

## Versuch 4 - Temperaturmessung

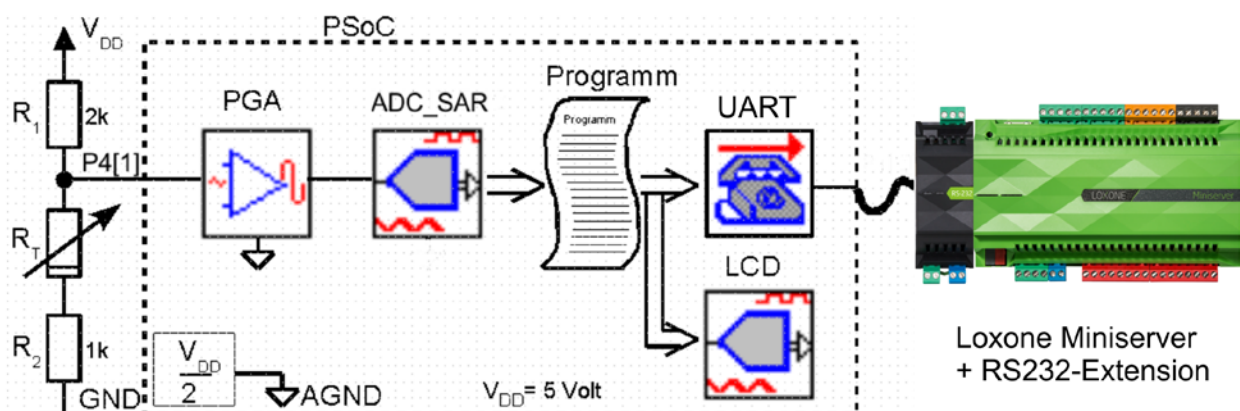
In diesem Versuch werden Sie die Umgebungstemperatur der Luft messen. Als Sensor steht Ihnen ein Thermistor zur Verfügung, der über einen Spannungsteiler eine temperaturabhängige, analoge Spannung liefert. Die PSoC-Experimentierplatine dient hierbei zur Anpassung und Wandlung der analogen Sensorspannung: Mit einem Analog-/Digitalwandler (ADW) wird das Sensorsignal digitalisiert, auf dem LCD-Panel ausgegeben und über die serielle Schnittstelle an die RS232-Extension des Loxone Miniservers gesendet. Im Miniserver sollen dann die Messwerte in °C umgerechnet und grafisch dargestellt werden.

### Thermistor KTY 83-110

Ein Thermistor ist ein temperaturabhängiger Widerstand auf Silizium-Halbleiterbasis. Thermistoren gibt es sowohl als NTC wie auch als PTC. Der KTY 83-110 ist ein preisgünstiger PTC der Firma Philips Semiconductors. Bei 25 °C hat er einen typischen Widerstand von 1000Ω und eine Toleranz von  $\pm 10\Omega$ . Er kann über einen Temperaturbereich von -55 °C bis +175 °C eingesetzt werden. Die Tabelle zeigt typische Widerstandswerte über einen kleinen Temperaturbereich. Der maximal zulässige Gleichstrom beträgt 10mA.

Temperatur [°C]	typ. Widerstand [Ω] $I_{con}=1mA$
-10	754
0	820
10	889
20	962
25	1000
30	1039
40	1118
50	1202
60	1288

### Versuchsaufbau / Blockschaltbild



Der Thermistor  $R_T$  bildet zusammen mit den Präzisionswiderständen  $R_1$  und  $R_2$  einen Spannungsteiler. Bei einer Temperatur von 25 °C hat der Thermistor einen Widerstand von 1kΩ und somit ergibt sich eine Sensorspannung von  $V_{DD}/2$ . Der programmierbare Verstärker (PGA) benötigt eine analoge Masse von  $AGND=V_{DD}/2$ .

Der Messbereich des AD-Wandlers ist  $V_{SS}$  bis  $V_{DD}$ . Über den UART-Baustein des PSoC werden die Daten per RS232 an den Loxone Miniserver gesendet.

