ISIMA

LINUX EMBARQUE

Gestion de capteur de température

Gestion et simulation d'un capteur de température en utilisant une plateforme embarquée de type Raspberry Pi.

BARBESANGE Benjamin – GARÇON Benoît

15/11/2015

Table des matières

Table des illustrations

INTRODUCTION

Ce projet s'inscrit dans le cursus de seconde année à l'ISIMA. Le but est d'implémenter la simulation d'un capteur de température sur une machine hôte. Ce capteur devra être géré par un serveur, en utilisant une carte embarquée de type Raspberry Pi.

L'affichage des résultats du capteur sera effectué par le biais d'un site web, également hébergé sur la carte Raspberry Pi. L'utilisateur sera également en mesure d'interagir avec le capteur, pour stopper l'acquisition de température ou modifier la fréquence d'acquisition.

Au fil des premiers TP de ce module, nous avons également appris à utiliser des modules dynamiques sous Unix. De plus, nous avons appréhendé l'utilisation de pilotes notamment pour gérer des LEDs que nous avions connectées sur le port GPIO du Raspberry Pi.

PRESENTATION DE L'ENVIRONNEMENT

Le PC hôte

Nous disposons d'un PC hôte sur lequel nous allons simuler le capteur de température. Il est équipé d'Ubuntu 12.04 et permet de communiquer avec la carte Raspberry Pi que nous détaillerons plus tard.

Sur cette machine, nous avons un accès internet ainsi que les droits d'administrateur afin de pouvoir gérer au mieux notre projet.

La carte Raspberry Pi

Description

Cette carte fonctionne comme un PC dans un format réduit. Elle dispose d'un processeur, d'une mémoire vive ainsi que d'une mémoire flash (carte mémoire). Nous disposons également de connectiques comme de l'USB, Ethernet, HDMI.

Pour pouvoir faire fonctionner cette carte, nous avons besoin de matériel :

- Un clavier (USB),
- Une souris (USB),
- Un écran,
- Un câble Ethernet,
- Un câble d'alimentation (USB),
- Un adaptateur VGA → HDMI,
- Un adaptateur Carte Mémoire → USB,
- Un câble RS232 → USB.

Afin d'établir la communication avec notre PC hôte, nous allons utiliser la connectique RS232.

Configuration

Afin de faire fonctionner notre capteur, il nous a été indispensable d'installer des paquets. Pour ce faire, nous avons dû configurer une connexion réseau dont voici les données :

- Adresse MAC : b8:27:eb:3d:c4:9d
- Adresse IP : 172.16.44.15
- Passerelle : 172.16.47.250
- DNS : 172.16.32.250 OU 172.16.32.252

Un serveur ftp est mis en place afin de faciliter les transferts de données entre le PC hôte et notre système embarqué. Son adresse est la suivante : ftp://172.16.44.15.

L'interaction avec ce système se faisant par le biais d'un navigateur internet, nous avons installé un serveur web ; dans notre cas, c'est APACHE2 couplé au paquet PHP5, afin de produire une page dynamique.

PRESENTATION DE LA SOLUTION

Vous trouverez tous les fichiers de ce projet dans l'archive jointe avec ce rapport si vous le consultez numériquement.

Capteur de température

Le code permettant de gérer ce capteur est trouvable dans le fichier *claude.c*. Un Makefile est disponible et permet de compiler ce fichier ainsi que la structure de file que nous utilisons.

Le capteur de température permet la création de trames fictives de températures générées aléatoirement. Les températures aléatoires ne le sont pas totalement et s'appuient sur les températures précédentes pour obtenir un résultat plus proche d'un résultat réel.

