Versuch 4 : I²C

In diesem Versuch ergänzen wir unser bisheriges Programm und verwenden die I2C-Schnittstelle. Über die I2C-Schnittstellle werden die Messdaten eines Temperatursensors ausgelesen. Außerdem wird über die I2C- Schnittstelle eine 4stellige Siebensegmentanzeige zur Visualisierung des Temperatur-Messwertes angesteuert.

I²C:
I2C (gesprochen "I-Quadrat-C" ist eine Zwei-Draht Kommunikationsschnittstelle. Zahlreiche Bausteine (Sensoren, Aktoren, Speicher, …) verwenden diese Schnittstelle und können damit an den STM32-Mikrocontroller angeschlossen werden. Die Schnittstelle ist ein serieller Bus und besteht aus einer Takt- und einer Datenleitung, sie erlaubt ca. 100 Teilnehmer. Jeder Teilnehmer hat eine 7-Bit Adresse. Nach erfolgreicher Adressierung können Daten bidirektional im synchronen Halbduplex-Verfahren mit 100 kBit/s ausgetauscht werden

Informieren Sie sich über I2C und über das Übertragungsprinzip. Eine kurze Beschreibung zum Thema I2C finden Sie in den Unterlagen zum Versuch sowie im Internet unter:

https://www.youtube.com/watch?v=6IAkYpmA1DQ

http://www.i2c-bus.org/

https://www.mikrocontroller.net/articles/I%C2%B2C

In diesem Versuch werden wir mit einer Bibliothek auf die I2C-Schnittstelle zugreifen, Sie brauchen dafür keine Kenntnisse über die I/O-Register des I2C-Teils des Prozessors (falls es Sie doch interessiert: siehe Seite 542 des Hardwaremanuals). Die I2C-Bibliothek besteht aus den Dateien I2C.c und I2C.h, es wird der I2C-Kanal 1 verwendet mit SCL an PortB6 und SDA an PortB7.

Am Ende sollten Sie ein funktionierendes Projekt haben, im dem alle wesentlichen Punkte der Versuche 2 bis 4 integriert sind!

Aufgabe 4_1 : I^2C- Temperatur

Als Temperatursensor wird der Baustein MCP9800 mit der Adresse 0x48 verwendet. Das Datenblatt finden Sie in den Unterlagen, für das Verständnis der Kommunikation mit dem Baustein ist das Kapitel 5 (Seite 13) wichtig.
Kurze Erklärung des Prinzips:
Der MCP9800 besitzt 4 Funktions-Register und ein Auswahl-Register (Registerpointer), das die Adresse (0-3) des ausgewählten Funktionsregisters enthält. Wenn man also auf eines der Funktionsregister zugreifen will, muss vorher der Registerpointer beschrieben werden.
Für uns interessant ist das Ambient Temperature Register (Register 0), ein 16-Bit Register, das im Highbyte die Temperatur in Grad Celsius und im Lowbyte die binären Nachkommastellen enthält (siehe Seite 16+17 im Datenblatt).
Zusätzlich wird noch das Config-Register verwendet (Register 1, Seite 18), hier werden in Bit 5 und 6 verschiedene Temperaturauflösungen eingestellt. Es kann zunächst die Voreinstellungen des Bausteins mit einer Auflösung von 0,5 C verwendet werden.

Aufgabe:
Verwenden Sie Ihr Programm aus dem letzten Versuch. Fügen Sie in das Projekt die C-Module I2C.c und I2C.h sowie die Module Temperatur (c und .h) ein.
Ergänzen Sie die Funktionen im Modul Temperatur.c so, dass die Temperatur ausgelesen werden kann und mit der Funktion set_TempRes die Auflösung geändert werden kann.

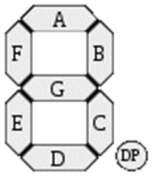
Die Execute-Funktion im Hauptprogramm soll um die Kommandos „tr.“ und „ts1.“ ergänzt werden.
Das Kommando „tr“ liest die Temperatur vom Temperatursensor aus und zeigt die Temperastur mittel putty an, das Kommando „tsx.“ Setzt die Auflösung der Temperaturmessung (x=0 => 0,5°C, x=1 => 0,25°C).

Analysieren Sie das I2C-Protokoll mit Hilfe des Oszis.
Optional: Ändern Sie die Auflösung des Temperatursensors mit den Kommandos „ts1“ bis ts3“.

Aufgabe 4_2 : I²C 7-Segment-Anzeige

Die 7-Segment Anzeige basiert auf dem Baustein SAA1064 mit der Adresse 0x38 (siehe Datenblatt in den Unterlagen). Für das Verständnis der Ansteuerung des Bausteins ist Seite 5+6 wesentlich.

Kurze Erklärung:



Der Baustein steuert vier 7-Segment-Anzeigen. Für jede Anzeige können über ein 8-Bit-Register die jeweiligen Segmente angesteuert werden (Bit0 = LED A, …, Bit 6 = LED G, Bit 7 = DP).
Der Bausteine verwendet 5 Register sowie einen Registerpointer (Instruction Byte). Wenn mehrere Register beschrieben werden sollen, muss der Registerpointer auf die Adresse des 1. Registers gesetzt werden, er wird automatisch pro Register um 1 erhöht.

