Séminaire Industriel
RICM

21 janvier 2010

Tutoriel Senslab

INRIA

Clément Burin des Roziers

Loïc Lemaître

Jean-Pierre Poutcheu

Sommaire

Introduction

Senslab est une plateforme d'expérimentation en réseaux de capteurs. Elle a pour but de faciliter et d'automatiser le déploiement, le test, et le déroulement d'applications de réseaux de capteurs sans fil.

Réseaux de capteurs sans fils

Un réseau de capteurs sans fil, l'objet d'étude et de test de la plateforme Senslab, est un ensemble de petits objets communicants entre eux, dotés de capacités de calcul et de mesure. Cela permet l'observation d'un phénomène physique à grande échelle, que ce soit au niveau densité de points de mesures ou au niveau étendue de la zone observée, avec un déploiement simplifié du fait de l'absence de fils les reliant (alimentation comme communication).

Applications types

Les domaines d'applications des réseaux de capteurs sans fil sont vastes et différents. Il est possible cependant d'en citer quelques uns, qui montrent cette diversité.

- **Régulation de l'arrosage sur des champs d'agriculture** : en mesurant en différents endroit l'humidité de la terre d'un champ d'agriculture, et en receuillant ces données, il est possible de déterminer les zones à irriguer en priorité ansi que celles pour lesquelles cela est inutile.
- **Surveillance d'une zone, détection d'intrus** : en équipant une zone à suveiller (bâtiment ou espace ouvert), il est possible de détecter le passage ou la proximité de personnes ou véhicules.
- **Monitoring sportif :** en dotant un sportif de différents capteurs situés à des endroits stratégiques du corps humain, on peut observer la performance de celui ci d'un manière plus précise qu'avec par exemple un ralenti vidéo, ce qui peut permettre d'établir de meilleurs techniques d'entrainements pour le sportif. Par exemple en plaçant des capteurs de mouvement sur les articulations d'un coureur de marathon.

Enjeux technologiques

Différents enjeux technogiques existent autour des réseaux de capteurs. Le perimer est l'autonomie énergétique. Les capteurs sans fil sont le plus souvent alimentés par une petite batterie, pour s'affranchir de toute connexion filare et permettre un déploiement vaste.

Dans le cadre d'un déploiement d'un grand nombre de capteurs sur une grande zone, il est bien concevable que le remplacement des batteries des capteurs ne peut devoir se faire fréquemment ! Une telle procédure serait couteuse en temps et argent, pénible, et produirait des arrêts momentanés sur le réseau.

Si au contraire les batteries ne peuvent pas être remplacées, il faut augmenter au maximum la durée de vie de chaque capteur (une durée de 5 ans avec 2 piles AA). Il faut donc limiter les facteurs consommants : mesures, calculs, communications radio.

D'autre part, le médium qu'est un canal radio est partagé entre tous les équipements l'utilisant. Il est

nécessaire d'organiser les éléments communicant pour rendre le canal utilisable, et fiabiliser au maximum les transmissions (couche 2 du modèle OSI : MAC).

A l'échelle du réseau, il faut organiser l'ensemble de ses noeuds pour permettre à chaque donnée d'arriver à la bonne destination, il s'agit de la notion de routage (couche 3 du modèle OSI : NETWORK).

Plateforme Senslab

La plateforme Senslab met à disposition des utilisateurs sur 4 sites en France un total de 1024 noeuds capteurs déployés, programmable et monitorables. Cette plateforme permet donc d'évaluer l'efficacité de solutions aux enjeux technologiques vu ci-dessus (consommation d'énergie, réussite des communications, ...)

De plus, le déploiement de programme sur l'ensemble des capteurs peut se faire de façon aisée, à distance, tout comme l'interaction avec chacun des capteurs.

Chacun des 1024 capteurs est accéssible indépendamment à l'utilisateur qui l'a réservé, et voici la liste des interactions qu'il est possible de faire :

- mesure de la consommation électrique;
- mesure de l'activité radio;
- mesure de la température et luminosité alentour;
- injection de bruit radio;
- génération de stimulii spécifiques;
- arrêt/démarrage/reset forcés (pour simuler une coupure réseau par exemple;
- communiquer avec le capteur via son port série (envoyer/recevoir des données).

Pour cela, chaque utilisateur Senslab a accès sur le site web de la plateforme à une interface lui permettant d'enregistrer des firmwares à tester, des profiles de mesures (fréquence des mesures de consommation etc..), et de créer des expérimentations à partir de ceux ci.

L'ensemble des noeuds étant utilisables par tous les utilisateurs, un système de réservation est mis en place. Il s'agit d'OAR, mais ce service n'est pas encore disponible à l'heure actuelle.

Une machine virtuelle est mise à disposition de chaque utilisateur sur le serveur de Senslab. C'est à partir de cette machine virtuelle qu'il est possible d'interagir avec une expérience en cours, ou plus précisémment avec chaqun des noeuds d'une expérience.

But

Le but de ce tutoriel est de prendre en main la plateforme, de bien comprendre les différents éléments la constituant, et de réaliser et tester quelques applications pour capteurs, et réseaux de capteurs.