Les trames suivent le format suivant :

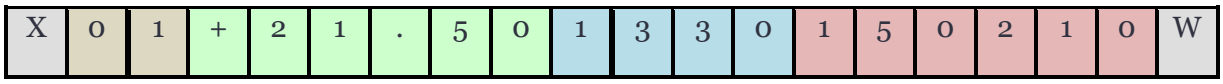

Figure 1 –Format de la trame de température

Les trames de température sont composées de 20 caractères :

- Le premier caractère (X) indique le début de la trame de température,
- Le deuxième et le troisième caractère identifient le type de la trame,
- Les 6 caractères suivants contiennent la température avec le signe,
- Les 4 suivants indiquent l'heure à laquelle la mesure a été réalisée,
- Les 6 suivants la date (JJMMAA) à laquelle la mesure a été réalisée,
- Enfin le dernier caractère 'W' marque la fin de la trame.

Le programme simulant le capteur est organisé en 3 threads :

- Un thread de lecture des trames,
- Un thread d'envoi de trames,
- Un thread générant les températures.

A ces threads il faut aussi ajouter le programme principal qui gère les différents ordres reçus par le serveur.

Nous utilisons ici des mutex afin de sécuriser l'écriture dans les files de réceptions et d'envoi des trames et l'accès aux booléens d'état des acquittements et du programme lui-même.

La première étape de ce programme est d'établir la connexion avec la carte embarquée par le port RS232.

Le programme va constamment envoyer des trames en attendant en retour un acquittement de réception. Si aucun acquittement n'est reçu au bout de 3 tentatives séparées de 200µs, alors nous abandonnons cette trame et nous passons à la trame suivante.

Dans le cas où l'on reçoit une trame de commande (changement de fréquence ou arrêt du capteur), on doit traiter celle-ci et envoyer un acquittement prioritairement, et donc laisser de côté l'envoi de trames pour le moment.

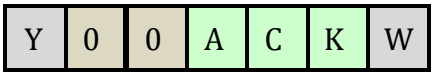

Figure 2 – Format d'une trame d'acquittement

La trame d'un acquittement de compose de la manière suivante :

- Le premier caractère 'Y' indique le début,
- Le deuxième et le troisième caractère identifient le type de la trame,
- Les 3 caractères suivants sont 'ACK',
- Le dernier caractère 'W' marque la fin de la trame.

Dans notre programme, la période d'échantillonnage maximale de génération des températures est de 50ms soit 20 Hz pour la fréquence, c'est un c as limite puisque certaines trames devront être renvoyées plusieurs fois pas le capteur. Nous poserons une limite fréquentielle basse à 5 000 ms (5s) soit 0.2 Hz, bien sûr cette valeur est arbitraire car il n'y a évidemment pas de limite fréquentielle minimale.

Maintenant que nous avons posé les bases, il est simple de comprendre le fonctionnement du capteur.

Le programme principal lance d'abord les threads d'envoi, de réception et de génération des trames de température. Tant qu'il n'a pas reçu l'ordre de s'éteindre, le programme principal va traiter les ordres reçus et attendre.

De son côté le thread d'envoi, va vérifier par le mutex la disponibilité d'accès de la file d'envoi et si c'est autorisé, il va envoyer le sommet de la file vers le serveur puis attendre son acquittement ou envoyer lui-même un acquittement.

Le thread de réception va servir de filtre aux messages reçu, il va déterminer si les trames reçues son correctes, si oui, si c'est un ordre, il va signaler au thread d'envoi d'envoyer un acquittement et mettre la trame dans la file de traitement pour être traitée par le programme principal sinon si c'est un acquittement, on le signal au thread d'envoi par un des booléen-mutex (=structure contenant un booléen et un mutex).

Le thread de génération des trames de températures est assez naïf. On va juste vérifier que l'accès à la file d'envoi est autorisé et on va y ajouter une trame assemblée par snprintf tous les intervalles de temps définis par l'utilisateur. Par défaut cet intervalle est minimal et vaut 20ms.

Serveur Raspberry Pi

Le programme principal du capteur est le fichier *manager.c*. Un Makefile permet de compiler simplement ce fichier ainsi que la structure de file. Le site web se trouve dans le répertoire *www/*.