## Datenübertragung per PSoC UART

Mit dem Usermodul UART ist es möglich, Daten über eine serielle Schnittstelle zu senden und zu empfangen. Es wird eine bidirektionale, virtuelle serielle Schnittstelle (UART: Universal Asynchronous Receiver Transmitter) realisiert, bei der Übertragungsraten von bis zu 4 Millionen Bits pro Sekunde möglich sind.

Über API-Funktionen können nicht nur einzelne Bytewerte (Lowlevel-APIs) übertragen werden, sondern ganze Zeichenketten (Highlevel-APIs), mit oder ohne Terminierungszeichen sowie HEX-Darstellung von 1- und 2-Bytewerten im ASCII-Format.

Stellen sie in den Design Wide Resources die Taktquellen entsprechend dem untenstehenden Bild ein.

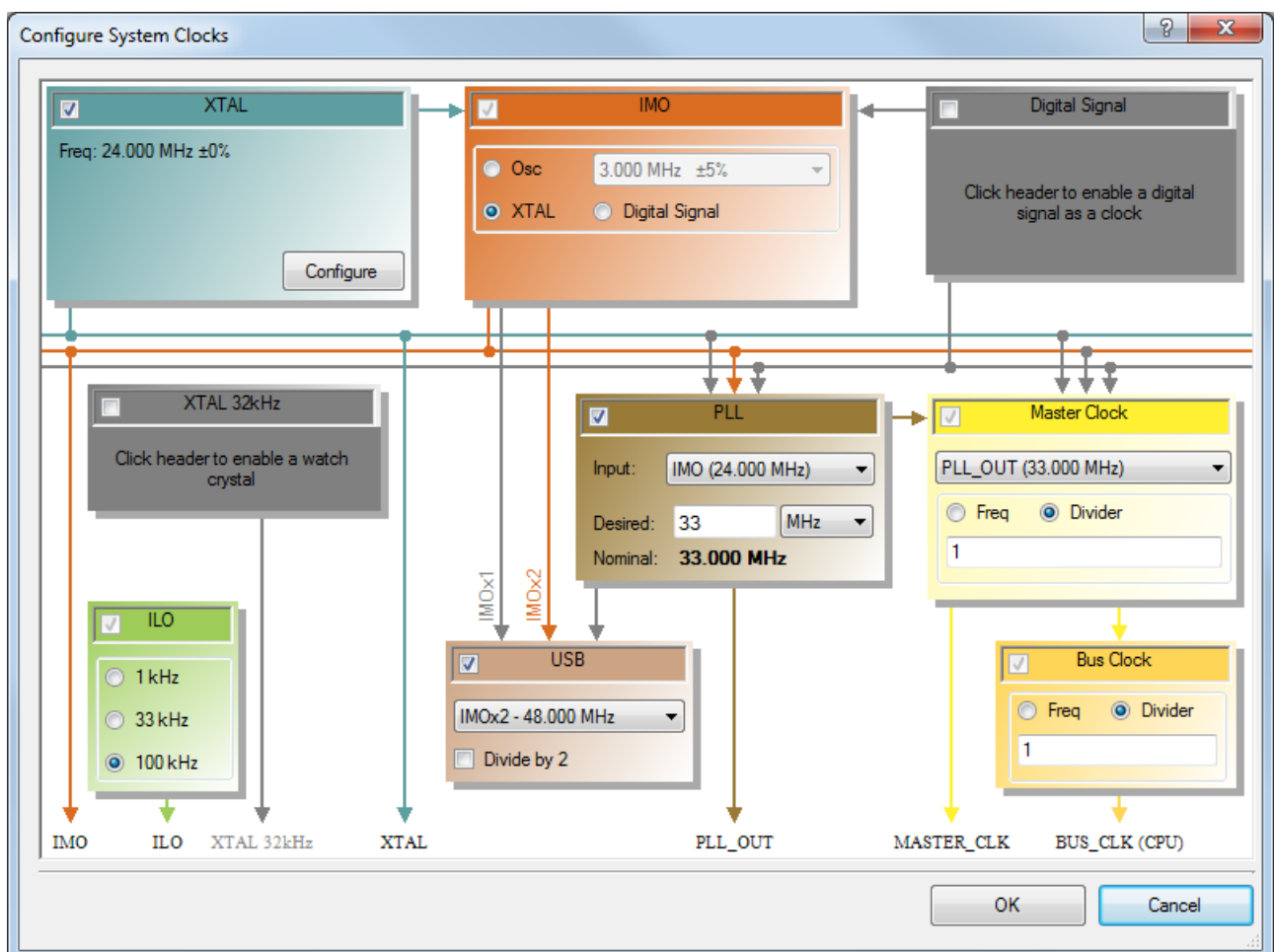


Abbildung 1: Systemweite Taktquellen

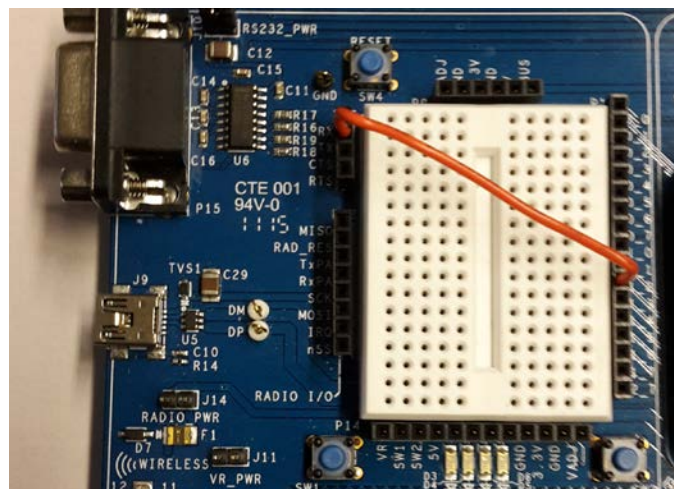
Ordnen Sie dem Anschluss TX\_OUT noch den Pin 1[2] zu und stellen sie sicher, dass er gemäß Abbildung 2 mit den Pegelwandlern für die neunpolige Buchse des seriellen Ports des PSoC-Boards verbunden ist.

### Programmbeispiel:

```
#include "stdio.h"
char8 buffer[30];
uint8 var;

void main()
{
...
    CyGlobalIntEnable; //enable global interrupts
    UART_Start();
...
    for(;;)
    {
        sprintf(buffer, "%x\r\n", var);
        UART_PutString(buffer);
    }
}
```

Weitere API-Funktionen, z.B. zur Übertragung von Zeichenketten (Strings) finden Sie in der Programmhilfe.



