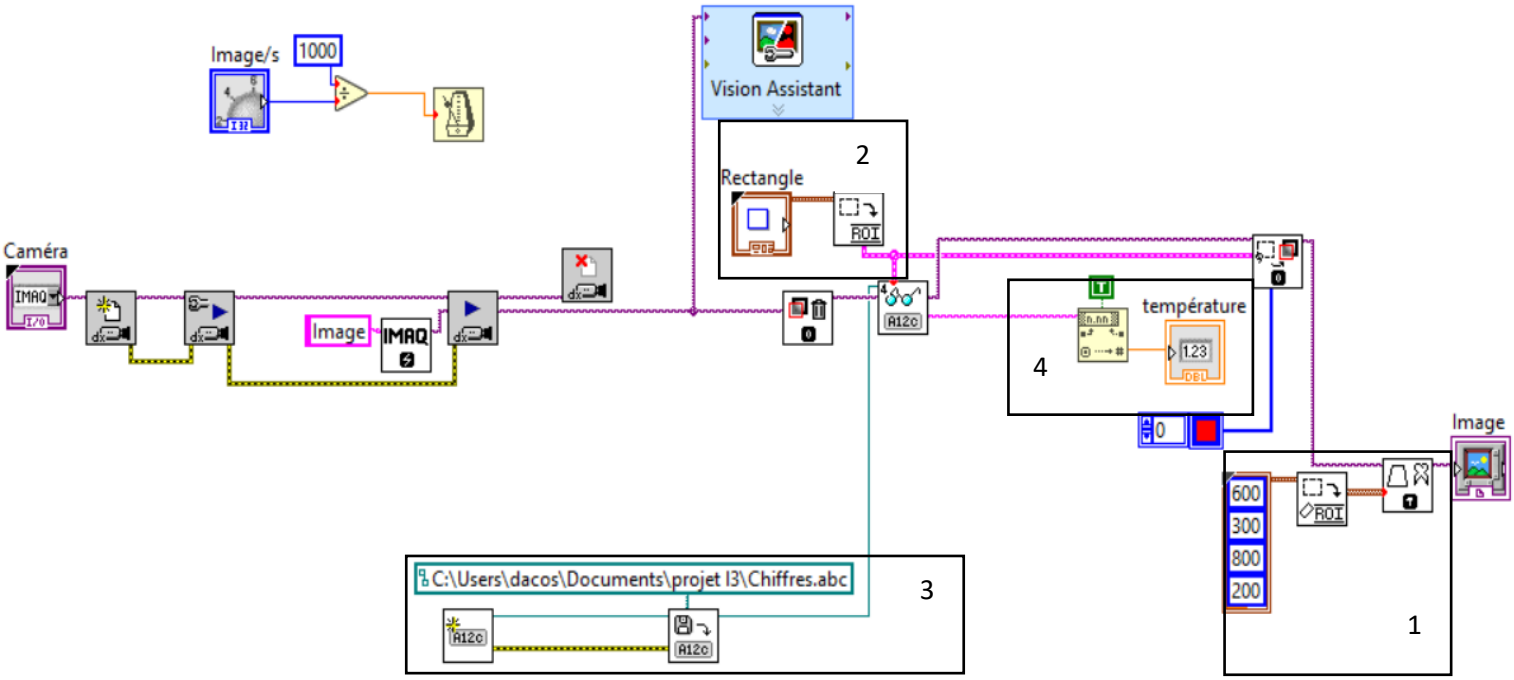
Dans cette première séquence, on initialise la valeur des indicateurs et des contrôles grâce à des variables locales.

Cette séquence nous permet aussi de rendre invisibles les fonctionnalités auxquelles l'utilisateur ne doit pas avoir accès au démarrage du programme grâce à des nœuds de propriété.

2) Sélection de la caméra et du port de la carte Arduino :

