Übung 1-4 – Übermitteln von CAN-Daten

Über einen Thermistor (Heißleiter, NTC (MF52-103 3435)) soll die Temperatur gemessen werden. Dabei wird der Spannungsabfall über den temperaturabhängigen Widerstand mithilfe eines Spannungsteilers gemessen.

Die gemessene Temperatur soll auf dem **CAN übertragen** werden. Dafür wird der Mikrokontroller **Aurix TC374** verwendet, der eine integrierte CAN-Schnittstelle bietet. Der **CAN-Transceiver „TLE9251VSJ“** von Infineon ist auf dem TC375 verbaut.

Schaltungsaufbau

- Spannungsteiler: Vorwiderstand (10 kΩ) an 5V und NTC (10 kΩ bei 25°C) an Masse
- Spannungsabgriff an Eingang P40.9 des Aurix TC375 mit einer Auflösung des ADC von 10 Bit bei Spannungsbereich von 0 . . . 5V

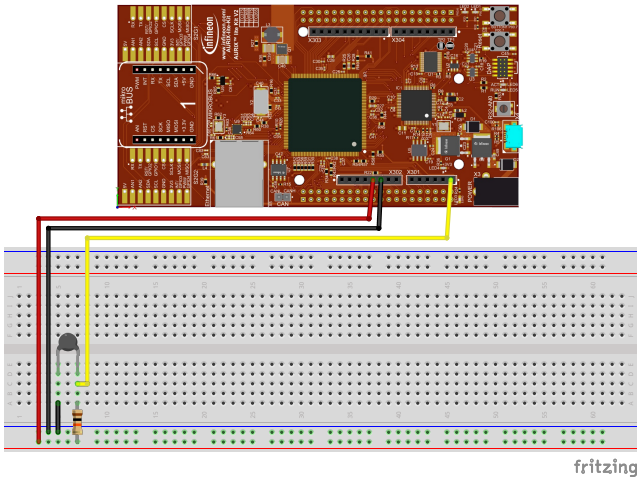


Abb. 1: Steckplatine

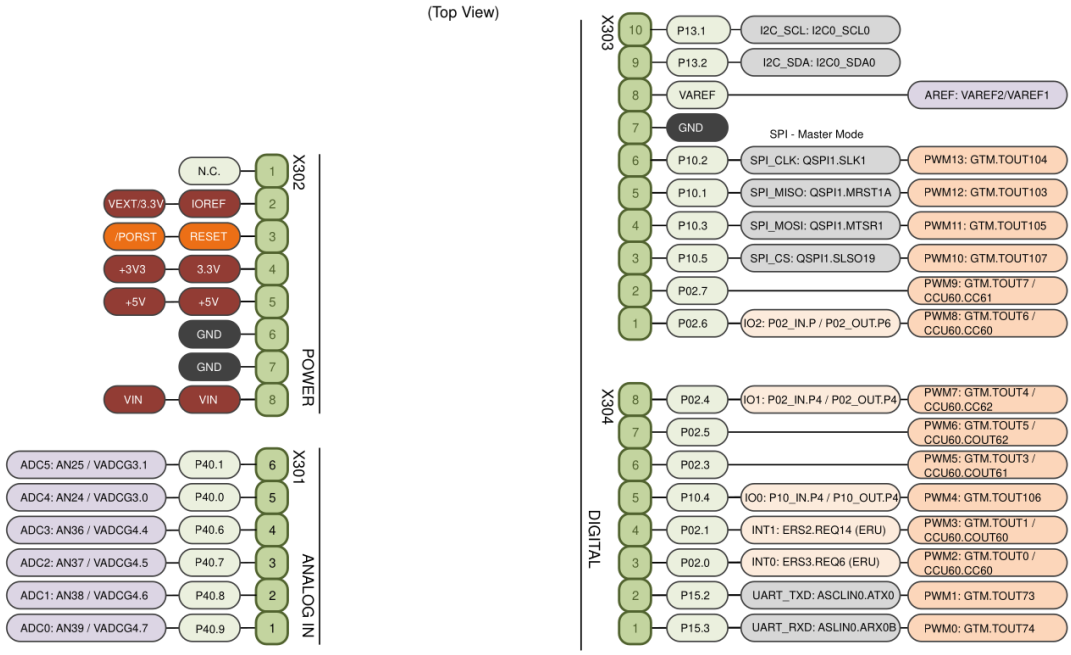
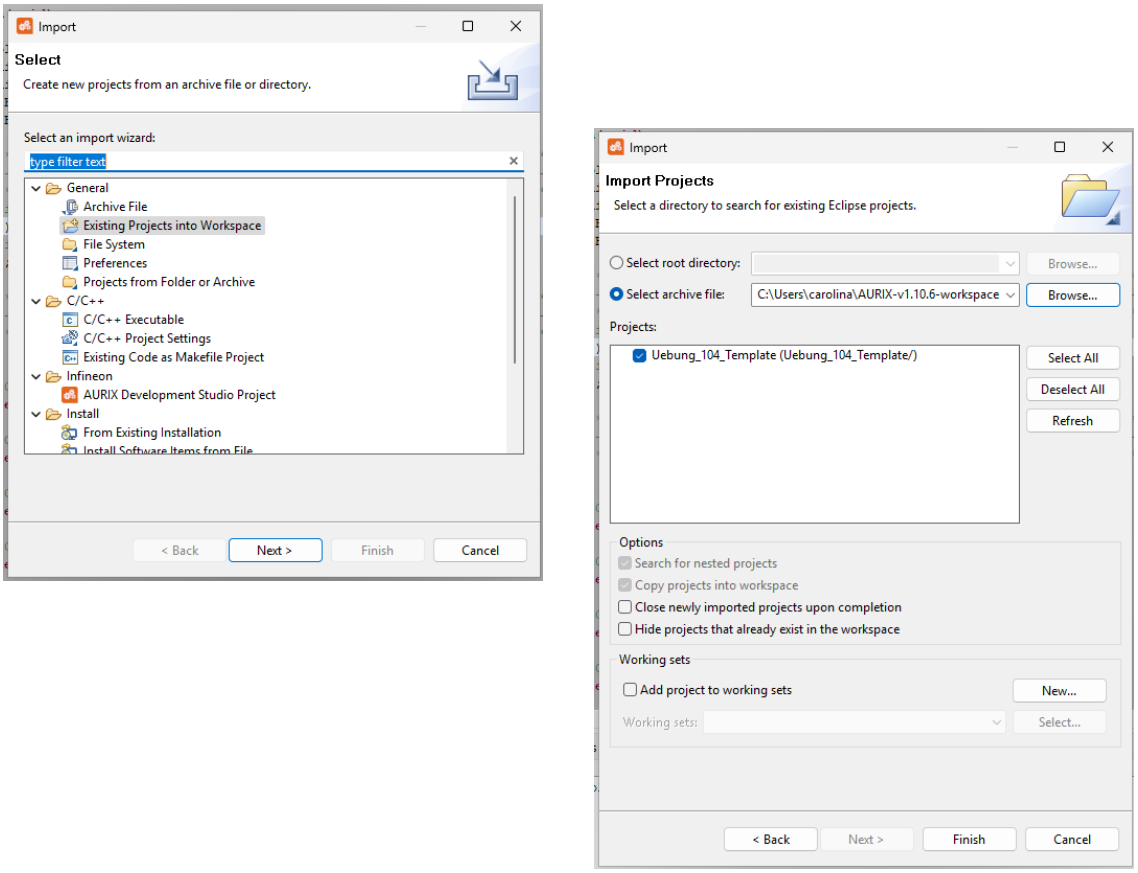