**Abbildung 2: RS232 Jumper PSoC-Board**

## AD-Wandler

Mit dem PSoC-Baustein und dessen Usermodulen können eine Vielzahl unterschiedlicher AD-Wandler (kurz ADW) realisiert werden. Der Einfachheit halber soll nur der ADC\_SAR erläutert werden.

Der ADC\_SAR ist ein 12-Bit ADW mit sukzessiver Approximation und Wandlungsraten zwischen 56 kHz bis 1 MHz. Der Eingangsspannungsbereich wird auf  $0.0 \pm V_{dda}/2$  festgelegt.

Eine Auflösung von 12-Bit und ein ADW-Wert in 16-Bit Zweierkomplementdarstellung bedeutet, dass die oberen 4 Bits bei negativen Werten "high" und bei positiven "low" sind.

Die Wandlungsrate wird über eine externe oder interne Taktquelle, gemäß folgender Formel, eingestellt:

$$\text{Wandlungsrate[kSPS]} = \text{Takt[kHz]} / 19$$

Die Auflösung bzw. das LSB (Least Significant Bit) des ADW wird wie folgt berechnet:

$$\text{LSB} = \text{MB} / (2^n - 1) \quad ; \text{MB: Messbereich in Volt}$$

$$; n: \text{Anzahl der Bits}$$

Für die Berechnung der gemessenen Spannung des AD-Wandlers gilt die Formel:

$$U = \text{sign}(\text{ADW}) * |\text{ADW}| * \text{LSB} + \text{LSB}/2 \quad ; \text{ADW: Zahlenwert des AD-Wandler}$$

$$; \quad \text{in Zweierkomplementdarstellung}$$

$$U = (\text{ADW} - 2^{(n-1)}) * \text{LSB} + \text{LSB}/2 \quad ; \text{ADW: Zahlenwert des AD-Wandler}$$

$$; \quad \text{im OffsetBinary-Format}$$

$$; n: \text{Anzahl der Bits}$$

Als Quelle für das Eingangssignal *Input* können die Ausgänge einiger anderer analoger Blöcke verwendet werden.

