MIKROPROZESSORPRAKTIKUM

SS2021

Einführung

Einfuehrung zum Mikroprozessorpraktikum

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

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

Inhaltsverzeichnis

Laborordnung

Sinn dieser Laborordnung ist die Festlegung von Regeln für die Benutzung der Labore in D10

Jede/Jeder ordentliche Studierende des Fachbereich wird in den Grundlagenveranstaltungen auf die gültige Laborordnung hingewiesen.

1. Der Notausschalter des Labors D10/0.32 befindet sich über dem Lichtschalter. Der Notausschalter ist im Notfall zu betätigen, um sämtliche elektrische Geräte stromlos zu schalten. Achten Sie bei der Einführung auf die entsprechenden Hinweise.
2. Im Labor darf maximal Paarweise an den Geräten gearbeitet werden. Das Labor ist für 8 Gruppen (je zwei Personen) ausgelegt.
3. Es ist nicht gestattet sich alleine im Labor aufzuhalten.
4. Die Ersthelfer sind Herr Rudi Scheitler (Raum 0.33 / Tel.: 38465), Herr Manfred Pester (Raum 0.33 / Tel.: 38428) Herr Sergio Vergata (Raum 0.37 / Tel.: 38491) und Herr Michael Sander (Raum 00.15 / Tel.: 38509). Wenden Sie sich bitte im Falle einer Verletzung direkt an eine dieser Personen . Das Erste-Hilfe-Material befindet sich im Eingangsbereich Südseite.
5. Es ist nicht gestattet Kabel zu entfernen, Gehäuse zu öffnen und Hardware zu installieren oder Änderungen an der Laborinfrastruktur vorzunehmen. Sollte etwas nicht funktionieren, oder es wird etwas benötigt, welches die vorhandene Infrastruktur nicht abdeckt, so wenden Sie sich an den Betreuer des Labors oder direkt an den zuständigen Laboringenieur Manfred Pester (Raum 0.33 / Tel.: 38428).
6. Melden Sie sich von benutzten Geräten am Ende Ihres Praktikum/Ihrer Übung ab. Beachten Sie die vom zuständigen Betreuer erhaltenen Anweisungen.
7. Speisen und Getränke sind an den Arbeitsplätzen im Labor nicht erlaubt.
8. Bei der Benutzung des Labordrucker ist Sorgfalt und Sparsamkeit oberstes Gebot.
9. Evtl. ausgestellte Dokumentationen dienen der Laborarbeit und müssen im Raum verbleiben.
10. Die Benutzung von Mobiltelefonen ist untersagt. Schalten Sie vor dem Betreten des Raumes die Geräte ab (Flugzeugmodus ist ok). In dringenden Fällen können Sie sich über das Labortelefon mit der Nummer 06151 1638433 anrufen lassen.
11. Hängen Sie Ihre Kleidung (Mäntel, Jacken, ..) an die dafür vorgesehenen Kleiderständer und nicht über die Stühle.
12. Deponieren Sie Taschen, Laptops u.s.w. nicht in den Gängen, sondern möglichst an den Seiten des Labors oder unter den Tischen.
13. Verlassen Sie Ihren Arbeitsplatz aufgeräumt! Müll gehört in die mehrfach vorhandenen Mülleimer, Altpapier in die dafür vorgesehene blaue Altpapierwanne.
14. Die Fluchtwege sind frei zu halten.

Bei Verstößen gegen die Laborordnung kann die Benutzungsberechtigung versagt werden.

Einleitung

Dieses Dokument soll Ihnen eine kurze Starthilfe in das Mikroprozessorpraktikum geben. Es richtet sich an jene, die noch keinerlei Erfahrung mit hardwarenaher Programmierung haben.

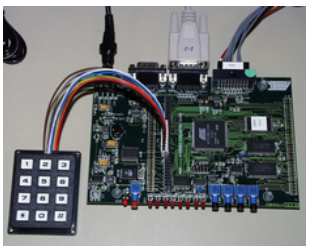
Zuerst werden wir kurz die Zielplattform betrachten, mit der Sie arbeiten werden, sowie die dazu notwendigen Entwicklungswerkzeuge in ihren wichtigsten Grundfunktionen vorstellen.

