


# HÖHERE TECHNISCHE BUNDESLEHRANSTALT HOLLABRUNN

Höhere Abteilung für Elektronik – Technische Informatik

Klasse/ Jahrgang: 5BHEL	Übungsbetreuer: Dipl. Ing. Josef Reisinger
Übungsnummer: Team #4	Übungstitel: CM3 Peripheral Library
Datum der Vorführung: /	Gruppe: Meichenitsch Tobias, Stella Florian
Datum der Abgabe: 16.04.21	Unterschrift: 

## Beurteilungskriterien

<b>Programm:</b>	<b>Punkte</b>
Programm Demonstration	
Erklärung Programmfunktionalität	
<b>Protokoll:</b>	<b>Punkte</b>
Pflichtenheft (Beschreibung Aufgabenstellung)	
Beschreibung SW Design (Flussdiagramm, Blockschaltbild,..)	
Dokumentation Programmcode	
Testplan (Beschreibung Testfälle)	
Kommentare / Bemerkungen	
<b>Summe Punkte</b>	

Note: \_\_\_\_\_

# POTI-ADC-UART2

## CM3 Peripheral Library

### 1.1 Inhaltsverzeichnis:

1.1	Inhaltsverzeichnis:.....	2
1.2	Produktanforderungen.....	2
1.3	Softwaredesign .....	3
1.4	Programmcode.....	4
1.5	Funktionstest:.....	6
1.6	Probleme.....	7
1.7	Erkenntnisse .....	7
1.8	Zeitaufwand.....	7

### 1.2 Produktanforderungen

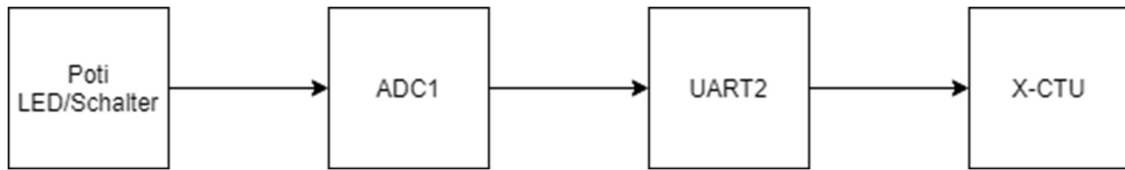
Es soll ein Programm entwickelt werden, welches die Analoge Ausgabe des Potentiometers der LED/Schalterplatine anhand des ADCs im Single Conversion Mode umwandelt und über die UART2 Schnittstelle ausgibt.

#### Pinbelegungen

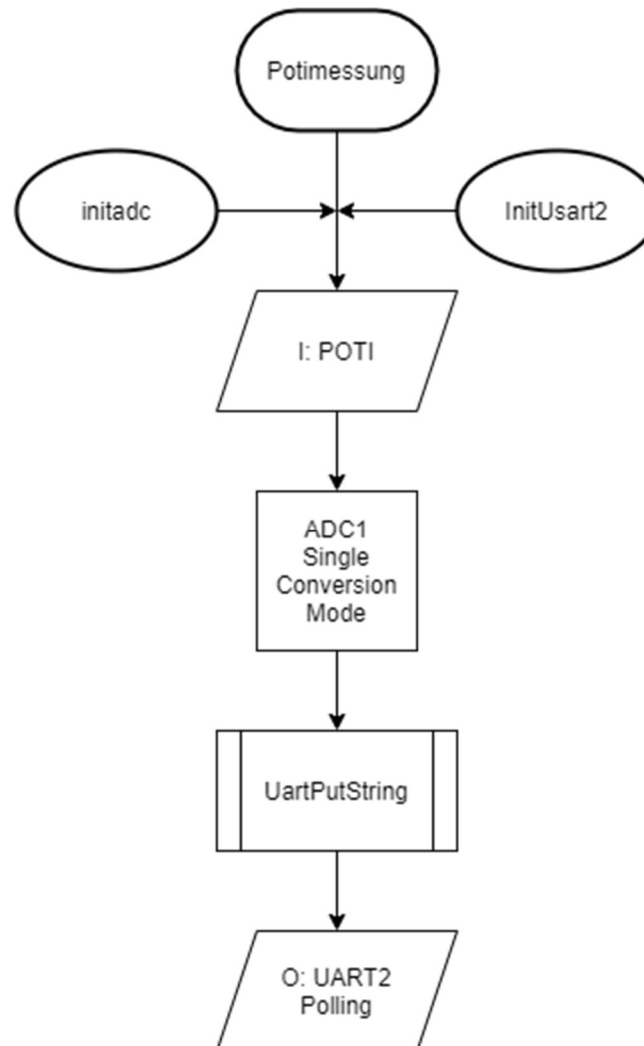
USART2		ADC1
Tx	Rx	Channel 9
PA2	PA3	PB1

