Konfigurieren des
MU-TC1

mit dem PPCAN-Editor

-

Ein Überblick mit Beispielen

Inhalt

Ausgabestand

Version	Änderungsgrund	Autor	Datum
0.1	Reviewexemplar	S. Schott	13.03.2009
0.2	Rechtschreibung, Reviewergebnisse	S. Schott	19.02.2010

Vorwort

Zielgruppe

Die Verwendung des PPCAN-Editors (anstelle der "Thermocouple-Configuration"-Software) setzt eine Mindest-Qualifikation des Benutzers hinsichtlich Hardware-Verständnis und Programmierkenntnissen voraus.
Dieses Tutorial wendet sich daher an Besitzer eines MU-TC1, die komplexere Konfigurationen des Geräts benötigen und über belastbare Grundkenntnisse der Elektronik und Informatik verfügen.

Zunächst sollte der für Kunden kostenlose PPCAN-Editor anhand dieses Tutorials getestet werden. Die Häufigkeit von Verständnisproblemen (technischer Natur !) beim Durcharbeiten des Dokuments kann möglicherweise als Entscheidungshilfe für einen zukünftigen Einsatz des Editors herangezogen werden. Im abschlägigen Fall bietet die PEAK-System Technik GmbH Ihren Kunden einen Konfigurationsservice gemäß detaillierter Spezifikation.

Technische Voraussetzungen

Zum sinnvollen Bearbeiten dieses Tutorials bzw. zum Lösen der Aufgaben steht ein Gerät MU TC1 inkl. Spannungsversorgung zur Verfügung. Dessen CAN-Bus ist über ein PCAN-PC-Interface an einen Computer angeschlossen und korrekt terminiert.

Ein passender Sensor steckt an Port 1a.

Ein Übertragungsnetz ist eingerichtet z.B. "Thermo_500k" mit 500kbit/s

Die Software PPCAN-Editor ist installiert.

Als CAN-Gegenstelle ist ein PCAN-View oder besser: ein PCAN-Explorer (Art.-Nr. IPES-005028) auf dem PC installiert.

Der Begriff Konfiguration

Die Mikrocontroller-basierten Geräte der Firma PEAK-System Technik GmbH bieten die Möglichkeit, alle verbauten Schnittstellen miteinander zu verknüpfen. Hierzu stellt die Firmware sog. Funktionsblöcke bereit, mit denen die Hardware-Ressourcen virtuell zu einer Konfiguration verschaltet werden. Zum Erstellen und Editieren von Konfigurationen stellt PEAK-System Technik GmbH den "PPCAN-Editor für Windows" mit jedem Gerät kostenlos bereit. Die so erstellte Datei mit der darin enthaltenen Konfiguration wird zunächst auf dem PC gespeichert, dann dem Gerät per Upload (via CAN) bekanntgegeben und dort nichtflüchtig gespeichert. Manche Geräte können mehrere Konfigurationen speichern: die jeweils gültige wird mittels eines Wahlschalters selektiert. Der Wahlschalter bestimmt gleichzeitig die Geräte-ID und den Speicherort der Konfiguration innerhalb des nichtflüchtigen Speichers.

Die mit dem PPCAN-Editor erstellten Dateien können mehrere Konfigurationen enthalten. Die Geräte-ID bestimmt dabei, welche davon bei Modulstart ausgeführt wird. Daraus ergibt sich z. B. die Möglichkeit, mehrere gleiche Geräte mit unterschiedlicher ID an einem CAN-Bus zu betreiben und die identische Konfigurations-Datei auf alle Geräte gleichzeitig zu uploaden. Die unterschiedliche ID sorgt dann dafür, dass jedes Gerät seine spezielle Konfiguration aus dem nichtflüchtigen Speicher lädt und dementsprechend seine Aufgabe ausführt.

Möglichkeiten der Konfiguration

Zur Verknüpfung von Schnittstellen gibt es die einfache Skalierung von Größen sowie die Methoden "CAN-Gateway-Dienste", "Default-Werte", "Funktionsblöcke", "ereignisgesteuertes CAN-Senden", "zeitgesteuerte Aktivitäten" und "Kennlinien". Bei Geräten mit nur einem CAN-Bus entfällt der Gateway-Dienst, und "zeitgesteuerte Aktivitäten" sind ebenfalls nicht mit jeder Hardware möglich. Die tatsächlich vorhandenen Ressourcen eines Geräts werden dem Editor durch eine spezielle Profildatei mitgeteilt, damit dieser die Verknüpfungsmöglichkeiten entsprechend freischaltet bzw. einschränkt.

Skalierung

Das grundlegendste Mittel der Manipulation von Größen ist die Verwendung der vier Grundrechenarten. Dies wird mit den Parametern SCALE und OFFSET gesteuert, die von der bekannten Geradengleichung aus der Mathematik übernommen sind. Dabei ist der Parameter SCALE für Multiplikation (wenn > 1) bzw. Division (wenn < 1) zuständig, der Parameter OFFSET bewirkt eine Addition (wenn > 0, positiv) bzw. Subtraktion (wenn < 0, negativ). Als neutrale Voreinstellung ist daher SCALE=1 und OFFSET=0 gewählt.

CAN-Gateway-Dienste

Eingehende Nachrichten auf dem einen CAN-Bus werden selektiv auf dem anderen CAN-Bus ausgegeben. Oder sie können auf dem gleichen CAN-Bus mit anderer ID ausgegeben werden (z. B. Umsetzung 11bit<->29bit). Oder eine eingehende Nachricht kann das Versenden einer beliebigen anderen Nachricht auslösen, wobei beide Nachrichten inhaltlich nichts miteinander zu tun haben.

Default-Werte

Durch hier festgelegte Parameter kann der Zustand des Moduls nach dem Einschalten bestimmt werden, wie z. B. die Baudrate des CAN-Bus, Aktivierung einer externen RS232-Schnittstelle, Aktivierung eines 5 V-Ausgangs für Sensorenversorgung, den log. Zustand von Leitungen, Leuchtdioden etc.

Funktionsblöcke

Falls die einfache Manipulation von Messgrößen etc. mittels Scale und Offset nicht ausreicht, stehen sog. Funktionsblöcke mit erheblich komplexeren Möglichkeiten zur Verfügung
Dies reicht vom Umsetzen per X/Y-Tabelle über Hysteresefunktionen, Delays, Zähler, Tiefpass-Filter, diverse mathematische und logische Operatoren bis zum komplexen PIDT1 Regelkreis. Funktionsblöcke können sequentiell oder bedingt abgearbeitet werden.

Ereignisgesteuertes Aussenden von CAN-Botschaften

Falls CAN-Botschaften nur bei bestimmten Gelegenheiten gesendet werden sollen, steht außerdem ein Repertoire an Triggerbedingungen zur Verfügung. CAN-Botschaften können außerdem von extern angefordert werden.

Kennlinien

Hier lassen sich 2..31 X-Werte als Eingangsparameter eintragen, denen je ein Y-Wert als Ausgangsparameter zugeordnet ist. Für den Wertebereich zwischen zwei X-Werten wird das Y-Ergebnis linear interpoliert. Anders ausgedrückt: So lassen sich (wie bei der Skalierung) SCALE und OFFSET für bis zu 32 Abschnitte eines Wertebereichs einzeln einstellen. Damit ist es möglich, Kurvensegmente in ihrer Steigung zu beeinflussen, um z. B. Plateaus zu definieren oder nicht-stetige Funktionen nachzubilden.

Geräteressourcen

Das Gerät " MU-TC1" ("Measuring Unit ThermoCouple 1", Art,-Nr. IPEH-002205) stellt folgende Ressourcen zum Verknüpfen bereit:

- Die Geräte-ID (4bit, 0..15 dez.)
  Sie kann im Innern des Geräts per Schalter verändert werden.

- 1 CAN-Bus (mit der Nummer 1, nicht 0 !!!)

- CAN Baudrate (10k, 20k, 33k3, 47k6, 50k, 83k3, 95k2, 100k, 125k, 250k, 500k, 1M)

- CAN-Botschaften (11bit oder 29bit)

- Information, ob und welche Messboards gesteckt sind (5bit, 0..31 pro Slot)
  0 = keine Karte gesteckt (leerer Slot)
  15 = Messkarte für 2 Thermoelemente Typ "K" (grün) gesteckt
  16 = Messkarte für 2 Thermoelemente Typ "J" (schwarz) gesteckt
  17 = Messkarte für 2 Thermoelemente Typ "T" (braun) gesteckt

- Die Temperatur des Kompensationssensors (=Referenztemperatur) jedes Messboards (insgesamt 4 Werte je 13bit, Auflösung 1/16°C)

- Die Temperaturen der beiden Messfühler pro Board (insgesamt 8 Werte je 16bit, 1/16°C)

- Je 2 Leuchtdioden pro Board (insgesamt 8, Status kann geschrieben und gelesen werden)

Aufgabenliste

Durch die Lösung der hier aufgelisteten Aufgaben lässt sich ein Überblick über die vielfältigen Möglichkeiten der PCAN-Hardware (hier speziell des MU-TC1) erhalten.

1 a) Setzen der CAN-Baudrate
1 b) Definieren von CAN-Botschaften
1 c) Ausgabe der Temperatur des Sensors an Port 1a in °C mit max. Auflösung
1 d) Exportieren eines Symbolfiles
1 e) Erstellen eines einfachen Instrument-Panels
1 f) Variation 1c: Ausgabe der Temperatur in °C mit 0,5° Auflösung
1 g) Variation 1c: Ausgabe der Temperatur in °F mit 1° Auflösung (ganzzahlig)
1 h) Verwendung der Kennlinienfunktion

2 a) Auswertung: Einschalten der LED, wenn Sensor gesteckt
2 b) Auswertung: Langsames Blinken der LED, wenn unterhalb Schwellwert
2 c) Variation 2b: LED ein, wenn unterhalb Schwellwert, mit Hysterese

3 a) On Event: Senden Alarm-Message, wenn außerhalb Fenster
3 b) On Event: Senden Temperatur nur, wenn Änderung um mind. 1°C

4 a) Diagnose: Ausgabe der Temperatur der internen Kompensationssensoren
4 b) Diagnose: Ausgabe der Modul-ID
4 c) Diagnose: Ausgabe des Kartentyps pro Slot
4 d) Diagnose: Externes Setzen der LEDs
4 e) Steuern der Blinkfunktion

5) On Request: CAN-Botschaften auf Anforderung

Lösungsweg mit Erläuterungen

Lösungsschritte zu Aufgabe 1 a

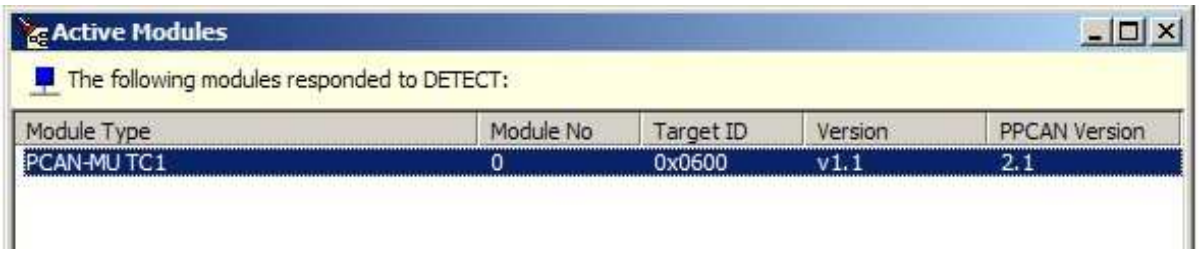
Aktion:
Öffnen des PPCAN-Editors durch Doppelklick

Aktion:
Verbinden des PPCAN-Editors mit dem Übertragungsnetz "Thermo_500k".
Dazu Menüpunkt "CAN" -> "Connect" anwählen.

Reaktion:
Das gewählte CAN-Netz erscheint in der Statuszeile des PPCAN-Editors (unten links).

Aktion:
Prüfen, ob das MU-TC-1 am CAN-Netzwerk gefunden werden kann:
Dazu Menüpunkt "Transmit" -> "Detect Modules" auswählen.

Reaktion:
Im Gutfall meldet sich das MU-TC1 mit einigen Statusinformationen.



Z. B. wird im Feld "Module No" die eingestellte Geräte-ID angezeigt (hier: 0).

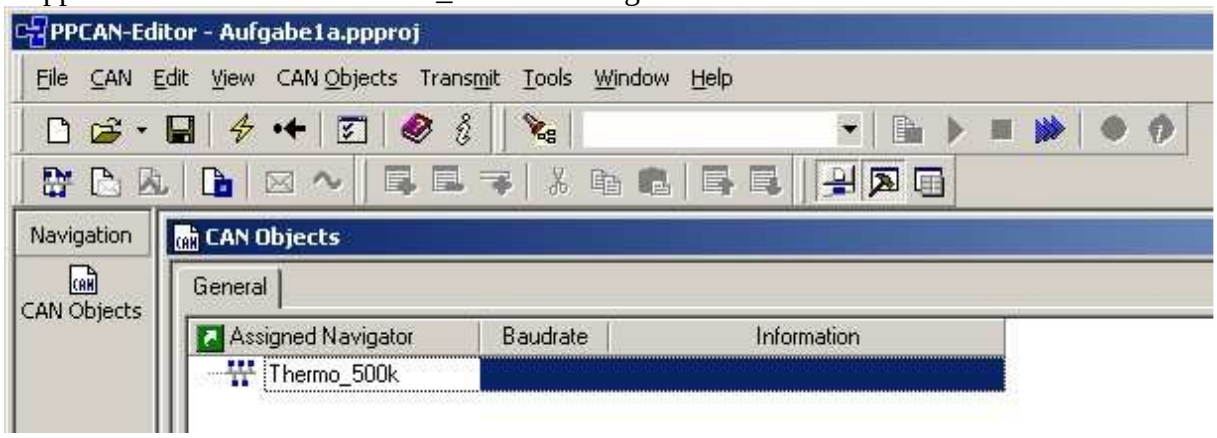
Aktion:
Anlegen einer leeren Konfigurationsdatei mit Menü "File" -> "New"

Reaktion:
Es erscheint ein leeres Fenster mit den globalen CAN-Objekten für alle später in der Datei enthaltenen Konfigurationen. Falls eine Datei mehrere Konfigurationen mit unterschiedlichen CAN-Objekten enthält, müssen -alle- zunächst hier definiert werden. Sie werden später selektiv in die verschiedenen Konfigurationen importiert.