Vous serez guidé pas-à-pas tout au long des différentes étapes, n'hésitez pas à modifier la procédure selon vos envie pour essayer des variantes.

Informations générales

Authentification

Un identifiant et un mot de passe vous ont été donnés, ils sont de la forme « userX » / « userX ». Ce couple identifiant/mot de passe est le même pour toutes les identifications de la plateforme, que ce soit sur le site internet, pour accéder à votre machine virtuelle, pour interagir avec vos expérimentations ou interroger la base de données contenant les mesures effectuées lors de vos expérimentations.

Dans toute la procédure de ce TP, nous utiliserons le login fictif « userX », n'oubliez pas de le remplacer par celui vous correspondant !!!

Expérimentation

Une expérimentation consiste en :

- un déploiement d'un ou plusieurs programmes (*firmware*) sur un ou plusieurs nœuds-capteurs, pour une certaine durée de temps ;
- une mesure régulière éventuelle sur tous les nœuds-capteurs déployés de la consommation électrique, de l'environnement physique (température, luminosité), et de l'activité radio (RSSI), avec une définition par capteur possible ;
- une interaction possible continue avec l'ensemble des capteurs, via leurs ports série;
- un ajustement possible de l'expérimentation en cours, via un client en ligne de commande, permettant de couper/allumer l'alimentation des capteurs, de leur charger un nouveau programme à tout moment, de reconfigurer les mesures régulières, etc...

Sont prévus deux moyens d'effectuer une expérimentation : via l'interface web, ou en ligne de commande dans une machine virtuelle, comme décris ci dessous.

Site Internet

Le site internet de la plateforme Senslab de Grenoble est hébergé à l'adresse suivante :

http://grenoble.senslab.info/SenslabPortalPolytech

Vous pouvez sur cette page demander la création d'un compte, puis une fois celui ci validé préparer une expérimentation, réserver des nœuds-capteurs, et démarrer cette expérimentation. Vous pouvez également consulter vos expérimentations passées pour en récupérer des informations.

Ce site internet n'est au moment du Tutoriel Polytech pas encore complètement fonctionnel. Vous pouvez utiliser la paire identifiant/mot de passe qui vous a été allouée pour vous connecter sur le site web et observer un aperçu de ses fonctionnalités.

Machine Virtuelle

Lorsqu'une demande de création de compte est validée, une machine virtuelle est créée pour chaque utilisateur. Son nom est l'identifiant de l'utilisateur préfixé par '*vm*'. C'est via celle-ci que toutes les interactions avec une expérimentation en cours peuvent être effectuées. Il est de plus possible de démarrer une expérimentation en ligne de commande. Il est à noter que chaque utilisateur peut avoir un accès root à sa machine virtuelle, lui permettant d'installer tous les paquets qu'il souhaite, dans la limite de la place qui lui est allouée.

La connexion à la machine virtuelle, se fait via SSH, en rebondissant d'abord sur une passerelle SSH hébergée sur la machine serveur de la plateforme Senslab de Grenoble. Voici la procédure de connexion:

1. A partir de n'importe quelle ordinateur connecté à internet, se connecter en SSH sur la passerelle de serveur, grenoble.senslab.info, en utilisant votre identifiant (*userX*) :
   $ ssh userX@grenoble.senslab.info
   userX@grenoble.senslab.info's password: [entrez ici votre mot de passe (*userX*)]
   userX@srvssh:~$
2. Il est maintenant possible de se connecter sur la machine virtuelle, dont le nom est '*vm*' suivi de l'identifiant :
   userX@srvssh:~$ ssh vmuserX
   [...]
   userX@vmuserX:~$
3. Si par exemple vous voulez un accès root, (pour installer de nouveaux paquets) il faut se connecter en ssh avec le login root. Cela est possible depuis la passerelle ssh, ou depuis la machine virtuelle directement :
   userX@vmuserX:~$ ssh root@localhost
   [...]
   vmuserX:~#

Nous voila donc connectés sur la machine virtuelle, où un certain nombre d'outils sont déjà installés. En particulier la chaine de développement *mspgcc* qui permet de compiler des programmes pour msp430, le microcontrôleur des nœuds-capteurs de Senslab :

userX@vmuserX:~$ msp430-<TAB><TAB>
msp430-addr2line msp430-g++ msp430-jtag msp430-ranlib
msp430-ar msp430-gcc msp430-jtag-unwrapped msp430-readelf
msp430-as msp430-gccbug msp430-ld msp430-run
msp430-bsl msp430-gcov msp430-miniterm msp430-size
msp430-c++ msp430-gdb msp430-nm msp430-strings
msp430-c++filt msp430-gdbproxy msp430-objcopy msp430-strip
msp430-cpp msp430-gdbtui msp430-objdump
msp430-dco msp430-gprof msp430-ram-usage

