Mikroprozessorpraktikum

Konstantin Bork, Kean Seng Liew, & Oliver Stein, Gruppe A, HWP8

01-02 GPIO Eingang

A01-02.1

> Erläutern Sie anhand der Schaltung des Tasters an der Portleitung PC5 die Funktionsweise für die Tastenzustände "gedrückt" und "nicht gedrückt".

Wenn der Taster "gedrückt" ist, schließt man einen Stromkreis, sodass Strom durch PC5 fließt und man ein "high"-Signal erhält. Ist der Taster "nicht gedrückt", ist der Stromkreis nicht geschlossen. Folglich fließt kein Strom an PC5 und das Signal ist "low".

A01-02.2

> Erstellen Sie die Funktionen init_taste_1() und init_taste_2() innerhalb der Datei aufgabe.c. Innerhalb der Funktionen sollen die betreffenden Portleitungen als Eingang konfiguriert werden. - Taste 1 ist an PC8 angeschlossen - Taste 2 ist an PC5 angeschlossen Notwendige Informationen dazu finden Sie hier. Kommentieren Sie die einzelnen Codezeilen in den Funktionen init_taste_1() und init_taste_2().

aufgabe.h

```
#ifndef __aufgabe_h__
#define __aufgabe_h__

//###########################################################################
//########## cmsis_lib include
//###########################################################################
#include "stm32f4xx_gpio.h"
#include "stm32f4xx_rcc.h"

//###########################################################################
//########## mpp_lib include
//###########################################################################
#include "led.h"
#include "taster.h"

//###########################################################################
//########## Eigene Funktionen, Macros und Variablen
//###########################################################################

//===========================================================================
```

```
// Macros
//=============================================================================
#define GR_LED_ON        (GPIO_SetBits(GPIOB, GPIO_Pin_2))
#define GR_LED_OFF       (GPIO_ResetBits(GPIOB, GPIO_Pin_2))
#define GR_LED_Toggle    (GPIO_ToggleBits(GPIOB, GPIO_Pin_2))

//=============================================================================
// Variablen
//=============================================================================

//=============================================================================
// Funktionen
//=============================================================================
// Aufgabe A01-01
extern void init_leds();

// Aufgabe A01-02
extern void init_taste_1();
extern void init_taste_2();

//=============================================================================
#endif
```

aufgabe.c

```
#include "aufgabe.h"
#include "test.h"

// Aufgabe A01-01.3
// Initialisiert die Portleitung der grünen LED und schaltet diese ein
void init_leds()
{
    // Setzt GPIO Port auf den Reset Zustand zurück
    GPIO_DeInit(GPIOB);

    // Taktquelle für die Peripherie aktivieren
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB, ENABLE);

    // Struct anlegen
    GPIO_InitTypeDef GPIO_InitStructure;

    // Struct Initialisieren setzt alle Leitungen auf
    // Eingang ohne PushPull
    GPIO_StructInit(&GPIO_InitStructure);

    // Die Funktionalität der Portleitungen festlegen
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_2;

    // Auswahl des I/O Mode
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT; // GPIO Output Mode

    // Auswahl der Speed
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_2MHz; // Low speed

    // Auswahl des Output Typs
    GPIO_InitStructure.GPIO_OType = GPIO_OType_PP; // PushPull
```

```
    // Auswahl des Push/Pull Typs
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL; // NoPull

    // PortLeitungen initialisieren
    GPIO_Init(GPIOB, &GPIO_InitStructure);

    // Schaltet die LED ein
    GR_LED_ON;

    // LED wurde initialisiert, LED ausschalten
    GR_LED_OFF;
}

// Aufgabe A01-02.2
// Funktion zur Initialisierung beider Tasten
void init_taste(uint16_t GPIO_Pin)
{
    // Setzt GPIO Port auf den Reset Zustand zurück
        GPIO_DeInit(GPIOC);

        // Taktquelle für die Peripherie aktivieren
        RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOC, ENABLE);

        // Struct anlegen
        GPIO_InitTypeDef GPIO_InitStructure;

        // Struct Initialisieren setzt alle Leitungen auf
        // Eingang ohne PushPull
        GPIO_StructInit(&GPIO_InitStructure);

        // Die Funktionalität der Portleitungen festlegen

        GPIO_InitStructure.GPIO_Pin = GPIO_Pin;

        // Auswahl des I/O Mode
        GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN; // GPIO Input Mode

        // Auswahl der Speed
        GPIO_InitStructure.GPIO_Speed = GPIO_Speed_2MHz; // Low speed

        // Auswahl des Output Typs
        GPIO_InitStructure.GPIO_OType = GPIO_OType_PP; // PushPull

        // Auswahl des Push/Pull Typs
        GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL; // NoPull

        // PortLeitungen initialisieren
        GPIO_Init(GPIOC, &GPIO_InitStructure);
}

// Initialisierung von Taste 1
void init_taste_1()
{
    init_taste(GPIO_Pin_8);
}

// Initialisierung von Taste 2
void init_taste_2()
```

```
{
    init_taste(GPIO_Pin_5);
}
```