Bedenken Sie bitte, dass dieses Dokument keinesfalls ausreichend ist, um die Übungsbeispiele zu lösen. Datenblätter sind nun einmal primäre Quellen im Bereich der hardwarenahen Programmierung. Im Mikroprozessorpraktikum kann und soll Ihnen die Beschäftigung damit nicht erspart bleiben. Wir wissen aus eigener Erfahrung, dass es oft mühsam ist, aus den mitunter nicht optimal aufbereiteten einschlägigen Datenblättern die wichtigsten Informationen herauszuholen. Dieses heraus filtern der wesentlichen Informationen in möglichst kurzer Zeit ist aber definitiv eine Fertigkeit, die beim Einstieg ins Berufsleben von Ihnen verlangt werden wird. Sie werden in Ihrer späteren Arbeitsumgebung auch nicht allmonatlich auf Firmenkosten nach Singapur verschifft, um eine persönliche Einführung zum Bauteil des Monats zu bekommen; auch werden Ihre Kollegen mit Ihnen keine Freude haben, wenn Sie die Hardware ständig im Detail erklärt haben wollen.

Andererseits handelt es sich hier um eine Lehrveranstaltung, daher sind Sie natürlich nicht auf sich alleine gestellt. Sie sollen nicht tagelang an ein und demselben Problem sitzen und letztlich frustriert unsere Geräte zum Fenster hinaus werfen. Als erste Anlaufstelle gibt es in den betreuten Übungszeiten Personal, welches Ihnen entsprechende Hinweise geben kann. Bedenken Sie aber bitte, dass es sich bei diesem Personal auch um keine Übermenschen handelt.

Hardware

Um die Programmieraufgaben möglichst vielfältig und abwechslungsreich gestalten zu können, haben wir uns für das Evaluationsboard AT91EB63 der Firma ATMEL entschieden. Durch die Möglichkeit Peripherie an das Board anzuschließen, werden Sie in verschiedenen Versuchen, die Funktionalität der einzelnen Komponenten eines modernen Mikrocontroller besser kennenlernen.



Controllerboard

Das Herzstück der Übungshardware ist das Evaluationboard AT91EB63. Die relevanten Komponenten darauf sind,

- der Kontroller AT91M63200
- zwei serielle Schnittstellen

  Die HOST-Schnittstelle wird zur Verbindung mit einem Rechner zwecks Kommunikationsschnittstelle genutzt.
- ein Resettaster
- vier 4 Taster
- acht Leuchtdioden
- 256KByte statischer Speicher (RAM)
- 2MByte Flash
- 2MByte serielles DatenFlash
- 64KBytes serielles EEPROM
- 2*32-pin EBI Erweiterungsverbinder
- 2*32-pin MPI Erweiterungsverbinder
- 2*32-pin Ein/Ausgabe Erweiterungsverbinder

  Hieran können angeschlossen werden:
  - Über ein Interface eine Saitenschwingwaage
  - Über ein Interface eine Kolbenhubpumpe
  - Ein Tastenfeld mit 3*4 Tasten
  - Eine Box (mit AVR-Mikrokontrollersystem) zur Simulation der Saitenschwingwaage und der Pumpe
- 20-pin JTAG Schnittstellenverbinder

Hieran angeschlossen ist im Labor ein BDI2000 Echtzeitemulatorsystem. Über den BDI2000 wird die Verbindung über das Netzwerk (Ethernet 10Mbit) zum Entwicklungsrechner hergestellt.

Für weitere Informationen schauen Sie im Benutzer Handbuch „AT91EB63 Evaluation Board" nach.

Der Mikrokontroller AT91M63200

Der AT91M63200 ist ein Mitglied der Atmel AT91 16/32-Bit Mikrokontrollerfamilie auf Basis des ARM7TDMI Prozessorkern. Der Prozessor hat eine hochperformante 32-Bit RISC-Architektur, unterstützt den hocheffizienten 16-Bit Befehlssatz (THUMB) und hat eine sehr geringe Verlustleistung. Um nicht alle Besonderheiten des Kontroller hier aufzählen zu müssen, verweisen wir an dieser Stelle auf die technische Dokumentation der Firma

ATMEL
AT91
ARM® Thumb®
Microcontrollers
AT91M63200

Rev. 1028A-11/99

Entwicklungsumgebung ARM-Toolchain und Aufgaben

Im Praktikum zur Rechnerarchitektur lernten wir eine ARM-Toolchain kennen. Mit dieser werden wir auch im Mikroprozessor-Praktikum arbeiten. Zur Wiederholung und Auffrischung beschäftigen Sie sich mit den Aufgaben von Termin 1 zum Praktikum Mikroprozessorsysteme SS2021. Diese Aufgaben können im ARM-Simulator des GDB (arm-eb63-elf-insight) getestet werden. Die Labore D10/0.32 und evtl. D10/0.35 stehen Ihnen während der freien Übungszeiten hierfür zur Verfügung.