Se trouvent également deux commandes senslab-*, qui vont permettre de déployer une expérimentation et d'interagir avec celle ci :

userX@vmuserX:~$ senslab- <TAB>
senslab-cli senslab-sub

La commande ***senslab-sub*** permet de lancer et de stopper une expérimentation. Elle se comporte comme un shell interactif si elle est appelée sans argument, et elle permet d'exécuter une sous-commande directement si cette sous commande et ses options sont passés en arguments :
userX@vmuserX:~$ senslab-sub
Senslab User Experiment Submitter CLI
>>

Il y a trois sous commandes d'intérêt : *config, submit, stop*. La première permet de configurer le serveur, l'identifiant et le mot de passe pour se connecter au serveur. Si l'on appelle cette fonction sans arguments, les informations enregistrées sont affichées :
>> **config**
server=experiment
user=userX
password=***

Il faut ensuite renseigner le mot de passe associé à votre identifiant. Exécutez à nouveau la sous commande config, avec l'option --user qui va ensuite vous demandez votre mot passe.
>> **config --user userX**
Enter password for user `userX`:
server=experiment
user=userX
password=****
>>

La configuration est stockée dans un fichier dans la machine virtuelle, le mot de passe ne sera donc plus demandé. Une fois cette configuration effectuée, les autres sous commandes vont pouvoir être appelées.
La sous commande submit permet la demande de soumission d'une expérimentation, voici sa structure :
>> **help submit**
Submit an experiment
usage: submit <filename>
<filename> the experiment archive to upload, it should be a .tar.gz

On, observe ici qu'il faut passer en argument un fichier, et plus exactement une archive tar-gz contenant les fichiers nécessaires à réaliser une expérimentation. La structure de cette archive est décrite ci dessous.

La sous commande *stop* permet d'arrêter l'expérimentation en cours :
>> **help stop**
Stop an experiment
usage: stop

Remarque: Dans le cadre du tutoriel, une seule expérimentation peut être lancée par un utilisateur. Il faudra donc toujours arrêter une expérimentation avant d'en déployer une autre.

La commande ***senslab-cli*** permet d'interagir avec les nœuds-capteur lorsqu'une expérimentation a lieu. Elle fonctionne de la même manière que ***senslab-sub***, à savoir en tant que shell interactif si exécutée sans argument. Elle utilise le même fichier de configuration que senslab-sub, donc il ne sera pas nécessaire de renseigner le mot de passe à nouveau. Par contre, si l'on exécute la sous commande *config* on se rend compte qu'il y a un nouveau champ :

>> **config**
server=experiment
user=userX
password=****
experiment=7

En effet, un champ 'experiment' est maintenant présent. Cela correspond au numéro (l'identifiant) de l'expérimentation en cours. Ce numéro est obtenu lorsqu'on exécute la sous commande submit de la commande senslab-sub. Il faudra donc mettre à jour cette valeur à chaque nouvelle expérimentation:

>> **config --experiment 39**
server=experiment
user=userX
password=****
experiment=39

La syntaxe des autres sous commandes, qui permettent l'interaction avec les nœuds-capteurs est majoritairement la même, et il est toujours possible d'appeler la sous commande *help* suivie du nom n'une autre sous commande pour en apprendre sa syntaxe :

>> **help start**
Usage: cli.py start [options] nodes

Options:
-h, --help show this help message and exit
--battery power on the open nodes with battery only

On retrouve donc de manière quasi systématique le nom de la sous commande (ici start), puis une série d'options commençant pas '-' ou '--', puis la liste des nœuds-capteurs à qui appliquer cette commande. Cette liste se décrit par les numéros des nœuds séparés par des espaces, ainsi que par des ensembles de nœuds sous la forme 10-20 pour les nœuds de 10 à 20 inclus.
Par exemple >> **start 1 3 5 10-20**

Voici une description des principales commandes disponibles:

- **start** : alimenter un ensemble de nœuds, soit par tension régulée, soit par batterie ;
- **stop** : coupe l'alimentation d'un ensemble de nœuds (coupe la tension et la batterie) ;
- **reset** : effectue un RESET des microcontrôleur d'un ensemble de nœuds ;
- **update** : permet d'uploader un firmware sur le serveur, puis de le charger sur un ensemble de

nœuds ;
- **measure** : effectue une mesure instantanée de différentes grandeurs sur un ensemble de nœuds ;
- **powerpoll** : configure la mesure régulière (polling) de la consommation d'un ensemble de nœuds ;
- **sensorpoll** : idem, sur les capteurs température et luminosité ;
- **radiopoll** : idem sur le polling radio RSSI ;

Comme mentionné ci dessus, il est possible de communiquer directement avec les nœuds-capteurs via leurs ports série. Ces derniers sont capturés et transférés par des sockets IP. En se connectant au serveur de la plateforme avec de simple sockets TCP, il est ainsi possible de communiquer avec les nœuds. Pour ce faire, le serveur écoute sur les port 30000 + (id du nœud), et il suffit de s'y connecter. Le nœud '1' d'un utilisateur est transféré par le port 30001. Un outil simple permettant de s'y connecter est *netcat* disponible sur les systèmes UNIX :
$ nc experiment 30001

Les données émises par le nœud-capteur 1 seront affichées ici, et si vous tapez des données, elles seront envoyées au capteur dès que vous pressez la touche « entrée ».