main.c

```
#include "main.h"
#include "aufgabe.h"

int main(void)
{
    // Initialisierung des Systems und des Clocksystems
    SystemInit();

    // SysTick initialisieren
    // jede ms erfolgt dann der Aufruf
    // des Handlers fuer den Interrupt SysTick_IRQn
    InitSysTick();

    // Initialisierung aller Portleitungen und Schnittstellen
    // Freigabe von Interrupten
    init_board();

    // Start der RTC  falls diese noch
    // nicht initialisiert war wird
    // die RTC mit der LSE-Taktquelle aktiviert
    start_RTC();

    // Initialisiere die grüne LED
    init_leds();

    // Initialisiere beide Tasten
    init_taste_1();
    init_taste_2();

    while(1)
    {

    }
}
```

A01-02.3

> Entwickeln Sie ein Programm, das folgende Funktionalität besitzt. - Wird die Taste 2 zweimal gedrückt (PC5) wird die grüne LED eingeschaltet. - Wird die Taste 1 einmal kurz gedrückt (PC8) wird die grüne LED ausgeschaltet.

aufgabe.h

```
#ifndef __aufgabe_h__
#define __aufgabe_h__
```

```
//#############################################################################
//########## cmsis_lib include
//#############################################################################
#include "stm32f4xx_gpio.h"
#include "stm32f4xx_rcc.h"

//#############################################################################
//########## mpp_lib include
//#############################################################################
#include "led.h"
#include "taster.h"

//#############################################################################
//########## Eigene Funktionen, Macros und Variablen
//#############################################################################

//=============================================================================
// Macros
//=============================================================================
#define GR_LED_ON        (GPIO_SetBits(GPIOB, GPIO_Pin_2))
#define GR_LED_OFF       (GPIO_ResetBits(GPIOB, GPIO_Pin_2))
#define GR_LED_Toggle    (GPIO_ToggleBits(GPIOB, GPIO_Pin_2))

//=============================================================================
// Variablen
//=============================================================================

//=============================================================================
// Funktionen
//=============================================================================
// Aufgabe A01-01
extern void init_leds();

// Aufgabe A01-02
extern void init_taste_1();
extern void init_taste_2();
extern uint8_t led_steuerung();

//=============================================================================
#endif
```

