Sciences et technologies de l'Industrie et du développement durable

SIN : Maquettage d'une solution en réponse à un cahier des charges

Module SIN 1.1 : Concevoir un système local et permettre le dialogue entre l'homme et la machine

Activité : TP3 – IOWarrior – Réalisation de l'IHM pour la station météo

IO-Warrior

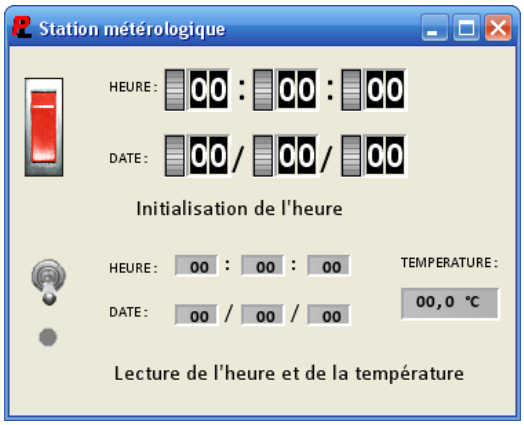
Generic universal I/O Controller for USB

Code Mercenaries

Sommaire

1 Présentation

Dans le but de répondre au cahier des charges fixé pour la station météorologique et le central, le StarterKit IOWarrior est utilisé avec des modules complémentaires RTC et Capteur.



La communication entre ces modules va se faire via une liaison I2C gérée par le composant IOW24.

L'utilisation du module RTC a été abordée dans l'activité précédente, il s'agit maintenant d'ajouter l'utilisation du capteur de température pour répondre au besoin de la station météorologique.

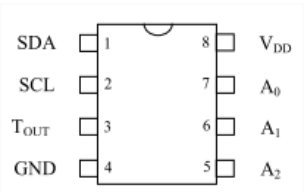
L'objectif est d'obtenir l'IHM suivante. Il faut pouvoir :

- ☒ initialiser l'heure
- ☒ lire l'heure
- ☒ afficher l'heure et la température

Pour cela, il faut compléter la DLL réalisée dans l'activité 2.

2 Le capteur de température DS1621

Le capteur de température est sous forme de circuit intégré : le DS1621.



Caractéristiques :

- ☒ Mesure de température de -55°C à +125°C
- ☒ Résultat sur 9 bits codé en complément à 2.
- ☒ Adresse I2C : 0b1001000 (les 3 bits de poids faibles sont configurables)

Il dispose de plusieurs registres contenant, notamment, le résultat de la conversion.

Instruction	Description	Commande de l'instruction	Données échangées
Lecture température	Lecture du résultat de la dernière conversion réalisée	0xAA	2 octets
Lecture compteur	Lecture de la variable Count_Remain	0xA8	1 octet
Lecture Slope	Lecture de la variable Count_Per_C	0xA9	1 octet
Start Convert T	Lancement de la conversion	0xEE	
Stop Convert T	Arrêt de la conversion	0x22	
Access Config	Ecriture ou lecture du registre de configuration	0xAC	1 octet

Le résultat est codé sur 9 bits sur 2 octets, de la manière suivante :

MSB

b8	b7	b6	b5	b4	b3	b2	b1

LSB

b0	X	X	X	X	X	X	X

Temp_Read

Pour obtenir une précision meilleure, il faut appliquer la formule suivante :

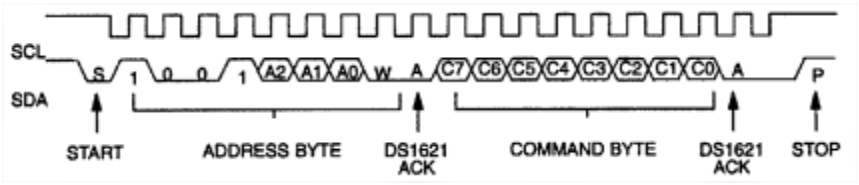
Le registre '*Configuration*' gère la conversion, ainsi que la sortie 'T_{OUT}' (cette sortie n'a pas d'utilité pour le projet).

b7	b6	b5	b4	b3	b2	b1	b0
DONE	THF	TLF	NVB	X	X	POL	1SHOT

☒ Bit '*DONE*' : Drapeau de fin de conversion (NL1 : conversion terminée).
☒ Bit '*1SHOT*' : Mode conversion continue (NL0) ou unique (NL1).