Reaktion:
Im Fenster ist bereits ein CAN-Bus definiert: "BUS_0", unter dem man globale CAN-Objekte hierarchisch anlegen kann. Eine Konfiguration ist damit aber noch nicht angelegt.

Definition (willkürlich):
Der Bus #0 erhält den Namen "Thermo_500k".

Aktion:
Doppelklick auf den Namen "Bus_0" und Eintragen des neuen Namens.



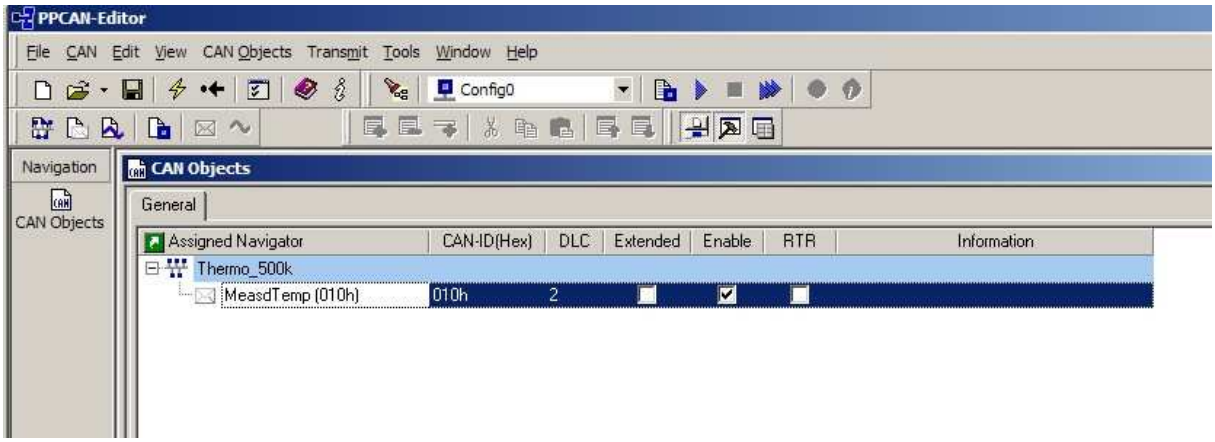
Busname: Thermo_500k
Baudrate: nur informativ, ohne Effekt
Information Eine Beschreibung, was diese Zeile tut (für später…)

Lösungsschritte zu Aufgabe 1 b

Information:
Um die geforderte Temperatur zu übertragen, genügt eine CAN-Botschaft mit 2 Bytes Länge, denn der entsprechende Messwert belegt 16 Bit.

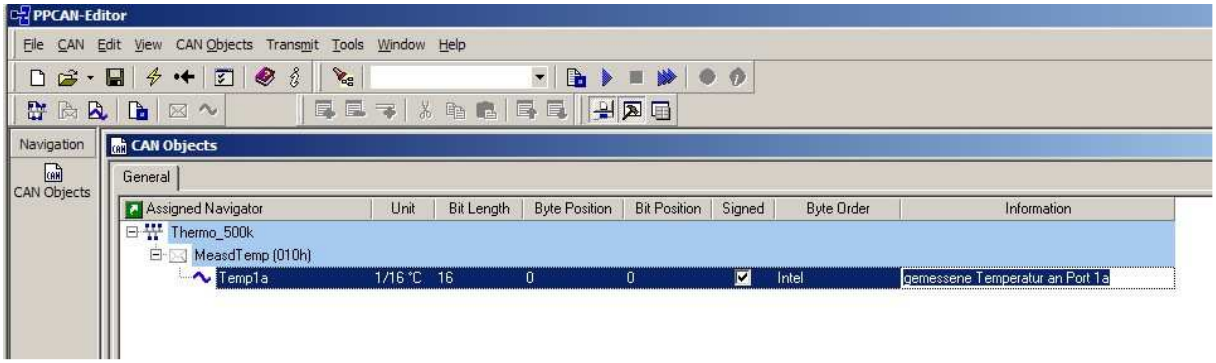
Definition (willkürlich):
Senden einer CAN-Botschaft "MeasdTemp" mit der ID 0x010, Länge 2 Bytes, zyklisch alle 300 msec auf dem Bus "Thermo_500k".

Aktion:
Dazu wird im Kontextmenü (Rechtsklick auf "Thermo_500k") der Punkt "Add a new Symbol" angewählt. Damit wird eine neue CAN-Botschaft auf dem Bus "Thermo_500k" definiert, deren Parameter jetzt noch ausgefüllt werden müssen:



Symbolname: "MeasdTemp"
ID: 0x010
DLC: 2
Extended: Nein, es genügt eine 11bit-ID
Enabled: Ja
RTR. Nein, die Botschaft soll -nicht- nur auf Anforderung gesendet werden
Information Eine Beschreibung, was diese Zeile tut (für später…)

Aktion:
Innerhalb der CAN-Botschaft muss ein 16bit breites Datenobjekt (= CAN-Signal) angelegt werden, das die gemessene Temperatur enthält. Dazu wird im Kontextmenü (Rechtsklick auf die CAN-Botschaft) der Punkt "Add a new Variable" angewählt, und dann die Parameter des Signals eingetragen.

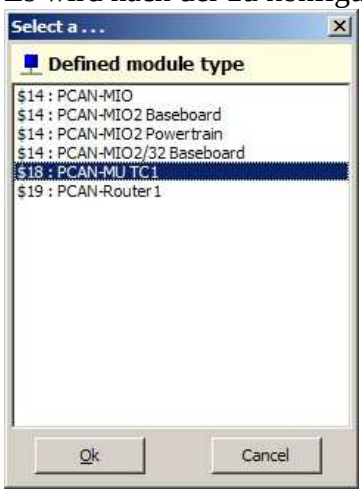


Variablenname: Temp1a
Unit: 1/16 °C (nur informativ)
Bit length 16
StartByte 0
StartBit 0
Signed Ja, vorzeichenbehaftet
(32767 ist größter positiver Wert, -32768 ist größter negativer Wert)
Byte Order Intel-Format (LSB in Byte 0 Bit 0, MSB in Byte 1 Bit 7)
Information Eine Beschreibung, was diese Zeile tut (für später…)

Information:
Die leere Hülle der CAN-Botschaft ist damit festgelegt, aber noch nicht einer physikalischen Datenquelle zugeordnet. Dazu muss eine Konfiguration erzeugt werden

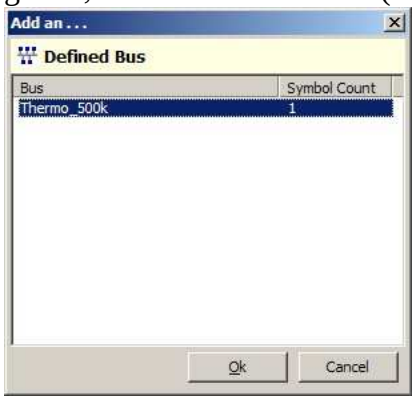
Lösungsschritte zu Aufgabe 1 c

Aktion:
Anlegen einer leeren Konfiguration innerhalb der Datei mit "Edit" -> "New Configuration"
Reaktion:
Es wird nach der zu konfigurierenden Hardware gefragt.



Information:
Der PPCAN-Editor ist für eine Vielzahl von konfigurierbaren PCAN-Geräten mit unterschiedlichsten Ressourcen ausgelegt. Für jede konfigurierbare PCAN-Hardware gibt es daher eine Liste der verfügbaren Ressourcen, die sog. **Profildatei**.
Aktion:
In diesem Fall ist das Profil für ein "MU TC1" auszuwählen.
Reaktion:
Es erscheint neben dem Kartei-Tab "General" ein neuer Tab mit dem Namen der Konfiguration "Config0 I/O". Außerdem ist im Navigationsfenster am linken Bildrand ein Icon "Config0" entstanden.

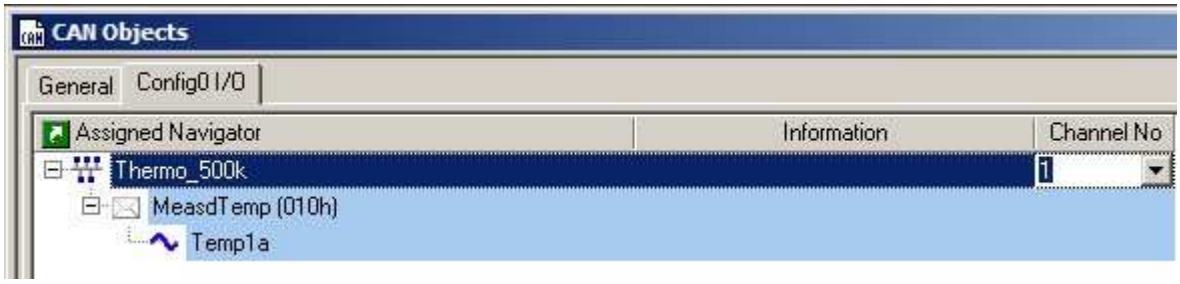
Aktion:
Die global definierte CAN-Botschaft soll in dieser Konfiguration verwendet werden. Sie muss also importiert werden. Dazu wird der neue Kartei-Tab "Config0 I/O" in den Vordergrund geholt, und im Kontextmenü (Rechtsklick) wird "Add defined bus" ausgewählt.



Reaktion:
Der global bereits definierte CAN-Bus "Thermo_500k" wird mitsamt seiner enthaltenen Botschaft "MeasdTemp" und der 16bit-Variablen "Temp1a" in die Konfiguration übernommen.

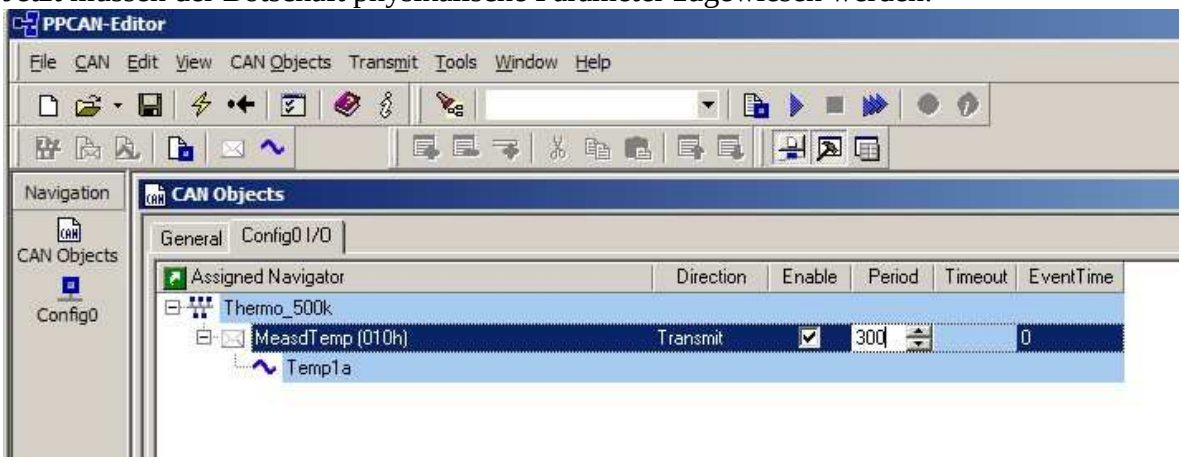
Aktion:

Wichtig: die Channel-No. beim MU-TC1 muss mit "1" versorgt werden, dies kann in der Listbox u. U. -nicht- ausgewählt werden (Channel 0 existiert beim MU-TC1 nicht) !!!



Aktion:

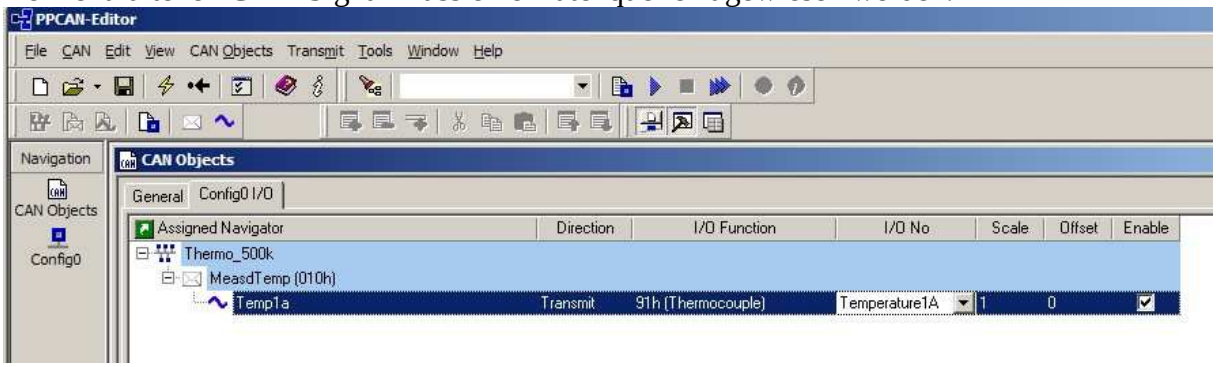
Jetzt müssen der Botschaft physikalische Parameter zugewiesen werden:



Direction: Transmit (Das MU-TC1 soll Sender sein)
Enable: Ja, diese Botschaft soll übermittelt werden
Period: 300 (Die Sendezykluszeit in msec)

Aktion:

Dem enthaltenen CAN-Signal muss eine Datenquelle zugewiesen werden.



I/O-Function: 91-Thermocouple (das ist die Datenquelle: Ein Thermopaar-Sensor)
I/O-Number: Temperature 1a (das ist der Sensor-Port, der die Daten liefert).
Scale: 1 (keine Verstärkung/Abschwächung der Daten, Multiplikation *1)
Offset: 0 (keine Anhebung/Absenkung des Wertebereichs, Addition 0)
Enable: Ja, dieses Signal (innerhalb der Botschaft) soll verwendet werden.

Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe abgeschlossen.
Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 1c" auf dem PC gespeichert.
Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload). Dazu muss der Menüpunkt "Transmit" -> "Send Configuration" angewählt werden (wichtig: am oberen Fensterrand im Toolbar muss die "Config0" in der Listbox eingestellt sein).

