Mise au point d'un réseau Xbee de mesure de températures à l'aide de Xbee S2C

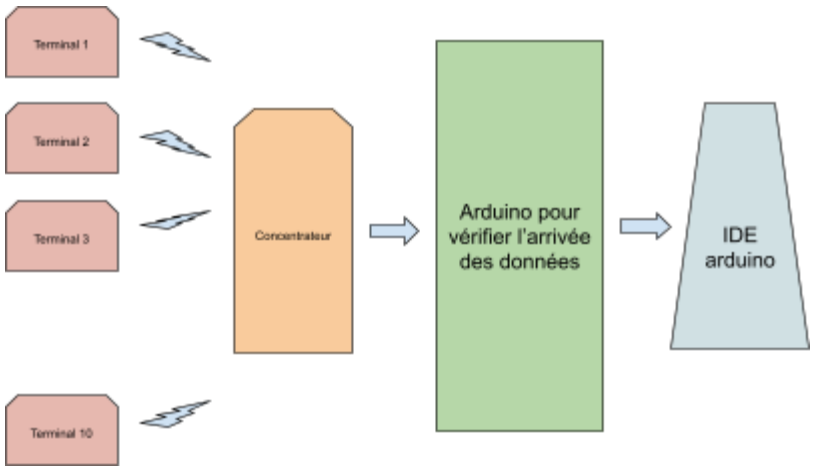
But :
vérifier le fonctionnement d'un ensemble de 10 Xbee émettant chacun une donnée analogique, vers un Xbee concentrateur.

Matériel utilisé :

- Xbee S2C (11)
- TMP36GZ (10) branché sur D1
- Arduino Micro

Architecture retenue :

10 terminaux composés chacun d'un Xbee S2C et d'une sonde de tempérture analogique (TMP36) envoient régulièrement la température mesurée à un Xbee concentrateur. La vérification de l'arrivée des informations se fait par l'intermédiaire d'un arduino, relié au concentrateur. Les informations transmises sont affichées dans la console de l'IDE Arduino.



Pour les essais, les terminaux sont mis en sommeil pendant 20 secondes puis se réveillent. Ils envoient alors une information analogique correspondant à une température mesurée sur un capteur TMP36GZ puis se rendorment.

Le concentrateur reçoit tous les messages et les transmet instantanément à l'arduino par liaison UART. La vitesse des échanges se fait à 9600 bauds.

Un petit programme Arduino place les données dans un tableau. Chaque case du tableau correspond à un terminal.

Toutes les 20 secondes, l'arduino affiche le contenu du tableau dans la console de l'IDE Arduino puis efface tout le contenu du tableau.

Cette procédure permet de vérifier qu'aucune donnée n'est perdue (collisions éventuelles entre émissions de 2 ou plusieurs terminaux en même temps).

Configuration des Xbee (terminaux et concentateur)

Cette configuration est réalisée avec le logiciel XCTU. Avant toute configuration, les Xbee ont été mis à jour avec le logiciel suivant :

- Product family : XB24C
- Function Set : Digimesh 2,4TH
- Firmware version : 9002 (Newest)

Puis les paramètres par défaut ont été chargés et écrits dans le Xbee.

Configuration des terminaux

Une fois la mise à jour réalisée, seuls les paramètres suivants ont été modifiés et enregistrés dans les terminaux :

- Dans "Networking and Security
  - ID : [ID_DU_RESEAU] [1]
- Dans "Addressing" :
  - DH et DL : adresse du concentrateur [2]
  - NI :"Terminal_xx"[3]
- Dans "Serial Interfacing"
  - AP : API Mode Without Escapes [1]
- Dans "I/O Settings" :
  - D0 : Disabled[0] (ADC[2]si deuxième capteur branché)
  - D1 : ADC[2]
- Dans "I/O Sampling" :
  - IR : 1388
- Dans "Sleep Command Options" :
  - SM : Async. Cyclic. Sleep [4]
  - SP : 7D0 (temps de mise en sommeil = 20 secondes)

Une fois modifiés, ces paramètres sont à transférer dans le Xbee (Bouton Write).