aufgabe.c

```
#include "aufgabe.h"
#include "test.h"

// Aufgabe A01-01.3
// Initialisiert die Portleitung der grünen LED und schaltet diese ein
void init_leds()
{
    // Setzt GPIO Port auf den Reset Zustand zurück
    GPIO_DeInit(GPIOB);

    // Taktquelle f�r die Peripherie aktivieren
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB, ENABLE);
```

```
    // Struct anlegen
    GPIO_InitTypeDef GPIO_InitStructure;

    // Struct Initialisieren setzt alle Leitungen auf
    // Eingang ohne PushPull
    GPIO_StructInit(&GPIO_InitStructure);

    // Die Funktionalit?t der Portleitungen festlegen

    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_2;

    // Auswahl des I/O Mode
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT; //GPIO Output Mode

    // Auswahl der Speed
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_2MHz; // Low speed

    // Auswahl des Output Typs
    GPIO_InitStructure.GPIO_OType = GPIO_OType_PP; // PushPull

    // Auswahl des Push/Pull Typs
    GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL; // NoPull

    // PortLeitungen initialisieren
    GPIO_Init(GPIOB, &GPIO_InitStructure);

    // Schaltet die LED ein
    GR_LED_ON;

    // LED wurde initialisiert, LED ausschalten
    GR_LED_OFF;
}

// Aufgabe A01-02.2
// Funktion zur Initialisierung beider Tasten
void init_taste(uint16_t GPIO_Pin)
{
    // Setzt GPIO Port auf den Reset Zustand zur?ck
        GPIO_DeInit(GPIOC);

        // Taktquelle für die Peripherie aktivieren
        RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOC, ENABLE);

        // Struct anlegen
        GPIO_InitTypeDef GPIO_InitStructure;

        // Struct Initialisieren setzt alle Leitungen auf
        // Eingang ohne PushPull
        GPIO_StructInit(&GPIO_InitStructure);

        // Die Funktionalit?t der Portleitungen festlegen
        GPIO_InitStructure.GPIO_Pin = GPIO_Pin;

        // Auswahl des I/O Mode
        GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN; // GPIO Input Mode

        // Auswahl der Speed
        GPIO_InitStructure.GPIO_Speed = GPIO_Speed_2MHz; // Low speed
```

```
        // Auswahl des Output Typs
        GPIO_InitStructure.GPIO_OType = GPIO_OType_PP; // PushPull

        // Auswahl des Push/Pull Typs
        GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL; // NoPull

        // PortLeitungen initialisieren
        GPIO_Init(GPIOC, &GPIO_InitStructure);
}

// Initialisierung von Taste 1
void init_taste_1()
{
    init_taste(GPIO_Pin_8);
}

// Initialisierung von Taste 2
void init_taste_2()
{
    init_taste(GPIO_Pin_5);
}

// Aufgabe A01-02.3
// Programm zum Kontrollieren der grünen LED mit den beiden Tasten
int led_steuerung()
{
    uint8_t     Byte = 0;

    Byte = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_5);

    if(Byte == Bit_SET)
    {
        return 1;
    }

    Byte = GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_8);

    if(Byte != Bit_SET)
    {
        return -1;
    }

    return 0;
}
```

main.c

```
#include "main.h"
#include "aufgabe.h"

int main(void)
{
    // Initialisierung des Systems und des Clocksystems
    SystemInit();

    // SysTick initialisieren
```

```
    // jede ms erfolgt dann der Aufruf
    // des Handlers fuer den Interrupt SysTick_IRQn
    InitSysTick();

    // Initialisierung aller Portleitungen und Schnittstellen
    // Freigabe von Interrupten
    init_board();

    // Start der RTC  falls diese noch
    // nicht initialisiert war wird
    // die RTC mit der LSE-Taktquelle aktiviert
    start_RTC();

    // Initialisiere die grüne LED
    init_leds();

    // Initialisiere beide Tasten
    init_taste_1();
    init_taste_2();

    // Initialisiere Zähler für Taste 2
    uint8_t taste_2_zaehler = 0;

    while(1)
    {
        int taste_2_gedrueckt = led_steuerung(); // Kontrolliere die grüne LED mit
beiden Tasten

        if (taste_2_gedrueckt == 1) {
            taste_2_zaehler++;
            if (taste_2_zaehler == 2) {
                GR_LED_ON;
                taste_2_zaehler = 0;
            }
        } else if (taste_2_gedrueckt == -1) {
            GR_LED_OFF;
            taste_2_zaehler = 0;
        }
    }
}
```