**HÖHERE TECHNISCHE BUNDESLEHRANSTALT**

**HOLLABRUNN**

Höhere Abteilung für Elektronik – Technische Informatik

|  |  |  |
| --- | --- | --- |
| Klasse / Jahrgang:  5BHEL | Gruppe:  10 | Übungsleiter:  Prof. Reisinger |
| Übungsnummer:  - | Übungstitel:  - | |
| Datum der Übung:  - | Teilnehmer:  Moritz Baldauf, Robert Radu | |
| Datum der Abgabe:  11.04.2021 | Schriftführer:  Baldauf, Radu | Unterschrift: |

**Allgemeiner Teil**

Übungsleiter: Prof. Reisinger

Klasse: 5BHEL

Schriftführer: Baldauf, Radu

Inhaltsverzeichnis

1 Aufgabenstellung 3

1.1 Individuelle Aufgabenstellung 3

1.2 Reale Anwendung 3

2 Blockschaltbild von Demoprogramm 4

2.1 Kurze Erläuterung wie Aufgabe gelöst wurde 4

3 Source-Code 4

4 Funktionsnachweis mit Screenshots & Erläuterung 9

# Aufgabenstellung

Generell ist es das Ziel in dieser Übung Demoprogramme für Peripheral Library des Mikrocontrollers STM32F10X zu entwickeln, die auch reale Anwendungen simulieren sollen. Dieses soll auf dem HTL eigenen Mikrocontrollersystem (Cortex M3) funktionieren. Zugriff auf Peripherie (GPIO, ADC, Joystick, …) darf außerdem auch nur über die Standard Peripheral Library erfolgen und das über Keil µVision (Entwicklungsumgebung). Zustandsänderungen müssen über den UART protokolliert werden.

## Individuelle Aufgabenstellung

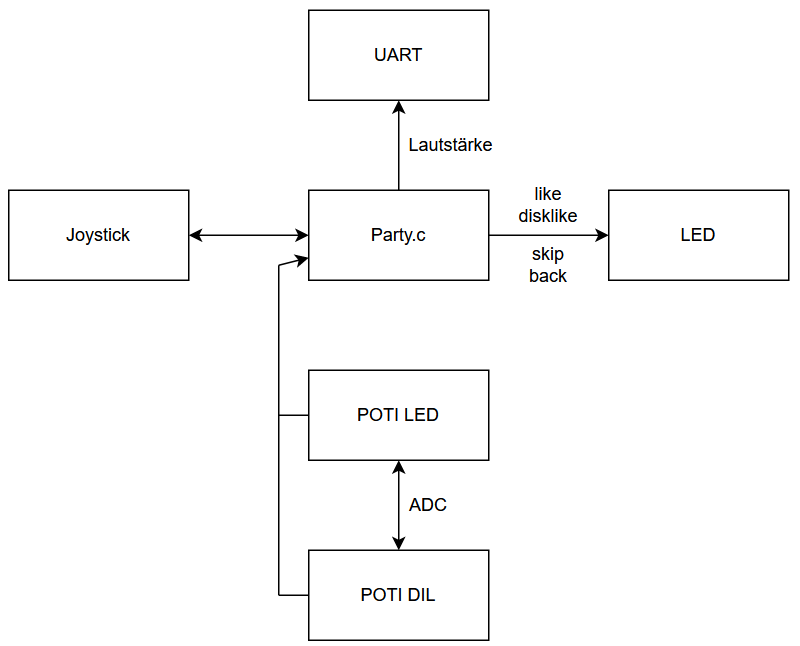
Folgende Peripherie (von CortexM3 Platine) muss verwendet werden:

* Joystick
* POTI (DIL Adapter)
* POTI LED/Schalterplatine
* ADC (ScanMode)
* LCD

## Reale Anwendung

Als Anwendung für die „echte Welt“ hat man hier einen Musik-Player verwirklicht. Mit den POTI-Bauelementen soll es möglich sein die Lautstärke zu verändern und diese muss über den UART sichtbar gemacht werden. Der Joystick soll außerdem die Songwahl steuern mit „skip“ und „back“, aber auch den Song selbst „liken“ oder „disliken“.