---

[1] La valeur ID_DU_RESEAU devra être la même pour tous les terminaux et le concentrateur.
[2] Cette adresse est composée des paramètres SH et SL du concentrateur qui doivent être saisis respectivement dans les paramètres DH et DL des terminaux.
[3] Ce nom est purement informel et permet d'identifier le Xbee lorsqu'il est connecté au logiciel XCTU. Dans ce cas, xx prend les valeurs 01 à 10 pour les 10 terminaux.

Configuration du concentrateur

Une fois la mise à jour réalisée, seuls les paramètres suivants ont été modifiés et enregistrés dans les terminaux :

- Dans “Networking and Security
  - ID : [ID_DU_RESEAU][4]
- Dans “Addressing” :
  - NI : “Concentrateur”[5]
- Dans “Serial Interfacing”
  - AP : API Mode Without Escapes [1]
- Dans “I/O Settings” :
  - D0 : Disabled[0]

Une fois modifiés, ces paramètres sont à transférer dans le Xbee.

Identification des modules Xbee

Désignation du module (ID)	SH	SL
Concentrateur	13 A2 00	DL du concentrateur
Terminal_01	13 A2 00	DL du terminal 01
Terminal_02	13 A2 00	DL du terminal 02
Terminal_03	13 A2 00	DL du terminal 03
Terminal_04	13 A2 00	DL du terminal 04
Terminal_05	13 A2 00	DL du terminal 05
Terminal_06	13 A2 00	DL du terminal 06
Terminal_07	13 A2 00	DL du terminal 07
Terminal_08	13 A2 00	DL du terminal 08
Terminal_09	13 A2 00	DL du terminal 09
Terminal_10	13 A2 00	DL du terminal 10

[4] La valeur ID_DU_RESEAU devra être la même pour tous les terminaux et le concentrateur.
[5] Ce nom est purement informel et permet d’identifier le Xbee lorsqu’il est connecté au logiciel XCTU.

Programme Arduino

Le programme de test utilisé est le suivant :