Présentation

Le Raspberry Pi se compose de différents éléments contribuant au fonctionnement général du capteur. Dans un premier temps, nous trouvons un programme qui ouvre la connexion au port RS232 et se compose de threads :

- Thread de lecture : attend les trames de températures et les acquittements de commandes,
- Thread d'écriture : envoie les commandes et les acquittements de trames de température,
- Thread de commandes : surveille les commandes envoyées à partir du serveur web.

Etant donné que certains threads vont se partager des informations comme les files d'attente des trames ou les booléens d'attente d'acquittements, nous avons décidé d'utiliser des mutex, qui vont ainsi éviter les soucis lors d'une écriture dans la même structure.

Acquisitions

Les acquisitions de trames de température débutent dans le thread de lecture où l'on va sans cesse lire les caractères envoyés. Lorsque l'on trouve le caractère de début de trame 'X', nous lisons les caractères suivants jusqu'au caractère terminal 'W' et les stockons dans une chaîne.

Si la chaîne correspond au format précédemment cité, (nous vérifions seulement le code de la trame), on la place dans la file de trames à afficher, puis on indique que l'on doit envoyer un acquittement avec un booléen.

Le thread d'écriture va ensuite à son tour voir qu'il faut envoyer un acquittement par le biais du booléen. On envoie cet acquittement puis on remodifie le booléen pour indiquer que l'on a envoyé l'acquittement. Juste après ceci, le thread va récupérer la trame dans la file correspondante et l'écrire dans un fichier qui sera lu en PHP pour être affichée sur le site.

Site web

Notre serveur étant équipé d'APACHE2 ainsi que de PHP5, nous pouvons implémenter l'interface web.

Pour ce faire, nous avons choisi la bibliothèque Bootstrap sb-admin. Nous avons choisi cette bibliothèque puisqu'elle permet de créer facilement des pages web dont le rendu est de qualité. De plus, son utilisation est très simple et permet de créer un rendu qui s'adapte automatiquement à la taille de l'écran sur lequel le site est affiché. Ceci rend possible la consultation sur mobile (web responsive).

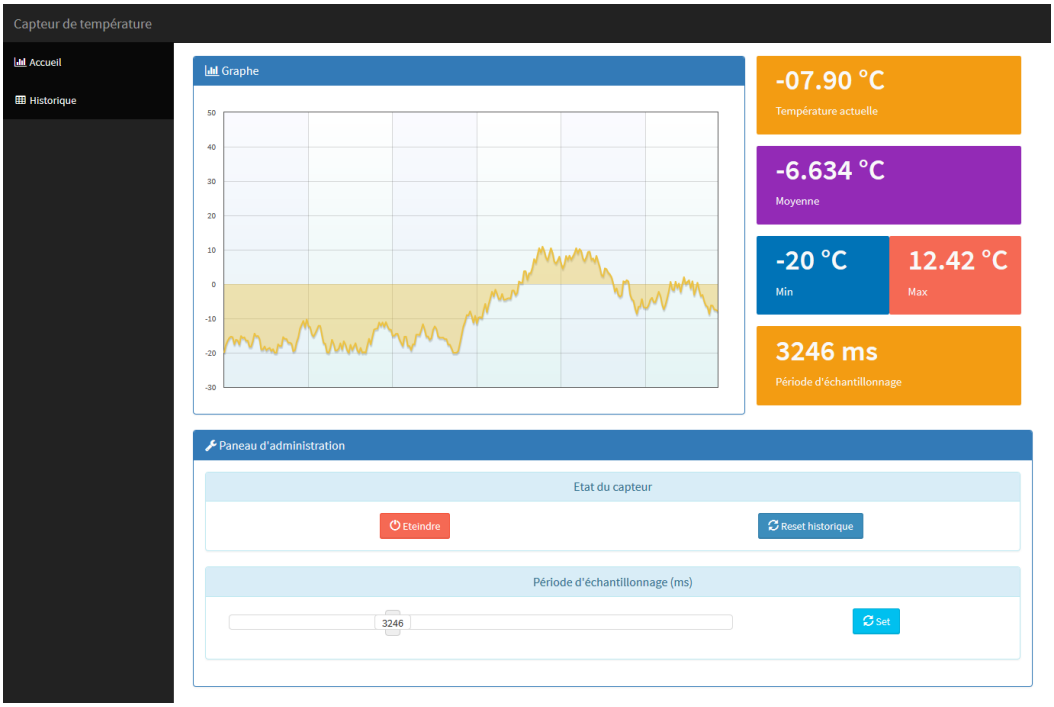

Figure 3 – Capture d'écran du site web

Comme nous pouvons le voir, il est possible de visualiser l'évolution de la température sur un graphe. Les panneaux sur la droite affichent de nombreuses valeurs utiles comme :