Sie möchten auf Ihrem eigenen Rechner arbeiten?
Besorgen Sie sich das angebotene aktuelle Image für RA und MPS. Dies finden Sie unter z:B.:
https://userv.fbi.h-da.de/LabDisk/MI/images/RA_MPS_SS2021.ova oder auf

userv-shell.fbi.h-da.de/share/LabDisk/MI/images/RA_MPS_SS2021.ova.

Auch im Internet oder im Wiki
(https://wiki.h-da.de/fbi/technische-systeme/index.php/Mikroprozessorsysteme_Historie#16._Oktober_2012_neues_Debian-Paket)
des Fachbereich Informatik findet man Informationen um sich eine eigene Toolchain zu installieren.

Zu jedem Termin für das Praktikum Mikroprozessorsysteme SS2021 gibt es einen Ordner mit Informationen und Beispielen. Sie müssen Ihre Praktikumstermine wahrnehmen. Nur im Labor für Mikroprozessorsysteme D10/0.32 steht Ihnen zur Lösung der Aufgaben Termin 1 bis Termin 6 die notwendige Hardware zur Verfügung.

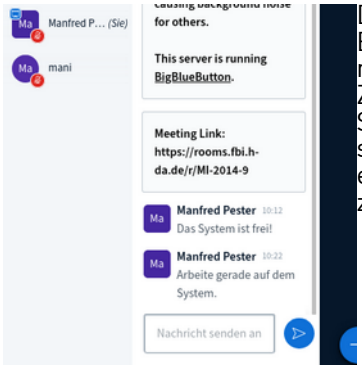
In Zeiten der Pandemie wurde die Laborinfrastruktur so erweitert, dass diese auch von externen Arbeitsplätzen nutzbar ist.

Pandemie bedingter externer Laborzugriff

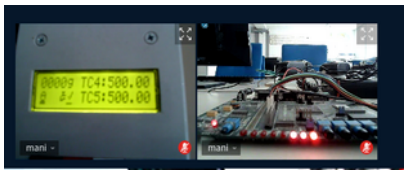
Es ist möglich die im Labor D10/00.32 aufgebauten, zur Verfügung gestellten Entwicklungssysteme auch über eine Netzwerkverbindung zu nutzen.

Hierzu schauen Sie unter:

https://rooms.fbi.h-da.de bei den persönlichen Räumen nach, ob Systeme MI-2014-x aktiviert sind. Gehen Sie in einen einem System zugehörigen Raum. Während der im Stundenplan/Raumplan angebotenen Laborzeiten (Praktika, offenen Labore) sollten auch Kamerabilder der Systeme (WaSim, LED's) übertragen werden. Im öffentlichen Chat sind Informationen über die Nutzung des zugehörigen Systems zu hinterlassen.

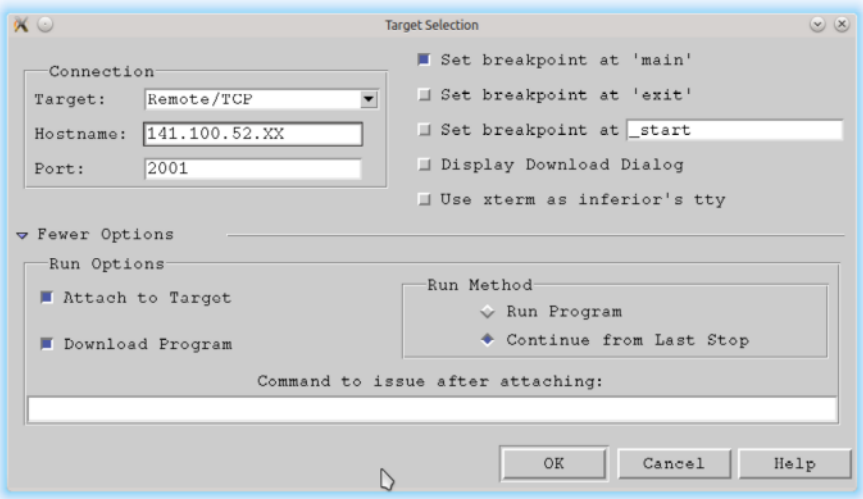


Die Nutzung eines Entwicklungssystem ist nur einmal möglich. Sollte es zu parallelen Zugriffen kommen, können die Systeme sich aufhängen. In einem solchen Fall muss das System durch einen Laborverantwortlichen zurückgesetzt werden.



