Annexe S.7

```c
//  fonctionsADC.c
//
//  Description : fonctions liée à l'ADC
//  Auteur : Perret Mélissa
//  Création : 09/05/2024
//  Modifications : --

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

//----------------------CONFIGURATION UART HARMONY---------------------------//
/*
 * Activer le UART -> CubeMX
 * Régler le BaudRate @ 9600 (comme signalé dans le datasheet)
 * Activer 8 bits de data, aucune parité et 1 bit de stop
 * Pamamètrer l'over Yampling à 16
 * Activer le control de flux
*/

#include "fonctionsADC.h"

//// Fonction LectureValeurAdcBrute (lecture de registre du ADC)
//// Description: lire la valeur brute de l'adc en fonction du channel
//// Entrées: Pointeur:ADC_HandleTypeDef hadc, uint8_t channel (numéro de channel à utiliser pour l'ADC)
//// Sorties: uint16_t (valeur brute de l'ADC, valeur comprise entre 0 à 4095)
uint16_t LectureValeurAdcBrute(ADC_HandleTypeDef* hadc, uint8_t channel)
{
  HAL_ADC_Stop(hadc); // Arrêt de l'ADC

  HAL_StatusTypeDef calibrationStatus = HAL_ADCEx_Calibration_Start(hadc); // Nouvelle calibration par précaution

  hadc->Instance->CHSELR = 1 << channel;  // Sélection du bon channel

  HAL_StatusTypeDef adcStatus = HAL_ADC_Start(hadc); // Démarrage de l'ADC

  // Realise une mesure du registre de l'ADC, sort de la fonction après un certain temps si pas possible (TIMEOUT_LECTURE_ADC_MS)
  while (HAL_ADC_PollForConversion(hadc, TIMEOUT_LECTURE_ADC_MS) != HAL_OK) // Attente fin de conversion
  {
  }

  uint32_t mesure = 0; // Déclaration variable locale (valeur mesurée)
  mesure = HAL_ADC_GetValue(hadc); // Sauvegarde de la mesure

  HAL_ADC_Stop(hadc); // Arrêt de l'ADC

  return mesure; // Retourne la valeur brute mesurée
}

//// Fonction ConversionValeurAdcEnVolt: conversion d'une valeur brute ADC en une tension
//// Description: conversion d'une valeur brute ADC (0 à 4095) en une tension (en Volt)
//// Entrées: uint16_t valeurADC (valeure brute ADC à convertir),
////          uint32_t pontDiviseurResistanceHaute (valeur de la resistance en haut dans le pont diviseur de tension)
////          uint32_t pontDiviseurResistanceBasse (valeur de la resistance en bas dans le pont diviseur de tension)
```

```
54  //// Sorties: float (valeur brute de l'ADC convertie en valeure tension)
55  float ConversionValeurAdcEnVolt(uint16_t valeurADC, uint32_t pontDiviseurResistanceHaute, uint32_t pontDiviseurResistanceBasse)
56  {
57    // Rappels ponts diviseur de tension:
58    // Vout = (Vin * R2) / (R1 + R2)
59    // Vin = (Vout / R2) * (R1 + R2)
60
61    uint32_t R1 = pontDiviseurResistanceHaute; // Plus simple pour lire le code qui suit (formules)
62    uint32_t R2 = pontDiviseurResistanceBasse; // Plus simple pour lire le code qui suit (formules)
63
64    float valeurAdcEnVolt = valeurADC * VREF_ADC_V / VAL_ADC_MAX; // Correspond au Vout du pont diviseur de tension
65    return  valeurAdcEnVolt * (R1 + R2) / R2; // Prise en compte du pont diviseur de tension pour retrouver le Vin
66  }
67
```