Reaktion:
Im Output-Window des PPCAN-Editors laufen nun verschiedene Statusmeldungen durch, die das Übertragungsprotokoll betreffen, und deren Bedeutung in anderen Dokumenten erläutert wird.

Reaktion:
Die Power-LED des MU-TC1 blinkt während der Übertragung und Verarbeitung der Konfigurationsdatei unrhythmisch. Sobald die LEDs an den Sensorports kurz aufblitzen, wurde die Konfiguration verarbeitet und ein automatischer Gerätereset durchgeführt. Danach blinkt die Power-LED mit 1 Hz, und das MU-TC1 ist mit seiner neuen Konfiguration betriebsbereit.

Reaktion:
Das MU-TC1 überträgt nun die gemessene Temperatur in 1/16 °C.

Lösungsschritte zu Aufgabe 1 d

Für Besitzer eines PCAN-Explorers (IPES-005028) ist es komfortabel, den übertragenen Wert mit Hilfe eines Symbolfiles zu dekodieren und lesbar darzustellen. Dazu dient ein Symbolfile, das vom PPCAN-Editor aus den CAN-Daten generiert und als Datei abgespeichert werden kann.

Aktion:
Durch Klicken auf Kartei-Tab "GENERAL" die CAN-Datenbasis auswählen.
Den Menüpunkt "File" -> "Export into Symbol File(s)" auswählen.
Einen Dateinamen und den Speicherort auf dem PC festlegen (z.B. "Aufgabe 1d").
Reaktion:
Es erscheint eine Aufforderung, die zu exportierenden CAN-Objekte auszuwählen:
Da in diesem Tutorial bisher noch nicht viele Objekte definiert wurden, kann der Button "Select_All" verwendet werden, um die zu exportierenden Objekte auszuwählen.
Aktion:
Der Button "Export" startet die Erzeugung des Symbolfiles.

Aktion:
Das Symbolfile kann nun im PCAN-Explorer mit "File" -> "Open" geladen und mit "File" -> "Apply" aktiviert werden.
Reaktion:
Im PCAN-Explorer erscheinen im Feld "Data" nun die beiden übertragenen Bytes des Messwerts, sowie der Versuch, diese Bytes zu einer Temperatur zu dekodieren.

Aktion:
Hier ist allerdings eine kl. Nacharbeit vorzunehmen:

Die Zeile im Symbolfile
```
"a=Temp1a      signed"
```

ist folgendermaßen zu ergänzen:
```
"a=Temp1a      signed /u:°C /f:0,0625"
```

Information:
Dies ist notwendig, weil zum einen ein physikalischer Messwert mit der Einheit "°C" übertragen wird, zum anderen weil dieser Messwert in 1/16 Grad-Schritten (=0,0625) aufgelöst ist.

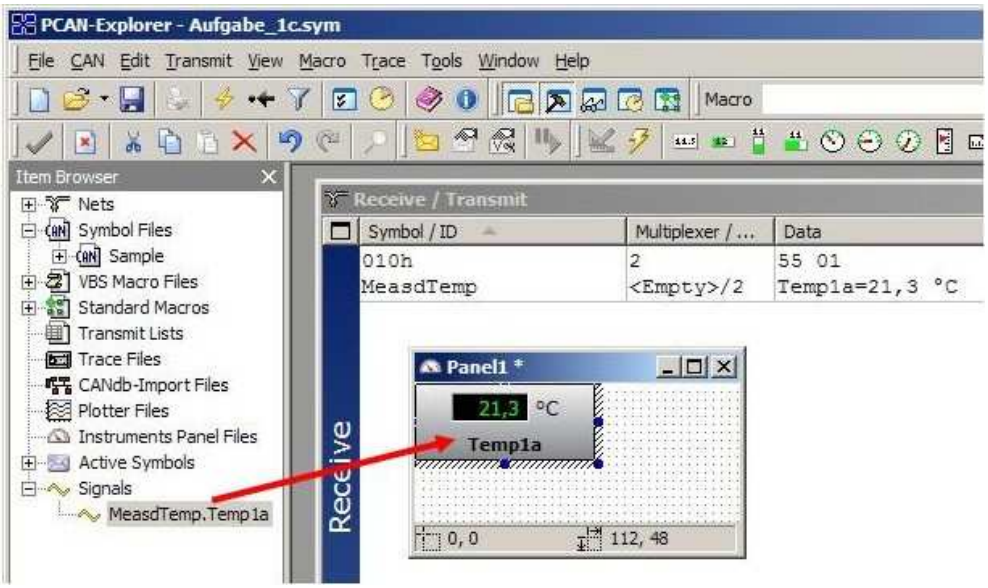
Reaktion:
Nach der manuellen Korrektur des Symbolfiles (mit Abspeichern und erneutem Apply) wird im Feld "Data" der aktuelle Messwert in °C angezeigt.

Lösungsschritte zu Aufgabe 1 e

Information:
Um als Besitzer eines PCAN-Explorers (IPES-005028) den Messwert noch grafisch aufzubereiten, kann ein sog. Instrumenten-Panel angelegt werden. Dazu muss aber das entsprechende PCAN-Explorer-AddIn "Instruments Panel" (IPES-005088) installiert sein.

Aktion:
Den Menüpunkt "Tools"-> "Instruments Panel" -> "Create Value Indicator". auswählen.
Reaktion:
Es wird ein leeres Panel mit einem digitalen Anzeigeinstrument dargestellt.

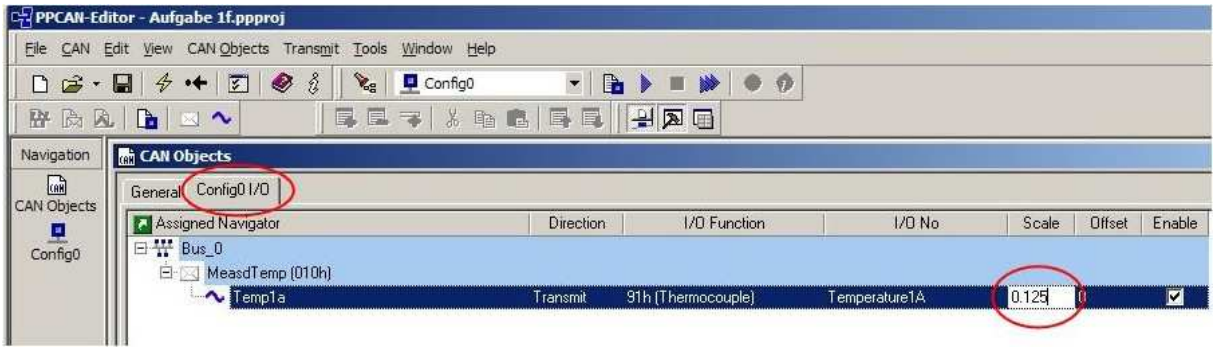
Aktion:
Diesem Instrument muss lediglich noch eine Datenquelle zugewiesen werden.
Dies geht per Drag und Drop: Man "zieht" (drag) mit der linken Maustaste die Variable "Temp1a" aus dem ItemBrowser über das Instrument und lässt sie dort fallen (drop).
Reaktion:
Das Instrument zeigt nun den Namen der Variablen und erhält die passende Formatierung.



Das Panel sollte auf dem PC abgespeichert werden ("Aufgabe 1e.ipf"), sinnvollerweise in der Nähe des dazu passenden Symbolfiles und der Konfiguration (z. B. ein sog. Projektverzeichnis).

Lösungsschritte zu Aufgabe 1 f

Information:
Die Auflösung 1/16 °C ist zwar die maximal mögliche, ihre Verwendung ist oft aber praxisfern. Stattdessen könnte z. B. in 0,5° Schritten gemessen werden. Dazu genügt die Manipulation des Scale-Wertes (/16 und *2), also eine Multiplikation mit Faktor 0,125.
Information:
Natürlich wird hardwareseitig weiterhin in 1/16°C gemessen, lediglich die Skalierung der übertragenen Daten wird verändert.
Aktion:
Im Kartei-Tab "Config0 I/O" das CAN-Signal "Temp1a" anklicken und den Wert Scale von 1 auf 0,125 ändern. Dadurch wird die Schrittzahl pro Grad auf 16 * 0,125 = 2 reduziert



Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe abgeschlossen..
Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 1f" auf dem PC gespeichert.
Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

Information:
In der CAN-Botschaft braucht man jetzt nur noch 1/8tel des Wertebereichs, man könnte also 3 bit einsparen: Die Variable "Temp1a" wäre nur noch 13bit breit (statt 16).
Information:
Entsprechend wäre dann ggf. das Symbolfile aus Aufgabe 1c abzuändern (und unter neuem Namen abzuspeichern):

```
Picture=aaaaaaaa aaaaaaaa
```

ändert sich in

```
Picture=aaaaaaaa ---aaaaa
```

und

```
a=Temp1a        signed  /u:°C /f:0,625
```

ändert sich in

```
a=Temp1a        signed  /u:°C /f:0,5
```

.

Lösungsschritte zu Aufgabe 1 g

Eine Temperatur in ganzen ° Fahrenheit anzuzeigen, ist ebenfalls durch richtige Verwendung von Scale und Offset möglich.

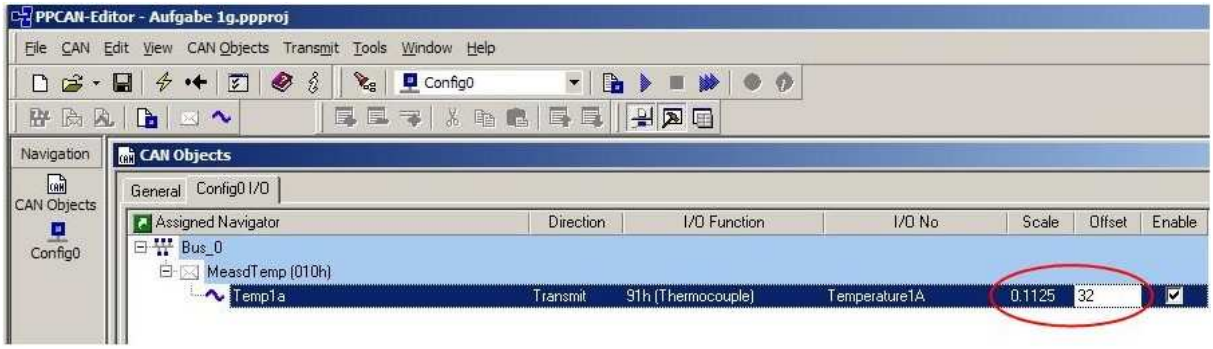
Information:
Das Gerät misst intern nach wie vor in 1/16° Celsius. Der Umrechnungsfaktor ist wie folgt:

$$T_{Fahrenheit} = ((T_{Celsius} \times 9) / 5) + 32$$

Daraus errechnet sich ein Scale-Wert von 1/16 *9 /5 = **0,1125**.
Der Offset-Wert von **32** wird 1/1 übernommen, da die Übertragung auf volle Grad umgerechnet wurde.

Information:
Bei z. B. ¼ Grad Auflösung müsste sowohl bei Scale als auch bei Offset ein Faktor 4 mitgerechnet werden (also Scale: 0,1125 *4 = **0,45** und Offset. 32 *4 = **128** eingetragen werden).

Aktion:
Im Kartei-Tab "Config0 I/O" das CAN-Signal "Temp1a" anklicken und den Wert "Scale" von 1 auf 0,1125 ändern, dann den Wert "Offset" von 0 auf 32 ändern.



Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe bereits abgeschlossen..

Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 1g" auf dem PC gespeichert.

Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

Information:
Entsprechend wäre dann auch ggf. das Symbolfile aus Aufgabe 1c abzuändern (und unter neuem Namen "Aufgabe 1g.sym" abzuspeichern):

```
a=Temp1a        signed  /u:°C /f:0,0625
```

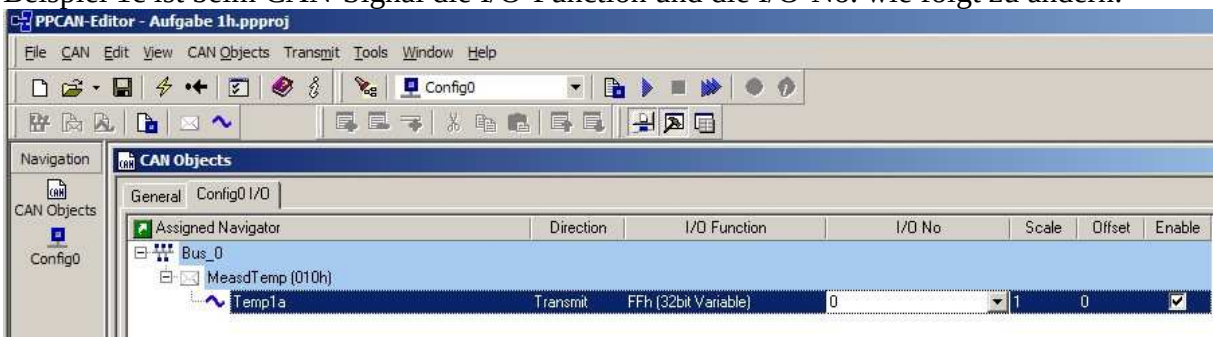
ändert sich in

```
a=Temp1a        signed  /u:°F
```

Lösungsschritte zu Aufgabe 1 h

Information:
Anwendungsbeispiele für Kennlinien: Der Zeiger für eine Kühlmittel-Temperaturanzeige soll "beruhigt" werden, z. B. zwischen 80° und 105°C immer den Wert 90°C anzeigen.
Oder: Der Vorlauf eines Tachometers soll (abgestuft) mit der Geschwindigkeit wachsen.