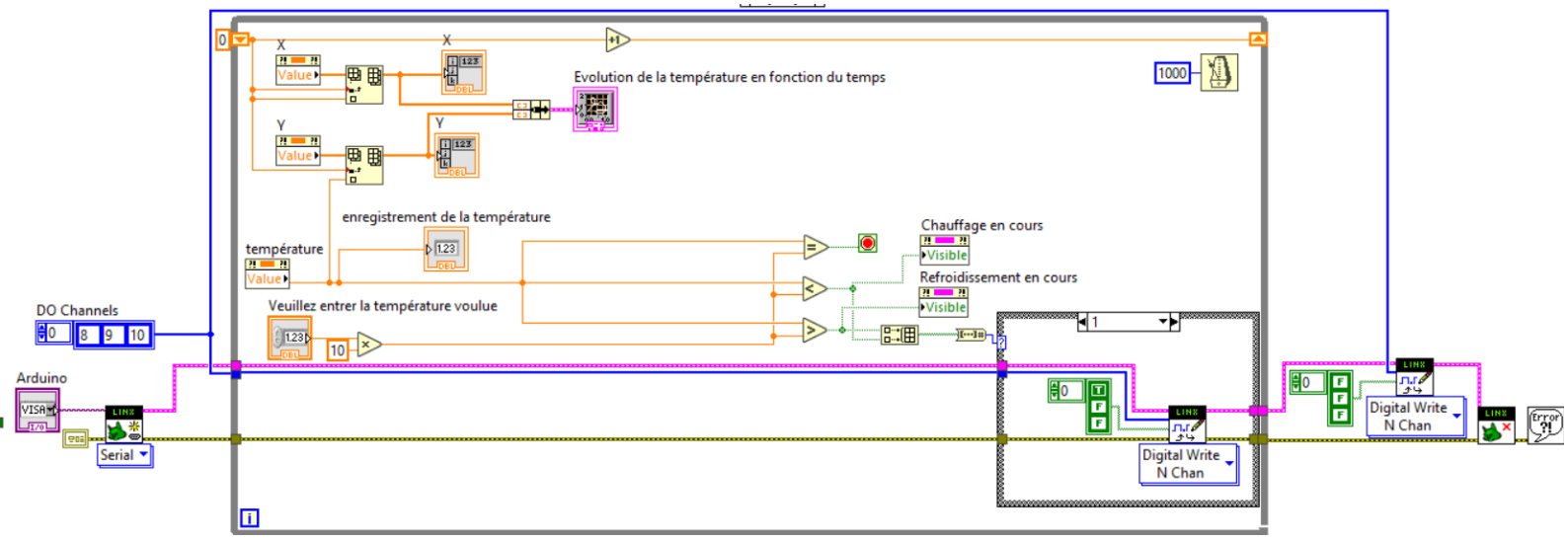


3) Affichage et lecture du thermomètre sur LABVIEW :

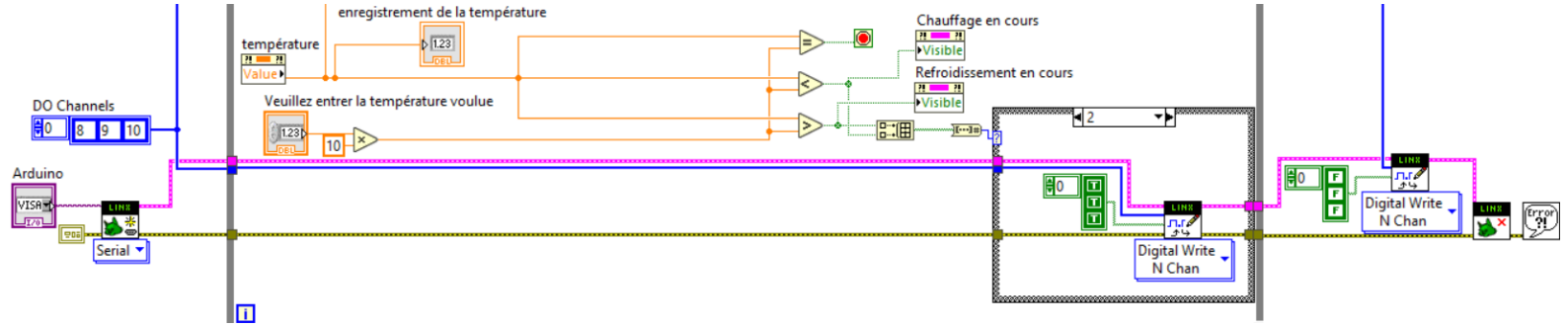


Ce programme consiste à récupérer une partie de l'image de la caméra (1) dans laquelle l'utilisateur va définir un rectangle qui délimitera la zone de lecture des données (2). Puis on utilise la base de données pour lire le texte (3) contenu dans la zone du rectangle et le convertir en donnée numérique (4) afin de pouvoir par la suite la comparer avec la valeur de la température choisie par la suite.

