h_da

HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

fbi
FACHBEREICH INFORMATIK

MIKROPROZESSORPRAKTIKUM

SS2022

Termin1

C-Programmierung für eingebettete Systeme

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

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

Laborordnung

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

Jeder ordentliche Student des Fachbereich wird in den Grundlagenveranstaltungen auf die gültige Laborordnung und die Brandschutzregeln 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 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) und Herr Sergio Vergata (Raum 0.37 / Tel.: 38491). 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 (außer USB-Sticks) 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. Fahren Sie die von Ihnen benutzten Geräte am Ende Ihres Praktikum/Ihrer Übung herunter und schalten diese aus, es sei denn Sie bekommen vom zuständigen Betreuer andere 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 168433 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.

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 mpsSS2022. 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.

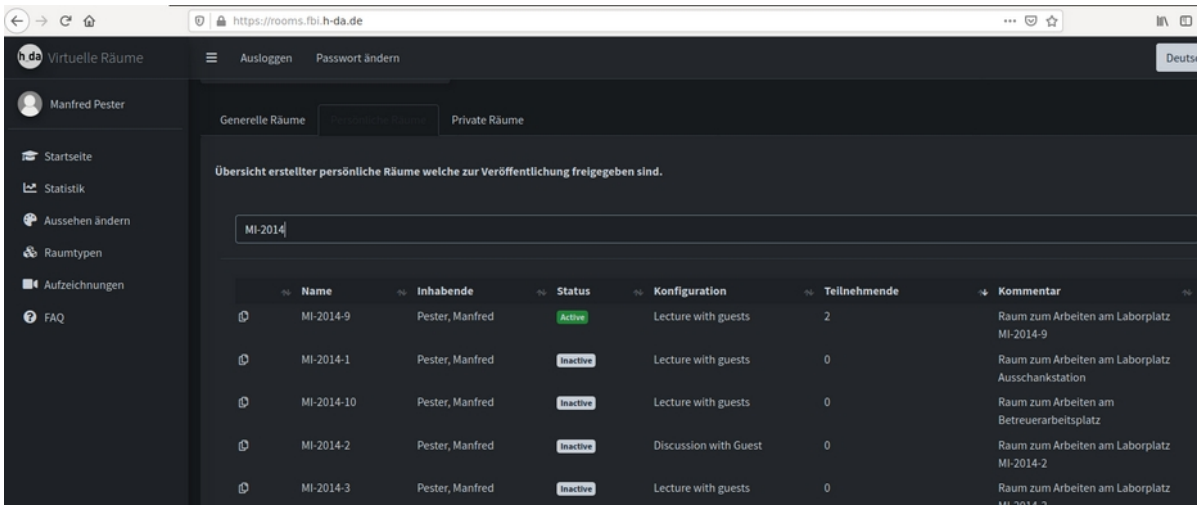
Aufgabe 4:

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.

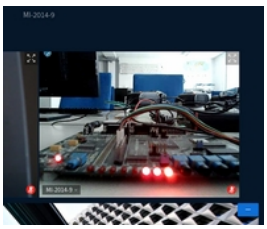
Infos während Pandemie-Semester

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.

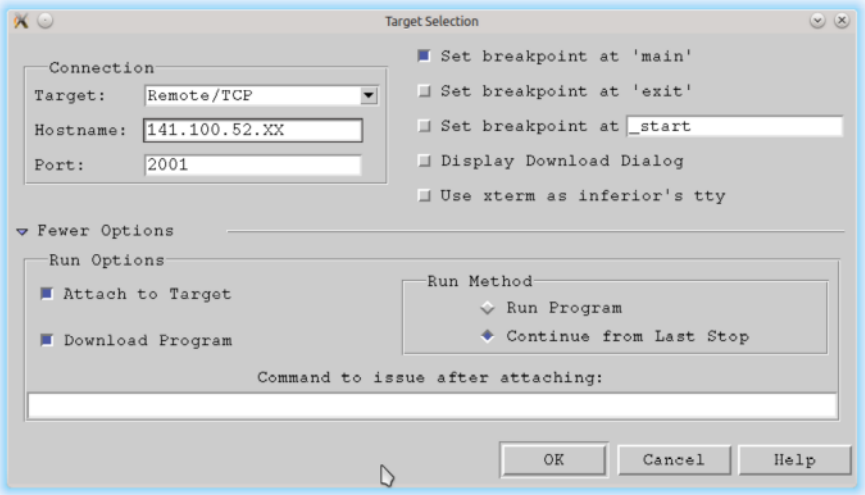


Gehen Sie in den zu einem Laborarbeitsplatz zugehörigen virtuellen Raum (z.B. MI-2014-9) und schauen/kontrollieren Sie ob das System gerade genutzt wird.



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 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 AT91M63200.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 Zusatzaufgabe:

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
```