### Programmbeispiel:

```
void main()
{
    ADC_Start();
    ADC_StartConvert();

    for(;;){
        if (ADC_IsEndConversion(ADC_WAIT_FOR_RESULT));
        {
            result = ADC_GetResult16(); //getting and saving ADC result
        }
    }
}
```

## Loxone Miniserver: Die RS232-Extension

Die RS232-Extension erlaubt es, mit Geräten über den Loxone Miniserver zu kommunizieren, die eine serielle Schnittstelle des Standards RS-232 besitzen. Das zu steuernden oder zu lesenden Peripheriegerät wird über eine dreipolige Klemme mit den Anschlüssen TxD (Sendeleitung für Signale vom Miniserver), RxD (Empfangsleitung des Miniservers für Sensorsignale) und einer Masseleitung an die Extension angeschlossen. Zwar fehlen der Extension die Anschlussmöglichkeiten für die im RS232-Standard definierten „Handshake“-Leitungen zur gezielten Kontrolle des Datenflusses zwischen den beiden Geräten, diese können in den meisten Fällen jedoch durch Softwarelösungen ersetzt werden.

Wie alle anderen Extensions für den Loxone Miniserver die Sie bereits in diesem Praktikum kennengelernt haben, kommuniziert die RS232-Extension über den LoxBus mit dem Miniserver.

In den Experimentierplätzen ist die serielle Schnittstelle der Extension fest mit einem 9-poligen Sub D-Stecker verkabelt, dessen Beschaltung mit der Buchse der PSoC-Platine übereinstimmt und daher direkt eingesteckt werden kann.

### Hinzufügen der Rs232-Extension in Loxone Config

Damit der Miniserver Daten über die serielle Schnittstelle empfangen kann, muss zuerst die RS232-Extension zu dem aktuellen Projekt hinzugefügt werden. Klicken Sie dazu auf den Reiter „**Miniserver**“ und dann in der oberen Menüleiste auf „**Peripherie Suchen**“. Markieren Sie nun die RS232-Extension in der Liste der gefundenen Peripheriegeräte und fügen sie mit einem Klick auf „**Gerät erstellen**“ dem aktuellen Projekt hinzu.