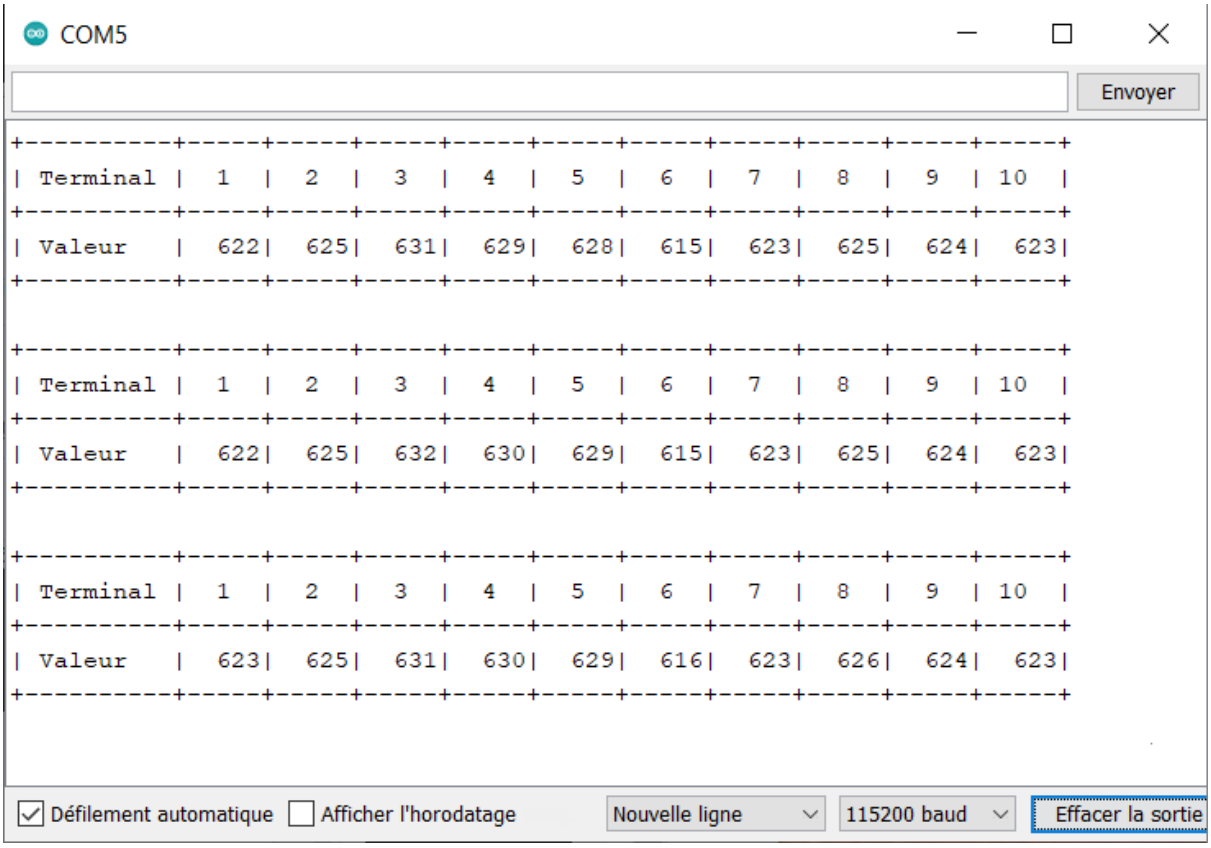
```
/*
*       V1 - reception des donnees brutes
*
*      Le programme attend une transmission sur UART(logiciel) (broche 8 en Rx)
*      Cette transmission venant d'un Xbee concentrateur
*
*      Elle peut relayer les informations venant de 10 Xbee (terminaux) mesurant
chacun une tension analogique (température)
*
*      La mesure correspondant à chaque terminal est stockée dans une des cases
d'un tableau
*      La correspondance "terminal" <=> "case du tableau" est assurée par
reconnaissance de l'adresse basse des Xbee
*      Ces adresses sont connues de l'arduino par les tableaux adress0 à
adress3.
*
*      Toutes les 20 secondes (période de repos des Xbee) on affiche le contenu
du tableau
*
*      Les valeurs affichées sont les valeurs lues par le CAN
*
*
*
*/
#include <SoftwareSerial.h>
SoftwareSerial SerialXbee(8, 9 ); // RX, TX

#define NB_TERM 10

int a=0;
int n=0;
int nb1, nb2;
int nbt = 0;
int fin=0;
unsigned int xx[30];
int resultat[NB_TERM];
int adr[NB_TERM];
```

```
//  Les 4 tableaux suivants contiennent les adresses DL des terminaux/
//  Celles ci sont stockées de la manière suivante :
//  Chaque tableau comporte 10 cases (1 par terminal)
//  le premier tableau contient le premier octet des 10 adresses DL des
//  terminaux (MSB)
//  Le deuxième tableau contient le deuxième octet des 10 adresses DL des
//  terminaux
//  etc. Exemple si l'adresse du terminal 1 est 0xAB45DE78
//  int adress0[] = {0xAB,......};
//  int adress1[] = {0x45,......};
//  int adress2[] = {0xDE,......};
//  int adress3[] = {0x78......};

int adress0[] = {......};
int adress1[] = {......};
int adress2[] = {......};
int adress3[] = {......};
unsigned long previous_millis=millis();
String s="";
/* ************************************************************************
SETUP */
void setup() {
  Serial.begin(115200);
  SerialXbee.begin(9600);