# Blockschaltbild von Demoprogramm



## Kurze Erläuterung wie Aufgabe gelöst wurde

Den Joystick kann man nach oben, unten, links oder rechts drücken und je nachdem wohin er gedrückt wird, gibt das Programm auf der LCD-Anzeige etwas anderes an, bei oben wird ein like ausgegeben, bei unten wird ein dislike ausgegeben, bei links ein back und bei rechts ein skip. Die Potentiometer sind beide für die Lautstärke zuständig, dafür muss man entweder nach links oder nach rechts drehen und die Lautstärke wird entweder kleiner oder größer.

# Source-Code

/\*

Name: Party.c

Author: Moritz Baldauf, Robert Radu

Description:

The two POTI's should be run by the ADC in Scanmode,

The joystick should give an message to the LCD if used,

all changes on the POTI's are shown on the LCD in a table

Version: 1.0

\*/

#include "Party.h"

#include "armv10\_std.h"

#include "stm32f10x\_conf.h"

/\*Pins for Joystick

Left:PC6

Down:PC7

Up:PC8

Right:PC9

\*/

//Initialise Pins for Joystick

GPIO\_InitTypeDef Joyleft **=**

**{**

**.**GPIO\_Pin **=** GPIO\_Pin\_6**,**

**.**GPIO\_Speed **=** GPIO\_Speed\_50MHz**,**

**.**GPIO\_Mode **=** GPIO\_Mode\_IPU

**};**

//Initialise Pins for Joystick

GPIO\_InitTypeDef JoyDown**=**

**{**

**.**GPIO\_Pin **=** GPIO\_Pin\_7**,**

**.**GPIO\_Speed **=** GPIO\_Speed\_50MHz**,**

**.**GPIO\_Mode **=** GPIO\_Mode\_IPU

**};**

//Initialise Pins for Joystick

GPIO\_InitTypeDef JoyUp **=**

**{**

**.**GPIO\_Pin **=** GPIO\_Pin\_8**,**

**.**GPIO\_Speed **=** GPIO\_Speed\_50MHz**,**

**.**GPIO\_Mode **=** GPIO\_Mode\_IPU

**};**

//Initialise Pins for Joystick

GPIO\_InitTypeDef JoyRight **=**

**{**

**.**GPIO\_Pin **=** GPIO\_Pin\_9**,**

**.**GPIO\_Speed **=** GPIO\_Speed\_50MHz**,**

**.**GPIO\_Mode **=** GPIO\_Mode\_IPU

**};**

//Internal peripheral Trigger

EXTI\_InitTypeDef Joyleft\_EXTI **=**

**{**

**.**EXTI\_Line **=** EXTI\_Line6**,**

**.**EXTI\_Mode **=** EXTI\_Mode\_Interrupt**,**

**.**EXTI\_Trigger **=** EXTI\_Trigger\_Falling**,**

**.**EXTI\_LineCmd **=** ENABLE

**};**

//Internal peripheral Trigger

EXTI\_InitTypeDef Joydown\_EXTI **=**

**{**

**.**EXTI\_Line **=** EXTI\_Line7**,**

**.**EXTI\_Mode **=** EXTI\_Mode\_Interrupt**,**

**.**EXTI\_Trigger **=** EXTI\_Trigger\_Falling**,**

**.**EXTI\_LineCmd **=** ENABLE

**};**

//Internal peripheral Trigger

EXTI\_InitTypeDef Joyup\_EXTI **=**

**{**

**.**EXTI\_Line **=** EXTI\_Line8**,**

**.**EXTI\_Mode **=** EXTI\_Mode\_Interrupt**,**

**.**EXTI\_Trigger **=** EXTI\_Trigger\_Falling**,**

**.**EXTI\_LineCmd **=** ENABLE

**};**

//Internal peripheral Trigger

EXTI\_InitTypeDef Joyright\_EXTI **=**