Description d'une expérimentation

Comme nous l'avons vu ci dessus, pour soumettre une experimentation via le client en ligne de commande senslab-sub, il faut fournir en paramètre une archive. Cette archive contient un fichier de description de l'expérimentation sous forme XML et qui s'appelle systématiquement 'experiment.xml', et l'ensemble des fichiers de firmware qui sont utilisées dans l'expérimentation, au format Intel HEX.

Voici un exemple de fichier experiment.xml décrivant une expérimentation :

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<experiment login="userX" ip="" id="1"
xmlns="http://www.senslab.info"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://www.senslab.info experiment.xsd">
     <profiles>
          <profile id="batt_conso">
            <cc1100/>
               <powersupply>battery</powersupply>
            <measure>
                <power values="voltage,current">100</power>
            </measure>
          </profile>
    </profiles>
     <nodes>
          <node userId="1" senslabId="1" profile="batt_conso"
firmware="ex1.hex" />
```

```
    </nodes>
</experiment>
```

Les paramètres importants sont les suivants :

- dans la balise **experiment :**
  - l'attribut **login** doit correspondre à l'identifiant utilisateur, ici 'userX';
  - l'attribut **ip** est inutilisé, on peut donc laisser un champ vide;
  - l'attribut **id** est aussi inutilisé, il faut cependant y laisser un numéro, laissez donc '1'.
- entre les balises **profiles** sont listés tous les profils de configuration utilisés dans l'expérimentation. Il peut y avoir entre 1 et N profils définis si N est le nombre de nœud-capteurs réservés. Ensuite, les nœuds capteurs réservés seront associés à un profil parmi ceux définis. Un profil décrit les éléments suivants:
  - le nom du profil doit être inscrit dans l'attribut **id** de la balise **profile** ;
  - les paramètres de configuration de la radio du nœud de contrôle en fonction de son type. A Grenoble les radios sont des CC1100, il faut donc ajouter la balise **<cc1100/>**. Cette balise peut avoir différents attributs indiquant la fréquence, la bande passante, la modulation à utiliser;
  - le mode d'alimentation du nœud-capteur dans la balise **powersupply**, la valeur doit être **dc** si le nœuds est alimenté par une source de tension externe, ou **battery** s'il doit être alimenté par batterie;
  - les mesures périodiques (polling) à effectuer sur le capteur, définies entre les balises **measure**, sont classées en trois catégories:
    - **power**, dont l'attribut **value** peut contenir un sous ensemble de voltage/current/power, et dont la valeur est la périod de polling en millisecondes, permet de mesurer la consommation électrique du nœud capteur;
    - **rssi**, pour mesurer l'activité radio à proximité du nœud capteur;
    - **sensor**, dont l'attribut **value** peut contenir un sous ensemble de temperature/luminosity, mesure la température et la luminosité à une période fixe;
- finalement entre les balises **nodes** se trouve la liste des nœuds capteurs réservés pour l'expérimentation. Chaque paire de balise **node** doit contenir les attributs suivants :
  - **userId** est le numéro du nœud du point de vue de l'utilisateur. C'est avec ce numéro qu'on peut ensuite interagir avec le nœud-capteur via le client en ligne de commande, et c'est sous ce même numéro que sont stockées les informations collectées dans la base de données. Cela permet sur un déploiement contenant 5 noeuds-capteurs, de travailler avec des noeuds numérotés 1, 2, 3, 4 et 5 indépendamment des numéros physiques (voir senslabId ci après) des noeuds réservés (qui peuvent être pseudo aléatoires) ;
  - **senslabId** est le numéro physique du nœud, qui permet à la plateforme de les identifier. Ce numéro a du être alloué à l'utilisateur, suite à une réservation ;
  - **profile** est le nom du profil défini ci dessus qui doit être utilisé pour la configuration du nœud de contrôle ;
  - **firmware** est le nom du fichier de firmware a utiliser pour programmer le nœud-capteur. Ce fichier doit également se trouver dans l'archive.

REMARQUE: dans le cadre du tutoriel, le système de réservation n'étant pas opérationnel, chaque utilisateur a accès à un cluster de 16 nœuds capteurs correspondant à son numéro d'utilisateur. Par exemple l'utilisateur *user8* a accès uniquement aux nœuds du cluster 8. La liste des numéros des nœuds-capteurs par cluster se trouve en annexe.