### 1.3 Softwaredesign

Blockschaltbild:



Flussdiagramm:



## 1.4 Programmcode

```
// File Name: main.c
// Author: Meichenitsch Tobias, Florian Stella
// Date: 10.04.21

#include "stm32f10x.h"
#include "stm32f10x_conf.h"
#include <stdio.h>
#include "stm32f10x_gpio.h"
#include "stm32f10x_rcc.h"
#include <stm32f10x_usart.h>
#include <stm32f10x_adc.h>
#include "stm32f10x_exti.h"
#include "string.h"

void UartPutString(USART_TypeDef *USARTx, char *str);

//USART2 Init
void InitUart2(void)
{
    GPIO_InitTypeDef gpio;
    USART_ClockInitTypeDef usartclock;
    USART_InitTypeDef usart;

    USART_DeInit(USART2);

    //Enable Clock
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_AFIO, ENABLE);
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_USART2, ENABLE);

    GPIO_StructInit(&gpio);

    //PA2 (Tx)
    gpio.GPIO_Mode = GPIO_Mode_AF_PP;
    gpio.GPIO_Pin = GPIO_Pin_2;
    gpio.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_Init(GPIOA, &gpio);

    //PA3 (Rx)
    gpio.GPIO_Mode = GPIO_Mode_IN_FLOATING;
    gpio.GPIO_Pin = GPIO_Pin_3;
    GPIO_Init(GPIOA, &gpio);

    //init USART2 cLock
    USART_ClockStructInit(&usartclock);
    usartclock.USART_CPHA = USART_CPHA_2Edge;
    USART_ClockInit(USART2, &usartclock);
}
```

```
//Setup USART2
USART_StructInit(&usart);
usart.USART_BaudRate = 115200;
usart.USART_WordLength = USART_WordLength_8b;
usart.USART_StopBits = USART_StopBits_1;
usart.USART_Parity = USART_Parity_No ;
usart.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
usart.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
USART_Init(USART2, &usart);

//enable USART2
USART_Cmd(USART2, ENABLE);
}

//USART String Output
void UartPutString(USART_TypeDef *USARTx, char *str)
{
    while (*str)
    {
        while (USART_GetFlagStatus(USARTx, USART_FLAG_TXE) == RESET);
        USART_SendData(USARTx, *str++);
    }
}

//ADC1 Init
void initadc()
{
    ADC_InitTypeDef adc_init;