4) Contrôle des relais et tracé du graphique représentant l'évolution de la température en fonction du temps :



a. Contrôle des relais :



Dans cette partie du VI, on peut retrouver le contrôle des relais (décrit dans la partie II.1)). Cependant, la sélection des relais à activer ne se fait plus en appuyant sur un bouton mais en comparant la valeur de la température relevée par la caméra avec la valeur de température choisie par l'utilisateur.

On doit multiplier par 10 car la lecture sur la caméra ne permet pas de relever le point (pour une température de 23.4°, le programme lira 234).

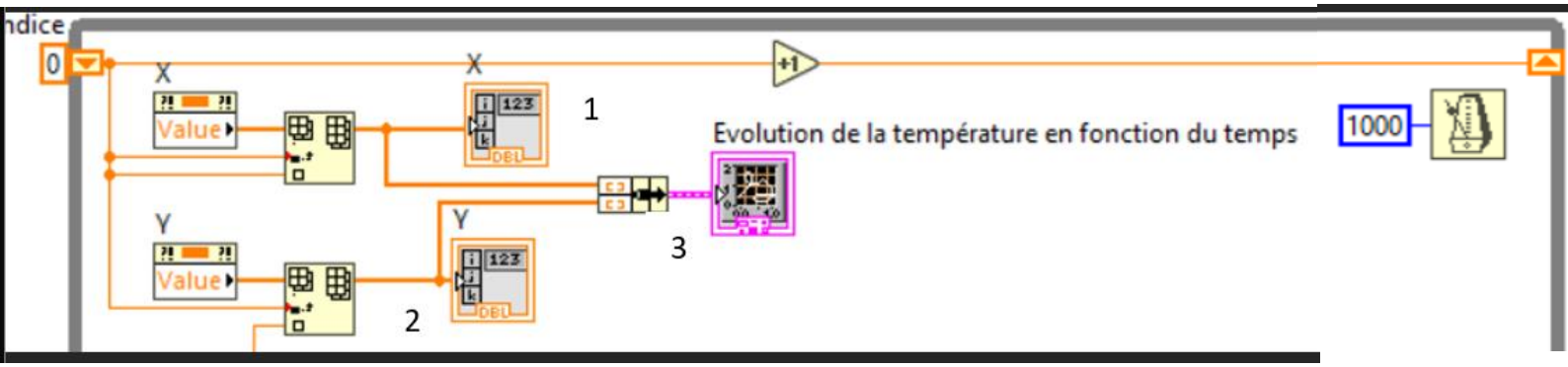
Pour comparer ces deux valeurs, on construit un tableau de booléen qui représente un nombre binaire, la première ligne correspondant au bit de poids faible et la dernière ligne correspondant au bit de poids fort. Il suffit ensuite de convertir cette valeur binaire en valeur décimale.

- Si la température relevée est inférieure à la température choisie par l'utilisateur le tableau de booléen correspond à la valeur 1 (01 en binaire). Un relais sera allumé et les deux autres seront éteints donc le peltier chauffe.

- Si la température relevée est supérieure à la température choisie par l'utilisateur le tableau de booléen correspond à la valeur 2 (10 en binaire). Les trois relais seront allumés donc le peltier refroidit.
- Enfin, si la température relevée est égale à la température choisie par l'utilisateur le programme exécute la condition par défaut qui correspond aux trois relais éteints.

On remarquera que l'on enregistre la température, ce qui nous servira plus loin dans le programme.

b. <u>Tracé du graphique représentant l'évolution de la température en fonction du temps :</u>



Pour tracer l'évolution de la température en fonction du temps, il faut créer deux tableaux. Un tableau qui contient les valeurs pour l'axe des abscisses X (1) et un tableau qui contient les valeurs pour l'axe des ordonnées Y (2).

On utilise donc deux fonctions (insert into array) qui permettent d'insérer une valeur dans un tableau pour un certain indice.

La sélection de l'indice se fait grâce au registre à décalage qui à chaque itération i de la boucle while ajoute 1 à la valeur de l'indice qu'il y avait à l'itération i-1 de cette boucle. La boucle while se répétant toutes les secondes l'indice correspond à la valeur du temps en seconde.

C'est pourquoi le tableau de l'axe des abscisses prend en entrée la même valeur que son indice. L'axe des abscisses représente ainsi l'axe du temps.

Le tableau de l'axe des ordonnées prend lui aussi les valeurs du registre à décalage comme indice mais il va associer à chaque indice la valeur de la température relevée grâce à la caméra.