2.1 Envoi d'une commande (Start/Stop convert)

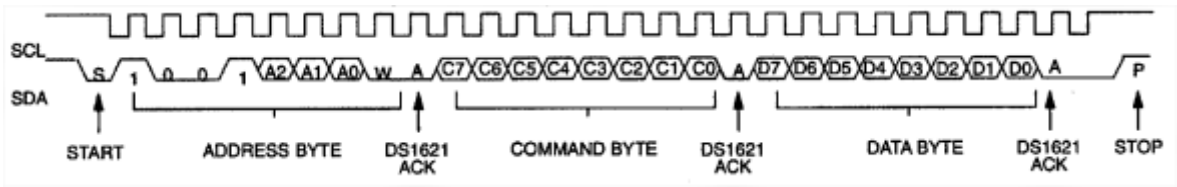
Le fabricant donne les informations présentes sur le bus I2C :



Le maître envoie les bits d'adresse du capteur et de l'opération (écriture). En réponse à l'acquittement du capteur, il envoie un octet de commande.

2.2 Ecriture dans le registre de configuration

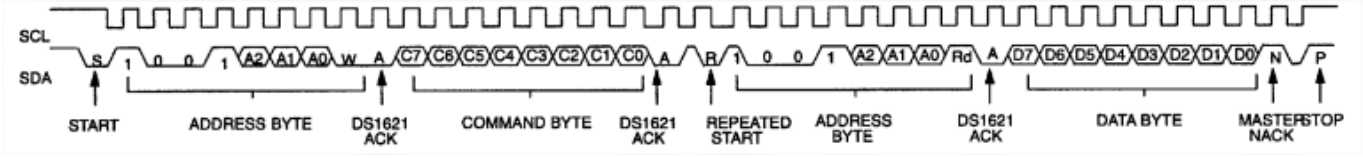
Le fabricant donne les informations présentes sur le bus I2C :



Le maître envoie les bits d'adresse du capteur et de l'opération (écriture). En réponse à l'acquittement du capteur, il envoie un octet de commande du registre de configuration (0xAC) puis un octet de données à placer dans ce registre.

2.3 Lecture du registre de configuration, du Slope, du compteur

Le fabricant donne les informations présentes sur le bus I2C :

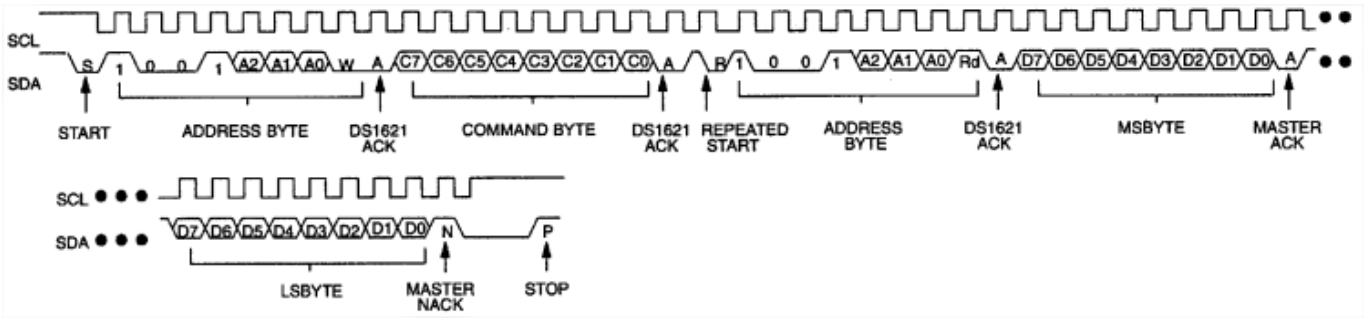


Le maître envoie les bits d'adresse du capteur et de l'opération (écriture). En réponse à l'acquittement du capteur, il envoie l'octet de commande du registre choisi. Après à l'acquittement du capteur, il envoie de nouveau un bit de start suivi des bits d'adresse et de l'opération (lecture cette fois). Le capteur envoie enfin le contenu du registre sélectionné.

