Mikroprozessorpraktikum

Konstantin Bork & Kean Seng Liew, Gruppe A, HWP8

06-02 Alarm

A06-02.1

> Es sollen zwei eigenständige Lösungen unter Nutzung des externen RTC Alarm A Event Interrupts erstellt werden, die folgende Funktionalität umsetzen. - Zu jeder 30igsten Sekunde in einer jeden Minute soll auf der USART2 die aktuelle Zeit ausgegeben werden. - Jeden Montag um 00:30:00 Uhr soll die aktuelle Zeit und das Datum ausgegeben werden.
>
> Bitte benutzen Sie zur Lösung die Möglichkeiten der RTC Alarm Konfiguration.

Auszug aufgabe.h

```
// Aufgabe A06-02.1
extern void alarm_bei_30_sekunden();
extern void alarm_am_montag();
extern void init_rtc_alarm_irq();
```

Auszug aufgabe.c

```
void alarm_bei_30_sekunden()
{
    RTC_AlarmTypeDef RTC_Alarm_Struct;

    // Alarmzeit setzen, nämlich bei der 30. Sekunde jeder Minute
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_H12      = RTC_H12_AM;
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_Hours    = 0x00;
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_Minutes  = 0x00;
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_Seconds  = 0x30;

    // Alarmmaske setzen, damit nur die Sekunden geprüft werden
    RTC_Alarm_Struct.RTC_AlarmMask          = RTC_AlarmMask_DateWeekDay
                                            | RTC_AlarmMask_Hours
                                            | RTC_AlarmMask_Minutes;

    // Alarm für den Tag oder Wochentag auswählen
    RTC_Alarm_Struct.RTC_AlarmDateWeekDaySel = RTC_AlarmDateWeekDaySel_WeekDay; //
Wochentag (Mo-So)

    // Alarm Tag oder Wochentag setzen
    RTC_Alarm_Struct.RTC_AlarmDateWeekDay    = RTC_Weekday_Monday;      //
Wochentag Mo
```

```
    // Konfiguration von Alarm A
    RTC_SetAlarm(RTC_Format_BCD, RTC_Alarm_A, &RTC_Alarm_Struct);
}

void alarm_am_montag()
{
    RTC_AlarmTypeDef RTC_Alarm_Struct;

    // Alarmzeit setzen um 00:30
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_H12      = RTC_H12_AM;
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_Hours    = 0x00;
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_Minutes  = 0x30;
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_Seconds  = 0x00;

    // Alarmmaske setzen
    RTC_Alarm_Struct.RTC_AlarmMask          = RTC_AlarmMask_None;

    // Alarm für den Wochentag auswählen
    RTC_Alarm_Struct.RTC_AlarmDateWeekDaySel = RTC_AlarmDateWeekDaySel_WeekDay; //
Wochentag (Mo-So)

    // Wochentag gemäß Aufgabe setzen
    RTC_Alarm_Struct.RTC_AlarmDateWeekDay    = RTC_Weekday_Monday;      //
Wochentag Mo

    // Konfiguration von Alarm A
    RTC_SetAlarm(RTC_Format_BCD, RTC_Alarm_A, &RTC_Alarm_Struct);
}

void init_rtc_alarm_irq()
{
    // RTC Alarm A Interruptkonfiguration

    // Anlegen der benötigten Structs
    EXTI_InitTypeDef EXTI_InitStructure;
    NVIC_InitTypeDef NVIC_InitStructure;

    // EXTI-Line Konfiguration
    EXTI_ClearITPendingBit(EXTI_Line17);
    EXTI_InitStructure.EXTI_Line = EXTI_Line17;
    EXTI_InitStructure.EXTI_Mode = EXTI_Mode_Interrupt;
    EXTI_InitStructure.EXTI_Trigger = EXTI_Trigger_Rising;
    EXTI_InitStructure.EXTI_LineCmd = ENABLE;
    EXTI_Init(&EXTI_InitStructure);

    // NIVC Konfiguration
    NVIC_InitStructure.NVIC_IRQChannel = RTC_Alarm_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);

    // Konfigurieren des Alarm A
    RTC_ITConfig(RTC_IT_ALRA, ENABLE);

    // RTC Alarm A freigeben
    RTC_AlarmCmd(RTC_Alarm_A, ENABLE);
```

```
    // Alarmflag löschen
    RTC_ClearFlag(RTC_FLAG_ALRAF);
}
```

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();

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

    // Je nachdem, welchen Alarm man benutzen will
    //alarm_bei_30_sekunden();
    alarm_am_montag();

    // Initialisiere den Alarm Interrupt
    init_rtc_alarm_irq();

    while(1)
    {

    }
}
```

Auszug interrupts.c

```
void RTC_Alarm_IRQHandler(void)
{
    //===== RTC_IT_ALRA: Alarm A interrupt
    if(RTC_GetITStatus(RTC_IT_ALRA) != RESET)
    {
        RTC_TimeTypeDef  RTC_Time_Aktuell;      //  Zeit
        RTC_DateTypeDef  RTC_Date_Aktuell;      //  Datum
        char data[50] = {0};

        // Dieser Teil wird für den Alarm am Montag benötigt und ist im anderen
Alarm
        // nicht vorhanden. Wenn man alarm_bei_30_sekunden() verwenden will, kann
man
        // die nächsten 3 Codezeilen entfernen oder auskommentieren
```