Dans cet exemple, l'utilisateur userX peut accéder depuis la machine vmuserX à cette expérimentation, qui comprend un profil nommé batt_conso. Ce profile définit une alimentation par batterie, et une mesure de la tension et du courant consommé toutes les 100ms. Ensuite un nœud est déclaré, repéré pour l'utilisateur par le numéro 1 et correspondant au nœud 1 de la plateforme. Ce nœud utilise pour profil le seul déclaré : batt_conso, et comme firmware un fichier nommé ex1.hex.

Partie Pratique

Exemple 1 : LED clignotante, et mesure de consommation

Dans ce premier exemple, vous allez compiler une application simple pour le nœud capteur WSN430, qui fait clignoter une LED présente sur la carte. Vous allez ensuite modifier un fichier experiment.xml pour qu'il décrive une expérimentation contenant un seul nœud appartenant à votre cluster, et configurer une mesure périodique de sa consommation électrique pour ensuite observer la courbe de consommation.

Voici la procédure à suivre:

- Connectez vous à votre machine virtuelle comme décrit ci-dessus.
  - **$ ssh userX@grenoble.senslab.info**
  - **userX@srvssh:~$ ssh vmuserX**
- Récupérer l'archive suivante, contenant tous les fichiers nécessaires au TP :
  - **userX@vmuserX:~$ wget http://www.senstools.info/packages/tp/wsn430.tar.gz**
- L'extraire:
  - **userX@vmuserX:~$ tar xzf wsn430.tar.gz**
- Vous disposez maintenant d'un dossier 'wsn430' dans lequel se trouvent un dossier 'drivers' contenant tous les drivers relatifs à la carte WSN430, utiles donc pour programmer des applications simplement, un dossier 'mac' qui contient de simples protocoles MAC de communication radio, qui vont être utiles pour le troisième exemple de ce TP, et les dossiers des exemples.
- Rendez vous dans le dossier du premier exemple :
  - **userX@vmuserX:~$ cd wsn430/ex1**
- Afficher le fichier source pour voir ce qu'il contient :
  - **userX@vmuserX:~/wsn430/ex1$ cat main.c**
- Compilez le grâce à make :
  - **userX@vmuserX:~/wsn430/ex1$ make**
- Le firmware ainsi compilé s'appelle ex1.hex, il faut maintenant éditer le fichier experiment.xml, grâce à votre éditeur de texte favori.
  - **userX@vmuserX:~/wsn430/ex1$ nano experiment.xml**
- Modifiez les champs importants du fichier, comme nous l'avons vu ci dessus:
  - mettez votre identifiant dans l'attribut login (userX);
  - changez le champ senslabId de la ligne <node>, en mettant un nœud qui appartient à votre cluster ;
  - vérifiez les autres champs ;
  - quittez nano (Ctrl-O, Ctrl-X).
- Créez une archive contenant le fichier experiment.xml et le firmware ex1.hex
  - **userX@vmuserX:~/wsn430/ex1$ tar czf exemple1.tar.gz experiment.xml ex1.hex**
- Exécutez la commande **senslab-sub** qui affiche l'invite de commande permettant de déployer

une expérimentation :
  - **userX@vmuserX:~/wsn430/ex1$ senslab-sub**
- Exécutez la sous commande config avec l'option --user pour configurer le client avec votre identifiant, et qu'il demande le mot de passe correct :
  - >> **config --user userX**
    Enter password for user `userX`:
- Déployez l'expérimentation grâce à l'archive que vous venez de faire :
  - >> **submit exemple1.tar.gz**
    Experiment started with id: 8
- Notez le numéro de l'expérimentation généré (ici 8), patientez quelques temps (une dizaine de secondes au moins), puis arrêtez l'expérimentation en cours:
  - >> **stop**
    Experiment stopped
- Quittez client
  - >> **quit**
- Dans un navigateur depuis votre ordinateur, visitez la page suivante :
  - **https://grenoble.senslab.info/phppgadmin/**
  - cliquez sur « SensLAB Grenoble PostgreSQL »
  - entrez votre identifiant/mot de passe
  - parcourez votre base et les schémas sous-jacents, vous devriez trouver un schéma nommé exp<expId> avec comme expId le numéro qui s'est affiché après avoir appelé la commande submit du client senslab-sub
  - dans cet schéma se trouvent autant de tables que de mesures qui peuvent être effectuées par la plateforme. Vous pouvez parcourir les tables data_voltage et data_current, qui devraient être les seules à avoir des données, puisque ce sont les seules à être mentionnées dans le fichier experiment.xml qui a été soumis.
- Nous allons maintenant extraire ces données de la base de données, et faire un graphe de la consommation sur une durée courte.
  - dans la machine virtuelle, dans le dossier ~/wsn430/ex1/ executez le script plotgen :
    - **userX@vmuserX:~/wsn430/ex1$ ./plotgen.py**
      usage: ./plotgen.py <expNum> <nodeNum> [<starttime> <duration>]
  - il faut donc indiquer le numéro de l'expérimentation, (ici 8), puis le numéro du nœud qui nous concerne, donc celui que nous avons mis dans le fichier experiment.xml (ici 1). il est possible d'afficher le graphe en commençant à un instant donné et pour une durée donnée. Essayez d'abord sur la totalité de l'expérimentation