**{**

**.**EXTI\_Line **=** EXTI\_Line9**,**

**.**EXTI\_Mode **=** EXTI\_Mode\_Interrupt**,**

**.**EXTI\_Trigger **=** EXTI\_Trigger\_Falling**,**

**.**EXTI\_LineCmd **=** ENABLE

**};**

//Vector interrupt control for Pins 5-9

NVIC\_InitTypeDef Joy\_NVIC **=**

**{**

**.**NVIC\_IRQChannel **=** EXTI9\_5\_IRQn**,**

**.**NVIC\_IRQChannelPreemptionPriority **=** 3**,**

**.**NVIC\_IRQChannelSubPriority **=** 0**,**

**.**NVIC\_IRQChannelCmd **=** ENABLE

**};**

//variable for value from ADC1 channel 9

\_\_IO uint16\_t Ch\_9 **=** 0**;**

//variable for value from ADC1 channel 14

\_\_IO uint16\_t Ch\_14 **=** 0**;**

//string buffer for UART transfer

char buffer**[**100**];**

//Protoype for ADC programm

void ch14**(**void**);**

void EXTI9\_5\_IRQHandler**()**

**{**

//Joystick up

**if(**EXTI\_GetFlagStatus**(**Joyup\_EXTI**.**EXTI\_Line**)** **==** SET**)**

**{**

//Print Like on UART and LCD

lcd\_init**();**

lcd\_set\_cursor**(**0**,**0**);**

sprintf**(&**buffer**[**0**],**"Like\r\n"**);**

lcd\_put\_string**(&**buffer**[**0**]);**

USART\_SendString**(**USART1**,**buffer**);**

EXTI\_ClearFlag**(**Joyup\_EXTI**.**EXTI\_Line**);**

**};**

//Joystick left

**if(**EXTI\_GetFlagStatus**(**Joydown\_EXTI**.**EXTI\_Line**)** **==** SET**)**

**{**

//Print Dislike on UART and LCD

lcd\_init**();**

lcd\_set\_cursor**(**0**,**0**);**

sprintf**(&**buffer**[**0**],**"Dislike\r\n"**);**

lcd\_put\_string**(&**buffer**[**0**]);**

USART\_SendString**(**USART1**,**buffer**);**

EXTI\_ClearFlag**(**Joydown\_EXTI**.**EXTI\_Line**);**

**};**

//Joystick right

**if(**EXTI\_GetFlagStatus**(**Joyright\_EXTI**.**EXTI\_Line**)** **==** SET**)**

**{**

//Print Skip on UART and LCD

lcd\_init**();**

lcd\_set\_cursor**(**0**,**0**);**

sprintf**(&**buffer**[**0**],**"Skip\r\n"**);**

lcd\_put\_string**(&**buffer**[**0**]);**

USART\_SendString**(**USART1**,**buffer**);**

EXTI\_ClearFlag**(**Joyright\_EXTI**.**EXTI\_Line**);**

**};**

//Joystick left

**if(**EXTI\_GetFlagStatus**(**Joyleft\_EXTI**.**EXTI\_Line**)** **==** SET**)**

**{**

//Print Back on UART and LCD

lcd\_init**();**

lcd\_set\_cursor**(**0**,**0**);**

sprintf**(&**buffer**[**0**],**"Back\r\n"**);**

lcd\_put\_string**(&**buffer**[**0**]);**

USART\_SendString**(**USART1**,**buffer**);**

EXTI\_ClearFlag**(**Joyleft\_EXTI**.**EXTI\_Line**);**

**};**

NVIC\_ClearPendingIRQ**(**EXTI9\_5\_IRQn**);**

**}**

/\*

Synchronise inputs before

processing input-data using

a semaphore rendezvous

\*/

void ch14 **()**

**{**

int perc **=** 0**;**

int old **=** 0**;**

**for(;;)** **{**

//get data from ADC1 channel14

Ch\_14 **=** ADC1ConvertedValues**[**1**];**