In Zeiten (Blöcken) von offiziellen Praktikumsterminen und/oder zu offenen Laborterminen sollten Sie im Raum D10/00.32 (https://rooms.fbi.h-da.de/r/D10/00.32) Menschen antreffen die Ihnen gerne helfen.

Werden die Systeme im Labor gestartet hat die Verbindung und die Initialisierung der Entwicklungssysteme funktioniert, wenn aus dem Lauflicht ein statisches Leuchten der rechten 3 LED (DS6 bis DS8) wurde. Sie können sich aus dem Debugger (arm-eb63-elf-insight) mit einem Target Remote/TCP mit einem System (AT91EB63) verbinden. Wählen Sie für den Hostnamen (z.B. MI-2014-x.fbi.h-da.de) oder die IP-Adresse (z.B. 141.100.52.x) des zu verwendenden Systems (x = Nummer des Systems). Als Port wählen Sie die 2001. Weitere Einstellungen entnehmen Sie der folgenden Abbildung.



ACHTUNG: Für eine Verbindung von zu hause aus ist zuvor ein VPN zu aktivieren.

Es darf immer nur eine Verbindung pro System bestehen.

Hinweis zu den Aufgaben / Termine

Bei den folgend aufgeführten Aufgaben kann es vorkommen, dass es sich nicht um die aktuellste Version handelt.

Die Aufgaben werden immer weiter entwickelt. Es können in den Aufgaben für das aktuelle Semester also Änderungen vorgenommen worden sein.

Bitte besorgen Sie sich die aktuellen Aufgaben für das Labor bei Ihrem zuständigen Professor oder dem Laboringenieur.

Beachten Sie die Hinweise in den Vorlesungen oder der ersten Laborveranstaltung.

Termin 1

Lernziele:

Mit den folgenden Versuchen sollen Sie die Sprache "C" einmal aus einer anderen Sicht kennen lernen. An ganz einfachen Programmen sollen Sie ermitteln, welchen Code ein Compiler erzeugt, wo welche Variablen abgelegt werden, welchen Einfluss die Optimierungsstufen haben, wie ein c*all by value / reference* in ARM Assembler umgesetzt wird. Wie auf Peripherie (System on Chip) zugegriffen wird.

Arbeitsverzeichnis:

Kopieren Sie sich das Verzeichnis mpsSS2021. Dort stehen im Unterverzeichnis Termin1 die Dateien *Termin1AufgabeX.c als* Programmgerüstbeispiele und *makefile* zur Verfügung.

Die Aufgaben 1 bis 4 dieses Termins können wie in der Veranstaltung Rechnerarchitektur gelernt auch im Target Simulator (also ohne die im Labor zur Verfügung gestellte Hardware) gelöst werden.

Aufgabe 1:

Legen Sie im C Programm zwei lokale integer Variablen an. Eine Variable initialisieren Sie mit einem Wert. Weisen Sie der zweiten Variablen im Code einen Wert zu. Übersetzen Sie Ihr Programm und schauen Sie sich den erzeugten Maschinen-Code (Assembler) an. Führen Sie das Programm im Debugger aus. Dokumentieren Sie den erzeugten Code.

Hinterfragen Sie Ihre Beobachtungen z. B. mit:

- Wo liegen die Werte der Variablen im Speicher?
- Wie kommen die Werte in den Speicher?
- Welche Register werden verwendet?
- Wie wird auf die Variablen zugegriffen?
- ..

Ändern Sie im Makefile die Optimierungsstufe und beobachten und dokumentieren Sie die Veränderungen.

Aufgabe 2:

Verwenden Sie nun statt der lokalen Variablen globale (initialisierte und nicht initialisierte) Variablen.

- Was ändert sich?
- ..

Ändern Sie auch nun die Optimierungsstufe und beobachten und dokumentieren Sie die Veränderungen.

Aufgabe 3:

Legen Sie nun zwei globale und zwei lokale integer Variablen an. Weisen Sie den Variablen Werte zu. Machen Sie nun Zuweisungen der Form „global=lokal" und/oder „lokal=global".

- Was stellen Sie fest?
- Wo liegen die Werte der Variablen im Speicher?
- ..

Ändern Sie im Makefile die Optimierung und beobachten und dokumentieren Sie die Auswirkungen.

