Annexe S.11

```c
//  mesures.c
//
//  Description : fonctions liées aux mesures de la sonde (température et humidté)
//  Auteur : Perret Mélissa
//  Création : 16/09/2024
//  Modifications : --

//  Version   : V1.0
/*-----------------------------------------------------------*/


#include "mesures.h"
#include "shtc3.h" // pour utiliser la sonde


//// Fonction EffectuerMesuresSonde (mesure témparture et humidité)
//// Description: mesure témparture et humidité
////              vérification dépassement des seuils
////              en cas de changement d'état, demande rafraichissement écran et l'envoie de la trame UART
//// Entrées: Pointeur:I2C_HandleTypeDef hi2c2, Pointeur:Mesures mesures, Tableau:DefinitionValeur valeursServeur
//// Sorties: Etat (état de l'alarme liée au dépassement des seuils)
Etat EffectuerMesuresSonde(I2C_HandleTypeDef* hi2c2, Mesures* mesures, DefinitionValeur valeursServeur[])
{
  if (shtc3_read_id(hi2c2)) // Si on détecte la sonde
  {
    // Effectuer les mesures
    if (shtc3_perform_measurements(hi2c2, &mesures->temperatureEntierActuelle, &mesures->humiditeEntierActuelle)) // Si la mesure est réussie
    {
      // Les résultats de la sonde sont transmis sous forme d'entier
      // Il s'agit en réalité de la valeur avec deux décimales, remis sous forme d'entier en étant multiplié par 100
      // On effectue l'opération inverse (division par 100) pour récupérer la valeur décimale
      mesures->temperatureActuelle = (float)mesures->temperatureEntierActuelle/100;
      mesures->humiditeActuelle = (float)mesures->humiditeEntierActuelle/100;

      // Pour éviter de rafraichisr l'écran E-paper trop souvent, on mesure l'écart entre la nouvelle valeur et la dernière valeur affiché
      // Si l'écart est supérieur à l'écart minimal on demandera le rafraichissement de l'écran
      // Les valeurs des écarts est transmises par le serveur

      // Ecart température
      double ecartTemperatureMinimal = valeursServeur[ECART_TEMPERATURE].valeur; // Ecart minimal de temperature pour effectuer un rafraichissement du E-paper
      float ecartTemperature = mesures->temperatureActuelle - mesures->temperatureAffichee; // Ecart entre la température mesurée et la température affichée
      if(ecartTemperature >= ecartTemperatureMinimal || ecartTemperature <= -ecartTemperatureMinimal) // Si l'écart est plus important que l'écart minimal
      {
        rafraichirEPaper = true; // Indiquer qu'un rafraichissement de l'écran est nécessaire
      }

      // Ecart humidité
      double ecartHumiditeMinimal= valeursServeur[ECART_HUMIDITE].valeur; // Ecart minimal d'humidité pour effectuer un rafraichissement du E-paper
      float ecartHumidite = mesures->humiditeActuelle - mesures->humiditeAffichee; // Ecart entre l'humidité mesurée et l'humidité affichée
      if(ecartHumidite >= ecartHumiditeMinimal || ecartHumidite <= -ecartHumiditeMinimal) // Si l'écart est plus important que l'écart minimal
```

```
51          {
52            rafraichirEPaper = true; // Indiquer qu'un rafraichissement de l'écran est nécessaire
53          }
54
55          // Vérification dépassement des seuils (pour définir le nouvel état de l'alarme dépassement des seuils)
56          bool depassementTemperature = mesures->temperatureActuelle < valeursServeur[SEUIL_TEMPERATURE_MIN].valeur || mesures->temperatureActuelle >
    valeursServeur[SEUIL_TEMPERATURE_MAX].valeur;
57          bool depassementHumidite = mesures->humiditeActuelle < valeursServeur[SEUIL_HUMIDITE_MIN].valeur || mesures->humiditeActuelle >
    valeursServeur[SEUIL_HUMIDITE_MAX].valeur;
58          Etat nouvelEtatSeuils = INDEFINI;
59          if(depassementTemperature || depassementHumidite)
60          {
61            nouvelEtatSeuils = ALARME; // Dépassement des seuils min/max (température et/ou humidité)
62          }
63          else
64          {
65            nouvelEtatSeuils  = OK; // Pas de dépassement des seuils min/max (température et humidité)
66          }
67
68          // Code de test qui change l'état de l'alarme dépassement des seuils à chaque réveil
69          // Permet de tester l'envoi des trames vers l'ESP
70          if(DEBUG_ALARME_SEUILS)
71          {
72            if(mesures->etatSeuils == ALARME) // Si l'état était ALARME on passe en OK
73            {
74              nouvelEtatSeuils = OK;
75            }
76            else
77            {
78              nouvelEtatSeuils = ALARME; // Si l'état n'était pas ALARME on passe en ALARME
79            }
80          }
81
82          return nouvelEtatSeuils;
83        }
84        else
85        {
86          return mesures->etatSeuils; // Mesure impossible, pas de changement d'état
87        }
88      }
89      else
90      {
91        return mesures->etatSeuils; // Mesure impossible, pas de changement d'état
92      }
93    }
94
```