Da man auf dem Schreibtisch aber nicht mit hohen Temperaturen oder Geschwindigkeiten experimentieren soll, wurde ein anderes (zugegeben praxisferneres) Beispiel gewählt, das zumindest den Mechanismus begreiflich macht.
Definition (willkürlich):
Der übertragene Messwert (=Temperatur) des Sensors von Port 1a soll im Bereich 0..50°C invertiert werden.
Information:
Die gemessene Temperatur wurde bisher direkt in das CAN-Signal geleitet und ggf. per Scale und Offset etwas formatiert. Bei komplexeren Umrechnungen muss ein Funktionsblock aus der Firmware herangezogen werden.
In diesem Fall einer, der den Messwert über eine Kennlinie führt (also jedem X-Wert einen neuen Y-Wert zuweist) und das Ergebnis in eine 32bit-Variable schreibt. Deren neuer Inhalt (der transformierte Messwert) kann jetzt in das CAN-Signal eingesetzt werden.
Aktion:
Beginnend mit dem letzten Schritt, wird in der CAN-Datenbasis nicht mehr direkt der Messwert von Port 1a gesendet, sondern der Inhalt der 32bit-Variablen #0. Basierend auf dem Beispiel 1c ist beim CAN-Signal die I/O-Function und die I/O-No. wie folgt zu ändern:



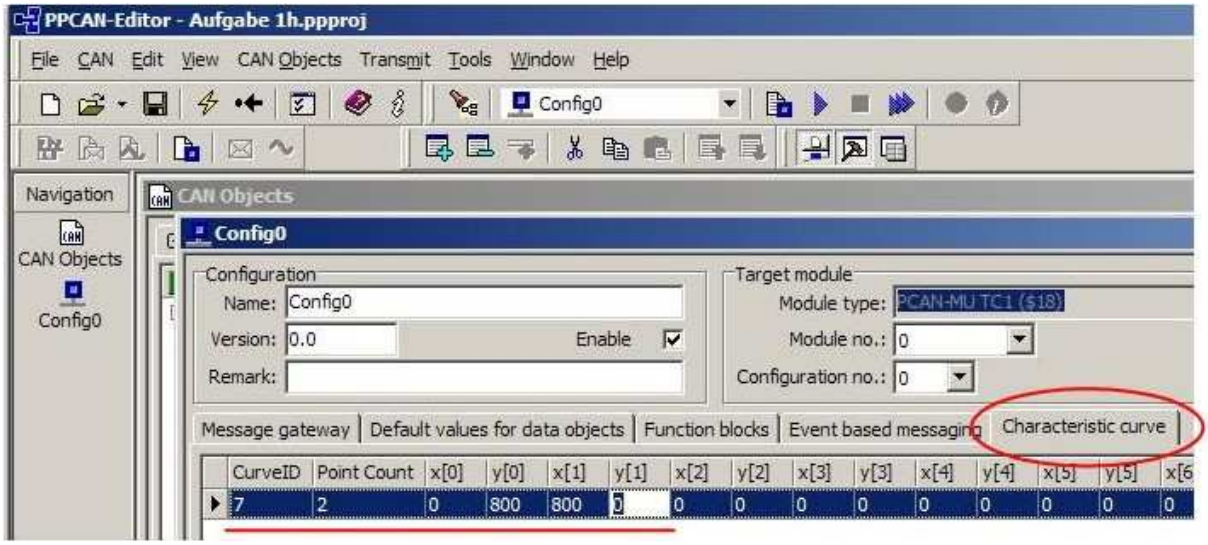
I/O-Function: "FF 32bit-Variable"
I/O-Number: 0

Information:
Als nächstes wird eine fallende Kennlinie (z. B. Nummer #7) eingerichtet, die 2 Punkte hat:

	X	Y
Punkt 1:	X = 0	Y = 800 (entspricht 50°C)
Punkt 2.	X = 800	Y = 0 (entspricht 0°C)

Aktion:
Dazu am linken Bildrand im Navigations-Fenster die Konfiguration "Config0" anklicken.
Reaktion:
Es öffnet sich ein neues Fenster mit dem Titel "Config0". Darin können komplexere Verknüpfungen von Ressourcen vorgenommen werden, die über das bloße Zuweisen und Skalieren hinausgehen.

Aktion:
In diesem Fenster den Kartei-Tab "Characteristic curve" auswählen, mit der rechten Maustaste das Kontextmenü öffnen und "Add Record" auswählen.
Reaktion:
Es erscheint eine Tabellenzeile, die eine Kennlinie repräsentiert und mit Werten befüllt werden muss.

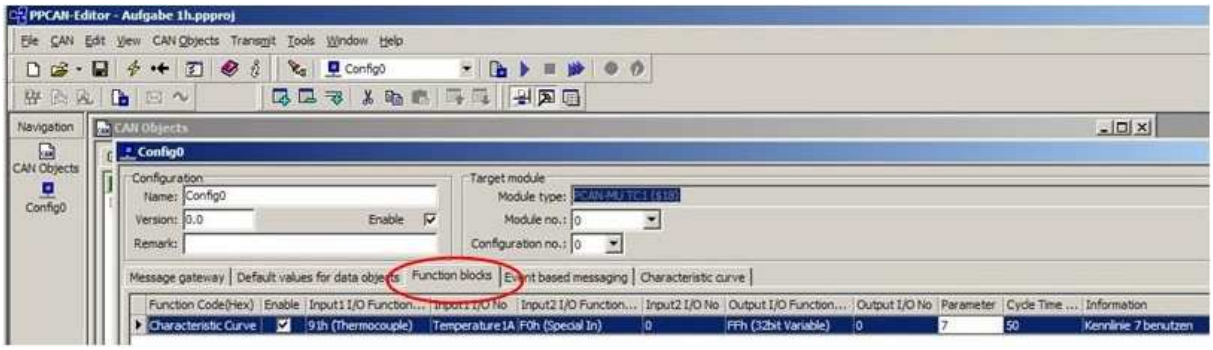


Curve-ID: 7 (eine willkürlich gewählte Nummer)
Point Count 2 (Anzahl der X/Y-Wertepaare)
Wertepaar 1: X=0 soll Y=800 ergeben
Wertepaar 2 X=800 soll Y=0 ergeben
(man erhält so eine fallende Gerade)
Information Eine Beschreibung, was diese Zeile tut (für später…)

Information:
Zuletzt muss dafür gesorgt werden, dass der Messwert zu den X-Koordinaten der Kennlinie gelangt, und der entsprechende Y-Wert in die 32bit-Variable #0 geschrieben wird, um auf dem CAN übermittelt zu werden. Dazu wird ein Funktionsblock benötigt, der die Handhabung der Kennlinie übernimmt.
Aktion:
Um einen neuen Funktionsblock anzulegen, wird im Kartei-Tab "Function blocks" mittels Kontextmenü (rechte Maustaste) "Add Record" durchgeführt.
Reaktion:
Es erscheint eine Tabellenzeile, die einen Funktionsblock repräsentiert und mit Werten befüllt werden muss.

Information:
Grundsätzlich hat jeder Funktionsblock zwei Eingänge (Operanden) und einen Ausgang (Ergebnis), jeweils bestehend aus I/O-Typ und I/O-Nummer, Weiterhin gibt es einen Hauptschalter (Enable) und eine Zykluszeit (wie oft wird das Ergebnis berechnet, in msec).

Aktion:
Der Funktionsblock wird gemäß folgendem Bild ausgefüllt:



Function Code:	Characteristic Curve (die Kennlinien-Bedien-Funktion)
Enable .	Ja, dieser Block soll aktiv sein
Eingang 1	"91-Thermocouple" und "Temperature1a" (die Quelle der X-Werte)
Eingang 2	"F0-Special In" und "none" (= nicht benutzt)
Ausgang	"FF-32bit-Variable" und "0" (das Y-Ergebnis in die 32bit-Variable #0)
Parameter	7 (die Nummer der bereits definierten Kennlinie)
Cycle time	z.B. 50 (die Umsetzung des Messwerts findet alle 50 msec statt)
Information	Eine Beschreibung, was diese Zeile tut (für später…)

Information:

Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe abgeschlossen..

Aktion:

Die Konfigurationsdatei wird als Projekt "Aufgabe 1h" auf dem PC gespeichert.

Aktion:

Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

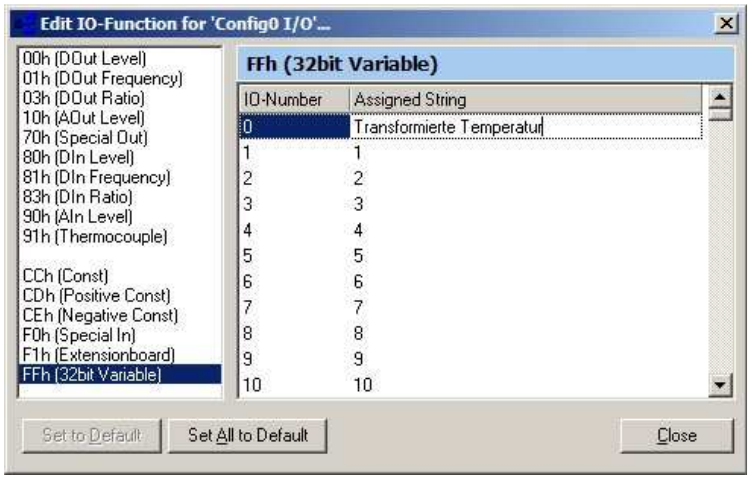
Reaktion:

Wenn man einen intakten Sensor in Port 1a steckt, wird eine Temperatur angezeigt, die *"50°C - des gemessenen Wertes"* entspricht. Wenn man den Sensor mit den Fingern erwärmt, fällt die angezeigte Temperatur entsprechend.

Tipp

Information:

Jeder 32bit-Variablen (und auch anderen Ressourcen) kann ein Klarname zugewiesen werden, indem man das Menü "Edit" -> "I/O-Function" aufruft.
Hier wird z. B. die 32bit-Variable #0 umbenannt in "Transformierte Temperatur".



Ab sofort kann man mit diesem Namen arbeiten, was Fehler durch Verwechslung einer Variablennummer erheblich reduziert.

Lösungsschritte zu Aufgabe 2 a

Wenn man die Konfigurationsdatei aus Aufgabe 1c erneut lädt, und den Temperatursensor vom Port 1a abzieht, wird ein Rohwert von 0x8000 übertragen.

Information:

Da Temperaturen unterhalb der Naturkonstante K_0 = -273,15°C nicht vorkommen können (dies entspräche Messwerten von 0x8000 bis 0xEEEC hex), wird dieser Wertebereich zum Darstellen von Fehlerbildern genutzt, z. B.:

0x8000: Keine Thermospannung gemessen: Sensor am Port nicht gesteckt oder Kabelbruch

0xE000: Karte nicht gesteckt bzw nicht erkannt

Die Messwerte vom Sensor können also anhand des Wertebereichs in gültig und ungültig unterschieden werden, und dies ist Teil der Lösung der gestellten Aufgabe.

Information:

Der mathematische Vergleich des Messwertes mit einer Konstanten (z.B. 0x8000) erfordert die Verwendung eines Funktionsblocks der Firmware.

Aktion:

Öffnen der Konfiguration aus Aufgabe 1c und speichern unter Aufgabe 2a.
Am linken Bildrand im Navigations-Fenster die Konfiguration "Config0" anklicken.

Reaktion:

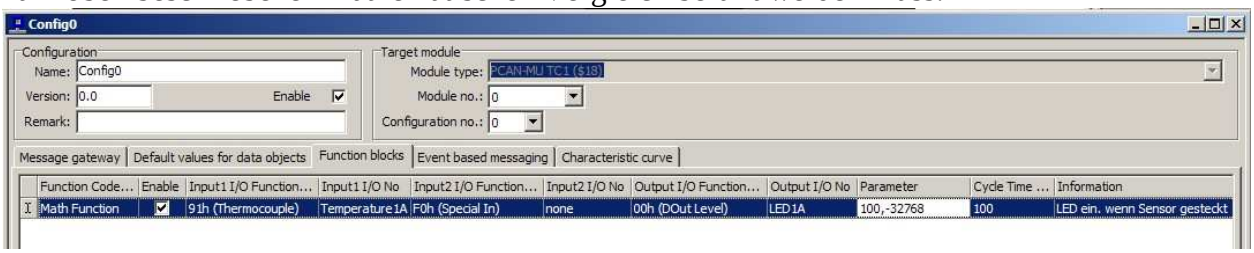
Es öffnet sich ein neues Fenster mit dem Titel "Config0" (wie von Aufgabe 1h bekannt).

Aktion:

Dort den Kartei-Tab "Function Blocks" auswählen, mit der rechten Maustaste das Kontext-menü öffnen und "Add Record" auswählen.

Reaktion:

Es erscheint eine Tabellenzeile, die einen Funktionsblock repräsentiert und mit den Werten zum oben beschriebenen mathematischen Vergleich befüllt werden muss.



Function Code:	MathFunction
Enable .	Ja, dieser Block soll aktiv sein
Eingang 1	"91-Thermocouple" und "Temperature1a"
Eingang 2	"F0-Special In" und "none" (= nicht benutzt)
Ausgang	"00-Dout Level" und "LED1a" (die Leuchtdiode zum Port 1a)
Parameter	Art der mathematischen Funktion: logischer Vergleich mit 0x8000: Als Funktion wird gewählt "Greater Const: Input1 > Const" und als Const. wird -32768 eingegeben. Das Ergebnis ist ein Bool'scher Wert (TRUE oder FALSE), der die LED steuert.
Cycle time	z.B. 100 (der logische Vergleich findet alle 100msec statt)
Information	Eine Beschreibung, was diese Zeile tut (für später…)

Information:

Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe abgeschlossen.

Aktion:

Die Konfigurationsdatei wird als Projekt "Aufgabe 2a" auf dem PC gespeichert.

Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

Reaktion:
Wenn man einen intakten Sensor in Port 1a steckt, leuchtet die LED,
wenn man den Sensor abzieht, geht die LED aus.