    GPIO_InitTypeDef gpio;

    //Peripheral Clock Enable
    RCC_APB2PeriphClockCmd(RCC_APB2ENR_IOPAEN, ENABLE);
    RCC_APB2PeriphClockCmd(RCC_APB2ENR_AFIOEN, ENABLE);
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOB, ENABLE);
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC1, ENABLE);

    //ADC1 Channel 9 (PB1, POT1 LED/Schalter)
    gpio.GPIO_Mode = GPIO_Mode_AIN;
    gpio.GPIO_Pin = GPIO_Pin_1;
    GPIO_Init(GPIOB, &gpio);

    //Setup ADC1
    adc_init.ADC_Mode = ADC_Mode_Independent;
    adc_init.ADC_ScanConvMode = DISABLE;
    adc_init.ADC_ContinuousConvMode = ENABLE; //Continous Conversion Mode
    adc_init.ADC_DataAlign = ADC_DataAlign_Right;
    adc_init.ADC_NbrOfChannel = 1;
    ADC_Init(ADC1, &adc_init);
}
```

```

ADC-RegularChannelConfig(ADC1, ADC_Channel_9, 1, ADC_SampleTime_28Cycles5);
ADC_ITConfig(ADC1, ADC_IT_EOC, ENABLE);
ADC_Cmd(ADC1, ENABLE);

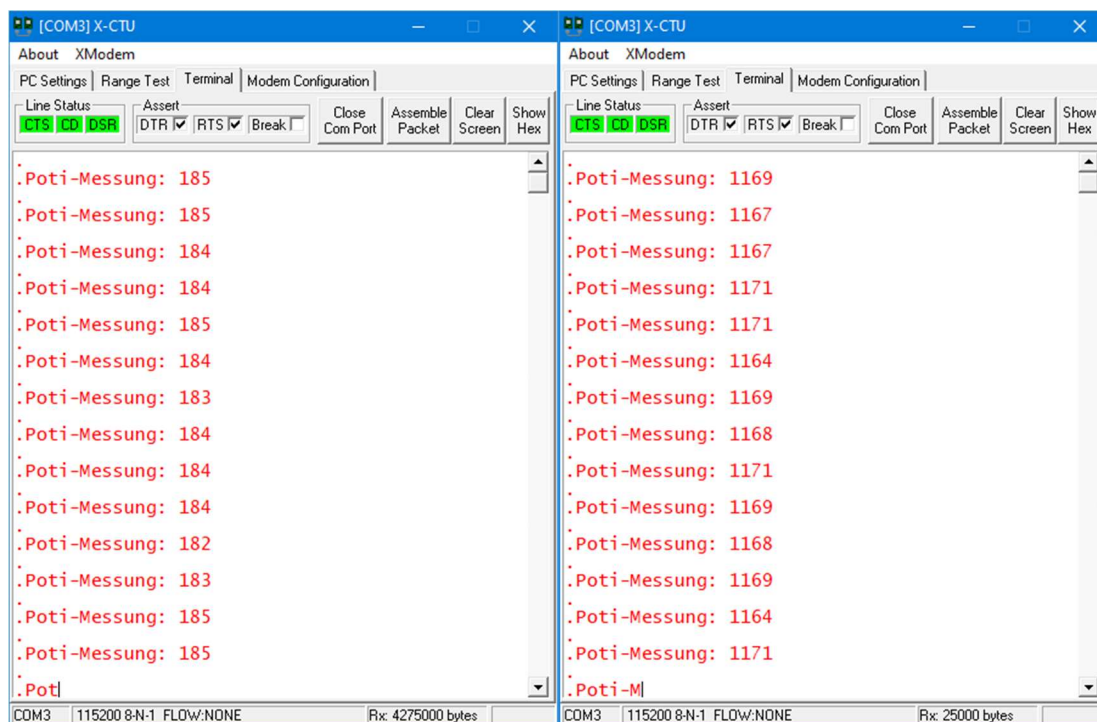
//ADC1 Calibration
ADC_ResetCalibration(ADC1);
while(ADC_GetResetCalibrationStatus(ADC1));
ADC_StartCalibration(ADC1);
while(ADC_GetCalibrationStatus(ADC1));
}

int main()
{
    char text[64];
    int value = 0;
    InitUsart2();
    initadc();
    while(1)
    {
        value = ADC_GetConversionValue(ADC1);
        sprintf(text, "\r\nPoti-Messung: %d\r\n", value);
        UartPutString(USART2, text);
    }
}

```

## 1.5 Funktionstest:

Um die Funktion zu testen wurde das POTI der Led/Schalterplatine gedreht und die Werte haben sich verändert.



## 1.6 Probleme

- UART2 Adapter Ursprünglich falsch angeschlossen

## 1.7 Erkenntnisse

- CM3 Standard Peripheral Librarys
- Hardware Reference Manual
- CM3 Arbeitsunterlagen

## 1.8 Zeitaufwand

<b>Tätigkeit</b>	<b>Aufwand</b>
Erstellung des Pflichtenhefts	0,5h
Erstellung des Systemdesign (Flussdiagramm bzw. Struktogramm und ev. UI Design)	0,75h
Programmcodierung	4h
Testen der Software	1,5h
Dokumentation (Protokoll)	2h
<b>Gesamt:</b>	<b>8,75h</b>