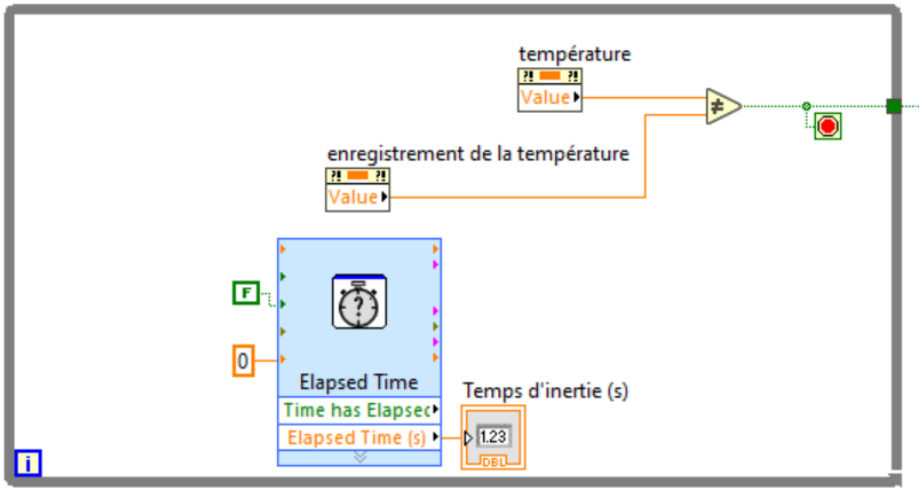
L'axe des ordonnées représente ainsi l'axe de la température.

Pour construire le graphique, il suffit d'assembler les deux tableaux dans un cluster (3) ce qui va permettre de tracer la température en fonction du temps.

5) Mesure du temps d'inertie :

Dans le cadre de notre projet, il est aussi intéressant de mesurer le temps pendant lequel la température reste inchangée une fois que l'on arrête de chauffer ou de refroidir (temps d'inertie). En effet, cela permet de connaître le temps disponible pour réaliser les mesures demandées dans le TP avant que la température ne change.

Pour cela nous utilisons le programme suivant :



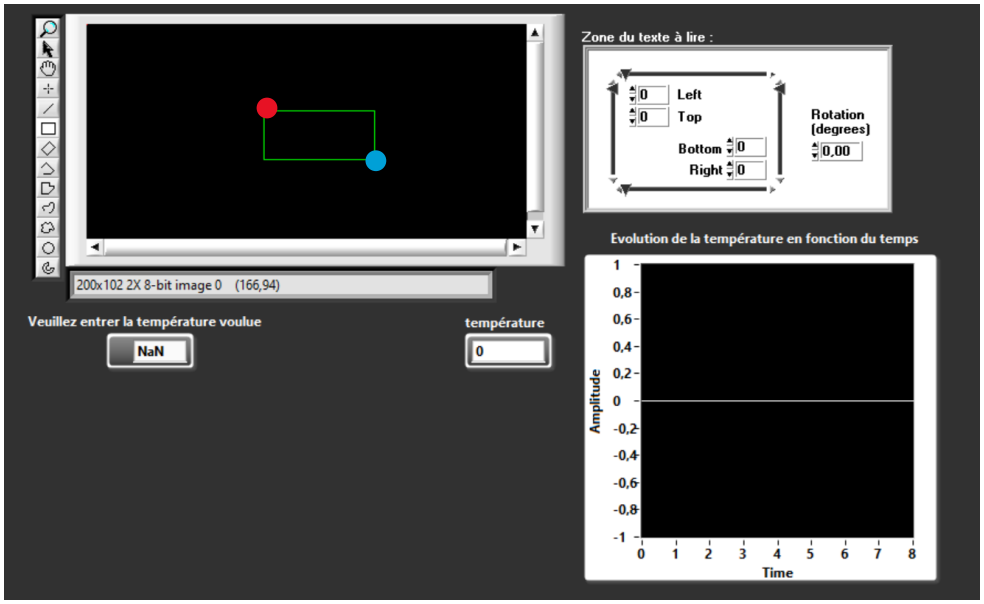
Il mesure le temps qui s'est écoulé entre le moment où la température a atteint la valeur souhaitée et le moment où elle augmente ou diminue. Pour cela, nous avons utilisé la fonction Elapsed Time qui démarre à zéro seconde et qui s'arrête quand la boucle while s'arrête.

La boucle while s'arrête quand la température que nous avions préalablement enregistrée ne correspond plus à la température lue sur le thermomètre.