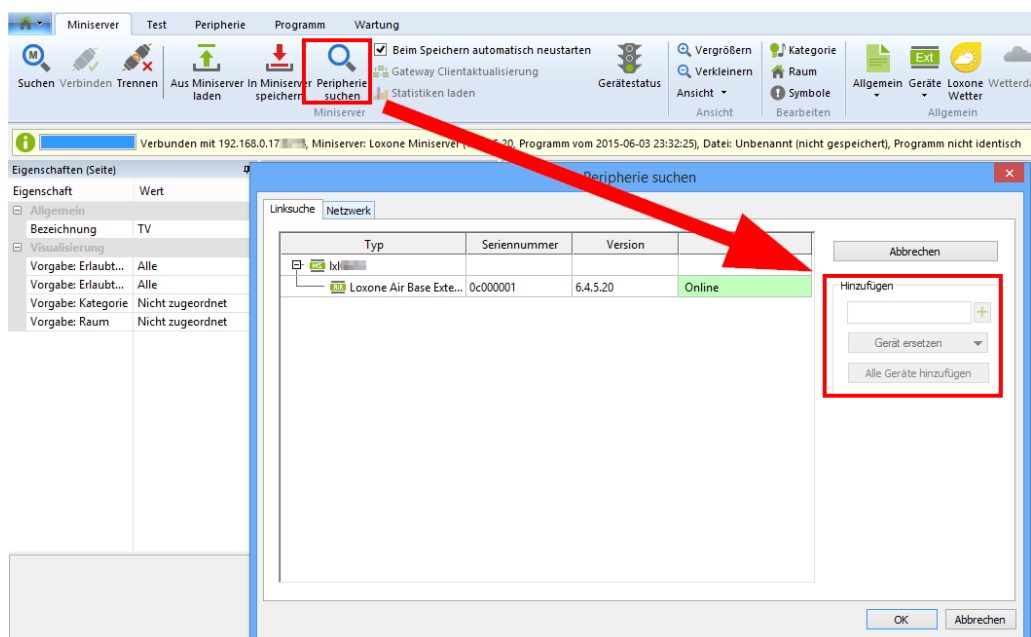


Abbildung 3: Hinzufügen einer Extension

### Miniserver: Datenempfang über die serielle Schnittstelle

Um Daten am Miniserver über die serielle Schnittstelle zu empfangen muss zuerst in Loxone Config die RS232-Extension gemäß dem verwendeten Protokoll eingestellt werden. Dazu die Extension im Peripherie-Baum auswählen und gemäß Abbildung 4 einstellen.