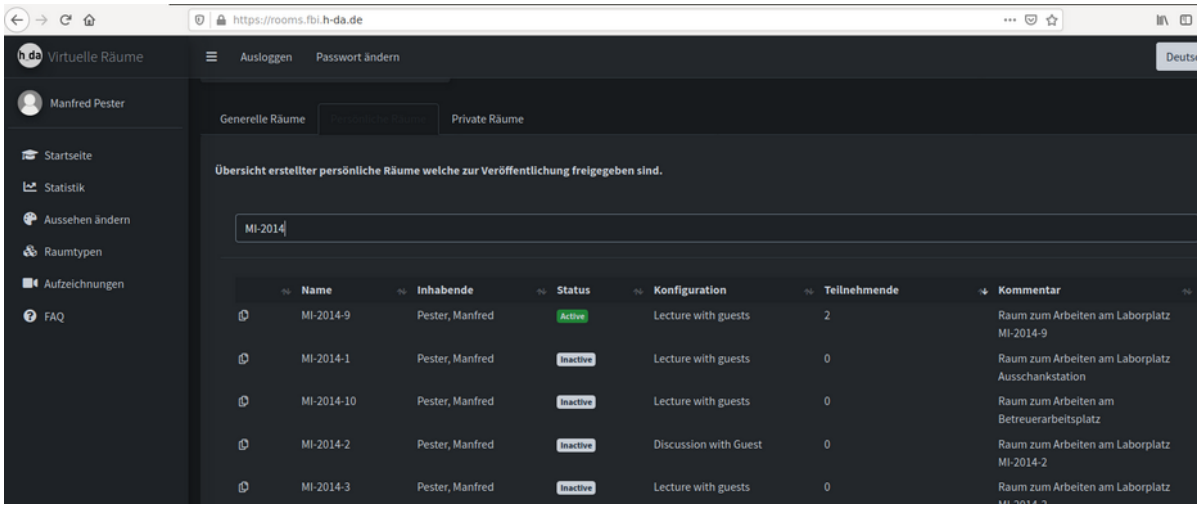
Aufgabe 4:

Vereinbaren Sie einen Funktionsprototypen für eine Funktion "int addition(int, int, int)", die 3 integer Parameter erwartet. Rufen Sie diese Funktion im Anschluss an die Zuweisung nach Aufgabe 2 und 3 mit den beiden lokalen und einer globalen Variablen auf. Dokumentieren Sie Ihre Beobachtungen. Ändern Sie im Makefile die Optimierung und beobachten und dokumentieren Sie die Auswirkungen.

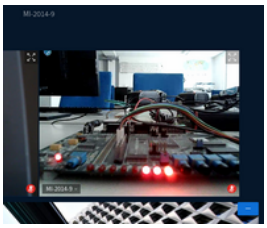
Infos während Pandemie-Semester

Es ist natürlich etwas mühsam/umständlich, aber machbar, die im Simulator nicht vorhandenen LED DS1-DS8, Tastendrücke KEY1 bis KEY3 durch sichten und manipulieren der Registerinhalte zu simulieren. Statt sich mit dem Target Simulator zu verbinden, können Sie im Mikroprozessorlabor auch die ARM7-Evaluationssysteme (AT91EB63) nutzen. Hierzu müssen die benötigten Systeme PC's und zugehöriges Entwicklungssysteme im Labor eingeschaltet/aktiv sein.

Schauen Sie nach, welches der Systeme im Labor aktiv ist. Dies ist über https://rooms.fbi.h-da.de möglich.

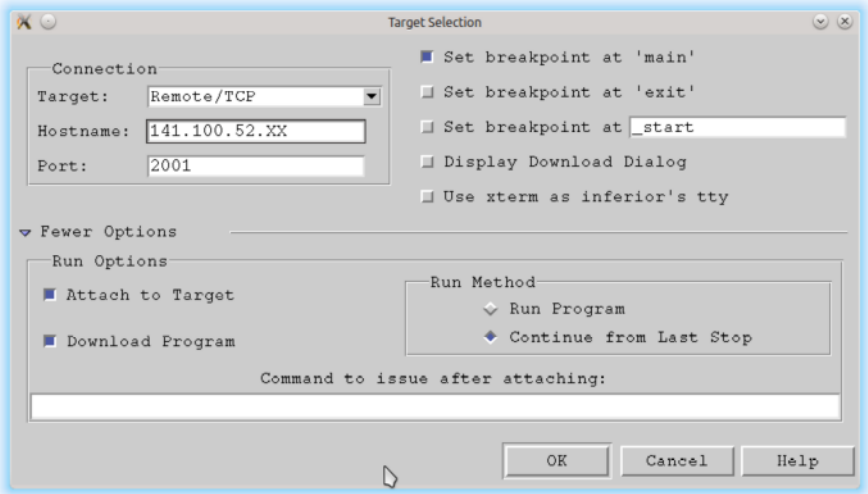


Gehen Sie in den zugehörigen virtuellen Raum (z.B. MI-2014-9) und schauen/kontrollieren Sie ob das System gerade genutzt wird.