```
        // Datum aus der RTC in das Struct laden
        RTC_GetDate(RTC_Format_BIN, &RTC_Date_Aktuell);
        sprintf(data,"\r\n%.2d-%.2d-%.2d-%.2d",RTC_Date_Aktuell.RTC_Year,
RTC_Date_Aktuell.RTC_Month, RTC_Date_Aktuell.RTC_Date,
RTC_Date_Aktuell.RTC_WeekDay);
        usart_2_print(data);

        // Ausgabe für beide Alarme

        // Zeit aus der RTC in das Struct laden
        RTC_GetTime(RTC_Format_BIN, &RTC_Time_Aktuell);
        sprintf(data,"\r\n%.2d:%.2d:%.2d:%.2d",RTC_Time_Aktuell.RTC_Hours,
RTC_Time_Aktuell.RTC_Minutes, RTC_Time_Aktuell.RTC_Seconds,
RTC_Time_Aktuell.RTC_H12);
        usart_2_print(data);

        RTC_ClearITPendingBit(RTC_IT_ALRA);
        EXTI_ClearITPendingBit(EXTI_Line17);
    }
}
```

A06-02.2

> Entwickeln Sie ein Programm das folgenden Anforderungen entspricht: - Die RTC soll einen externen RTC Alarm A Event (EXTI_Line17) Interrupt generieren. - Der Interrupt soll alle 25 Sekunden ausgelöst werden. - In der ISR soll die grüne LED getoggelt werden und die aktuelle RTC Zeit auf die USART2 ausgegeben werden.

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();

    // 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_irq();
    init_taste_2_irq();

    // Initialisiere USART2 für die Ausgabe
    init_usart_2_irq();

    init_nvic();

    // Initialisiere hier den ersten Alarm
    RTC_AlarmCmd(RTC_Alarm_A, DISABLE);
    alarm_alle_25_sekunden();
    init_rtc_alarm_irq();

    while(1)
    {

    }
}
```

Auszug aufgabe.h

```
// Aufgabe A06-02.2
void alarm_alle_25_sekunden();
```

Auszug aufgabe.c

```
// Nimmt die aktuellen Sekunden, addiert 25 dazu und nimmt das Modulo 60,
// damit die Sekunden im erlaubten Bereich von 0 bis einschließlich 59 bleiben
uint8_t berechne_sekunden(uint8_t aktuelle_sekunden) {
    return (aktuelle_sekunden + 25) % 60;
}

void alarm_alle_25_sekunden()
{
    RTC_TimeTypeDef  RTC_Time_Aktuell;      //  Zeit
    RTC_AlarmTypeDef RTC_Alarm_Struct;

    RTC_GetTime(RTC_Format_BIN, &RTC_Time_Aktuell); // aktuelle Zeit bekommen

    // Alarmzeit setzen..
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_H12      = RTC_H12_AM;
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_Hours    = RTC_Time_Aktuell.RTC_Hours;
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_Minutes  = RTC_Time_Aktuell.RTC_Minutes;
    RTC_Alarm_Struct.RTC_AlarmTime.RTC_Seconds  =
berechne_sekunden(RTC_Time_Aktuell.RTC_Seconds);

    // Alarmmaske setzen, damit nur die Sekunden betrachtet werden
    RTC_Alarm_Struct.RTC_AlarmMask          = RTC_AlarmMask_DateWeekDay
                                            | RTC_AlarmMask_Hours
                                            | RTC_AlarmMask_Minutes;

    // Alarm für den Tag oder Wochentag auswählen
    RTC_Alarm_Struct.RTC_AlarmDateWeekDaySel = RTC_AlarmDateWeekDaySel_Date; // Tag
(1-31)
```

```
    // Alarm Tag oder Wochentag setzen
    RTC_Alarm_Struct.RTC_AlarmDateWeekDay    = 0x01; // Tag 0x01...0x31

    // Konfiguration von Alarm A
    RTC_SetAlarm(RTC_Format_BIN, RTC_Alarm_A, &RTC_Alarm_Struct);
}
```

Auszug interrupts.c

```
void RTC_Alarm_IRQHandler(void)
{
    //===== RTC_IT_ALRA: Alarm A interrupt
    if(RTC_GetITStatus(RTC_IT_ALRA) != RESET)
    {
        RTC_TimeTypeDef  RTC_Time_Aktuell;      //  Zeit
        char data[50] = {0};

        GR_LED_Toggle;

        // Zeit aus der RTC in das Struct laden
        RTC_GetTime(RTC_Format_BIN, &RTC_Time_Aktuell);
        sprintf(data,"\r\n%.2d:%.2d:%.2d:%.2d",RTC_Time_Aktuell.RTC_Hours,
RTC_Time_Aktuell.RTC_Minutes, RTC_Time_Aktuell.RTC_Seconds,
RTC_Time_Aktuell.RTC_H12);
        usart_2_print(data);

        RTC_ClearITPendingBit(RTC_IT_ALRA);
        EXTI_ClearITPendingBit(EXTI_Line17);

        // Reinitialisiere den Alarm, damit in 25 Sekunden der nächste erfolgt
        RTC_AlarmCmd(RTC_Alarm_A, DISABLE);
        alarm_alle_25_sekunden();
        RTC_AlarmCmd(RTC_Alarm_A, ENABLE);
    }
}
```