Multiplexing mit dem Multifunctionshield

## Dokumentation

* 1. Neues Programm: Vorlage: mbed OS Blinky LED Hello World
  2. Import der Bibliothek
     + Offline MBED-Studio: Import Library. URL der Library: <https://os.mbed.com/users/jack1930/code/MFS_Display_HAL/>
     + Online MBED: Import Library, Suchen mit GSOE

MFS\_Display\_HAL doppelklicken

* 1. Einbinden der Bibliothek in main.cpp

/\* mbed Microcontroller Library

 \* Copyright (c) 2019 ARM Limited

 \* SPDX-License-Identifier: Apache-2.0

 \*/

#include "mbed.h"

**#include "MFS.h" //Einbinden der Bibliothek**

// Blinking rate in milliseconds

#define BLINKING\_RATE     500ms

* 1. Globale Deklaration des Multifunctionshields  
     **MFS display; //Anstatt PortOut display verwenden wir MFS display**
  2. Verwendung

Bit0 a

Bit1 b

Bit2 c

Bit3 d

Bit4 e

Bit5 f

Bit6 g

Bit7 dp

a

g

d

f

e

b

c

a

g

d

f

e

b

c

a

g

d

f

e

b

c

a

g

d

f

e

b

c

Bit12 Digit 0

Bit13 Digit 1

Bit14 Digit 2

Bit15 Digit 3

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| display | Dig3 | Dig2 | Dig1 | Dig0 | 0 | 0 | 0 | 0 | dp | g | f | e | d | c | b | a |

char seg7[10]={0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F};

Beispiele: display=0b0001000000111111; //Gibt eine 0 auf Digit 0 aus  
display=0x2000+seg7[3]; //gibt 3 auf Digit 1 aus

## Stufe 1: Zahl auf Digit 0 ausgeben

Programmgerüst:

#include "mbed.h"

#include "MFS.h"

MFS display;

char seg7[10]={0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F};

int main()

{

// Initialise the digital pin LED1 as an output

DigitalOut led(LED1);

while (true) {

display=0x1000+seg7[5]; //gibt eine 5 auf Digit 0 aus

led = !led;

HAL\_Delay(500);

}

}

* 1. Geben Sie ebenso eine 3 auf Digit 1 aus
  2. Ergänzen sie eine Zählvariable, die im Sekundentakt immer von 0 bis 9 zählt und dann wieder von vorne beginnt

## Stufe 2: Die Zahlen 0 .. 3 nacheinander auf die Digits 0 .. 3 mit externem Interrupt ausgeben.

* 1. Die Ausgabe erfolgt mit externem Interrupt an Taste PA1
     + Global
       1. InterruptIn Taste(PA\_1);
       2. int dig=0; //Stellenzähler
       3. int digTab[4]={0x1000,0x2000,0x4000,0x8000};
     + ISR void TasteISR()
       1. Ausgabe display=digTab[dig]+seg7[dig]
       2. dig++
       3. falls dig=4 dann dig wieder bei 0 beginnen
     + in main
       1. Externen Interrupt für fallende Flanke initialisieren
       2. Die Testbefehle aus Stufe 1 entfernen
  2. Multiplexbetrieb: Schließen Sie die Brücke, damit werden mit hoher Frequenz Interrupts ausgelöst. 

## Stufe 3: Ausgabe des analogen Poti-Einstellwertes 4-stellig auf das Display

* 1. Ergänzungen global
     + AnalogIn poti(PA\_0); //Analogeingang an Portin PA\_0
     + int ausgabewert=0; //Diese Zahl soll ausgegeben werden
  2. In der ISR
     + Berechnung der lokalen Variablen int stelle (Einer, Zehner, Hunderter und Tausender
     + Wenn dig=0 (Einerstelle) dann stelle=ausgabewert%10;
     + Wenn dig=1 (Zehnerstelle) dann stelle=ausgabewert/10%10;
     + Usw.
     + Ausgabe: display=digTab[dig]+seg7[stelle]
     + dig erhöhen bis 3, bei 4 von vorne
  3. main
     + In der Endlosschleife: ausgabewert=poti\*10000;   
       //poti geht von 0..1, die Anzeige soll aber von 0-9999 gehen.

## Beantworten Sie folgende Fragen

* 1. Warum bemerkt das Auge den Multiplexbetrieb nicht?
  2. Was bewirkt die Anweisung: **stelle=ausgabewert/10%10;**
  3. Wie funktioniert die Anweisung: display=digTab[dig]+seg7[stelle]