  while (!Serial) {
    delay(100);
  }
  Serial.println ("Ready");
}

/* ************************************************************************
LOOP */
void loop() {
  if (SerialXbee.available() > 0) {
    n=0;
    a=SerialXbee.read();
    if (a!=126) {
      SerialXbee.readString();
    } else {
      while (!SerialXbee.available() > 0) {
        delayMicroseconds(10);
      }
      nb1=SerialXbee.read();
      while (!SerialXbee.available() > 0) {
        delayMicroseconds(10);
      }
      nb2=SerialXbee.read();
      //
      nbt=nb1*256+nb2+1;  // +1 pour le check sum
      //Serial.print("Nb de caracteres a suivre : ");
      //Serial.println(nbt);
```

```
      for (int i=0; i<nbt; i++) {
        while (!SerialXbee.available() > 0) {
          delayMicroseconds(10);
        }
        xx[i]=SerialXbee.read();
      }
    }
    for (int i=0; i<NB_TERM; i++) {
      if (xx[5]==adress0[i] && xx[6]==adress1[i] && xx[7]==adress2[i] &&
xx[8]==adress3[i] ) {
        resultat[i]=xx[16]*256 + xx[17];
        break;
      }
    }

    if (millis() - previous_millis>20000) {
      previous_millis=millis();
      // affiche le tableau des résultats
      Serial.println
("+----------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+");
      Serial.println ("| Terminal |  1  |  2  |  3  |  4  |  5  |  6  |  7  |  8
|  9  | 10  |");
      Serial.println
("+----------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+");
      Serial.print   ("| Valeur   |");
      for (int i=0; i<NB_TERM; i++) {
        // formate la sortie sur 5 chr
        s="       "+String(resultat[i]);
        s=s.substring(s.length()-5);
        Serial.print(s+"|");
        //Serial.print(resultat[i],HEX);
        resultat[i]=0;
      }
      Serial.println("");
      Serial.println
("+----------+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+");
      Serial.println("");
    }
  }
}
```