IV. Description et utilisation de la face avant du VI :

1) Sélection du port de l'Arduino, de la Caméra et de nombre d'images par seconde souhaitées :

2) Sélection de la zone de lecture, de la température souhaitée et affichage de l'évolution de la température :



Pour sélectionner la zone de lecture des caractères, il faut placer le curseur au niveau des deux points rouge et bleu puis rentrer les cordonnées de ces deux points dans la zone du texte à lire.

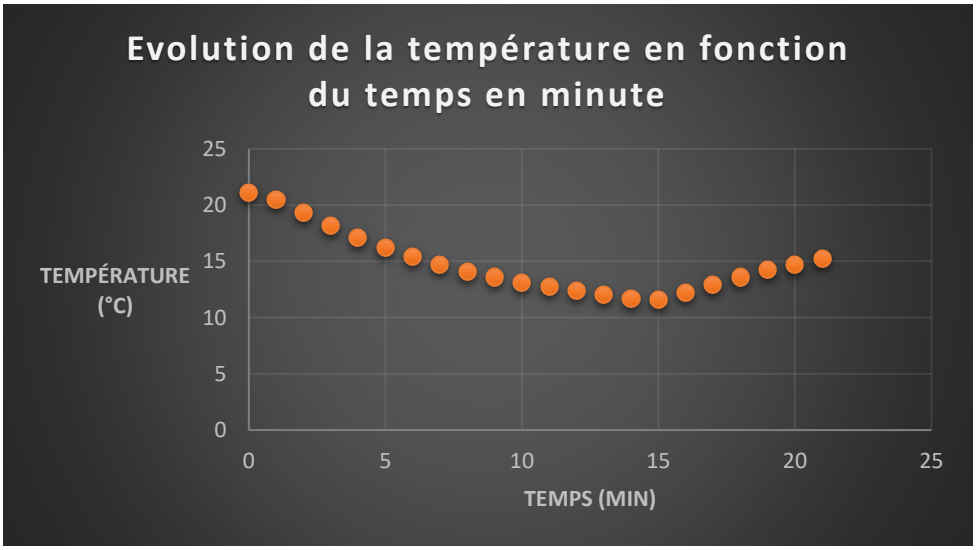
Les coordonnées x et y du point rouge correspondent respectivement aux valeurs left et top.

Les coordonnées x et y du point bleu correspondent respectivement aux valeurs right et bottom.

V. Difficultés rencontrées et résultats attendus :

Notre programme fonctionne, cependant la qualité de la caméra ne nous permet pas d'obtenir les valeurs de température.

Nous pouvons tout de même savoir que la courbe que le programme doit tracer ressemble à cette courbe obtenue grâce à des mesures faites manuellement :



On remarquera que ces mesures ont été réalisées en appliquant une tension de 12V et sans les ventilateurs. La courbe que nous devrions obtenir aurait une pente moins importante. La température devrait dans notre cas évoluer moins rapidement.

Une autre difficulté a été de devoir reconnecter plusieurs fois la carte Arduino à LABVIEW. En effet, normalement la liaison avec la carte n'est nécessaire qu'une seule fois. Cependant, nous avons pu constater qu'une erreur apparaissait à chaque nouvelle exécution du programme.

Malheureusement, nous n'avons pas pu résoudre ce problème ou en déterminer la cause.

VI. Améliorations possibles :

Pour améliorer le projet, nous avons pensé aux améliorations suivantes :

1) Amélioration de la caméra :

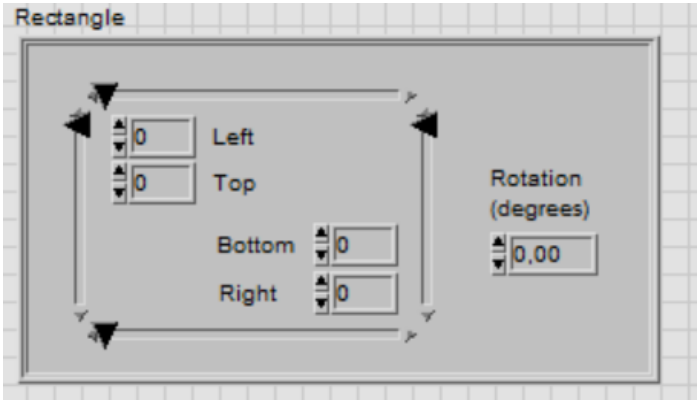
Pour un fonctionnement encore plus efficace du programme, il faudrait une caméra plus performante. En effet, cela permettrait de réaliser des captures des valeurs du thermomètre numérique plus nettes. De plus, cela permettrait à notre programme de reconnaitre plus facilement les chiffres stockés dans la base de données afin de produire une meilleure identification des valeurs et ainsi pouvoir tracer le graphique représentant l'évolution de la température en fonction du temps.