2.4 Lecture de la température

Le fabricant donne les informations présentes sur le bus I2C :

Le maître envoie les bits d'adresse du capteur et de l'opération (écriture). En réponse à l'acquittement du capteur, il envoie l'octet de commande des registres contenant la valeur de la température (0xAA). Après à l'acquittement du capteur, il envoie de nouveau un bit de start suivi des bits d'adresse et de l'opération (lecture cette fois). Le capteur envoie enfin le contenu des registres sélectionné.



3 Manipulation

La programmation de la DLL va se faire en parallèle avec la programmation sous ProfiLab Expert et les relevés de trame.

- Ouvrir le projet '*DLL_RTC.dev*' réalisé pendant l'activité 2 et le sauvegarder sous un autre nom.

3.1 Définition des entrées et des sorties de la DLL

On souhaite maintenant ajouter la mesure et l'affichage de la température. Pour cela, il faut ajouter une sortie à la DLL.

- Modifier les fichiers '*dllmain.cpp*' et '*dll.h*' pour ajouter 1 sortie nommée '*Temp*'.
- Compiler le projet.

3.1.1 Programmation graphique

- Ouvrir le programme graphique, établi pendant l'activité 2, et le sauvegarder sous un nouveau nom.
- Modifier le fichier importé dans le composant DLL et vérifier que la nouvelle sortie a été prise en compte.
- Ajouter un composant '*Afficheur numérique*' pour l'affiche de la température et modifier l'IHM comme présenté dans le cahier des charges.

3.2 Initialisation et arrêt de la DLL

Il faut compléter les fonctions '*CSimStart*' et '*CSimStop*' afin de tenir compte de la présence du capteur de température.

Au départ, il faut initialiser le capteur. On souhaite une mesure toutes les 30 secondes. Il faut donc initialiser à un mode de conversion unique.

3.2.1 Fonction CSimStart de la DLL

En plus d'initialiser la communication entre le PC et le composant IOW24 et d'ouvrir la liaison I2C, il faut initialiser le capteur de température.

Pour répondre au cahier des charges, il faut réaliser une mesure horodatée toutes les 30 secondes. Il faut donc initialiser à un mode de conversion unique.

- Dans le fichier '*fonction.cpp*', créer une fonction '*InitDS1621*' Cette fonction retourne un booléen et passe en paramètre l'identificateur du composant IOW24. :

  BOOLEAN InitDS1621(IOWKIT_HANDLE IOWarrior)

- A partir de la présentation du capteur de température et notamment du paragraphe 2.2, indiquer le nombre d'octet à envoyer sur la liaison I2C, ainsi que leur valeur pour le configurer
  .
- En prenant exemple sur la fonction '*InitRTC*', écrire le programme de la fonction '*InitDS1621*'.

- Compléter enfin la fonction '*CSimStart*' pour réaliser la configuration du capteur. Il faut tenir compte d'une éventuelle erreur d'écriture.

3.2.2 Fonction CSimStop de la DLL

La fonction 'CSimStop' reste inchangée. Elle doit fermer la liaison I2C et clore la communication entre le PC et le IOW24.

- Compiler le projet.

3.2.3 Programmation graphique

- Ouvrir le document de programmation graphique précédent. Il n'y a pas de modification à apporter à l'IHM.

- Exécuter le programme. Vérifier que l'initialisation du capteur de température se fait correctement.

3.2.4 Relevés de trames I2C

- Proposer une méthode pour relever la trame d'initialisation du capteur de température avec l'analyseur logique ou l'oscilloscope.

Utiliser la ressource sur l'analyseur logique ou l'oscilloscope MSO2024 pour suivre la méthode de relevé de chronogrammes.

- Relever la trame et vérifier les informations présentes. On pourra ajouter des MessageBox dans le programme de la DLL.

3.3 Comportement de la DLL

Cette partie définit le comportement de la DLL en fonctionnement.

Toutes les 30 secondes, une mesure horodatée de la température doit être effectuée.