    - **userX@vmuserX:~/wsn430/ex1$ ./plotgen.py 8 1**
  - Votre mot de passe est à nouveau demandé, pour interroger la base de données, puis un fichier 'cv_plot.png' est généré. Vous pouvez essayer de l'afficher si le Xforwarding est activé :
    - **userX@vmuserX:~/wsn430/ex1$ display cv_plot.png**
  - Sinon vous pouvez le télécharger sur votre ordinateur via scp, en tappant dans un terminal :
    - **$ scp userX@grenoble.senslab.info:wsn430/ex1/cv_plot.png .**

- Si le graphe n'est pas clair, à cause d'une durée trop longue, relancez le script plotgen dans la machine virtuelle avec les paramètres optionels, par exemple pour utiliser 10 secondes données à partir de la 5° seconde de l'expérimentation :
  - **userX@vmuserX:~/wsn430/ex1$ ./plotgen.py 8 1 5 10**
- Répétez la procédure pour afficher le graphe, et vous devriez observer un beau créneau de courant, correspondant à la LED qui s'allume et qui s'éteind.

Question Bonus : modifiez le code source pour faire clignoter une autre LED, à un autre rythme, et répétez l'expérience pour obtenir une courbe de courant contenant plus de paliers. (Doc sur les drivers WSN430 sur: www.senstools.info/doxygen/)

Exemple 2 : communication via le port série

Ce second exemple va vous permettre de mettre en application la communication avec un nœud-capteur via son port série. Le code source capteur se trouve dans le dossier ~/wsn430/ex2 que vous avez du télécharger et extraire lors de l'exécution de l'exemple 1.

- Placez vous dans le dossier de l'exemple 2 :
  - **userX@vmuserX:~$ cd ~/wsn430/ex2**
- Étudiez le code qui s'exécutera sur le capteur, pour en comprendre le sens :
  - **userX@vmuserX:~/wsn430/ex2$ nano main.c**
- Compilez cet exemple grâce à la commande make :
  - **userX@vmuserX:~/wsn430/ex2$ make**
- Vous vous retrouvez maintenant avec un fichier compilé au format Intel HEX nommé 'ex2.hex'. Il faut a présent éditer le fichier experiment.xml afin de pouvoir créer une archive de description d'expérimentation, et la soumettre pour déployer cet exemple. Comme vous l'avez peut être remarqué il n'y a pas de fichier experiment.xml fourni avec cet exemple, il va donc falloir en écrire un nouveau. Ou plutôt récupérer celui de l'exemple 1, et le modifier pour cet exemple ! Copiez donc l'autre fichier dans ce répertoire :
  - **userX@vmuserX:~/wsn430/ex2$ cp ../ex1/experiment.xml .**
- Puis éditez le avec votre éditeur de texte favori :
  - **userX@vmuserX:~/wsn430/ex2$ nano experiment.xml**
- Normalement la configuration de identifiant et du nom de la machine virtuelle a déjà été faite dans le premier exemple. Vous devez maintenant éditer le profil correspondant à l'alimentation et aux mesures effectuées sur le noeur capteur que vous allez déployer. Renommez l"id' du profil par *no_measure*, changer la valeur entre les balises powersupply pour mettre *dc* ce qui indique une alimentation externe régulée, et supprimez la ligne entre les balises *measures*, car nous ne sommes pas cette fois intéressés pour faire des mesures de consommation (attention, il faut quand même laisser <measures</measures>).
- Changez ensuite la ligne concernant le seul nœud réservé, en modifiant si cela vous chante le *userId* et/ou le *senslabId* tant que ce dernier se trouve dans votre cluster de nœuds. Vous pouvez bien entendu laisser les numéros de l'exemple 1, et nous conseillons de toujours numéroter les nœuds par le champ *userId* en commençant par la valeur 1.
- Il faut ensuite mettre à jour la valeur du profil utilisé pour le nœud, donc mettre *no_measure*, et changer le nom du firmware à utiliser, en remplaçant *ex1.hex* par *ex2.hex* .
- Sauvegardez et quittez l'éditeur de texte (Ctrl-O, Entrée, Ctrl-X sous nano).
- Créez une archive pour le déploiement:
  - **userX@vmuserX:~/wsn430/ex2$ tar czf exemple2.tar.gz experiment.xml ex2.hex**

- Déployez l'expérimentation. (Il est possible d'exécuter une sous-commande d'un client en ligne de commande de Senslab sans entrer dans le shell, en tappant la sous-commande et ses arguments directement dans la ligne de commande) :
  - **userX@vmuserX:~/wsn430/ex2$ senslab-sub submit exemple2.tar.gz**
    Experiment started with id: 10
- Notez l'identifiant (id) de l'expérimentation, puis connectez vous au port série du capteur déployé. Pour ce faire, il faut se connecter en utilisant une socket TCP au serveur Senslab, ici 'experiment', sur le port numéro 30000 plus l'id du nœud auquel on souhaite se connecter. L'id correspond au *userId* précisé dans le fichier experiment.xml, nous avons dans cet exemple choisi le nœud 1, donc nous nous connectons sur le port 30001 :