In Zeiten (Blöcken) von offiziellen Praktikumsterminen und/oder zu offenen Laborterminen sollten Sie im Raum D10/00.32 (https://rooms.fbi.h-da.de/r/D10/00.32) Menschen antreffen die Ihnen gerne helfen.

Werden die Systeme im Labor gestartet hat die Verbindung und die Initialisierung der Entwicklungssysteme funktioniert, wenn aus dem Lauflicht ein statisches Leuchten der rechten 3 LED (DS6 bis DS8) wurde. Sie können sich aus dem Debugger (arm-eb63-elf-insight) mit einem Target Remote/TCP mit einem System (AT91EB63) verbinden. Wählen Sie für den Hostnamen (z.B. MI-2014-9.fbi.h-da.de) oder die IP-Adresse (z.B. 141.100.52.9) des zu verwendenden Systems. Als Port wählen Sie die 2001. Weitere Einstellungen entnehmen Sie der folgenden Abbildung.



ACHTUNG: Für eine Verbindung von zu hause aus ist zuvor ein VPN zu aktivieren.

Es darf immer nur eine Verbindung pro System bestehen.

Aufgabe 5:

Wir werden uns nun mit Registern der PIO des hier eingesetzten Mikrocontroller (siehe auch Doku AT91M6320.pdf) beschäftigen.

Name	Adresse	Bedeutung
PIOB_PER	0xFFFF0000	PIOB Port Enable Register
PIOB_OER	0xFFFF0010	PIOB Output Enable Register
PIOB_SODR	0xFFFF0030	PIOB Set Output Data Register
PIOB_CODR	0xFFFF0034	PIOB Clear Output Data Register

Schreiben Sie zunächst den Wert 0x100 ins PIOB_PER und dann ins PIOB_OER. Danach schreiben Sie nacheinander den Wert 0x100 einige Male alternierend ins PIOB_SODR und PIOB_CODR. Dadurch sollte die LED DS1 auf dem Board AT91EB63 an und aus gehen. Testen Sie die Funktion im Debugger mit Einzelschritten aus. Im gegebenen Programmbeispiel Termin1Aufgabe5.c sind verschiedene Möglichkeiten des Beschreibens/Lesens der Register der PIO gezeigt. Beschäftigen Sie sich mit den verschiedenen Programmiermöglichkeiten.

- Welche Variante würden Sie bevorzugen und warum?
- Beobachten Sie wann die LED an und aus geht. Schauen Sie sich hierzu die Doku an. Beschreiben Sie Ihre Beobachtungen/Erkenntnisse.

Aufgabe 6:

Versuchen Sie nun Ihr Programm auf minimale Programmgröße zu bringen.

- Wie viele Byte Speicher benötigen Sie für die Initialisierung und um die LED in einer Endlosschleife blinken zu lassen?
- Aus wie vielen Assemblerbefehlen besteht Ihr minimiertes Programm?

Aufgabe 7:

Erstellen Sie zu diesem Termin ein Protokoll mit den Lösungen zu den Aufgaben und Ihren Erkenntnissen. Das Protokoll sollen Sie zu den nächsten Terminen vorlegen können. Denken Sie auch daran, dass Sie Programmteile in den nächsten Praktika nochmals benötigen. Beschäftigen Sie sich baldigst auch mit den Aufgaben der nächsten Termine. Im letzten Termin sollten Sie in der Lage sein, die gestellte Projektaufgabe zu realisieren.