3.3.1 Fonction CCalculate

La fonction '*CCalculate*' doit :

- ☒ Initialiser la RTC si l'entrée '*Init*' est au niveau logique haut. Les valeurs présentes sur les entrées de la DLL sont envoyées vers la RTC (travail effectué dans l'activité 2).
- ☒ Lire le capteur de température et la RTC toutes les 30 secondes. On considérera que toutes les 30 secondes, l'entrée RD passe au niveau logique haut.
- ☒ Calculer la température en utilisant seulement les 8 bits de poids forts du résultat de la conversion.
- ☒ Afficher la température et l'heure.
- ☒ Vérifier s'il y a erreur à chaque transmission.

- Dans le fichier '*fonction.cpp*', créer 3 fonctions '*Lancement_Conversion*', '*Attente_Conversion*' et '*Lecture_Temperature*'.

- En prenant exemple sur les autres fonctions et à l'aide de la présentation du composant DS1621, écrire le programme de chaque fonction créée.

- Compléter la fonction '*CCalculate*' pour répondre au cahier des charges.

- Compiler le projet pour obtenir la DLL.

3.3.2 Programmation graphique

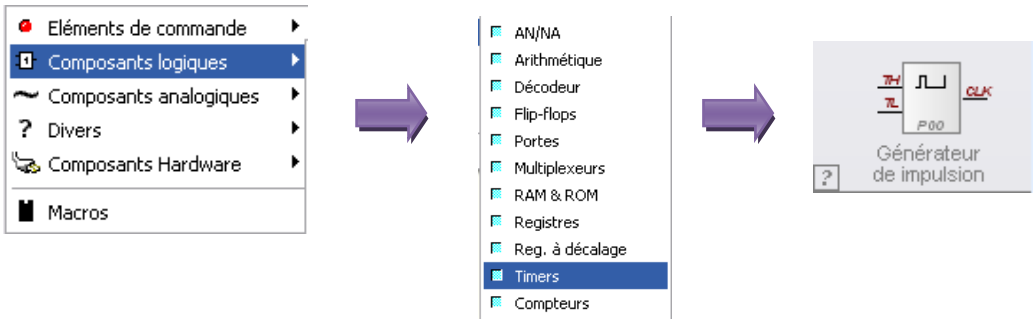
- Ouvrir le document de programmation graphique précédent.

3.3.2.1 Mesure toutes les 30 secondes

Le cahier des charges impose une mesure toute les 30 secondes. Il faut donc un niveau haut toutes les 30 secondes sur l'entrée RD.

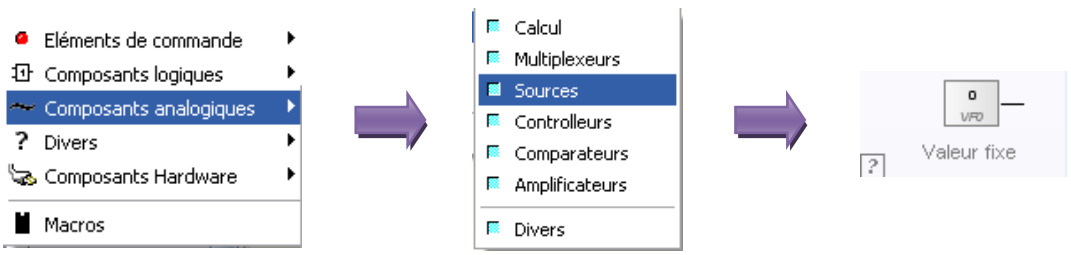
Pour cela, il faut générer un signal périodique présent sur l'entrée RD lorsque l'interrupteur de lecture est activé.

