

Externer Interrupt

Aufgabe 1: Was wäre wenn es keine Klingel gäbe?

Situation: Ihr seid zuhause beim Bügeln und erwartet wichtigen Besuch. Eine genaue Uhrzeit habt Ihr nicht vereinbart. Euer Besuch kann telefonisch, oder sonst wie, nicht erreicht werden. Bügeln muss aber dringend sein.

Szenario 1: Alle Klingel (Haustür, Handy usw.) funktionieren

Szenario 2: Alle Klingeln sind ausgefallen.

Diskutiert im Plenum folgende Fragen:

a) Wie verhaltet Ihr Euch in beiden Szenarien?

b) Bei einem **Interrupt** reagiert Ihr auf ein Ereignis, beim **Polling** müsst Ihr ständig die Eingänge prüfen. Begründet, wie Ihr die beiden Szenarien zuordnet.

c) Gebt für beide Verfahren, Polling und Interrupt, Vor- und Nachteile an.

Aufgabe 2: Schauen Sie sich die Videos an und lösen Sie folgende Aufgaben mit Externem Interrupt

- a) Bei einem Tastendruck an PA_1 soll eine ihren Zustand wechseln: Aus -> Taste -> An -> Taste -> Aus ...
- b) Bei Tastendruck an PA_1 soll die LED eingeschaltet werden, bei loslassen soll sie wieder ausgehen
- c) Bei Tastendruck auf Taste PA_1 soll die LED angehen, bei Tastendruck auf Taste PA_6 soll sie wieder ausgehen.
- d) Bei Tastendruck auf Taste PA_1 soll GPIOC PortC hochzählen, bei Tastendruck auf Taste PA_6 soll PortC herunterzählen
- e) Reaktionsspiel: Im Hauptprogramm wandert eine 1, zyklisch wiederholt von PC_0 nach PC_7 und wieder zurück. Die Wartezeit soll jeweils 0,25 Sekunden betragen.

Interrupt-Service-Routine:

Wenn bei Tastendruck an PA_1 gerade PC_3 leuchtet soll ein Trefferzähler int treffer=0; hochgezählt werden. Bei einem Fehlschuss wird die Trefferzahl um 1 heruntergezählt. Minimum=0. Die Anzahl der Treffer soll auf dem LCD-Display angezeigt werden. Beachten Sie: PA_1 benötigt PullDown.

Beachten Sie weiterhin: Die Deklarationen von Ports, treffer, LCD müssen global sein;

```
#include "mbed.h"
```

```
#include "platform/mbed_thread.h"
```

```
#include "LCD.h"
```

```
// Blinking rate in milliseconds
```

```
#define BLINKING_RATE_MS 250
```

```
//globale Deklarationen
```

```
int treffer=0;
```

```
Lcd mylcd;
```

```
PortOut Lauflicht(PortC,0xFF);
```

```
...
```

- f) Ampel: Jeweils 3 LED: Rot, gelb und grün an PortC bilden die Ampeln von Haupt- und Nebenstraße.

In Hauptprogramm werden nacheinander, zyklisch wiederholt, zunächst die Hauptstraße, dann die Nebenstraße rot, rot-gelb-grün-gelb-rot geschaltet. Die übrigen beiden LEDs rot und grün bilden die Fußgängerampeln. Die ISR der Taste PA_1 (mit PullDown) setzt eine globale bool-Variable bool gedrueckt=false; auf true. Wenn gedrueckt == true, wird im Hauptprogramm, an passender Stelle, die Fußgängerampel auf grün und wieder auf rot geschaltet. Die globale Variable gedrueckt muss dabei natürlich wieder auf false gesetzt werden.

Aufgabe 3: Kontrollfragen:

1. Wie wird ein Interruptprogramm (ISR) aufgerufen?
2. Wie wird der PA_6 als Interruptquelle festgelegt?
3. Wie legen Sie ein Unterprogramm als Interruptserviceroutine (ISR) für PA_6 steigende Flanke fest? Wie für fallende Flanke?
4. Welche Auswirkung hat das Eintreten eines Interruptereignisses auf das Hauptprogramm?
5. Wo im Programm steht die Interruptinitialisierung?
6. Wie wird der Interrupt freigegeben?
7. Welche Wirkung hat die Interruptfreigabe?
8. Wie wird der Interrupt wieder gesperrt?