Aufgaben:

4.2.1 Binden Sie das Modul Display in Ihr Projekt ein. Das Modul enthält zwei Funktionen zum Ansteuern des Displays.
Verwenden Sie zunächst die Funktion Display_I2C und geben damit die Bitmuster zur Ansteuerung der vier Anzeigen über I2C an den Displaybaustein. Fügen Sie dazu in die Funktion ExecuteCmd das Kommando „dba,b,c,d.“ ein, wobei a,b,,c und d für eine Hexzahl steht. Überlegen Sie sich vorab, welche Hex-Zahlen eingegeben werden müssen, um das Wort ESEL bzw. AFFE zu schreiben.
TIP: Hex-Eingabe mit sscanf(inputBuffer,"db%x,%x,%x,%x.",&digit[0],&digit[1],&digit[2],&digit[3]);

4.2.2 Ergänzen Sie nun die Funktion Display_Zahl, um eine Zahl mit Dezimalpunkt am Display auszugeben. Das Kommando in ExecuteCmd dafür ist „dz1234,2.“ um 12.34 (1234 ist der Zahlenwert, 2 steht für 2 Nachkommastellen) auszugeben.

Aufgabe 4_3 : Temperatur anzeigen

Nun soll zyklisch die Temperatur abgefragt und im Display angezeigt werden. Rechnen Sie dazu die Temperatur so um, dass sie auf dem 4-stelligen Display mit Dezimalpunkt angezeigt werden kann.
Zusätzlich soll der Schrittmotor als analoge Temperaturanzeige verwendet werden. Dazu müssen Sie aus dem Temperaturwert die Sollposition des Schrittmotors berechnen und die Skalierung so bemessen, dass der Temperaturbereich von 20 Grad Celsius bis ca. 30 Grad Celsius auf eine 180 Grad Drehung des Schrittmotors abgebildet wird.

Der zyklische Aufruf erfolgt am einfachsten, wenn Sie im Handler des Systick-Timer, der bereits für den Schrittmotor benötigt wird, eine globale Variable Ticks hochzählen und im Hauptprogramm diese Variable abfragen. Die Temperaturmessung mit Anzeige soll im Hauptprogramm alle 0,5 Sekunden aufgerufen werden.

I2C- Bibliothek:
Folgende Funktionen werden für die *I2C-Bus Kommunikation* zur Verfügung gestellt:

- *i2c.c:*

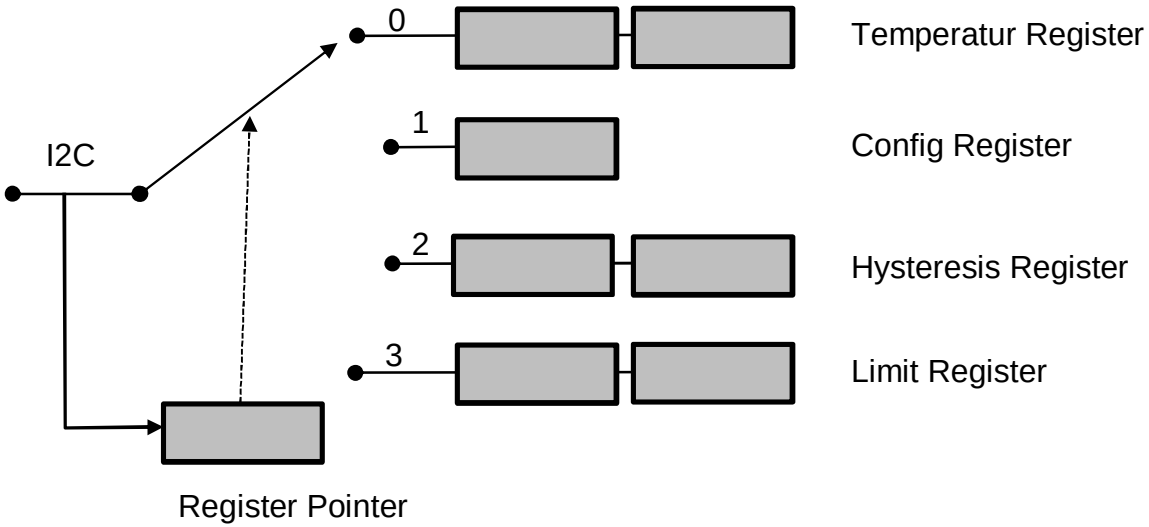
```
/* ================================================================
   I2C Init Routine
   ================================================================ */
void I2C_init(
         unsigned int clock                 // Systembus Taktrate
         ,unsigned char address             // 7-Bit I2C Adresse des Masters (µC)
)

/* ================================================================
   I2C Write n Byte From Buffer Routine
   ================================================================ */
void I2C_write(
         const unsigned char * buf          // Puffer der zu sendenden Byte
         ,unsigned int n                    // Anzahl zu sendender Byte
         ,unsigned char dest                // 7-Bit I2C Adresse des Busteilnehmers
)

/* ================================================================
   I2C Read n Byte To Buffer Routine
   ================================================================ */
void I2C_read(
         unsigned char * buf                // Puffer für eingelesene Byte
         ,unsigned int n                    // Anzahl zu lesender Byte
         ,unsigned char dest                // 7-Bit I2C Adresse des Busteilnehmers
)
```

Typische Werte:

clock: SystemCoreClock, APB Bustakt in Hz (i.A. bei uns 24 MHz)
I2C-Adresse des Busmasters (µC): 0x08 (d.h., niedriger als jeder Slave)
I2C-Adresse der Busteilnehmer: aus den Datenblättern entnehmen, muss die *7-Bit Adresse* sein!
Puffer buf: ausreichend lang für Daten und Befehle machen, mindestens n Byte!

Hinweise

Bei der Verwendung des RegisterPointers im Temperatursensor hilft die Vorstellung, den RegisterPointer wie einen Umschalter zu verwenden.



Um also die Temperatur auszulesen muss vorher der Register Pointer mit 0 beschrieben werden. Anschließend können die 2 Bytes des Temperaturregisters gelesen werden