- Sélectionner la librairie '*Composants logiques*' puis '*Timers*'. Sélectionner le composant '*Générateur d'impulsion'* :



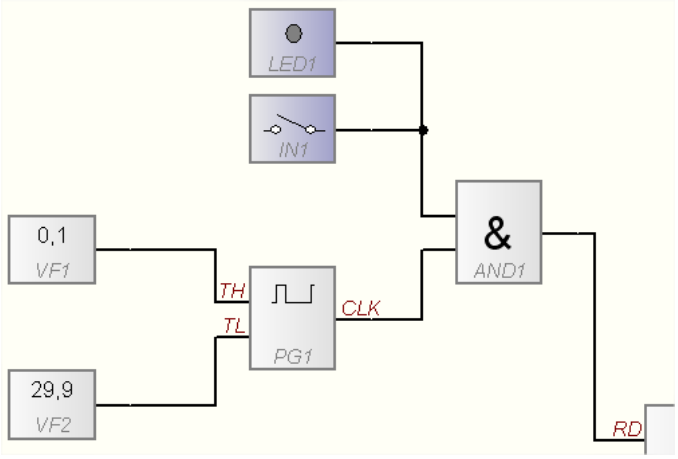
Ce composant génère un signal périodique dont les durées à l'état bas (TL) et à l'état haut(TH) sont programmables. Pour obtenir une période de 30 secondes, on prendra TL = 29.9s et TH = 0.1s

- Sélectionner la librairie '*Composants analogiques*' puis '*Sources*'. Sélectionner le composant '*Valeur fixe'* :



- Connecter ce composant sur les entrées du générateur d'impulsion et configurer les valeurs TH et TL en double cliquant dessus.
- Placer un composant 'AND' reliant la sortie du générateur d'impulsion et l'interrupteur de lecture.

Le schéma est le suivant :

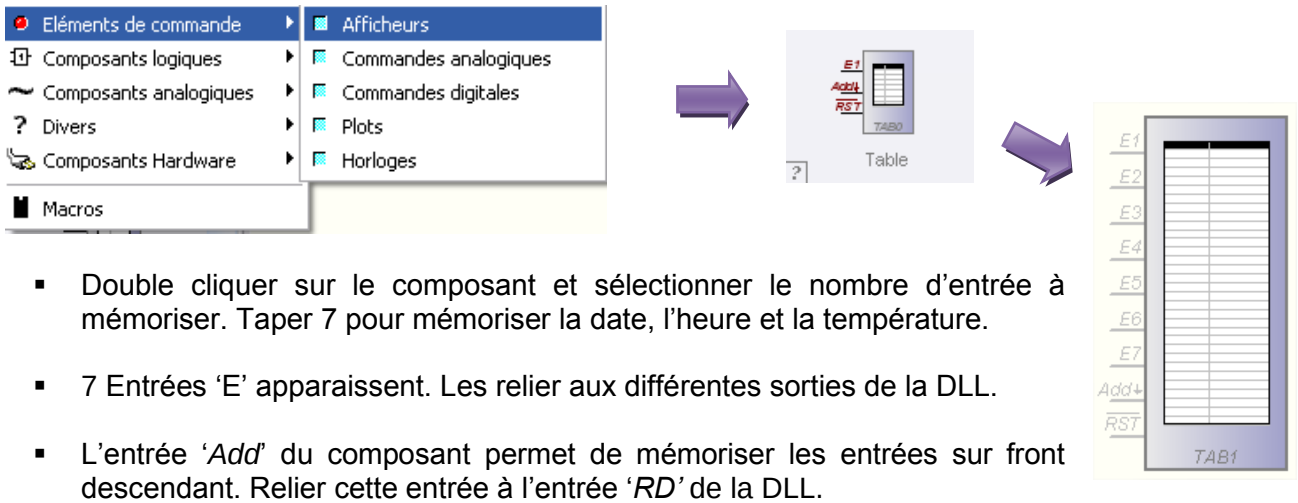


- Exécuter le programme et valider l'affichage de la température, de la date et de l'heure.