Exemple d’affichage

L’affichage se présente sous la forme d’un tableau dune ligne à 10 colonnes. Chaque colonne représente la valeur d’un capteur (donc la valeur remontée par un terminal).



Ceci permet de s'assurer que les terminaux émettent bien et que les informations reçues par le concentrateur sont correctement ségréguées.

Vérification des mesures / conversion en températures

Fonctionnement du CAN des Xbee S2 : Digital and analog sampling using Xbee Radios
Pour des problèmes de compatibilité avec d'autres modules, la pin Vref peut être connectée à Vcc mais c'est inutile pour le S2. Le convertisseur possède sa propre tension de référence interne à 1,2V (Xbee/Xbee-PRO S2C Zigbee Modules page 150 I/O sampling).

Pour connaitre la mesure réalisée en mV par le CAN, il faut appliquer la formule suivante :

AD(mV) = (A/D reading * 1200mV) / 1023

Le TMP36GZ renvoie une tension proportionnelle à la température mesurée. La courbe à l'équation suivante :

V(mV) = 500 + 10 * T(°C)

Pour déduire la température en fonction de la tension mesurée on applique la formule suivante :

T(°C) = (V(mV) - 500) /10)

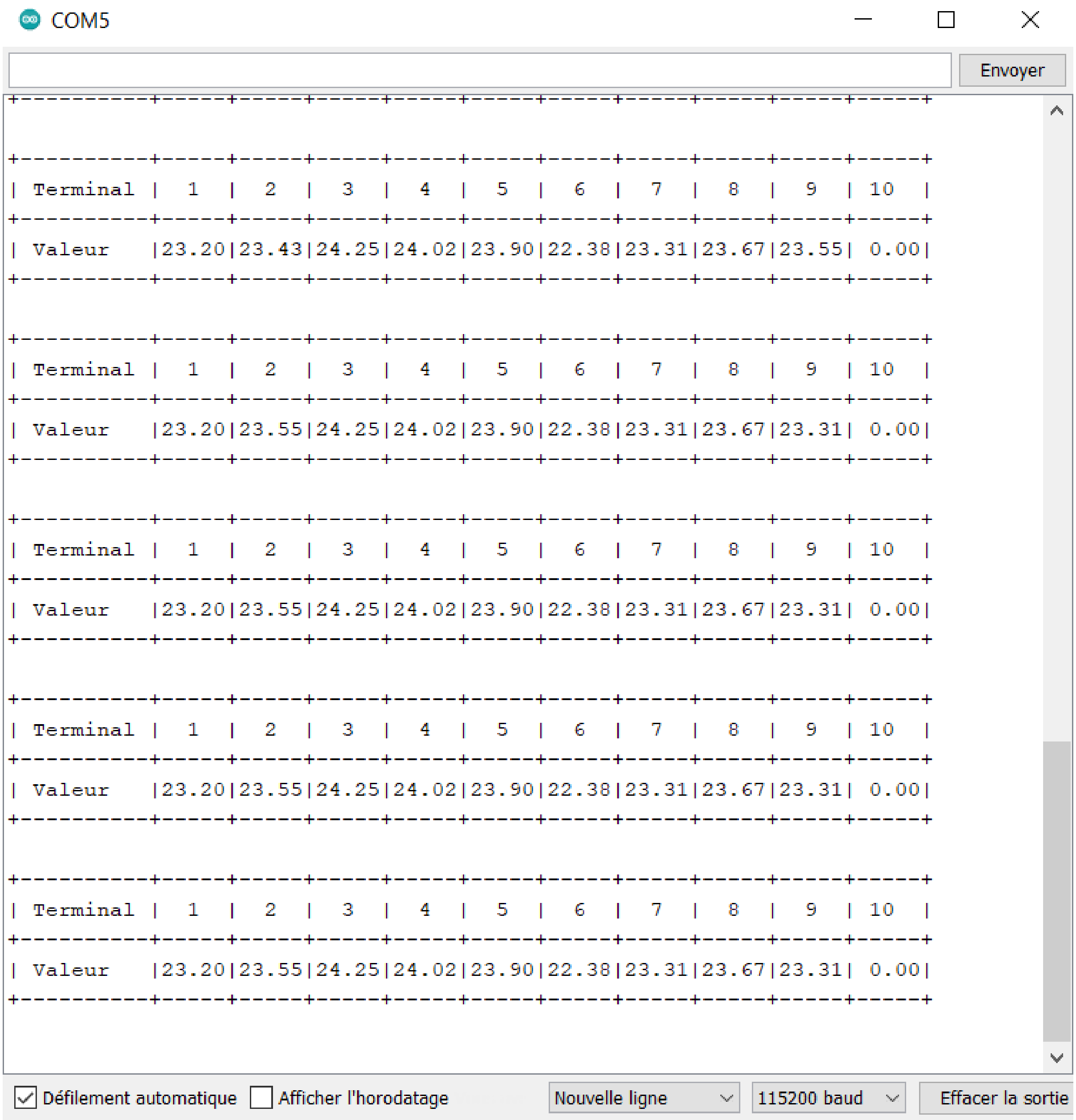
Ces conversions ont été appliquées dans le programme précédent. Ce qui donne le programme suivant :

```
/*
*
*
*      Le programme attend une transmission sur UART(logiciel) (broche 8
en Rx)
*      Cette transmission venant d'un Xbee concentrateur
*
*      Elle peut relayer les informations venant de 10 Xbee (terminaux)
mesurant chacun une tension analogique (température)
*
*      La mesure correspondant à chaque terminal est stockée dans une
des cases d'un tableau
*      La correspondance "terminal" <=> "case du tableau" est assurée
par reconnaissance de l'adresse basse des Xbee
*      Ces adresses sont connues de l'arduino par les tableaux adress0 à
adress3.
*
*      Toutes les 20 secondes (période de repos des Xbee) on affiche le
contenu du tableau
*
*      Les valeurs affichées sont les valeurs lues par le CAN
*
*
*
*/
#include <SoftwareSerial.h>
SoftwareSerial SerialXbee(8, 9 ); // RX, TX

#define NB_TERM 10

int a=0;
int n=0;
int nb1, nb2;
int nbt = 0;
int fin=0;
unsigned int xx[30];
float resultat[NB_TERM];
int adr[NB_TERM];
int adress0[] = {......};
int adress1[] = {......};
int adress2[] = {......};
int adress3[] = {......};

unsigned long previous_millis=millis();
