



MIKROPROZESSORPRAKTIKUM

WS2020

Termin 2

C-Programmierung für eingebettete Systeme Pointer, Peripherie, PIO,
Interrupt

Name, Vorname	Matrikelnummer	Anmerkungen
Datum	Raster (z.B. Mi3x)	Testat/Datum

Legende: V: Vorbereitung, D: Durchführung, P: Protokoll/Dokumentation, T: Testat

Lernziele:

Der Umgang mit digitalen Ein- und Ausgängen und Unterbrechungen (Interrupt).

Arbeitsverzeichnis:

Kopieren Sie sich aus dem Ordner „./mnt/Originale“ auf dem Laborarbeitsplatzrechner oder aus dem Ordner „<https://userv.fbi.h-da.de/LabDisk/MI/>“ den Ordner mpsWS2020. Dort finden Sie zu jedem Termin vorgegebene Dateien.

Vorgehensweise:

Speichern Sie zu jeder gestellten Aufgabe Ihre Lösung unter Termin2AufgabeX.c (X steht für Nummer der Aufgabe). In den angebotenen offenen Laboren bekommen Sie die Möglichkeit Ihre Lösungen praktisch vorzuführen und zu testen und/oder mit den Lösungen Ihrer Kommilitonen zu vergleichen/zus diskutieren.

Zum zugehörigen Praktikumstermin sollen Sie Ihre Lösungen präsentieren können.

Aufgabe 1:

Wir werden uns nach Termin 1 nun nochmals mit den Registern der PIOB des hier eingesetzten Mikrocontroller (AT91M63200) beschäftigen. Durch die strukturierte Anordnung der vielen verschiedenen Register der Peripherie des Mikrocontroller ist es möglich durch angelegte Strukturen sich die Programmierung und den Umgang mit der Peripherie zu vereinfachen. Schauen Sie sich Termin2Aufgabe1.c an. Testen und beschreiben Sie die Funktion des Programms. Hierzu schauen Sie sich die Dokumentation zum Board AT91EB63 (AT91EB63.pdf) und die Doku des eingesetzten Mikrocontrollers AT91M63200 (AT91M63200 (Complete).pdf) an.

Mit welchen Anweisung werden die LED's ein- bzw. ausgeschaltet und warum?

..

Auf welchen Speicherstellen können Sie erkennen ob eine LED ein- oder ausgeschaltet sein sollte?

..

Aufgabe 2:

Gut, wir können nun die Leuchtdioden (DS1..DS8) kontrollieren. Ändern und erweitern Sie das Programm so, dass die LED DS1 durch drücken der Taste SW1 eingeschaltet und durch drücken der Taste SW2 ausgeschaltet wird. Speichern Sie Ihre Lösung unter Termin2Aufgabe2.c. Sollten die Tasten nicht funktionieren, so überprüfen Sie ob im Power Management Controller (PMC) der Clock für PIOB eingeschaltet ist.

In welchen Registern müssen welche Bit gesetzt sein/werden, damit der Zustand der Tasten über das Pin Data Status Register (PDSR) erfasst werden kann?

..

Auf welchen Speicherstellen können Sie erkennen ob eine Taste gedrückt ist?

..

Aufgabe 3:

Erweitern Sie das nächste Programm so, dass zusätzlich die LED DS2 mit ca. 0,5Hz (einfache Zeitschleife programmieren) blinkt.

Was erwarten Sie, wie Tastendrucke an SW1 und SW2 reagieren?

..

Untersuchen Sie die Funktion ihres Programms auch mit verschiedenen Optimierungsstufen.

..

Aufgabe 4:

Schreiben Sie für die Tasten SW1 und SW2 eine passende Interruptserviceroutine. Erklären Sie die Wechsel der ARM-Betriebsmodi.

Wie reagieren nun Ihre Tastendrucke an SW1 und SW2?

..

Hat das Bedienen der Tasten Einfluss auf die Blinkfrequenz von DS2?

..

Sollten Sie in der Vorlesung noch kein Interrupt behandelt haben, kann diese Aufgabe auch nach Termin3 verschoben werden.

Aufgabe 5:

Erstellen Sie zu diesem Termin ein Protokoll mit den Lösungen zu den Aufgaben und Ihren Erkenntnissen. Das Protokoll sollen Sie zum nächsten Termin vorlegen können. Denken Sie daran, dass einige Programmteile nochmals benötigt werden könnten.

Programmgerüstbeispiele:

```
// Loesung zu Termin 2
// Aufgabe 1
// Namen: _____; _____
// Matr.: _____; _____
// vom : _____
//

#include "../h/pmc.h"
#include "../h/pio.h"

int main(void)
{
    StructPMC* pmcbase = PMC_BASE;           // Basisadresse des PMC
    StructPIO* piobaseB = PIOB_BASE;         // Basisadresse PIO B
    piobaseB->PIO_PER = ALL_LEDS;
    piobaseB->PIO_OER = ALL_LEDS;

    while(1)
    {
        piobaseB->PIO_CODR = ALL_LEDS;
        piobaseB->PIO_SODR = ALL_LEDS;
    }
    return 0;
}
```

```
// Loesung zu Termin2
// Aufgabe 4
// Namen: _____; _____
// Matr.: _____; _____
// vom : _____
//

#include "../h/pmc.h"
#include "../h/pio.h"
#include "../h/aic.h"..

void taste_irq_handler (void) __attribute__((interrupt));

void taste_irq_handler (void)
{
}

int main(void)
{
    return 0;
}
```