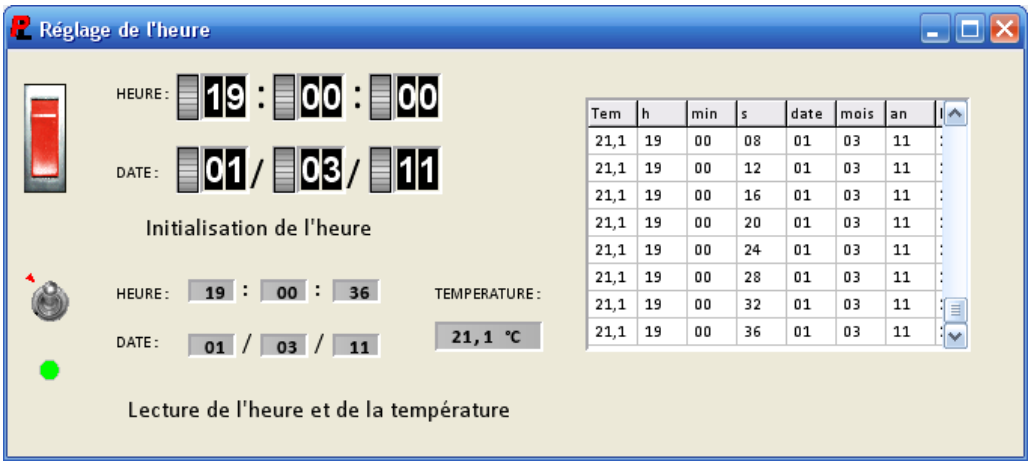
3.3.2.2 Enregistrement des mesures

Le cahier des charges impose un enregistrement des mesures horodatées. Il est possible d'enregistrer les mesures dans un fichier de type texte, Excel, Word.

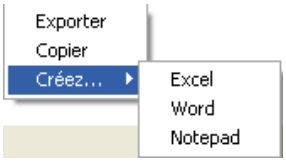
- Sélectionner la librairie '*Eléments de commande*' puis '*Afficheurs*'. Sélectionner le composant '*Table'* :



- Double cliquer sur le composant et sélectionner le nombre d'entrée à mémoriser. Taper 7 pour mémoriser la date, l'heure et la température.
- 7 Entrées 'E' apparaissent. Les relier aux différentes sorties de la DLL.
- L'entrée '*Add*' du composant permet de mémoriser les entrées sur front descendant. Relier cette entrée à l'entrée '*RD'* de la DLL.
- Modifier l'IHM. Double cliquer sur la table et nommer les colonnes, définir la largeur des colonnes, ainsi que le format des informations.



Pendant l'exécution du programme, cliquer droit sur la table un menu contextuel apparaît. Cliquer sur '*Créez...*' pour enregistrer les données de la table sous le format désiré.

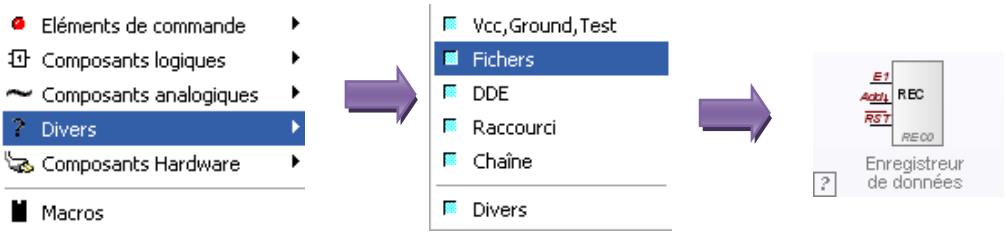


Remarques :

- ☒ Les données ne sont par enregistrées dans le fichier au fur et à mesure.
- ☒ La table a une contenance de 16000 valeurs par colonne.

Il est possible de mémoriser les données dans un fichier automatiquement.

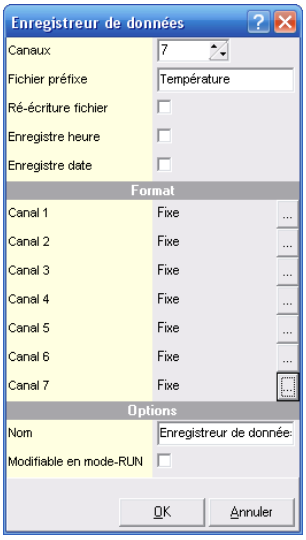
- Sélectionner la librairie '*Divers*' puis '*Fichiers*'. Sélectionner le composant '*Enregistreur de données'*. Le mettre à la place du composant '*Table'*.



- Double cliquer sur le composant et sélectionner le nombre d'entrée à mémoriser. Taper 7 pour mémoriser la date, l'heure et la température.