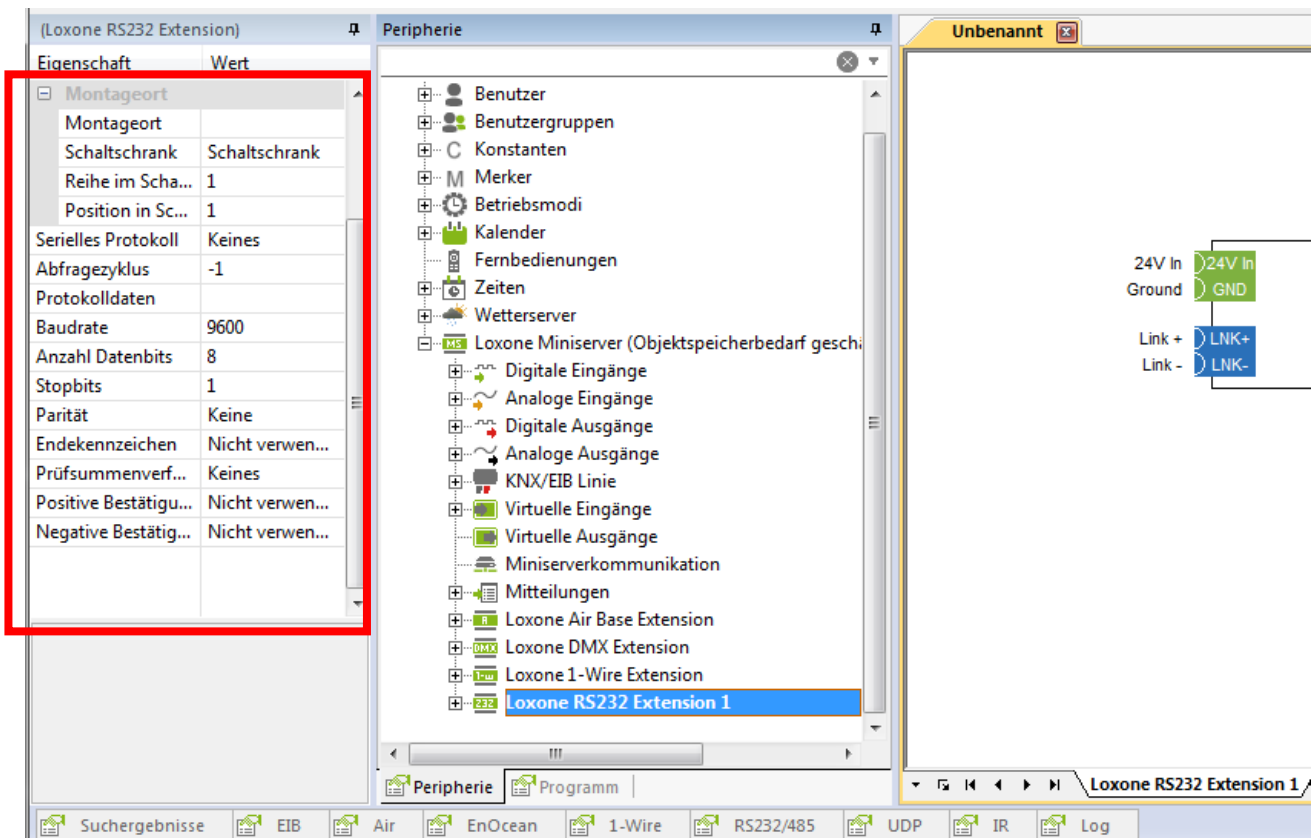


Abbildung 4: Einstellungen RS232-Extension

Das Auslesen der vom Peripheriegerät empfangenen Daten, die Verarbeitung und logische Verknüpfung mit anderen Daten in der Ablaufsteuerung und die Visualisierung der Daten geschieht ähnlich wie das Vorgehen bei der Verwendung der 1-Wire-Extension durch die Definition eines oder mehrerer RS232-Sensoren. Ein neuer Sensor wird mit der Schaltfläche „**Sensoren und Aktoren**“ -> „**RS232-Sensor**“ im Reiter „**Peripherie**“ erstellt und erscheint dann im Peripherie-Baum unter dem Eintrag der RS232-Extension.

Mit einem Klick auf den Eintrag des Sensors erscheint das Eigenschaftsfeld. Die wichtigste Eigenschaft ist hier der Eintrag „Befehlserkennung“, da er das Verhalten des Sensorbausteins festlegt. Der Miniserver vergleicht die hier eingetragene Zahlenfolge mit dem über die serielle Schnittstelle empfangenen Datenstrom und gibt bei Übereinstimmung den erkannten Wert („Analogsensor“) oder einen digitalen Puls („Digitalsensor“, Checkbox „Als Digitaleingang verwenden“ markiert) am Ausgang des Sensorbausteins aus.



### Beispiele:

#### Fall 1:

Empfangener Datenstrom: „CMD01 20,5 OK\n\r“  
Befehlskennung: „CMD01 \v OK\n\r“

Ausgabe Analogsensor: 20,5  
Ausgabe Digitalsensor: Puls

#### Fall 2:

Empfangener Datenstrom: „CMD01 1024 OK\n\r“  
Befehlskennung: „CMD01 \v OK\n\r“

Ausgabe Analogsensor: 1024  
Ausgabe Digitalsensor: Puls

#### Fall 3:

Empfangener Datenstrom: „CMD02 20,5 OK\n\r“  
Befehlskennung: „CMD01 \v OK\n\r“

Ausgabe Analogsensor: Keine Änderung des Ausgangswertes  
Ausgabe Digitalsensor: 0

Die Befehlskennung lässt hierbei eine Reihe von Sonderzeichen zu, die zur Formatierung der erkannten Werte verwendet werden. Diese sind durch das Präfix „\“ gekennzeichnet. Im Fall des Beispiels steht „\v“ („value“) für die Übernahme eines Wertes aus einer ASCII-kodierten Zeichenfolge.

Mögliche Sonderzeichen sind:

\h	Wertetext als Hexadezimalzahl übernehmen
\x	Hexadezimal z.B. \x09 für 0x09 oder \x01\x02\x03\x04 für 0x01020304
\1 (\2, \3, \4)	Der Numerische Wert des empfangenen Bytes wird als Byte (LSB) einer Zahl mit der Wertigkeit 1 (2,3,4) am Ausgang übernommen. Bsp.: Empfangen: „0xA9 0xF1 0x04“ Befehlskennung: „\3\2\1“ Wert am Ausgang: 0x04F1A9
\v	Aus einer ASCII Zeichenfolge wird der Wert übernommen. Die Kommastrichen müssen durch einen Beistrich oder durch einen Punkt getrennt sein.
\#	Irgendeine Zahl
\r\n	Carriage Return/Line Feed (neue Zeile)

\a	Irgendein Buchstabe
\t	Tabulator (0x09)

Da wie bei allen Extensions die Kommunikation der RS232-Extension mit dem Miniserver über den LoxBus stattfindet, verzögert eine häufige Abfrage von Sensorwerten/mit hoher Frequenz ankommende Sensorwerte das Ausführen der Programmierung und die Abfrage anderer Extensions. Als guter Wert hat sich ein Sensorwert pro Sekunde ergeben.