2) Amélioration de la mobilité du matériel :

Pour éviter d'endommager les composants électroniques, le matériel ou encore de risquer de débrancher un fil en cas de déplacement du système, il serait plus judicieux d'encapsuler le matériel dans une boite en réservant une place précise pour chaque composant. De plus, cela permettrait de garder une position fixe pour la caméra et ainsi éviter de devoir redéfinir la zone de lecture des caractères à chaque utilisation du matériel.

3) Coordonnées du rectangle :

Avant chaque exécution du programme l'utilisateur doit définir une zone de capture pour la lecture de caractères, ce qui est laborieux. En effet, il doit lancer une première fois le programme afin de capturer une image du thermomètre puis rentrer les coordonnées du rectangle dans cet encadré :



Il doit ensuite arrêter le programme avant de le relancer.
Cela pose plusieurs problèmes, comme par exemple le fait de devoir recommencer cette tâche à chaque fois que l'on bouge la caméra.
Pour y remédier, il faudrait trouver un moyen de rendre les coordonnées du rectangle dynamique c'est-à-dire de les adapter automatiquement en mettant les bonnes coordonnées pour chaque déplacement de la caméra.

Une autre solution serait de fixer le matériel comme nous l'avons expliqué précédemment car ainsi les coordonnées du rectangle seraient constantes.

VII. Pistes pour la suite du projet :

Nous avons pu automatiser la partie contrôle du peltier et acquisition de la température du TP. Cependant, il reste plusieurs étapes dans ce TP.

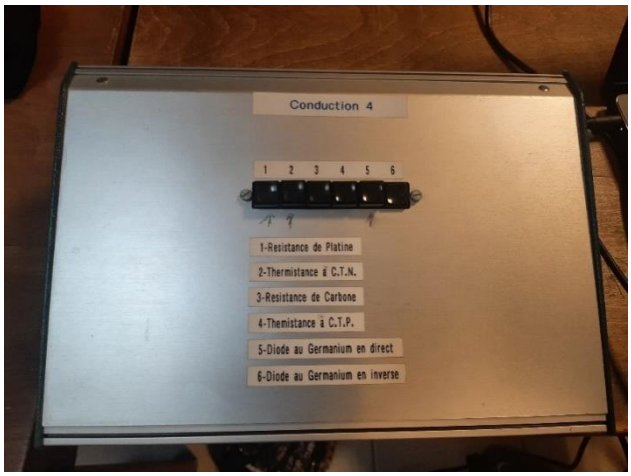
Il serait donc envisageable par la suite de mesurer automatiquement les caractéristiques I(V) des composants suivants :

- Résistances de Platine et de Carbone.
- Thermistances à C.T.N. et à C.T.P.
- Diode au Germanium en direct et en inverse.
- Diode à effet Zener en inverse

En effet, pour le moment les étudiants doivent réaliser les mesures de ces caractéristiques manuellement, ce qui est long et fastidieux. Automatiser les manipulations leur permettrait ainsi de pouvoir se focaliser sur l'interprétation des résultats.

Pour cela, il faudrait :

- Automatiser la commutation entre les sondes car elle se réalise manuellement pour le moment grâce à ce boîtier :



- Tracer les graphiques d'évolution des caractéristiques I(V) des sondes en fonction de la température

VIII. Conclusion :

Nous avons réussi à atteindre la plupart des objectifs que nous nous étions fixés. Malheureusement, la qualité de la caméra ne nous a pas permis d'obtenir le tracé de l'évolution de la température en fonction du temps ou le temps d'inertie.

De plus, ce projet nous a permis d'apprendre à maitriser de nouvelles fonctionnalités de LabVIEW comme LINX ou Vision Assistant. Cela nous a aussi permis de nous confronter à la réalisation d'un projet en respectant un cahier des charges précis dans un délais imparti.

Enfin, le fait de réaliser un projet pour pouvoir aider des étudiants dans la réalisation de leur TP a été une grande source de motivation car cela nous a donné un objectif concret à atteindre.