- La valeur courante de température ;
- La moyenne de température depuis le début de l'historique ;
- Les valeurs minimum et maximum prises par la température ;
- La période d'échantillonnage.

Ces informations sont complétées par un panneau se situant sous le graphe et qui permet quant à lui de stopper le capteur, de réinitialiser l'historique qui peut vite devenir gourmand en espace mémoire et de changer sa fréquence d'acquisition au moyen d'un slider.

Etant donné que la connexion internet est établie de manière locale à l'ISIMA, il n'est pas possible d'accéder à cette page en dehors de l'école. Cependant, une reconfiguration de la connexion permettrait d'y accéder n'importe où dans le monde.

Comme demandé pour ce projet, nous avons donc réalisé une historisation des mesures prises par le capteur. Pour cela nous avons choisi de ne pas opté pour une base de données qui aurait été bien trop gourmande en ressources bien qu'un peut plus efficace sur le traitement des données. Mais pour l'historisation, un simple fichier texte suffisait donc.

Dans un premier temps nous avions pensé à ne garder que les N dernières mesures mais cela devenait vite très lourd d'effacer à chaque fois les valeurs les plus anciennes. De plus l'utilisateur aura parfois besoin d'un historique bien plus long que ce qui aurait pu lui être proposé. C'est pourquoi nous avons décidé de mettre un bouton sur l'interface pour redémarrer l'historique, ce qui donne le pouvoir à l'utilisateur.

Pour visualiser cet historique c'est très simple, il suffit d'aller dans l'onglet historique et sur cette nouvelle page apparaitra le tableau non dynamique des mesures prises. Ce tableau se constitue ainsi :

- Numéro de la mesure ;
- Date de la mesure ;
- Valeur de la mesure.

En voici donc une capture d'écran :