  - **userX@vmuserX:~/wsn430/ex2$ nc experiment 30001**
- Rien ne se passe ! Car le nœud a déjà démarré avec le programme joint, et est en attente. Pour s'en assurer, nous allons le redémarrer. Pour faire cela, gardez le terminal communiquant avec le port série du nœud ouvert, et ouvrez un autre terminal. Dans ce dernier, connectez vous en SSH à la passerelle SSH, puis à votre machine virtuelle à nouveau :
  - **$ ssh userX@grenoble.senslab.info**
  - **userX@srvgwssh:~$ ssh vmuserX**
- Exécutez la commande senslab-cli lançant le shell permettant d'interagir avec les nœuds de son expérimentation :
  - **userX@vmuserX:~$ senslab-cli**
- Il est maintenant nécessaire de reconfigurer les paramètres de ce client, pour lui indiquer le numéro de l'expérimentation en cours :
  - **Senslab User CLI**
    >> config --experiment 10
    server=grenoble.senslab.info
    user=userX
    password=****
    experiment=10
- Vous avez maintenant la possibilité d'interagir avec les nœuds de votre expérimentation. Nous souhaitons ici redémarrer le nœud 1 (le seul déployé), ce qui se fait avec la sous commande *reset*, tappez donc ce qui suit :
  - **>> reset 1**
    Success: [1]
- La ligne suivante indique que la commande a été effectuée avec succès pour le nœud 1. Et dans l'autre terminal, exécutant la commande *nc*, qui jusqu'à présent n'affichait rien, devrait avoir affiché le texte suivant :
  - **userX@vmuserX:~/wsn430/ex2$ nc grenoble.senslab.info 30001**
    Senslab TP Ex2: UART
    Type command
    t: temperature measure
    l: luminosity measure
    cmd >
- Comme décrit par le code source du fichier *main.c*, le nœud capteur attend en entrée sur son

port série un caractère pour effectuer une mesure : 't' pour une mesure de température, 'l' pour une mesure de luminosité. Tappez soit l'un soit l'autre, puis validez par la touche Entrée :

- **cmd > t**
  Temperature measure: 28.500
  cmd >

- Et voilà, vous avez effectué votre première interaction bidirectionnelle avec un nœud capteur via l'interface Senslab ! Vous pouvez tester les deux commandes disponibles, puis interrompre la commande *nc* discuttant avec le nœud capteur en tappant Ctrl-C, puis arrêter l'expérimentation avec la commande *senslab-sub* :
  - **userX@vmuserX:~/wsn430/ex2$ senslab-sub stop**
    Experiment stopped

Question Bonus : Modifier le code source du nœud capteur pour ajouter une commande 'i' qui lorsqu'elle est exécutée affiche la valeur d'une variable valant 0 au démarrage, et s'incrémentant à chaque appel.

Exemple 3 : communication radio

Ce troisième et dernier exemple va vous permettre de tester les communications radio entre les nœuds-capteurs de votre déploiement, en déclenchant l'émission de messages radio via le port série des nœuds-capteurs, et en observant les réceptions de la même manière.

- Allez dans le dossier ~/wsn430/ex3 pour ce troisième exemple :
  - **userX@vmuserX:~/wsn430/ex2$ $ cd ~/wsn430/ex3**
- Éditez le fichier code source *main.c* avec votre éditeur favori :
  - **userX@vmuserX:~/wsn430/ex3$ nano main.c**
- Parcourez ce fichier pour en comprendre le fonctionnement. Il y a un point important à avoir en tête : tous les participants aux TP qui vont exécuter ce programme en même temps vont avoir les mêmes configuration radio et les mêmes types de messages échangés, et si tout le monde utilise ce programme tel quel, vous allez recevoir les messages radios des autres groupes, et vice-versa. Pour palier à ce problème potentiel, il va vous falloir utiliser un canal radio différent pour chaque utilisateur. Repérez dans le fichier ***main.c*** **l'appel de la fonction** *mac_init*. Le seul argument de cette fonction est le numéro du canal radio à utilisé, comme défini arbitrairement par la simple implémentation d'un protocole MAC radio de type CSMA. Il vous est donc suggéré de modifier cette ligne en remplaçant la valeur par défaut par le numéro de votre identifiant (*userX* devrait choisir le canal radio 1, et ainsi de suite) :
  - **...**
    mac_init(1);
    ...
- Enregistrez et quittez, avec les commandes Ctrl-O, Ctrl-X.
- Copiez le fichier *experiment.xml* de l'exemple 2 dans le dossier de l'exemple 3 :

- **userX@vmuserX:~/wsn430/ex3$ cp ../ex2/experiment.xml .**

- Éditez le :
  - **userX@vmuserX:~/wsn430/ex3$ nano experiment.xml**
