h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

fbi
FACHBEREICH INFORMATIK

RECHNERARCHITEKTUR

SS2017

Termin 6

Speicherkonzept, Datenaufteilung, initialisierte und unitialisierte Variablen, Konstanten, Assemblersyntax (global, lokal, labels)

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

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

Ziele:

Das Anlegen von initialisierten und nicht initialisierten Daten. Die Sektionen für Programm (.text) Daten (.data) und die uninitialisierte Sektion (.comm). Der Zugriff auf Daten in verschiedenen Sektionen und Zeiger auf Daten.

Arbeitsverzeichnis:

Kopieren Sie sich das Verzeichnis, welches Ihnen im Praktikum zur Verfügung gestellt wird, in Ihr persönliches Arbeitsverzeichnis. Dort stehen Ihnen dann alle benötigten Dateien, um die Aufgabenstellung lösen zu können, zur Verfügung.

Aufgabenstellung

Es soll eine verkettete Liste von initialisierten Datenstrukturen (Senderspeicher) in der Data-Section (.data) angelegt werden. Im Programm soll eine weitere uninitialisierte Datenstruktur (weiterer Senderspeicher) initialisiert und an die verkettete Liste eingefügt werden.

Aufgabe

Legen Sie in der Datensektion eine verkettete Liste mit vier Elementen (Senderspeicher1 .. 4) mit folgender Struktur an.

- Zeiger auf die nächste Senderspeicherstelle
- Zeiger auf eine 32 Bit große vorzeichenlose Zahl für Senderfrequenz
- Zeiger auf einen String für Sendername
- Zeiger auf ASCII-Zeichen für Speichertaste

Reservieren Sie im uninitialisierten Bereich (.comm) Speicher für ein weiteres Element (Senderspeicher5) mit der beschriebenen Struktur.

Schreiben Sie in Assembler ein Programm, welches das neue Element Senderspeicher5 mit den folgenden Werten initialisiert.

- Zeiger auf Senderspeicher1
- Zeiger auf Frequenz (500)
- Zeiger auf Sendername („Mein Sender“)
- Zeiger auf Zeichen der Speichertaste („E“)

Fügen Sie das neue Element dann an die verkettete Liste an/ein.

Vorbereitung:

Assembler Direktiven zum Anlegen von Daten. Labels. Symbolische Konstanten. Die Sektionen eines Assembler Programms. Speicherung von Konstanten und Strings. Nutzung von Konstanten in Assembler Programmen. Direkte und indirekte Adressierung mit und ohne Offset. Pointer. Alignement von Daten.

Testat:

Nachweis der Funktion des Programms durch Nutzung der gegebenen Entwicklungswerkzeuge.

Erstellen Sie eine Speicherkarte Ihres Programm und der zugehörigen Daten wie Sie diese im Simulator vorfinden.

Kurzes Fachgespräch am Anfang (Vorbereitung) und am Ende der Veranstaltung.

```
// Vorschlag eines Programmgeruest zur Aufgabe Termin6 SS2017
        .file   "term6.S"
// Daten zu den vier vorgegebenen Sendern sind im ROM abgelegt
        .section .rodata
        .align  2       @ Anfangsadresse durch vier teilbar
// Sendernamenliste
Sender1:        .ascii  "HR3\0"
        .align  2
Sender2:        .ascii  "SWR3\0"
        .align  2
```

```
Sender3:        .ascii    "Deutschlandfunk\0"
        .align    2
Sender4:        .ascii    "HR4\0"

// Senderfrequenzliste
Frequenz1:      .word           100
Frequenz2:      .word           200
Frequenz3:      .word           300
Frequenz4:      .word           400
// Speichertasten
Taste1:  .byte          'A'
Taste2:  .byte          'B'
Taste3:  .byte          'C'
Taste4:  .byte          'D'
Taste5:  .byte          'E'
// Die Senderspeicher mit entsprechender Struktur im initialisierten Datenbereich
        .data
        .align    2

        .type     Senderspeicher1, object
        .size     Senderspeicher1, 16
Senderspeicher1:
        .word     Senderspeicher2 @ Zeiger auf nächstes Element
        .word     Frequenz1               @ Zeiger auf Frequenz1
        .word     Sender1         @ Zeiger auf Sender1
        .word     Taste1          @ Zeiger auf zu belegende Taste

        .type     Senderspeicher2, object
        .size     Senderspeicher2, 16
Senderspeicher2:
        .word     Senderspeicher3 @ Zeiger auf nächstes Element
        .word     Frequenz2               @ Zeiger auf Frequenz2
        .word     Sender2         @ Zeiger auf Sender2
        .word     Taste2          @ Zeiger auf zu belegende Taste

        .type     Senderspeicher3, object
        .size     Senderspeicher3, 16
....
....
        .text
        .align    2
        .global   main
        .type     main,function
main:
// Hier das geforderte Programm einfuegen
....
        mov       pc, lr
// Speicher für einen 5ten Senderspeicher reservieren
        .comm     Senderspeicher5, 16
// Programmende
```