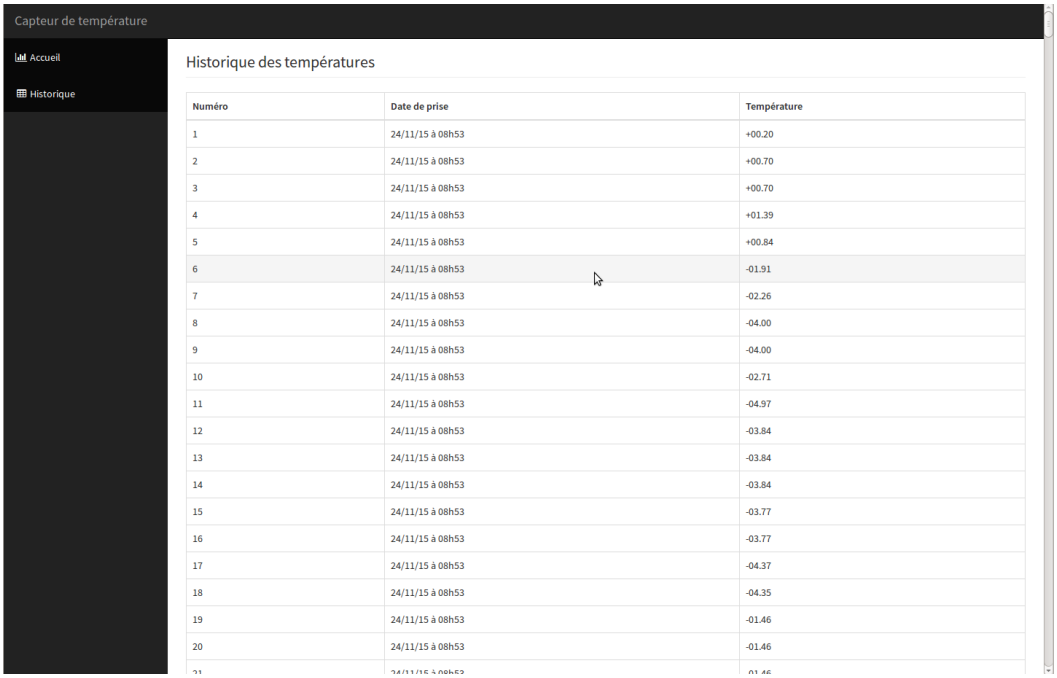

Figure 4 - Capture d'écran de l'historique

Interactions

Les interactions sont possibles par le biais du site web présenté précédemment.

Si l'on souhaite stopper le capteur, il suffit de cliquer sur le bouton correspondant. Ceci va appeler un script PHP correspondant, qui va aller écrire dans un fichier la trame STOP à traiter dans le programme.

Z	0	2	0	0	0	0	6	0	W

Figure 5 – Format d'une trame de changement de fréquence

La trame de changement de fréquence se compose de la manière suivante :

- Le premier caractère 'Z' indique le début de la trame,
- Le deuxième et troisième caractère identifient le type de trame,
- Les 6 caractères suivants indiquent la période d'échantillonnage (en ms)
- Enfin le caractère 'W' marque la fin de la trame.

Q	0	3	S	T	O	P	W

Figure 6 - Format d'une trame stop

La trame d'extinction du capteur se compose de la manière suivante :

- Le premier caractère 'Q' indique le début de la trame,
- Le deuxième et le troisième caractère identifient le type de trame,
- Les 4 caractères suivants correspondent au mot 'STOP',
- Enfin le caractère 'W' indique la fin de trame.

Notre programme de gestion va se charger de lire cette trame et de la vérifier. Dans le thread d'écriture, nous allons mettre un booléen en attente d'acquittement. Tant que l'on ne reçoit pas d'acquittement, nous tentons d'envoyer la trame à nouveau en attendant 200 µs à chaque envoi et avec un maximum de 3 tentatives, après lesquelles nous ignorons cette commande et passons à la suivante si c'est possible.

L'acquittement sera reçu par le thread de lecture en détectant le code de celui-ci. Si l'acquittement est correct, le booléen d'attente est remodifié pour indiquer que l'on a l'acquittement et que l'on peut passer à la commande suivante.

A chaque fois que l'on lancera le programme, la fréquence d'acquisition est de 50ms malgré les indications du site. Il faut donc penser à changer la période d'échantillonnage à chaque relance du programme du capteur.

Le changement de température se fait de la même manière. C'est seulement la construction de la trame à envoyer qui est différente.

CONCLUSION

La finalité de ce projet est qu'il est très complet. En effet, nous avons dû réfléchir à l'organisation de nos différents programmes afin qu'ils collaborent.

Après une phase de test et de découverte du matériel, nous avons configurés les différents éléments dont nous nous sommes servis, tels que la connexion internet, le serveur web.

Il en résulte une application permettant effectivement de visualiser les résultats du capteur de température dans un navigateur, pour toute personne connaissant l'adresse de celui-ci. De plus, l'utilisateur est capable d'interagir en temps réel avec le système, soit pour le mettre en marche, soit pour le stopper ; ou encore changer la fréquence d'acquisition.

Ce projet permet la gestion d'un capteur de température, cependant le fonctionnement serait similaire si l'on souhaitait implémenter la gestion d'arrosage de plantes, ou encore pour des applications de domotique.

Pour aller plus loin nous pourrions proposer de développer un véritable capteur en lieu et place de l'ordinateur le simulant afin de pouvoir aboutir à un outil réel et utilisable. Nous pourrions aussi rendre accessible le serveur à bien plus de monde que le réseau local de la salle de TP pour pouvoir partager à un grand nombre de personnes notre travail.