Beispiel eines makefile zu Termin1 Aufgabe1

```
# zu verwendete Quelldatei
FILE = Termin1Aufgabe1
# Toolchain
TOOLCHAIN = arm-eb63-elf-
# Compiler
COMPILER = gcc
# Linker/Binder
LINKER = ld
# Debugger
DEBUGGER = insight
# Optimierungsstufe
OPTI = 0

# Bauen
all:
# uebersetzen der Quelldatei (FILE).c
# Der Schalter -c erzeugt nur die Objektdatei aus der Quelldatei ohne zu binden
# Der Schalter -g in gcc fügt Debugging-Code in die kompilierten Objektdateien ein
# Der Schalter -O gibt die zu verwendete Optimierungsstufe (0..3,s) an
# Der Schalter -I weist gcc an, das Verzeichnis include in den Include-Pfad einzufügen
# Der Schalter -L weist gcc an, das Verzeichnis lib in den Library-Pfad einzutragen.
# Der Schalter -S weist gcc an, eine Datei mit Assemblercode zu erzeugen.

	$(TOOLCHAIN)$(COMPILER) -c -g -O$(OPTI) $(FILE).c -I ../h

# Der Schalter -S erzeugt eine Assemblerdatei aus der Quelldatei
	$(TOOLCHAIN)$(COMPILER) -S -O$(OPTI) $(FILE).c

# Erzeugen weitere benoetitgen Objektdateien
	$(TOOLCHAIN)$(COMPILER) -c -g -O$(OPTI) ../boot/swi.S -o swi.o -I ../h
	$(TOOLCHAIN)$(COMPILER) -c -g -O$(OPTI) ../boot/boot_ice.S -o boot_ice.o -I ../h

# Binden fuer die RAM-Version
	$(TOOLCHAIN)$(LINKER) -Ttext 0x02000000 boot_ice.o swi.o $(FILE).o -o $(FILE).elf

# Debugger starten
debug:
	$(TOOLCHAIN)$(DEBUGGER) $(FILE).elf

# Aufraeumen
clean:
	rm *.o
	rm $(FILE).s
	rm $(FILE).elf

# BDI2000 zuruecksetzen (funktioniert nur wenn auf dem zugehörigen PC gearbeitet wird)
bdireset:
	bash bdi_reset
```

Termin2

Lernziele:

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

Arbeitsverzeichnis:

Kopieren Sie sich aus dem Ordner userv-shell.fbi.h-da.de/share/LabDisk/MI/" den Ordner mpsSS2021. 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/zu 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 und/oder in welchen Registern 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 und/oder in welchen Registern 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 Tastendrücke 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 Tastendrücke 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.

Beschäftigen Sie sich auch schon mit der Aufgabenstellung von Termin6.

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;
}
```

Termin 3

Lernziele:

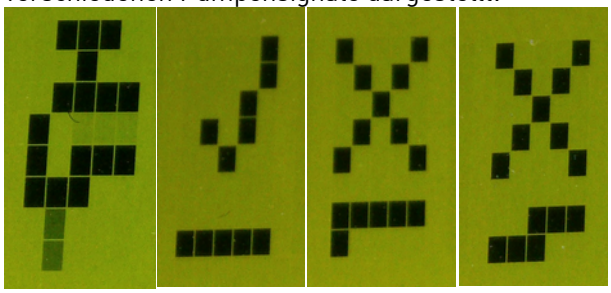
Mit den folgenden Versuchen sollen Sie lernen, wie Sie aus der Sprache "C" Peripherie (z. B. Timer) von modernen Mikrocontrollern nutzen.

Arbeitsverzeichnis:

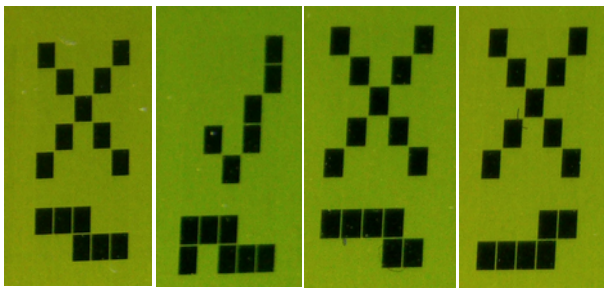
Kopieren Sie sich aus dem Ordner „sftp://stxxyyyy@userv-shell.fbi.h-da.de/share/LabDisk/MI/" den Ordner mpsSS2021. Dort finden Sie zu jedem Termin vorgegebene Dateien.

Weitere Infos:

Ab dem Sommersemester 2011 stehen an jedem Laborarbeitsplatz WaSim (Waagensimulatoren) zur Verfügung. Mit diesen angeschlossenen WaSim kann auch das Pumpensignal überprüft werden. Im Display werden mit folgenden Symbolen die verschiedenen Pumpensignale dargestellt:



es pumpt Gewicht nimmt zu | *kein Signal* | *dauer high* | *Frequenz zu hoch*



Frequenz zu niedrig | *Frequenz richtig* | *Highpegel zu lang* | *Lowpegel zu lang*

Aufgabe 8:

Es soll eine Kolbenhubpumpe, welche über PA1/TIOA3 angesteuert wird, betrieben werden. Die Pumpe benötigt ein symmetrisches Rechtecksignal mit einer Frequenz von ca. 50Hz. Sie könnten eine Zeitschleife programmieren. Hiermit würden Sie aber den Prozessor blockieren (Erinnerung an Termin 2). Besser ist es, Sie initialisieren einen Timer (Timer3) so, dass dieser selbstständig das Signal für die Pumpe an TIOA3 erzeugt.

ACHTUNG: Die Pumpe darf kein Dauerhighsignal erhalten.

Vervollständigen Sie die das gegebene Programm *Termin3Aufgabe1.c* entsprechend. Ergänzen und berichtigen Sie auch die Kommentare.

Könnte der eingesetzte Timer bei dem benötigten symmetrischen Signal auch anders betrieben werden?

Zeigen Sie die Berechnung der benötigten Werte um die Frequenz und das Pulsweitenverhältnis richtig einzustellen.

..

Aufgabe 9:

Erweitern Sie Ihr Programm so, dass die Pumpe durch Betätigung von Tasten eingeschaltet und abgeschaltet werden kann.

Welche Möglichkeiten haben Sie gefunden, um das Pumpensignal ein- bzw. auszuschalten?

..

Für welche Lösung entscheiden Sie sich und warum?

..

Aufgabe 10:

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. Isolieren Sie die Routinen/ Funktionen, welche für die nächsten Termine (siehe speziell Termin6), noch benötigt werden.