- Nous allons garder le même profil de configuration à savoir le *no_measure*, que nous allons appliquer à plusieurs nœuds-capteurs cette fois ci. Dupliquez donc la ligne contenant la balise *<node>* de manière à en obtenir 4. Ceci peut être effectué avec **nano** en positionnant le curseur au début de la ligne en question, puis en tapant Ctrl-K, et en tapant Ctrl-U le nombre de fois qu'il faut.
- Adaptez ensuite les numéros des 4 lignes de descriptions de nœuds-capteurs, pour avoir les *userId* allant de 1 à 4, associés à 4 *senslabId* appartenant à votre cluster de nœuds (cd Annexe). Vérifiez que la valeur de profil correspond bien au seul profil défini (ici *no_measure*), et changez les valeurs de *firmware* par 'ex3.hex'.
- Quittez l'éditeur (sous nano avec Ctrl-O, Ctrl-X), puis compilez l'application avec make :
  - **userX@vmuserX:~/wsn430/ex3$ make**
- Créez l'archive avec le fichier de description d'expérimentation et le fichier de firmware :
  - **userX@vmuserX:~/wsn430/ex3$ tar czf exemple3.tar.gz experiment.xml ex3.hex**
- Démarrez l'expérimentation avec la commande senslab-sub :
  - **userX@vmuserX:~/wsn430/ex3$ senslab-sub submit exemple3.tar.gz**
    Experiment started with id: 16
- Connectez vous avec *nc* au port série du nœud-capteur 1 :
  - **userX@vmuserX:~/wsn430/ex3$ nc experiment 30001**
- Ouvrez 3 autres terminaux, dans chacun d'entre eux connectez vous en SSH sur la passerelle SSH puis sur votre machine virtuelle, et connectez vous au port série des nœuds-capteurs 2, 3 et 4 de la même manière. Ici encore, rien ne devrait apparaitre. Ouvrez donc un cinquième terminal, connectez vous en SSH à votre machine virtuelle, et exécutez la commande senslab-cli pour interagir avec les nœuds :
  - **userX@vmuserX:~$ senslab-cli**
- Exécutez la sous commande config pour mettre à jour le numéro d'expérimentation :
  - **>> config --experimentation 16**
- Redémarrez tous les nœuds-capteurs de votre déploiement. Au lieu de spécifier la liste des nœuds (1 2 3 4), il est possible d'utiliser le mot clé 'all' pour indiquer tous les nœuds du déploiement :
  - **>> reset all**
    Success: [1, 2, 3, 4]
- La liste des nœuds pour lesquels la commande a été effectuée avec succès est affichée, donc ici

les 4. Vous devriez voir apparaître dans les quatre terminaux connectés aux quatre ports série le message de démarrage des nœuds.

- Vous pouvez donc maintenant entrer la commande 's' dans les terminaux, et voir les messages être reçus par les autres capteurs.
- Stoppez l'expérimentation avec la commande senslab-sub une fois les tests effectués.
    - **userX@vmuserX:~$ senlsab-stop**
      Experiment stopped
- Si vous aviez laissé les terminaux ouvert communiquant avec les nœuds capteurs, vous avez pu observer qu'ils ont redémarré après avoir lancé la commande stop, puis que les connexions TCP avec le serveur se sont fermées automatiquement, c'est la procédure de fin d'expérimentation !

Question Bonus:

Faire un Token Ring.

ANNEXE: Répartition des nœuds par cluster

Voici la liste des nœuds autorisés par cluster, leur regroupement ayant été fait de façon géographique, à savoir les nœuds d'un cluster sont garantis d'être au plus proche les uns des autres :

cluster 1 : { 1, 2, 3, 4, 12, 13, 14, 15, 16, 26, 27, 28, 29, 30, 40, 41 }

cluster 2 : { 5, 6, 7, 8, 17, 18, 19, 20, 31, 32, 33, 34, 35, 42, 43, 44 }

cluster 3 : { 9, 10, 11, 21, 22, 23, 24, 25, 36, 37, 38, 39, 45, 46, 71, 72 }

cluster 4 : { 57, 58, 59, 60, 68, 69, 70, 81, 82, 83, 84, 85, 95, 96, 97, 98 }

cluster 5 : { 48, 49, 50, 51, 52, 61, 62, 63, 64, 74, 75, 76, 77, 89, 90, 91 }

cluster 6 : { 53, 54, 55, 56, 65, 66, 67, 73, 78, 79, 80, 92, 93, 94, 100, 102 }

cluster 7 : { 104, 105, 106, 107, 108, 111, 112, 113, 114, 117, 118, 119, 120, 122, 123, 124 }

cluster 8 : { 47, 86, 87, 88, 99, 101, 103, 109, 110, 115, 116, 121, 125, 126, 128, 256 }

cluster 9 : { 127, 129, 146, 160, 172, 173, 186, 187, 204, 205, 206, 207, 208, 209, 210, 211 }

cluster 10 : { 130, 131, 132, 133, 147, 148, 149, 161, 162, 163, 164, 174, 175, 188, 189, 190 }

cluster 11 : { 182, 183, 184, 185, 197, 199, 200, 201, 202, 203, 212, 213, 214, 215, 216, 217 }

cluster 12 : { 139, 140, 141, 142, 143, 145, 154, 155, 156, 157, 158, 159, 170, 171, 180, 181 }

cluster 13 : { 134, 135, 136, 137, 138, 144, 150, 151, 152, 153, 168, 169, 176, 177, 178, 179 }

cluster 14 : { 165, 166, 167, 191, 192, 193, 194, 195, 196, 198, 218, 222, 223, 227, 231, 234 }

cluster 15 : { 219, 220, 221, 224, 225, 226, 228, 229, 230, 232, 233, 235, 236, 237, 238, 241 }

cluster 16 : { 239, 240, 242, 243, 244, 245, 246, 247, 248, 249, 250, 252, 252, 253, 254, 255 }