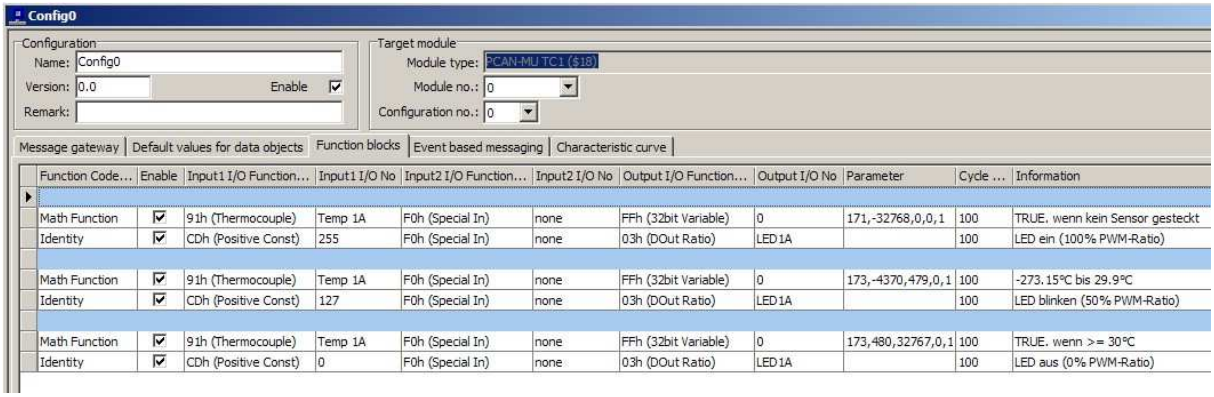
Lösungsschritte zu Aufgabe 2 b

Definition (willkürlich):
Die Leuchtdiode LED 1a soll blinken, wenn der Messwert 30°C unterschreitet.
Sie soll aus sein, wenn der Messwert mindestens 30°C erreicht.
Sie soll weiterhin dauerhaft leuchten, wenn kein Sensor gesteckt ist.

Information:
In diesem Beispiel wird ein bedingtes Ausführen von Funktionsblöcken demonstriert: hier sind 3 verschiedene Messbereiche zu prüfen: Wenn das Ergebnis TRUE ist, wird die Folgezeile ausgeführt, um die LED entsprechend zu setzen Bei Ergebnis FALSE wird die Folgezeile übersprungen. Die Anzahl der zu überspringenden Zeilen wird im "Parameter"-Feld der jeweiligen Bereichsprüfung angegeben (hier: jeweils 1). So lassen sich recht übersichtlich CASE-ähnliche Strukturen erstellen.

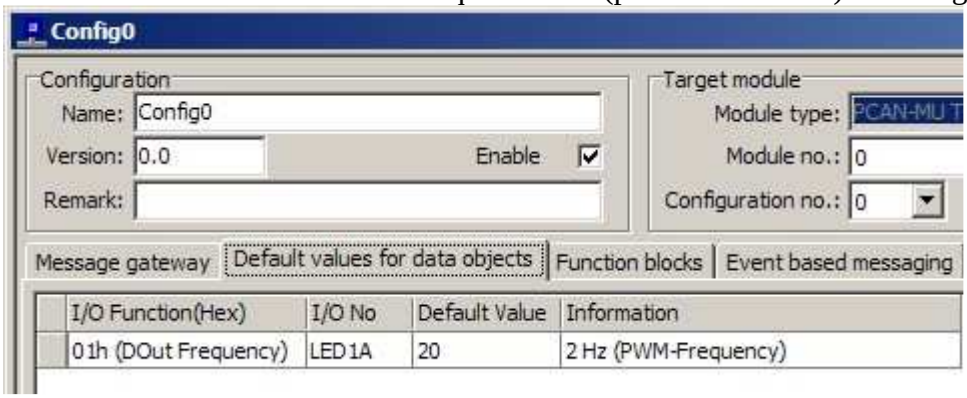
Information:
Blinken wird über die Funktion "PWM" realisiert. Es wird per Defaultwert eine feste PWM-Frequenz eingestellt, und das Tastverhältnis entscheidet über "aus", "blinken" oder "ein"-Zustand der Leuchtdiode 1a. Die Frequenz kann in 0,1 Hz-Schritten zwischen 0,1 und 10 Hz eingestellt werden, das Tastverhältnis (="Ratio") wird von 0 bis 255 eingestellt. Der Wert 127 entspricht dabei 50 %.

Aktion:
Es sind 3 Funktionsblöcke für die 3 Bereichsprüfungen notwendig, sowie jeweils ein zugeordneter Funktionsblock zum Setzen des PWM-Ratio (=der entsprechenden Blinkart).
Zur Gliederung einer Konfiguration können Trennzeilen eingefügt werden (Menü "Edit" -> "Insert Space Record").



Function Code...	Enable	Input1 I/O Function...	Input1 I/O No	Input2 I/O Function...	Input2 I/O No	Output I/O Function...	Output I/O No	Parameter	Cycle ...	Information
Math Function	☑	91h (Thermocouple)	Temp 1A	F0h (Special In)	none	FFh (32bit Variable)	0	171,-32768,0,0,1	100	TRUE. wenn kein Sensor gesteckt
Identity	☑	CDh (Positive Const)	255	F0h (Special In)	none	03h (DOut Ratio)	LED1A		100	LED ein (100% PWM-Ratio)
Math Function	☑	91h (Thermocouple)	Temp 1A	F0h (Special In)	none	FFh (32bit Variable)	0	173,-4370,479,0,1	100	-273.15°C bis 29.9°C
Identity	☑	CDh (Positive Const)	127	F0h (Special In)	none	03h (DOut Ratio)	LED1A		100	LED blinken (50% PWM-Ratio)
Math Function	☑	91h (Thermocouple)	Temp 1A	F0h (Special In)	none	FFh (32bit Variable)	0	173,480,32767,0,1	100	TRUE. wenn >= 30°C
Identity	☑	CDh (Positive Const)	0	F0h (Special In)	none	03h (DOut Ratio)	LED1A		100	LED aus (0% PWM-Ratio)

Abschließend ist noch die Blinkfrequenz 2 Hz (per Default-Wert) festzulegen.



Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe abgeschlossen..

Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 2b" auf dem PC gespeichert.

Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

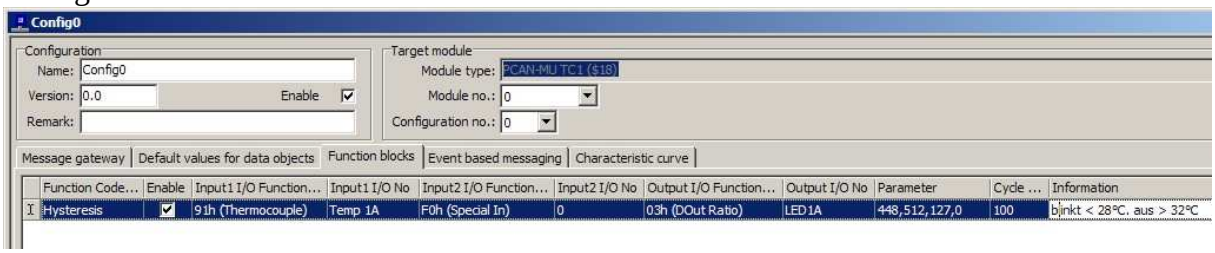
Reaktion:
Wenn kein Sensor in Port 1a steckt, leuchtet die LED.
Wenn ein Sensor gesteckt ist und die Messtemperatur unterhalb 30°C ist, blinkt die LED.
Steigt die Temperatur auf mindestens 30°C, geht die LED aus..

Lösungsschritte zu Aufgabe 2 c

Definition (willkürlich):
Die Leuchtdiode LED1a soll blinken, bis der steigende Messwert 32°C erreicht.
Sie soll aus bleiben, bis der fallende Messwert 28°C unterschreitet, darunter wieder blinken.

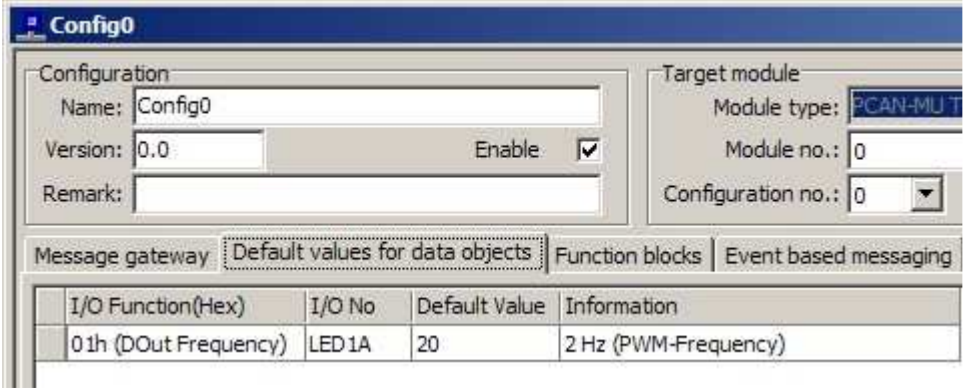
Information:
Blinken wird über die Funktion "PWM" realisiert. Es wird per Defaultwert eine feste PWM-Frequenz eingestellt, und das Tastverhältnis entscheidet über "aus" oder "blinken"-Zustand der Leuchtdiode 1a.

Aktion:
Öffnen der Konfiguration aus Aufgabe 1c und speichern unter Aufgabe 2c.
Einfügen eines Funktionsblocks und Ausfüllen der Felder.



Function Code: Hysterese
Input I/O-Function: 90-Thermocouple (Auswerten der Meßtemperatur)
Input I/O-Number: Temp1a
Output I/O-Function: PWM-Ratio (Ergebnis wirkt auf Ratio der LED)
Output I/O-Number: LED1a
Hysterese-Parameter: 448 (= 28°C), 512 (= 32°C), 127 (= 50 % Ratio), 0 (= LED aus)

Abschließend ist noch die Blinkfrequenz 2 Hz (per Default-Wert) festzulegen.



Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe bereits abgeschlossen..

Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 2c" auf dem PC gespeichert.

Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

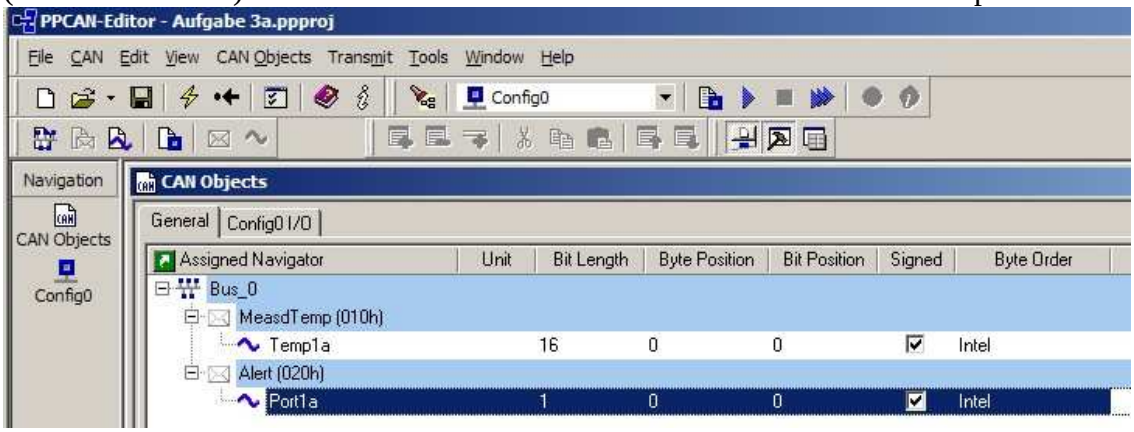
Reaktion:
Erwärmen des Sensors: Das Blinken wird erst oberhalb 32°C beendet.
Loslassen/Abkühlen des Sensors das Blinken beginnt erst wieder unterhalb 28°C.

Lösungsschritte zu Aufgabe 3 a

Information:
Im folgenden Beispiel (basierend auf Aufgabe 1c) soll eine separate CAN-Botschaft gesendet werden, wenn der Messwert einen bestimmten Bereich verläßt.
Dazu muss zunächst die CAN-Datenbasis um eine zusätzliche Botschaft erweitert werden
Definition (willkürlich):
Senden einer CAN-Botschaft "Alert" mit der ID 0x020, Länge 1 Byte auf dem Bus "Thermo_500k", aber nur wenn Sensor an Port 1a den Bereich 28..33°C verläßt,
Diese Botschaft enthält ein Signal "ErrFlag".
Aktion:
Im Kartei-Tab "General" zunächst die globale CAN-Datenbasis erweitern (um die Hülle der neuen Alarm-Botschaft). Dazu mit dem Kontextmenü (Mausklick rechts) "Add a new Symbol" ausführen und die neue Zeile entsprechend befüllen.

Symbolname: "Alert"
ID: 20h
DLC: 1
Extended: Nein, es genügt eine 11bit-ID
Enabled: Ja
RTR. Nein, die Botschaft soll -nicht- nur auf Anforderung gesendet werden

Aktion:
Innerhalb der CAN-Botschaft ein neues Signal anlegen Dazu mit dem Kontextmenü (Mausklick rechts) "Add a new Variable" ausführen und die neue Zeile entsprechend befüllen.

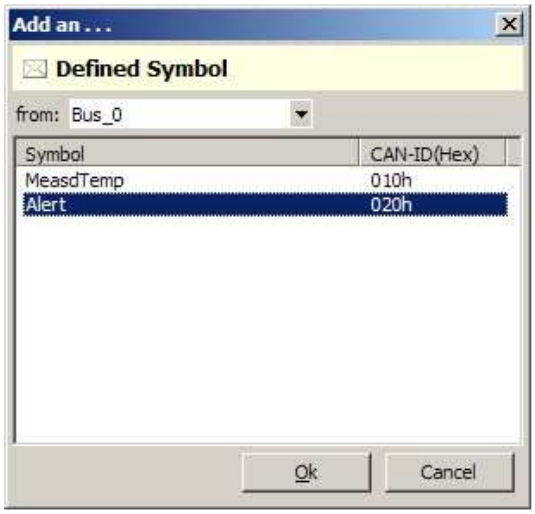


Variablenname:	Port1a
Unit:	-
Bit length	1
StartByte	0
StartBit	0
Signed	Ja (spielt keine Rolle bei 1-Bit-Werten)
Byte Order	Intel-Format (spielt keine Rolle bei 1-Bit-Werten)

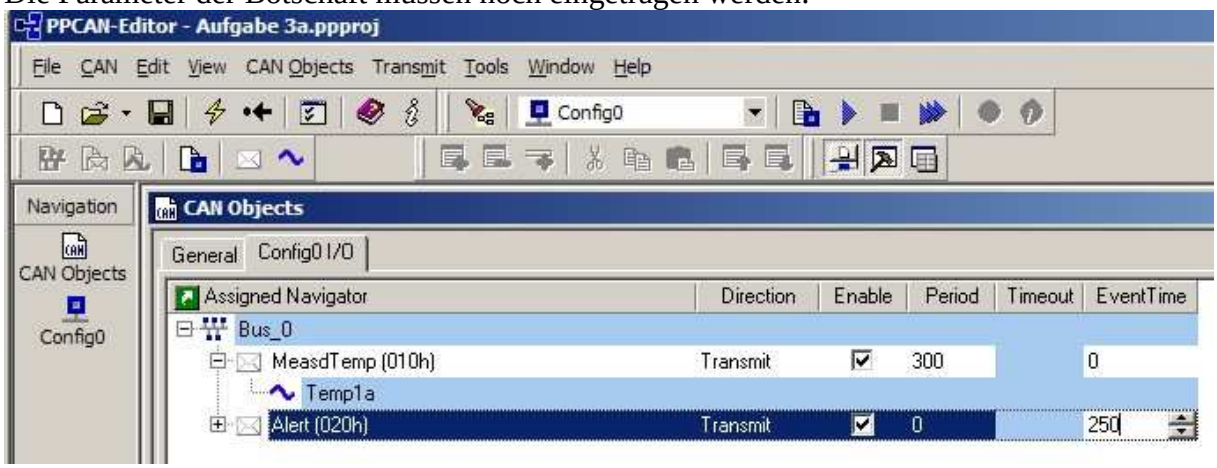
Information:
Die leere Hülle der neuen CAN-Botschaft ist damit festgelegt Sie muss als nächstes in die Konfiguration "Config0" importiert und dort mit Daten befüllt werden.