Abb. 2: Schaltplan Mapping of Arduino Functions to AURIX™ Pin Functions

Softwareerstellung

Verwenden Sie als Basis das Aurix-Development-Studio-Template `Uebung_104_Template.zip`. Importieren sie das Projekt.

`File > Import > General > Existing Projects into Workspace > Select archive file > Browse > Finish`



(a) **(b)**

Abb. 3: Importieren eines Projektes in Aurix Studio

Bearbeiten sie die Datei namens `Cpu0_Main.c`.

Initialisierung: (`void core0_main(void)`)

1. Initialisierung des CAN-Controllers mittels `initMCMCAN()`
2. Initialisierung des EVADC-Modul mittels `initEVADC()`

Endlosschleife: (`while(1)`)

1. Temperatur des Thermistors in Kelvin berechnen
2. Spannungswert am Pin A39 mittels `readEVADC()` einlesen.
3. Gemessene Spannung berechnen
4. Widerstand des Thermistors anhand Spannungsteilerregel bestimmen
5. Temperatur aus Widerstandswert bestimmen gemäß Formel

$$T(R) = \frac{1}{\frac{1}{B}\ln(\frac{R}{R_{25}}) + \frac{1}{T_{25}}}$$

wobei B – Sensorkonstante, R_{25} – Widerstand des Thermistors bei 25°C, T_{25} – Temperatur in Kelvin (25°C)

Ermitteln Sie die Sensorkonstante (B-Wert) aus dem Datenblatt des Thermistors (Siehe OPAL).

6. Ausgeben des Temperaturwert auf einer CAN-Botschaft mittels
   `transmitCanMessage(unsigned int message, uint32 messageId)`
   Beachte: Der CAN-ID lautet 100h + <Testplatznummer> – Somit sendet bspw. der Testplatz 4 seine Temperatur unter der CAN-ID **`104h`** und der Testplatz 10 mit der ID **`10Ah`**

7. Die Übertragung der Temperatur soll zyklisch aller 1000 ms erfolgen

Anhang

Arduino-Pins: (siehe `can.h`) **EVADC Channel and Group** (siehe `can.h`)