float mesure = 0; // lecture du CAN
float tension = 0; // conversion en tension
```

```
float temper = 0; // convesion en température
String s="";
/*
**********************************************************************
**** SETUP */
void setup() {
  Serial.begin(115200);
  SerialXbee.begin(9600);
  while (!Serial) {
    delay(100);
  }
  Serial.println ("Ready");
}

/*
**********************************************************************
**** LOOP */
void loop() {
  if (SerialXbee.available() > 0) {
    n=0;
    a=SerialXbee.read();
    if (a!=126) {
      SerialXbee.readString();
    } else {
      while (!SerialXbee.available() > 0) {
        delayMicroseconds(10);
      }
      nb1=SerialXbee.read();
      while (!SerialXbee.available() > 0) {
        delayMicroseconds(10);
      }
      nb2=SerialXbee.read();
      //
      nbt=nb1*256+nb2+1;  // +1 pour le check sum
      //Serial.print("Nb de caracteres a suivre : ");
      //Serial.println(nbt);
      for (int i=0; i<nbt; i++) {
        while (!SerialXbee.available() > 0) {
          delayMicroseconds(10);
        }
        xx[i]=SerialXbee.read();
      }
    }
    for (int i=0; i<NB_TERM; i++) {
      if (xx[5]==adress0[i] && xx[6]==adress1[i] && xx[7]==adress2[i] &&
xx[8]==adress3[i] ) {
```

```
        mesure = xx[16]*256 + xx[17];
        tension = (mesure * 1200) / 1023;
        temper = (tension - 500) / 10 ;
        resultat[i]=temper;
        break;
      }
    }

    if (millis() - previous_millis>20000) {
      previous_millis=millis();
      // affiche le tableau des résultats
      Serial.println
("+----------+-----+-----+-----+-----+-----+-----+-----+-----+-----+----
-+");
      Serial.println ("| Terminal |  1  |  2  |  3  |  4  |  5  |  6  |
7  |  8  |  9  | 10  |");
      Serial.println
("+----------+-----+-----+-----+-----+-----+-----+-----+-----+-----+----
-+");
      Serial.print   ("| Valeur   |");
      for (int i=0; i<NB_TERM; i++) {
        // formate la sortie sur 5 chr
        s="       "+String(resultat[i]);
        s=s.substring(s.length()-5);
        Serial.print(s+"|");
        //Serial.print(resultat[i],HEX);
        resultat[i]=0;
      }
      Serial.println("");
      Serial.println
("+----------+-----+-----+-----+-----+-----+-----+-----+-----+-----+----
-+");
      Serial.println("");
    }
  }
}
```