Aktion:
Dazu den Kartei-Tab "Config0-I/O" anklicken und dort im Kontextmenü (Rechte Maustaste) "Add defined Symbol" ausführen.

Reaktion:
Es wird gefragt, welches der definierten Symbole hier eingefügt werden soll: In diesem Fall die "Alert"-Botschaft inklusive des CAN-Signals "Port1a".



Aktion:
Die Parameter der Botschaft müssen noch eingetragen werden:



Direction: Transmit (Das MU-TC1 soll Sender sein)

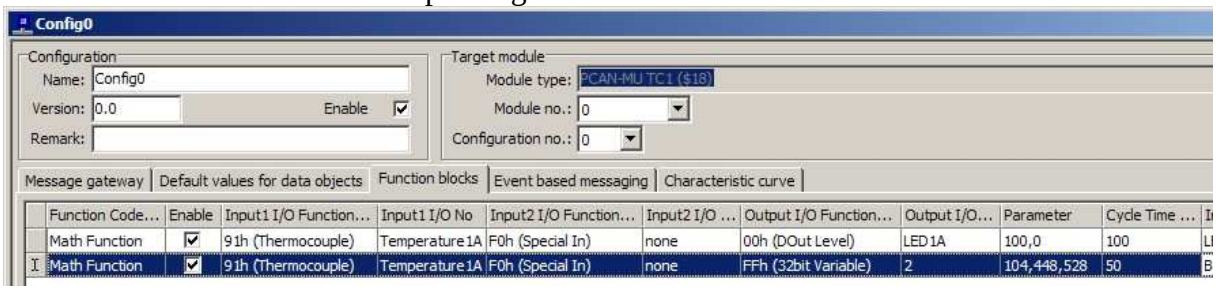
Enable: Ja, diese Botschaft soll übermittelt werden
Period: 0 (Kein zyklisches Senden, sondern nur im Alarmfall)
Event Time: 250msec (wenn der Alarm statisch anliegt, ist dies der Mindestabstand zwischen zwei aufeinanderfolgenden Alert-Botschaften)

Aktion:
Die Parameter des CAN-Signals müssen noch eingetragen werden:

I/O-Function: "FF 32bit-Variable" (Datenquelle: Eine Variable, die das Error-Flag enthält)
I/O-Number: 2 (die Variablen-Nummer, ein Name wäre sicherlich besser).
Scale: 1
Offset: 0
Enable: Ja, dieses Signal (innerhalb der Botschaft) soll verwendet werden.

Information:
Es fehlen jetzt noch zwei Schritte zur Lösung:
Es muss zyklisch geprüft werden, ob die an Port 1a gemessene Temperatur den Wertebereich verlassen hat (Bereichsprüfung mit Bool'schem Wert als Ergebnis), und abhängig davon muss das Senden der Botschaft "Alert" getriggert werden.

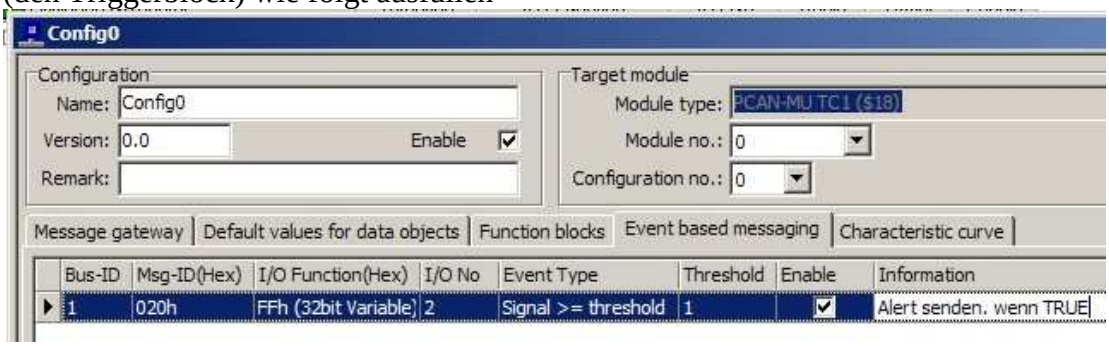
Aktion:
Zur Bereichsprüfung ist die Verwendung eines Funktionsblocks notwendig. Um diesen zu erstellen, muss am linken Bildrand im Navigations-Fenster die Konfiguration "Config0" angeklickt werden.
Reaktion:
Es öffnet sich das bereits bekannte Fenster mit dem Titel "Config0".
Aktion:
Dort den Kartei-Tab "Function Blocks" auswählen, mit der rechten Maustaste das Kontextmenü öffnen und "Add Record" auswählen.
Reaktion:
Es erscheint eine Tabellenzeile, die einen Funktionsblock repräsentiert und mit den Werten zur oben beschriebenen Bereichsprüfung befüllt werden muss.



Function Code: MathFunction

Enable .	Ja, dieser Block soll aktiv sein
Eingang 1	"91-Thermocouple" und "Temperature1a"
Eingang 2	"F0-Special In" und "none" (= nicht benutzt)
Ausgang	"FF 32bit-Variable " und #2 (willkürlich gewählt)
Parameter	Art der mathem. Funktion: TRUE, wenn ausserhalb fester Grenzen: Untergrenze 448 (=28°C), Obergrenze 528 (=33°C)
Cycle time	z.B. 50 (der logische Vergleich findet alle 50msec statt)
Information	Eine Beschreibung, was diese Zeile tut (für später…)

Aktion:
Abschließend soll anhand des Bool'schen Wertes in der Variablen #2 eine CAN-Botschaft getriggert werden. Dazu den Kartei-Tab "Event based Messaging " anklicken, mit der rechten Maustaste das Kontextmenü öffnen und "Add Record" auswählen. Die neu entstandene Zeile (den Triggerblock) wie folgt ausfüllen



Bus-ID:	#1 (Es ist im General-Tab nur dieser Bus definiert)
Msg-ID:	0x020 (diese CAN-Botschaft soll im Alarm-Fall gesendet werden)
I/O-Function:	Die 32bit-Variable #2, die das Ergebnis des Bereichschecks enthält
Event Type:	Wenn das Ergebnis >= 1 (also TRUE) ist
Threshold:	Die erwähnte 1 (= TRUE)
Enable:	Ja, der Trigger soll wirksam sein
Information	Eine Beschreibung, was diese Zeile tut (für später…)

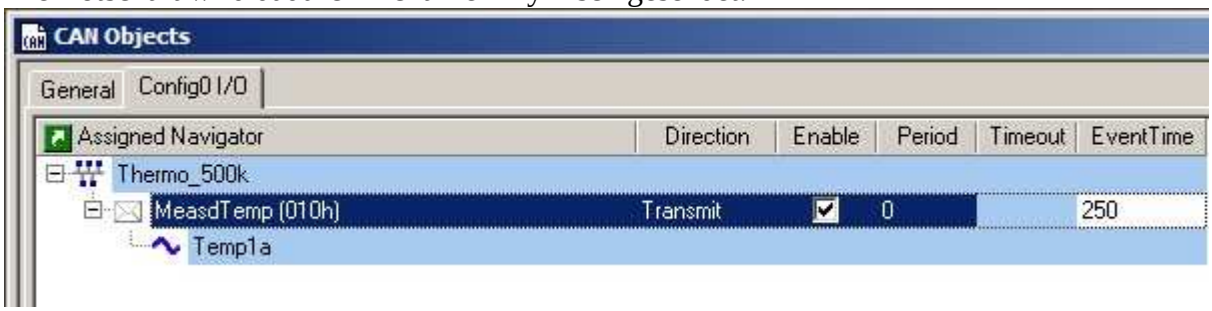
Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe abgeschlossen.
Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 3a" auf dem PC gespeichert.
Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).
Reaktion:
Bei einer Raumtemperatur unterhalb 28°C wird im Abstand von 250msec die Botschaft "Alert" (ID=0x020) gesendet. Wenn man den Sensor mit dem Finger erwärmt, bleibt die Botschaft ab 28°C stehen, bis 33 °C überschritten sind. Dann wird erneut die Botschaft gesendet, d.h. es gibt -keinen- Alarm zwischen 28 und 33°C.

Information:
Je nach Jahreszeit / Temperatur können andere Temperaturschwellen sinnvoll sein.
Information:
Man könnte in der Alert-Botschaft auch die Meßtemperatur übertragen, oder ein Oberhalb-/Unterhalb-Flag.

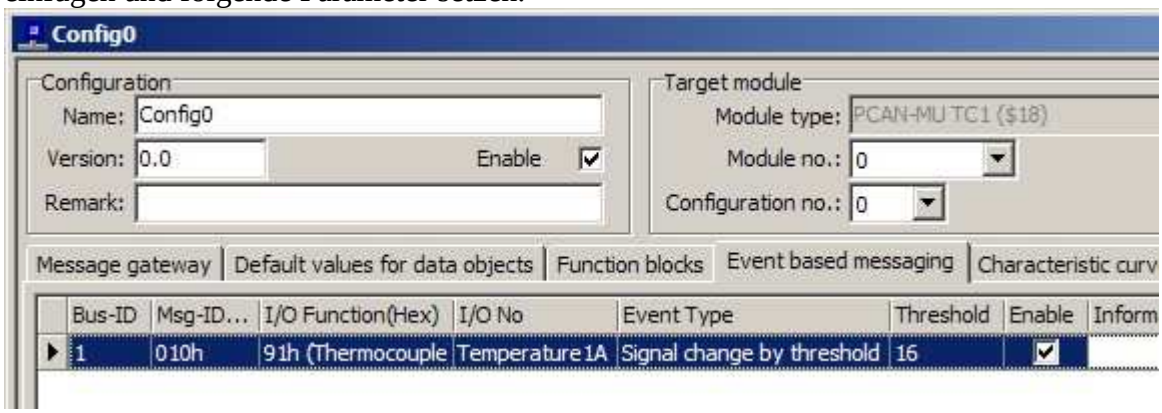
Lösungsschritte zu Aufgabe 3 b

Information:
Im folgenden Beispiel (basierend auf Aufgabe 1c) soll die CAN-Botschaft "MeasdTemp" nur noch dann gesendet werden, wenn sich die gemessene Temperatur um mindestens 1° Celsius verändert hat.
Dazu muss die Botschaft in der CAN-Datenbasis modifiziert und ein entsprechender Sende-Trigger definiert werden.

Aktion:
Im Kartei-Tab "Config0 I/O" zunächst die CAN-Botschaft "MeasdTemp" so umstellen, dass die Sendeperiode auf 0 und stattdessen ein Mindestabstand von 250msec gesetzt wird.
Die Botschaft wird dadurch nicht mehr zyklisch gesendet.



Damit die Botschaft dennoch hin und wieder gesendet wird, muss noch der Auslöser definiert werden. Dazu im Kartei-Tab "Event based messaging" mit "Add record" eine neue Zeile einfügen und folgende Parameter setzen.



Bus-ID: #1 (Es ist im General-Tab nur dieser Bus definiert)
Message-ID: 0x010 (die Temperatur-Botschaft soll gesendet werden)
I/O-Function: "91-Thermocouple" (ein Temperatur-Messwert soll bewertet werden)
I/O_Number: "Temp1a" (und zwar die gemessene Temperatur an Port 1a)
Event type: Wenn sich der Messwert um einen bestimmten Betrag (=Threshold) ändert
Threshold: 16 (entspricht 1°C, da immer in 1/16tel Grad Auflösung gemessen wird)
Enable: Ja, der Trigger soll wirksam sein
Information Eine Beschreibung, was diese Zeile tut (für später…)

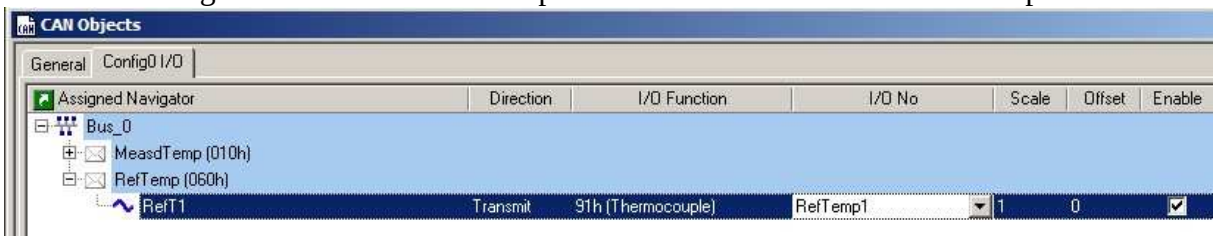
Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe abgeschlossen.

Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 3b" auf dem PC gespeichert.
Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

Reaktion:
Die CAN-Botschaft "MeasdTemp" wird jetzt nur noch gesendet, wenn sich der Sensor erwärmt/abkühlt oder gezogen/gesteckt wird.