```
// Loesung zu Termin 3
// Aufgabe 1
// von:
// vom:
//
#include "../h/pmc.h"
#include "../h/tc.h"
#include "../h/pio.h"
#include "../h/aic.h"

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

// Interruptserviceroutine für die Tasten SW1 und SW2
void taste_irq_handler (void)
{
 StructPIO* piobaseB   = PIOB_BASE;                 // Basisadresse PIO B
 StructAIC* aicbase    = AIC_BASE;                  //__

// ab hier entsprechend der Aufgabestellung ergänzen
// ************************************************


        aicbase->AIC_EOICR = piobaseB->PIO_ISR;     //__
}

// Timer3 initialisieren
void Timer3_init( void )
{
        StructPMC* pmcbase = PMC_BASE;              // Basisadresse des PMC
        StructTC* timerbase3  = TCB3_BASE;          // Basisadressse TC Block 1
        StructPIO* piobaseA   = PIOA_BASE;          // Basisadresse PIO B

        pmcbase→PMC_PCER = 0x2200;                  // Peripheral Clocks einschalten für PIOA und TC3

        timerbase3->TC_CCR = TC_CLKDIS;             // Disable Clock

 // Initialize the mode of the timer 3
        timerbase3->TC_CMR =
                TC_ACPC_CLEAR_OUTPUT  |     // ACPC         : Register C clear TIOA
                TC_ACPA_SET_OUTPUT    |     // ACPA         : Register A set TIOA
                TC_WAVE               |     // WAVE         : Waveform mode
                TC_CPCTRG             |     // CPCTRG       : Register C compare trigger enable
                TC_CLKS_MCK1024;            // TCCLKS       : MCKI / 1024

 // Initialize the counter:
        timerbase3->TC_RA = 400;    // hier sind noch die richtigen Werte zu ermitteln
        timerbase3->TC_RC = 800;    // Die Pumpe soll mit einem 50Hz Signal betrieben werden

 // Starten des Timer und Ausgabe eines Low-Pegel an die Pumpe
        timerbase3->TC_CCR = TC_CLKEN ;     //__
        timerbase3->TC_CCR = TC_SWTRG ;     //__
        piobaseA->PIO_PER  = (1<<PIOTIOA3) ;  //__
        piobaseA->PIO_OER  = (1<<PIOTIOA3) ;  //__
        piobaseA->PIO_CODR = (1<<PIOTIOA3) ;  //__
}

int main(void)
{
        StructPMC* pmcbase = PMC_BASE;      // Basisadresse des PMC
        StructPIO* piobaseA = PIOA_BASE;    // Basisadresse PIO A
        StructPIO* piobaseB = PIOB_BASE;    // Basisadresse PIO B


// ab hier entsprechend der Aufgabestellung ergänzen
//************************************************


        return 0;
}
```