- Définir le format de l'information pour chaque entrée, le nom di fichier d'enregistrement. Il est possible d'enregistrer la date et l'heure du PC.

- 7 Entrées 'E' apparaissent. Les relier aux différentes sorties de la DLL.



L'entrée '*Add*' du composant permet de mémoriser les entrées sur front descendant. Relier cette entrée à l'entrée '*RD'* de la DLL.

La table a disparu de l'IHM.

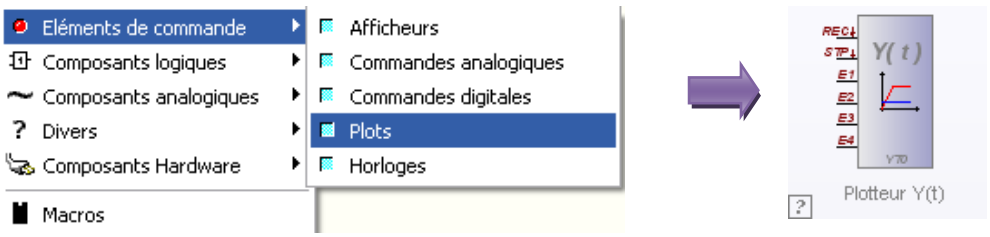
Lors de l'exécution du programme, les données sont enregistrées dans le fichier à chaque front descendant sur l'entrée '*Add'*. Le fichier est fermé dès l'arrêt du programme.

- Retrouver le fichier créé dans le dossier suivant :

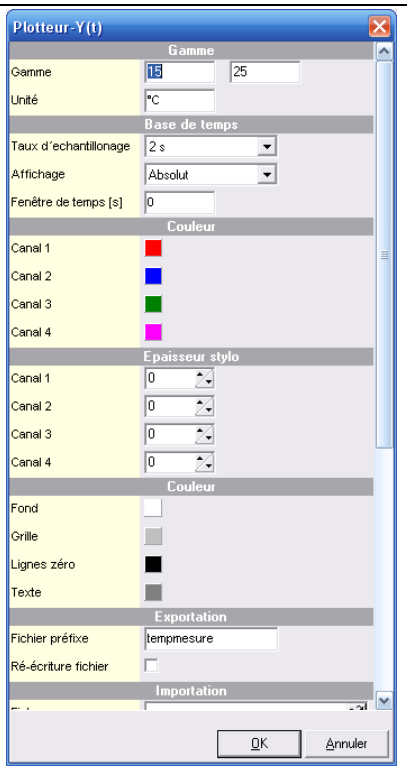
C:\Program Files\ProfiLab-Expert40\Data

Pour observer l'évolution de la température en en fonction du temps, on va ajouter un graphe à l'IHM.

- Sélectionner la librairie '*Eléments de commande*' puis '*Plots*'. Sélectionner le composant '*Plotteur Y(t)'*.



Ce composant peut tracer 4 signaux simultanément. L'entrée 'REC' permet l'enregistrement de la valeur du signal à tracer, sur front descendant. L'entrée 'STP' stoppe le tracé sur front descendant.



- Dans l'IHM apparaît un graphe. Double cliquer sur le graphe et sélectionner la plage des valeurs mesurées, l'unité, le taux d'échantillonnage...
- Lancer le programme. Le chronogramme débute à partir du premier front descendant sur l'entrée '*REC*'.

3.3.3 Relevés des trames I2C

Selon le matériel à disposition, on pourra utiliser un oscilloscope ou un analyseur logique

- Configurer le matériel pour relever et décoder une trame I2C en mode monocoup.
- Relever la trame I2C lors de la lecture de la température. On pourra ajouter des MessageBox dans le programme de la DLL.

4 Amélioration de la conversion

La valeur de la température est calculée à partir du résultat issu du capteur (8 bits de poids forts)

- Modifier le programme de la DLL pour calculer la température à partir de l'expression donnée dans le paragraphe 2.
- Proposer une méthode de mesure pour relever les différentes valeurs mesurées (Count_Per_C, Temp_Read et Count_Remain). On pourra ajouter des MessageBox dans le programme de la DLL.
- Réaliser les mesures.
- Vérifier le résultat affiché sur l'IHM.