Lösungsschritte zu Aufgabe 4 a

Information:
Die zur Temperaturmessung herangezogene Thermospannung an der Spitze des Meßsensors kann durch verschiedene Einflüsse merklich verfälscht werden.
Eine wesentliche Fehlerquelle sind die metallischen Kontakte der 8 Anschlußports. Hier entstehen parasitäre Thermospannungen z.B. durch Lötstellen etc., deren Einfluß mit zunehmender Umgebungstemperatur wächst.
Zur Kompensation wird die Temperatur an diesen Kontaktstellen gemessen und das mathematische Äquivalent einer entgegengesetzten Spannung in den Messwert eingerechnet.
Die Temperatur an den Kontaktstellen (sog. **Referenztemperatur**) kann ausgelesen werden.
Definition (willkürlich):
Senden einer CAN-Botschaft "RefTemp" mit der ID 0x060, Länge 8 Bytes auf dem Bus "Thermo_500k", Diese Botschaft enthält das Signal "RefT1", das die Referenztemperatur der linken Karte (Steckplatz #1) enthält.
Aktion:
Öffnen der Konfiguration aus Aufgabe 1c und speichern unter Aufgabe 4a.
Anlegen einer neuen Botschaft "RefTemp" im General-Tab (wie bereits bekannt)
Anlegen eines neuen CAN-Signals (13Bit breit) im General-Tab (wie bereits bekannt)
Importieren der Botschaft inkl. des Signals in die Konfiguration (wie bereits bekannt)
Befüllen des Signals mit der Referenztemperatur 1 anstelle der Meßsensor-Temperatur 1a.



Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe bereits abgeschlossen..
Ein versierter PPCAN-Programmierer braucht ca. 40 Sekunden zum Erstellen der Konfiguration.
Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 4a" auf dem PC gespeichert.
Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

Reaktion:
Es wird in der Botschaft "RefTemp" (mit der ID 0x060) die Temperatur übertragen, die an den Sensorports 1a und 1b gemessen wird. Sie sollte etwa der Raumtemperatur entsprechen.

Lösungsschritte zu Aufgabe 4 b

Information:
Die Modul-ID ist ein 4bit-Wert, der ab Werk auf 0 eingestellt ist und im Innern des MU-TC1 per DIP-Schalter verändert werden kann. Sie hat verschiedene Funktionen, z.B. wählt sie aus einer Datei mit mehreren Konfigurationen die der Schalterstellung entsprechende aus.
Bei unerklärlichem Verhalten einer neuen Konfiguration ist einer der ersten Debug-Schritte das Feststellen der Modul-ID. Es könnte nämlich sein, dass man die eine Konfiguration erfolglos editiert, weil jedesmal die andere ausgeführt wird.

Definition (willkürlich):
Senden einer CAN-Botschaft "Diagnosis" mit der ID 0x080, Länge 8 Bytes auf dem Bus "Thermo_500k", Diese Botschaft enthält das Signal "ModuleId", das die derzeit eingestellte Modul-ID anzeigt.
Aktion:
Öffnen der Konfiguration aus Aufgabe 1c und speichern unter Aufgabe 4b.
Anlegen einer neuen Botschaft "Diagnosis" im General-Tab (wie bereits bekannt)
Anlegen eines neuen CAN-Signals (4Bit breit, unsigned) im General-Tab (wie bekannt)
Importieren der Botschaft inkl. des Signals in die Konfiguration (wie bereits bekannt)
Befüllen des Signals mit der Modul-ID anstelle der Meßsensor-Temperatur 1a.

I/O-Function: "F0-Special In" (Datenquelle: Eine von mehreren Statusinformationen)
I/O-Number: Module-Id (hier wurde der I/O-Nummer 16 bereits ein Name zugeordnet).
Scale: 1
Offset: 0
Enable: Ja.

Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe bereits abgeschlossen..
Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 4b" auf dem PC gespeichert.
Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

Reaktion:
Es wird in der Botschaft "Diagnosis" (mit der ID 0x080) die Modul-Id übertragen.

Information:
Eine Änderung der Modul-Id wird erst nach Reset des Geräts aktiv (z.B. Power Off/On).

Lösungsschritte zu Aufgabe 4 c

Information:

Das Controllerboard des MU-TC-1 kann 31 verschiedene Meßkartentypen erkennen. Der entsprechende Wert 0..31 kann gelesen und angezeigt werden. Derzeit sind definiert:

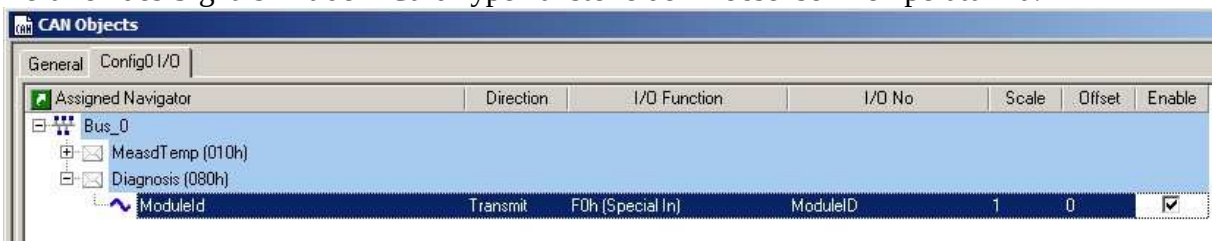
```
 0 = keine Karte gesteckt (leerer Slot)
15 = Messkarte für 2 Thermoelemente Typ "K" (grün) gesteckt
16 = Messkarte für 2 Thermoelemente Typ "J" (schwarz) gesteckt
17 = Messkarte für 2 Thermoelemente Typ "T" (braun) gesteckt
```

Definition (willkürlich):

Senden einer CAN-Botschaft "Diagnosis" mit der ID 0x080, Länge 8 Bytes auf dem Bus "Thermo_500k", Diese Botschaft enthält das Signal "CardType1", das den Typ der in Slot #1 erkannten Karte angibt.

Aktion:

Öffnen der Konfiguration aus Aufgabe 4b und speichern unter Aufgabe 4c.
Hinzufügen eines neuen CAN-Signals (5Bit breit, unsigned) im General-Tab (wie bekannt)
Importieren des Signals in die Konfiguration (wie bereits bekannt)
Befüllen des Signals mit dem CardType1 anstelle der Meßsensor-Temperatur 1a.



I/O-Function: "F1-ExtensionBoard" (Datenquelle: Eine von mehreren Statusinformationen)
I/O-Number: Slot 0 (hier wurde der I/O-Nummer 0 bereits ein Name zugeordnet).
Scale: 1
Offset: 0
Enable: Ja.

Information:

Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe bereits abgeschlossen..

Aktion:

Die Konfigurationsdatei wird als Projekt "Aufgabe 4c" auf dem PC gespeichert.

Aktion:

Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

Reaktion:

Es wird in der Botschaft "Diagnosis" (mit der ID 0x080) außer der Modul-Id (in Byte 0) noch der Kartentyp im Slot 0 (in Byte 4) übertragen.

Information:

Der folgende Tipp ermöglicht die Umwandlung des Byte 4 in Klartext.

Tipp

Information:

Im PCAN-Explorer kann eine Nummernfolge in Klartext umgewandelt werden.

Bezogen auf die Aufgabe 4c kann der Wertebereich 0..31 entsprechend zu Kartentypen dekodiert werden:

Im Symbolfile gibt es den Abschnitt ENUMS: hier können Aufzählungen definiert werden.

```
{ENUMS}
enum BoardType(15="K(green)", 16="J(black)", 17="T(brown)", 0="empty")
```

In der Botschaftsdefinition können die definierten ENUMs dann verwendet werden.

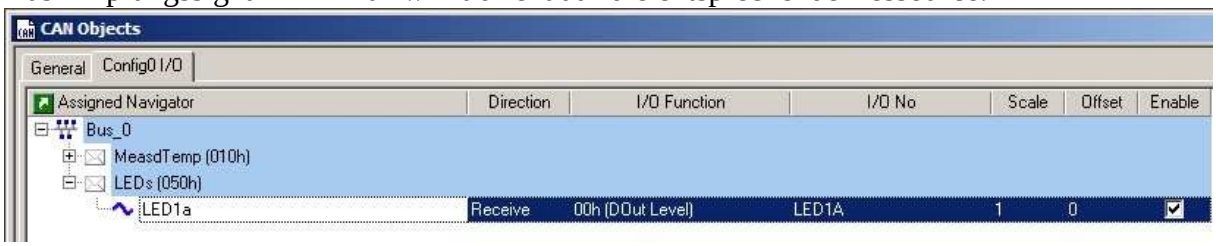
```
{RECEIVE}
[Diagnosis]
ID=80h
Picture=----aaaa -------- -------- -------- ---bbbbb -------- -------- --------
a=Module-ID  unsigned
b=Board-0    unsigned /e:BoardType
```

Lösungsschritte zu Aufgabe 4 d

Information:
Das Setzen der Leuchtdioden aufgrund von geräteinternen Entscheidungen wurde bereits in Aufgabe 2 gezeigt. Des Setzen der LEDs von außerhalb durch eine CAN-Empfangsbotschaft ist ebenfalls möglich.
Definition (willkürlich):
Empfangen einer CAN-Botschaft "LEDs" mit der ID 0x050, Länge 4 Bytes auf dem Bus "Thermo_500k", Diese Botschaft enthält das Signal "LED1a", das mit der LED für den Messfühler-Port 1a korrespondiert. Wenn die Empfangsbotschaft ausbleibt, soll die LED nach 1 Sekunde ausgehen
Aktion:
Wieder basierend auf dem Projekt 1c wird der CAN-Datenbasis eine Empfangsbotschaft hinzugefügt:
Öffnen der Konfiguration aus Aufgabe 1c und speichern unter Aufgabe 4d.
Anlegen einer neuen Botschaft "LEDs" im General-Tab (wie bereits bekannt)
Anlegen eines neuen CAN-Signals (1Bit breit) im General-Tab (wie bereits bekannt)
Importieren der Botschaft inkl. des Signals in die Konfiguration (wie bereits bekannt)
Die Botschaft LEDs ist als Empfangsbotschaft zu deklarieren und hat deshalb keine Sendeperiode.
Das TimeOut für die Botschaft beträgt 1000 msec., danach wird der Default-Wert wirksam.

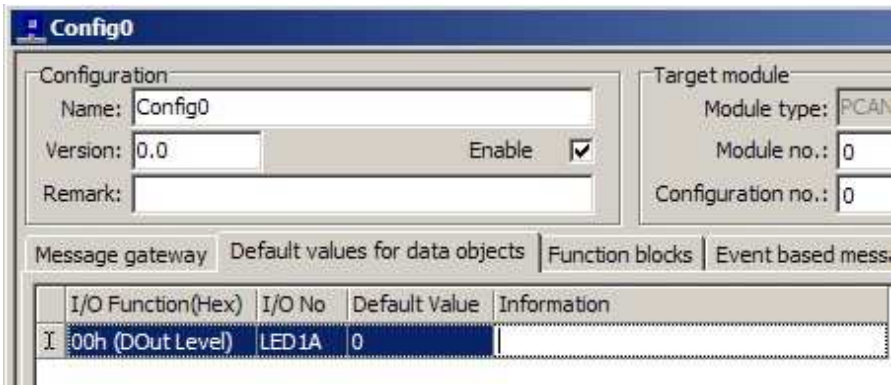
Direction: Receive (diese Botschaft kommt von außen und wird ausgewertet)
Enable: Ja, die Botschaft soll verwendet werden
Period: Sendezyklus, entfällt bei Receive-Botschaften
Timeout: 1000msec. (danach wird der Default-Wert angenommen)
EventTime: Mindestabstand, entfällt bei Receive-Botschaften

Das Empfangssignal "LED1a" wirkt direkt auf die entsprechende Ressource.



I/O-Function: "00-Dout Level" (eine der Leuchtdioden)
I/O-Number: "LED1a" (die Leuchtdiode zum Port 1a)
Scale: 1
Offset: 0
Enable . Ja, dieses Signal soll verwendet werden

Abschließend muss noch der Defaultwert für die Leuchtdiode gesetzt werden: Bei PowerOn und im Timeout-Fall soll die LED aus sein: Default = 0.



I/O-Function: "00-Dout Level" (eine der Leuchtdioden)
I/O-Number: "LED1a" (die Leuchtdiode zum Port 1a)
Default-Value: 0 (LED soll aus sein)
Information Eine Beschreibung, was diese Zeile tut (für später…)

Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe bereits abgeschlossen..
Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 4d" auf dem PC gespeichert.
Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

Aktion:
Es wird vom PCAN-Explorer (oder dem PCAN-View) eine CAN-Botschaft mit der ID 0x050 gesendet, in der Byte 0 Bit 0 =1 gesetzt ist.

Reaktion:
Die Leuchtdiode geht an.
Wird innerhalb 1 Sekunde eine weitere Botschaft gesendet, bleibt die Leuchtdiode an.
Wird für mehr als eine Sekunde die Botschaft nicht gesendet, tritt der Default-Wert in Kraft und die Leuchtdiode geht aus.

Information:
Eine entsprechende Sendebotschaft kann im PCAN-Explorer-Symbolfile definiert werden:

```
{SEND}
[Leds]
ID=50h
//       Byte 0   Byte 1   Byte 2   Byte 3   Byte 4   Byte 5   Byte 6   Byte 7
//      76543210 76543210 76543210 76543210 76543210 76543210 76543210 76543210
Picture=-------a -------- -------- --------
a=Led1a bit
```

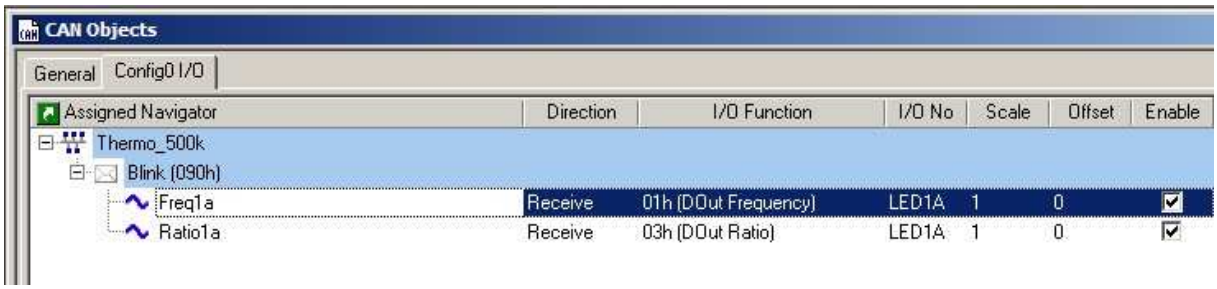
Lösungsschritte zu Aufgabe 4 e

Information:
Blinken von Leuchtdioden kann auch über eine PWM Funktion realisiert werden. Dazu stehen (wie bei PWM zu erwarten) die Parameter "Frequenz" und "Ratio" zur Verfügung.