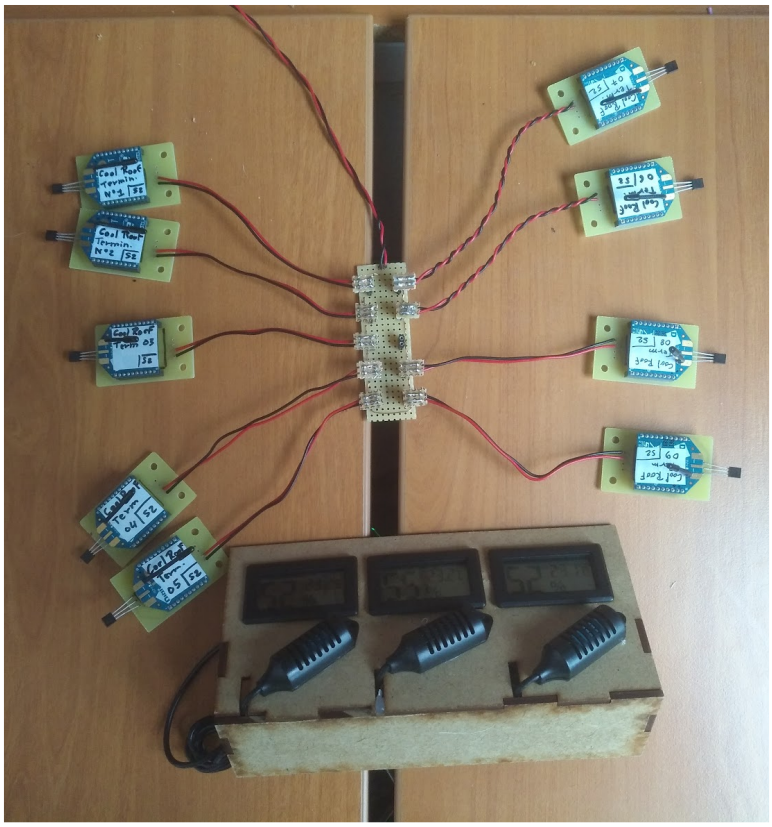
Et quelques résultats :




```
+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+

+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+
| Terminal |  1  |  2  |  3  |  4  |  5  |  6  |  7  |  8  |  9  | 10  |
+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+
| Valeur   |23.20|23.43|24.25|24.02|23.90|22.38|23.31|23.67|23.55| 0.00|
+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+

+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+
| Terminal |  1  |  2  |  3  |  4  |  5  |  6  |  7  |  8  |  9  | 10  |
+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+
| Valeur   |23.20|23.55|24.25|24.02|23.90|22.38|23.31|23.67|23.31| 0.00|
+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+

+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+
| Terminal |  1  |  2  |  3  |  4  |  5  |  6  |  7  |  8  |  9  | 10  |
+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+
| Valeur   |23.20|23.55|24.25|24.02|23.90|22.38|23.31|23.67|23.31| 0.00|
+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+

+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+
| Terminal |  1  |  2  |  3  |  4  |  5  |  6  |  7  |  8  |  9  | 10  |
+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+
| Valeur   |23.20|23.55|24.25|24.02|23.90|22.38|23.31|23.67|23.31| 0.00|
+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+

+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+
| Terminal |  1  |  2  |  3  |  4  |  5  |  6  |  7  |  8  |  9  | 10  |
+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+
| Valeur   |23.20|23.55|24.25|24.02|23.90|22.38|23.31|23.67|23.31| 0.00|
+----------+-----+-----+-----+-----+-----+-----+-----+-----+------+
```

Vérification des mesures

Pour cette vérification les 9 capteurs ont été reliés à la même source d'alimentation (une alimentation de PC capable de délivrer 34A sous 3,3v). Ils sont tous situés à proximité les uns des autres et un thermomètre extérieur composé de 3 sondes DHT22 donne une température de référence. (un capteur manque, le test ne sera réalisé que sur 9 capteurs).



Le tableau suivant donne les comparaisons de mesures dans diverses configurations.
En vert les température dans la game de tolérance de 1/10ème de degré

T° ref	Term01	Term02	Term03	Term04	Term05	Term06	Term07	Term08	Term09	Ecart max
23,1	23,2	23,55	24,25	24,02	23,90	22,38	23,55	23,67	23,31	1,87
22,5	21,44	21,79	22,84	22,84	21,91	21,09	22,14	22,14	21,44	1,75
6,8	7,6	7,48	8,53	8,18	7,6	6,3	8,06	7,36	6,30	2,23

La précision du capteur est donnée à +/- 2°.
Les mesures sont donc largement dans les spécifications du composant.