### Loxone Config: Der RS232-Monitor

Ein wichtiges Werkzeug zur Analyse der mit der RS232-Extension empfangenen Daten und zur Fehlersuche ist der RS232-Monitor in Loxone Config. Zum Starten wird erst die RS232-Extension im Peripheriebaum ausgewählt und dann im Reiter „**Miniserver**“ die Checkbox „**RS232/RS485-Monitor starten und anzeigen**“ gesetzt (Abbildung 5). Im unteren Teil des Bildschirms öffnet sich nun ein Fenster, das alle über die serielle Schnittstelle empfangenen Daten anzeigt und bei korrekt definierter Befehlserkennung dekodiert.



Abbildung 5: Loxone Config: RS232-Monitor

## Temperaturberechnung und Darstellung

Die Berechnung der Temperatur könnte auch der Mikrocontroller des PSoC erledigen. Da die Multiplikationen und Divisionen mit Fließkommazahlen jedoch etwas aufwändiger ist, ist es einfacher, diese im Miniserver durch das Einfügen der entsprechenden Blöcke in Loxone Config zu berechnen.

Zur Messwertverarbeitung und –darstellung verwenden Sie Ihr Programm aus „Versuch 2 – Sensordaten mit Loxone Config“ und fügen die Berechnung der Thermistor-Temperatur auf den Seiten „Sensorerkennung“ und „Umrechnung Sensorwert“ hinzu.

Wie in Versuch 2 bereits erwähnt, beinhaltet das 16-Bit Datenwort einen 4-Bit Steuercode und ein 12-Bit AD-Wert im Zweierkomplement-Format. Nachdem Sie das 16-Bit Datenwort aufgesplittet haben, sollten Sie den 12-Bit AD-Wert in eine reelle Zahl umwandeln. Anschließend berechnen Sie in diesem Versuch die Thermistorspannung und den –widerstand.

Die Berechnung der vom Thermistor gemessenen Temperatur geschieht in zwei Schritten. Zuerst berechnen Sie entsprechend den Aufgaben in der Versuchsvorbereitung den Wert des Thermistors in  $\Omega$  aus dem gewandelten Digitalwert. Im zweiten Schritt wird die Temperatur in  $^{\circ}\text{C}$  aus dem Widerstandswert in  $\Omega$  mithilfe der Steinhart-Hart-Gleichung berechnet. Die sogenannte Steinhart-Hart-Gleichung ist eine an Thermistoren angepasste logarithmische Polynominterpolation dritten Grades nach der Formel:



$$T = 1 / (A + B \cdot \ln(R) + C \cdot \ln(R)^3) - 273.15 \quad (1)$$

Der quadratische Term hat in der Praxis keine Bedeutung, da er die Genauigkeit nur minimal erhöht. Die konstanten Koeffizienten A, B und C wurden über 3 typische Widerstands/Temperatur-Paare (bei 0°C, 25°C und 50°C) aus dem Datenblatt des Thermistors berechnet.

Koeffizient	Wert
A	$24920,4935 \cdot 10^{-6}$
B	$-3944,7852 \cdot 10^{-6}$
C	$17,2416 \cdot 10^{-6}$

## Aufgabenstellung

Jede gelöste Teilaufgabe muss immer zuerst getestet werden, bevor die nächste bearbeitet wird. Sie sollten sich an die vorgegebene Reihenfolge halten und bei jeder Teilaufgabe mit der Markierung "→**Betreuer**" einem Betreuer das Programm zeigen und das Projekt unter einem anderen Namen fortführen.

- 1) Legen Sie ein neues Projekt an und stellen die globalen Parameter so ein, wie im vorherigen Versuch.
- 2) Verwenden Sie einen PGA mit einem Verstärkungsfaktor von 8. Überlegen Sie, wie Sie den Verstärker testen können.
- 3) Für das UART Usermodul müssen die Taktquellen entsprechend der Versuchsbeschreibung eingestellt werden.
- 4) Inkrementieren oder dekrementieren Sie eine Testvariable in einer Schleife und prüfen Sie die Übertragung an die RS232-Extension und den Miniserver mit dem „RS232-Monitor“ in Loxone Config.