//get data from ADC1 channel9

Ch\_9 **=** ADC1ConvertedValues**[**0**];**

//calculate combined percentage of both variable resistors

perc **=** **((**Ch\_14 **+** Ch\_9**)\***100**)/**8190**;**

//Only print if Changes

//Potis are not totally exact so there is a tolerance for changes

**if(!((**perc **>=** old**-**1**)&&(**perc **<=** old**+**1**)))**

**{**

old **=** perc**;**

//prepare string buffer for UART

sprintf**(&**buffer**[**0**],**"Lautstaerke:%d\r\n"**,**perc**);**

//send string via UART

USART\_SendString**(**USART1**,**buffer**);**

lcd\_init**();**

lcd\_clear**();**

//shows procentage on LCD screen

lcd\_bargraphXY**(**1**,**0**,**perc**);**

**}**

**}**

**}**

int main **(**void**)** **{**

//Enalbe Peripheral Clock

RCC\_APB2PeriphClockCmd**(**RCC\_APB2Periph\_GPIOC**,** ENABLE**);**

RCC\_APB2PeriphClockCmd**(**RCC\_APB2Periph\_AFIO**,** ENABLE**);**

//Initilaice GPIO Port and Pins

GPIO\_Init**(**GPIOC**,** **&**Joyleft**);**

GPIO\_Init**(**GPIOC**,** **&**JoyRight**);**

GPIO\_Init**(**GPIOC**,** **&**JoyDown**);**

GPIO\_Init**(**GPIOC**,** **&**JoyUp**);**

//Initilaice External Interrupt for IE-Event

AFIO**->**EXTICR**[**6**/**4**]** **|=** **(**AFIO\_EXTICR2\_EXTI6\_PC**);**

AFIO**->**EXTICR**[**7**/**4**]** **|=** **(**AFIO\_EXTICR2\_EXTI7\_PC**);**

AFIO**->**EXTICR**[**8**/**4**]** **|=** **(**AFIO\_EXTICR3\_EXTI8\_PC**);**

AFIO**->**EXTICR**[**9**/**4**]** **|=** **(**AFIO\_EXTICR3\_EXTI9\_PC**);**

EXTI\_Init**(&**Joyup\_EXTI**);**

EXTI\_Init**(&**Joydown\_EXTI**);**

EXTI\_Init**(&**Joyright\_EXTI**);**

EXTI\_Init**(&**Joyleft\_EXTI**);**

NVIC\_Init**(&**Joy\_NVIC**);**

//load DMA config

DMA\_Config**();**

//initialize ADC1

init\_uart**();**//initialize UART

ADC1\_Init**();**

//Clear Console

USART\_SendData**(**USART1**,**12**);**

USART\_SendString**(**USART1**,**

"\r\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\r\n Musikanlage \r\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\r\n"**);**

SystemCoreClockUpdate**();**

ch14**();**//start programm for potis

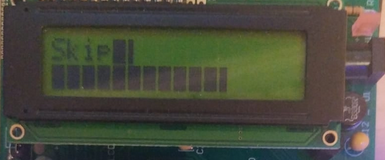
**while(**1**);**

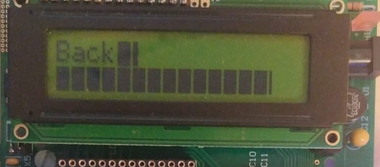
**}**

# Funktionsnachweis mit Screenshots & Erläuterung

Hier kann man klar erkennen wie eines der beiden POTI’s verwendet wird, da sich die Lautstärke verändert. Wird direkt ausgegeben bei Veränderung, aber auch wenn es keine gibt





Der Joystick wurde nach oben gedrückt 🡪 Der Song wird geliked.

Der Joystick wurde nach unten gedrückt 🡪 Der Song wird gedisliked.

Der Joystick wurde nach rechts gedrückt 🡪 Song wird übersprungen.

Der Joystick wurde nach links gedrückt 🡪 Ein Song davor wird abgespielt.