Anhang 1 Interrupt-Anweisungen:

Anweisung	Bedeutung
InterruptIn name(Port);	Ein Port wird als Interruptquelle mit Namen <i>name</i> festgelegt Beispiel: InterruptIn taste(PA_1);
name.mode(PullDown);	Der Interrupteingang bekommt einen PullDown
name.rise(&isr)	Bei einer steigenden Flanke am Port wird das Unterprogramm mit dem Namen isr (Interrupt Service Routine) automatisch aufgerufen
name.fall(&isr)	Bei einer fallenden Flanke am Port wird das Unterprogramm mit dem Namen isr (Interrupt Service Routine) automatisch aufgerufen
name.enable_irq()	Freigabe des Interrupts. Nach dieser Anweisung reagiert der Mikrocontroller auf die fallende oder steigende Flanke indem er automatisch die ISR aufruft
name.disable_irq()	Sperre des Interrupts. Der Mikrocontroller reagiert nicht mehr auf Ereignisse am Port.

Anhang 2: Nützliche Anweisungen

Anweisung	Bedeutung
PortOut Lauflicht(PortC,0xFF)	Die Bits PC_0 .. PC_7 werden als 8-Bit-Ausgangsport festgelegt
Lauflicht=1;	Bit0 wird auf 1 gesetzt
Lauflicht=Lauflicht<<1;	Das Bitmuster in Lauflicht wird um eine Stelle nach links verschoben
If (Lauflicht==0) Lauflicht=1;	Von vorne

Anhang 2:

Hinweise zur Programmierung:

Da die Ausgabe auf das LCD-Display erfolgen soll benötigen Sie folgende Library:

Ergänzen Sie in der main.cpp an den entsprechenden Stellen:

The screenshot shows the Mbed IDE interface. At the top, the 'Import' button is highlighted with a red box and an arrow. Below it, the 'Import Wizard' dialog is open, showing the 'Libraries' tab. The search bar contains 'LCD_i2c_GSOE'. A red box labeled 'Doppelklick' points to the 'LCD_i2c_GSOE' entry in the list of published libraries. To the right, a red box says 'Library LCD_i2c_GSOE suchen'. Below the wizard, the 'main.cpp' file is open. A red box labeled 'Library mit #include „LCD.h“ einbinden' points to the line `#include "LCD.h"`. Another red box labeled 'LCD-Display-Objekt mit Namen z.B. mylcd deklarieren und erzeugen' points to the line `lcd mylcd;`. A third red box labeled 'Befehle:' contains a list of commands to be added to the `main()` function.

Import

Import Wizard

Import a library from os.mbed.com

Select library from the list. You can also drag&drop them in your workspace.
[Click here](#) to import from URL.

Programs Libraries Bookmarked Upload

Listing published libraries: LCD_i2c_GSOE

Name	Author	Imports	Modified	Description
★ LCD_i2c_GSOE	Jack Hansdampf	1	12 Mar 2020	LCD Baseshield GSOE

Doppelklick

Library LCD_i2c_GSOE suchen

main.cpp

```
1 /* mbed Microcontroller Library
2 * Copyright (c) 2019 ARM Limited
3 * SPDX-License-Identifier: Apache-2.0
4 */
5
6 #include "mbed.h"
7 #include "platform/mbed_t
8 #include "LCD.h"
9
10 int main()
11 {
12
13     // Initialise
14     mylcd.clear();
15     mylcd.cursorpos(0);
16     mylcd.printf("Hallo %c", 'x');
17     PortIn spalten(PortB, 0b1111);
18     spalten.mode(PullUp);
19 }
```

Library mit #include „LCD.h“ einbinden

LCD-Display-Objekt mit Namen z.B. mylcd deklarieren und erzeugen

Befehle:

- `mylcd.clear();` //löscht das Display
- `mylcd.cursorpos(wert);` //plaziert den Cursor
0..15: 1.Zeile, 64..79 2.Zeile (64=0x40)
- `mylcd.printf(Formatstring,Werte);`
//Ausgabe. Doku: Internet printf