→ **Betreuer**

- 5) Fügen sie den Eingangspin ihres Sensors in die Programmierung ein. Für den ADW ADC\_SAR verwenden sie eine der berechneten Wandlungsraten.
- 6) Der ADW speichert das Ergebnis der Wandlung als Integerwert im 2er-Komplement. Sichern Sie diesen Wert in einer Variablen. Fügen Sie dem High-Byte den Steuercode 1h für Sensor 1 in den oberen vier Bits hinzu (siehe auch Versuch 2). Anschließend übertragen Sie die vier Hexadezimalziffern als ASCII-Zeichen über die RS232-Extension an den Miniserver und testen diese mit dem „RS232-Monitor“ in Loxone Config.
- 7) Erstellen Sie entsprechend der Anleitung einen neuen RS232-Analogsensor der den übertragenen Wert korrekt dekodiert.
- 8) Mit dem CyDelay(int ms); sollten Sie Ihre Programmschleife so verzögern, dass einmal pro Sekunde ein neuer Wert über die RS232-Schnittstelle gesendet wird.

→ **Betreuer**

- 9) Erstellen Sie Ihr *LoxoneConfig*-Programm zur Berechnung und Darstellung der Sensortemperatur unter Verwendung Ihres Programms aus Versuch 2. Beachten Sie dabei den verwendeten Steuercode. Zur Glättung des Signals führen Sie am Ende eine Mittelwertbildung ein. Der gewandelte Temperaturwert soll in der Visualisierung/dem Webinterface erscheinen und über eine Statistik verfügen. Vergleichen sie den gemessenen Temperaturwert mit dem 1-Wire Temperatursensor und dem 0-10 V Temperatursensor des Experimentierplatzes.
- 10) Ermitteln Sie den Temperatur-Offsetfehler, indem Sie den Thermistor durch einen Präzisionswiderstand ersetzen. Auf der Sensorplatine muss dazu nur der Jumper JP1 in Position Cal. gesteckt werden. Damit erhalten Sie einen Spannungsteiler mit dem gleichen Teilverhältnis wie mit dem Thermistor bei 25°C. Diesen Wert verwenden Sie als Offset für ihre Temperaturberechnung.

→ **Betreuer**

- 11) Auf dem LCD-Panel geben Sie in der ersten Zeile den ADW-Wert hexadezimal aus.  
In der zweiten Zeile geben Sie den AD-Wert mit Vorzeichen und Betrag aus

→ **Betreuer**

- 12) Teilen Sie den Temperaturbereich zwischen 20 °C und 40 °C in vier Bereiche auf.  
Schreiben Sie Ihren Quellcode so um, dass auf dem LCD-Panel angezeigt wird, in welchem Temperaturbereich die gemessene Temperatur liegt. Bsp.: die gemessene Temperatur beträgt 27°C. Die Ausgabe auf dem LCD-Panel könnte lauten: "25 < T.Mess < 30".

*Hinweis: Um sicher zu gehen, dass die richtige Temperatur angezeigt wird, müssen Sie eventuell den Temperaturoffset berücksichtigen, welchen Sie in 9) ermittelt haben.*

- 13) Als Erweiterung zu der LCD-Anzeige können Sie den 4 Temperaturbereichen unterschiedlich schnelle LED-Leuchtfrequenzen zuordnen. Zum Beispiel für den Bereich A 1Hz, Bereich B 2Hz, usw.

Als Einstellung für die Taktquelle  $f = 1\text{kHz}$  verwenden und mittels einer PWM den Takt weiter herunter zu teilen. Durch Manipulation der „Period“ und „Compare“ Werte lässt sich die Frequenz und die Intensität der LED beeinflussen.

→ **Betreuer**

### **Hinweise und Tipps:**

Gehen Sie Schrittweise vor und testen auf korrekte Funktion.

- Z.B. können Sie an den Eingang des PGA einen DAC anschließen und über LEDs die Funktionsweise prüfen. Dazu muss der PGA-Ausgang mit einem LED-Pin verbunden werden. Zum Anschluss an den ADW kann diese Verbindung wieder gelöscht werden.
- Verwenden Sie zur Datenübertragung zunächst definierte, konstante Werte
- Auf dem LCD können Sie statt des AD-Wertes zunächst eine Testvariable (2 Byte) ausgeben