	Bereich	**Auflösung**
Frequenz	0,1..10 Hz	0,1 Hz
Ratio (Puls-Pause-Verhältnis)	0..255 entspr 0..100%	1

Definition (willkürlich):
Empfangen einer CAN-Botschaft "Blink" mit der ID 0x090, Länge 2 Bytes auf dem Bus "Thermo_500k", Diese Botschaft enthält die beiden 8bit-Signale "Freq1a" und "Ratio1a", die direkt die LED1a beeinflussen.

Aktion:
Es wird eine neue Empfangsbotschaft inkl. der beiden Signale angelegt. Diese werden direkt auf die I/O-Funktionen "Frequency" und "Ratio" der LED 1a geschrieben.



Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe bereits abgeschlossen..
Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 4e" auf dem PC gespeichert.
Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

Aktion:
Es wird vom PCAN-Explorer (oder dem PCAN-View) eine 2 Bytes lange CAN-Botschaft mit der ID 0x090 gesendet, in der Byte 0 und Byte 1 mit entspr. Werten für Frequency und Ratio versorgt sind.

Reaktion:
Die Leuchtdiode blinkt entsprechend der eingestellten Werte.

Information:
Eine entsprechende Sendebotschaft kann im PCAN-Explorer Symbolfile zur "Aufgabe 4e" definiert werden:

```
{SEND}
 [Blink]
ID=90h
//       Byte 0   Byte 1   Byte 2   Byte 3   Byte 4   Byte 5   Byte 6   Byte 7
//      76543210 76543210 76543210 76543210 76543210 76543210 76543210 76543210
Picture=aaaaaaaa bbbbbbbb
a=Freq1a unsigned
b=Ratio1a unsigned
```

Information:
Für Besitzer eines PCAN-Explorers (Art.-Nr. IPES-005028): Eine bessere Bedienbarkeit der beiden Parameter wird durch ein Instrument Panel erreicht. Dazu muss jedoch das PCAN-Explorer-AddIn "Instruments Panel (Art.-Nr. IPES-005088) installiert sein.

Aktion:
Den Menüpunkt "Tools"-> "Instruments Panel" -> "Create vertical Slider Control" auswählen.

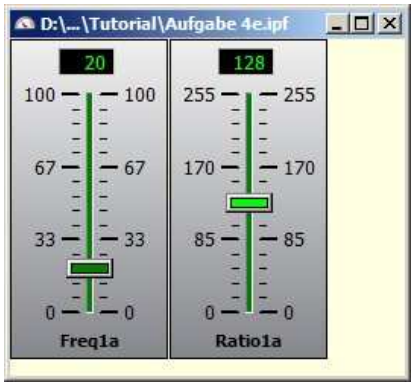
Reaktion:
Es wird ein leeres Panel mit einem senkrechten Schieberegler dargestellt.

Aktion:
Erneut den Menüpunkt "Tools"-> "Instruments Panel" -> "Create vertical Slider Control". auswählen.

Reaktion:
Im Instrument Panel erscheint ein zweiter Schieberegler, den man neben den ersten plaziert.

Aktion:
Den beiden Schiebereglern muss lediglich noch ein Sendeparameter zugewiesen werden. Dies geht per Drag und Drop: Man "zieht" (drag) mit der linken Maustaste die Variablen "Freq1a" und "Ratio1a" nacheinander über je einen Schieberegler und lässt sie dort fallen (drop).

Reaktion:
Die Schieberegler zeigen nun den Namen der Variablen und erhalten eine passende Formatierung (evtl. ist der Wertebereich des Frequency-Regler manuell auf "100".zu begrenzen. Dies kann im Kontextmenü des Reglers unter "Properties" eingestellt werden).



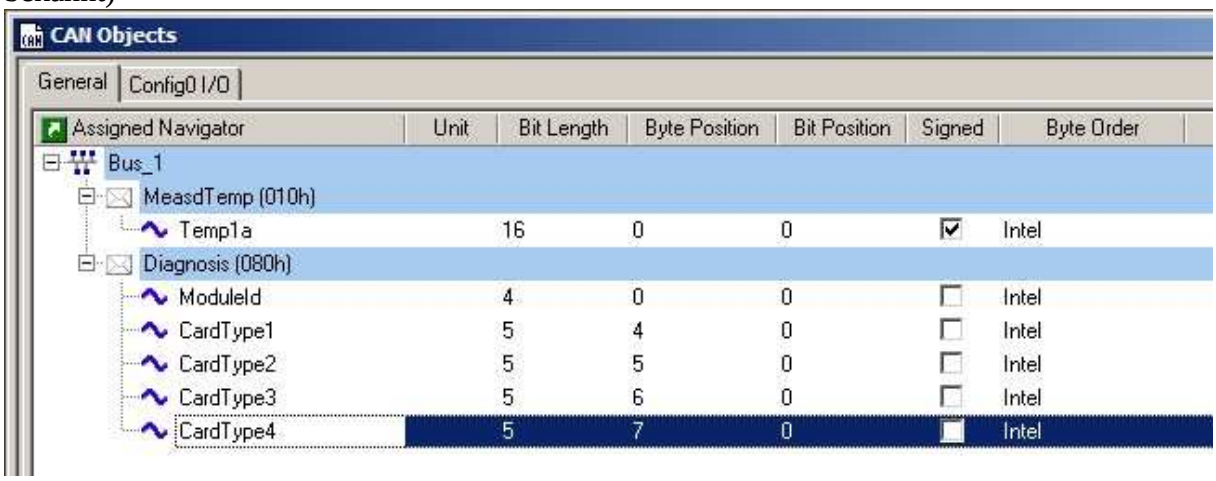
Aktion:
Das Panel sollte als "Aufgabe 4e.ipf" auf dem PC abgespeichert werden, sinnvollerweise in der Nähe des dazu passenden Symbolfiles und der Konfiguration (z.B. ein sog. Projektverzeichnis).

Lösungsschritte zu Aufgabe 5

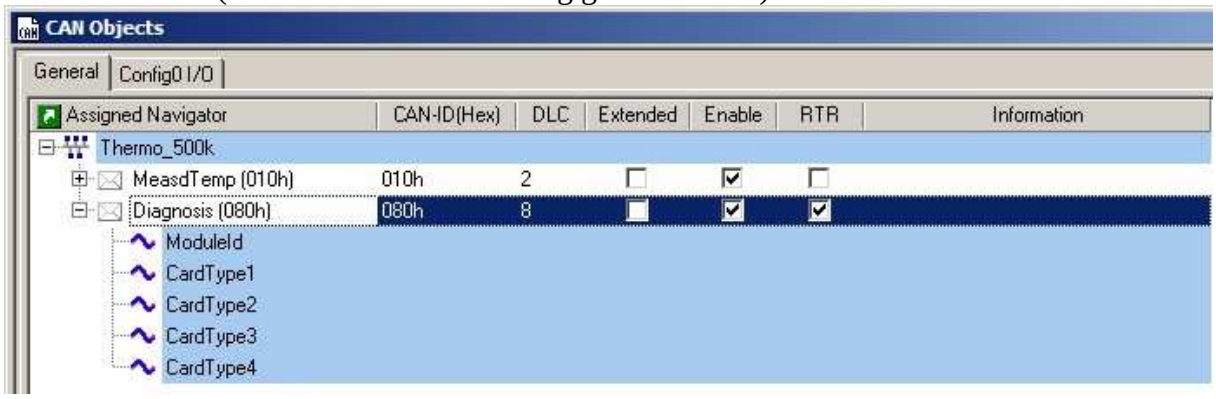
Information:
Eine Statusinformation soll nur auf Anfrage ausgegeben werden. Ausgehend von Aufgabe 4c sollen die Typen aller gesteckten Karten über CAN ausgegeben werden.

Definition (willkürlich):
Senden einer CAN-Botschaft "Diagnosis" mit der ID 0x080, Länge 8 Bytes auf dem Bus "Thermo_500k", aber nur auf Anforderung von extern: "Remote Transmission Request". Diese Botschaft enthält die Signale "CardType1" bis "CardType4", die den Typ der in den Slots erkannten Karten angibt.

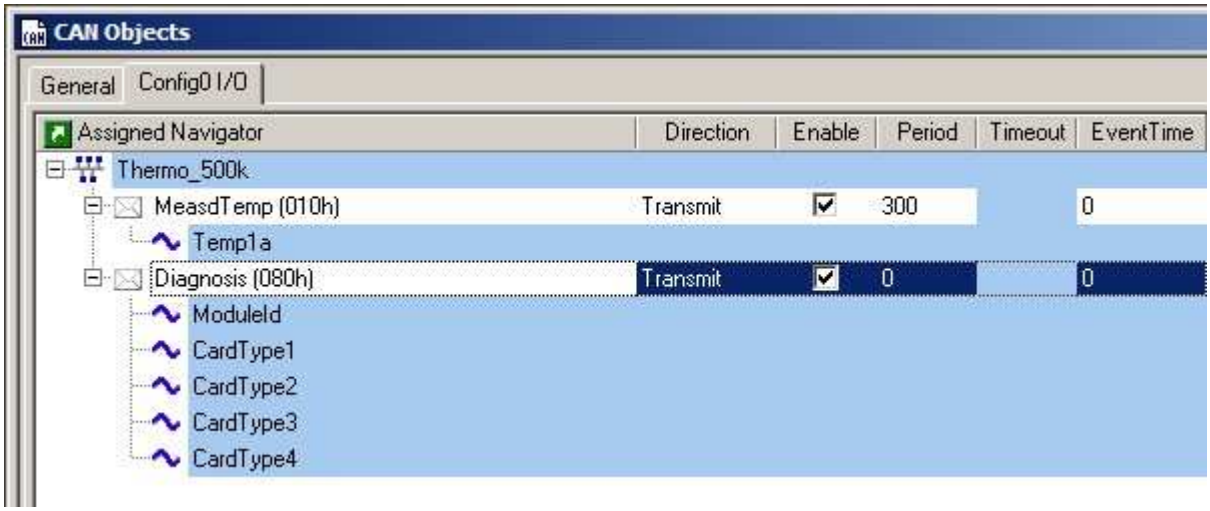
Aktion:
Öffnen der Konfiguration aus Aufgabe 4c und speichern unter Aufgabe 5.
Hinzufügen von 3 neuen CAN-Signalen (je 5Bit breit, unsigned) im General-Tab (wie bereits bekannt)



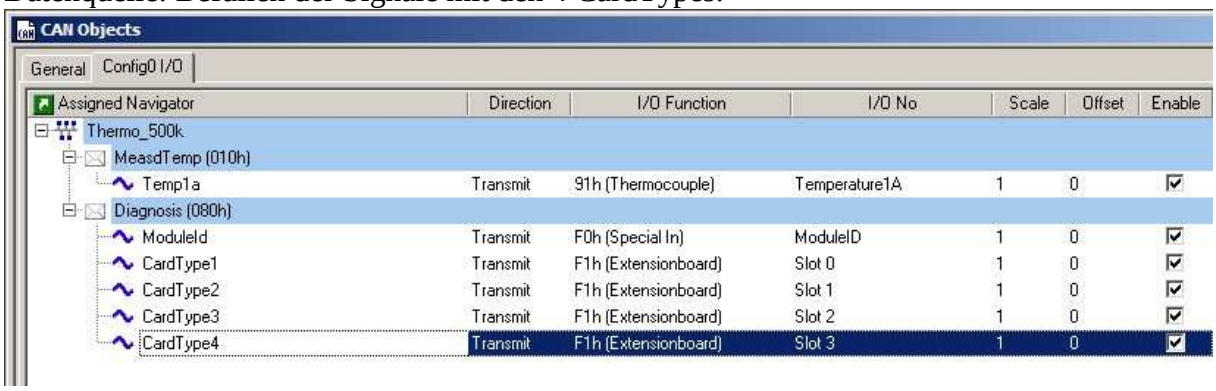
Umstellen der CAN-Botschaft auf "Transmission Request" durch Setzen des RTR-Häkchens in der Botschaft (damit nur auf Anforderung gesendet wird).



Importieren der Signale in die Konfiguration (wie bereits bekannt) und Setzen des Sendezyklus auf 0. Damit wird die Botschaft nicht mehr selbsttätig gesendet.



Datenquelle: Befüllen der Signale mit den 4 CardTypes.



Information:
Damit sind die Konfigurationsarbeiten zur Lösung dieser Aufgabe abgeschlossen..

Aktion:
Die Konfigurationsdatei wird als Projekt "Aufgabe 5" auf dem PC gespeichert.

Aktion:
Die Konfigurationsdatei wird über den CAN-Bus an das MU-TC1 übermittelt (Upload).

Aktion:
Es wird vom PCAN-Explorer (oder dem PCAN-View) eine CAN-Botschaft mit der ID 0x080 und der Länge DLC = 0 gesendet.

Reaktion:
Das MU-TC1 antwortet, indem es die definierte 8 Bytes lange Botschaft sendet, die u. a. die 4 Kartentypen enthält.

Aktion:
Im PCAN-Explorer-Symbolfile kann die Botschaft "Diagnosis" entsprechend ergänzt werden.

```
[Diagnosis]
ID=80h
//       Byte 0   Byte 1   Byte 2   Byte 3   Byte 4   Byte 5   Byte 6   Byte 7
//      76543210 76543210 76543210 76543210 76543210 76543210 76543210 76543210
Picture=----aaaa -------- -------- -------- ---eeeee ---fffff ---ggggg ---hhhhh
a=Module-ID unsigned
e=Board-0 unsigned /e:BoardType
f=Board-1 unsigned /e:BoardType
g=Board-2 unsigned /e:BoardType
h=Board-3 unsigned /e:BoardType
```

Referenzen und weiterführende Literatur

ThermoCouple MU-TC1 Hardware Handbuch

PPCAN-Editor Online-Hilfe

Referenzliste der Funktionsblöcke

PCAN-View Online-Hilfe
oder
PCAN-Explorer Online-Hilfe

PPCAN-Protokoll Referenz