UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR TECHNISCHE INFORMATIK

Übungsblatt 8

Stack und Unterprogramme

Vorlesung *Technische Grundlagen der Informatik 1*, Sommersemester 2020
Erstellt von Dr.-Ing. Kristian Ehlers

Verwenden Sie für Ihre Implementierungen einen beliebigen Editor und simulieren Sie ihren Assembler-Code anschließend im **ITImega16-Simulator**. Ihnen steht im Moodle für die Übung entsprechendes **Template** zur Verfügung.

Es seien hiermit noch folgende Anmerkung zur Nutzung des **ITImega16-Simulators** gegeben:

- Um die Namensdefinitionen wie z.B. DDRA, PORTA, PINA usw. nutzen zu können, müssen diese mithilfe des Includes `.include "m16def.inc"` bereitgestellt werden.
- Die Konfigurationsdateien der Peripherie müssen auf `layout.js` enden.
- Die Direktiven `.list` bzw. `.nolist` erlauben es, nachfolgenden Quellcode im Simulator anzuzeigen oder auszublenden.

Aufgabe 1 Rekursive Berechnung der Fibonacci-Zahlen

Auch in dieser Aufgabe sollen die Ihnen bereits bekannten Fibonacci-Zahlen mit Hilfe des ATmega16 berechnet werden. Zur Erinnerung sei die Definition der unendlichen Folge der Fibonacci-Zahlen hier nochmal gegeben:

$$F(0) = 0,\ F(1) = 1,\ F(n) = F(n-1) + F(n-2) \quad \forall n \in \mathbb{N} \wedge n \geq 2.$$

In dieser Übung soll ein rekursiver Algorithmus zur Berechnung der Fibonacci-Zahlen implementiert werden. Der folgende Pseudocode gibt eine entsprechende Rechenvorschrift an:

```
byte FIB_REKURSIV(byte Prev, byte Preprev, byte K)
{
  if(K = 0)
  {
    return Preprev;
  }
  else
  {
    return FIB_REKURSIV(Prev + Preprev, Prev, K-1);
  }
}
```

Die beiden ersten Parameter enthalten sozusagen die vorherigen beiden Fibonacci-Zahlen. Um die k-te Fibonacci-Zahl zu berechnen, muss die im Pseudocode beschriebene Funktion folgendermaßen aufgerufen werden: $F(k) =$ `FIB_REKURSIV`$(1, 0, k)$.

(a) Anlegen eines Projektes und Initialisierungen

Definieren Sie sich gegebenenfalls verwendete Konstanten oder Namensdefinitionen einzelner Register. Für die Rückgabe des Ergebnisses soll das später zu realisierende Unterprogramm das Register `R20` unter dem Namen `Result` nutzen.

(b) Implementierung des Hauptprogramms

Beginnen Sie nun die eigentliche Programmierung. Dem in der nachfolgenden Teilaufgabe zu implementierenden Unterprogramm `FIB_REKURSIV` sollen seine Parameter in Registern übergeben werden. Nehmen Sie die initiale Belegung dieser Register zu Beginn des Hauptprogramms `MAIN` vor. Rufen Sie anschließend das Unterprogramm `FIB_REKURSIV` auf und verharren Sie nach der Berechnung und Rückkehr in das Hauptprogramm in einer Endlosschleife.

(c) Implementierung des Unterprogramms

Implementieren Sie nun das Unterprogramm `FIB_REKURSIV`. Dieses soll sich am oben gegebenen Pseudocode orientieren und die Berechnung der k-ten Fibonacci-Zahl realisieren. Die Parameterübergabe für das Unterprogramm soll über Register erfolgen und das Resultat vom Unterprogramm im `Result` Register zurückgegeben werden. Sorgen Sie dafür, dass alle im Unterprogramm genutzten Register einschließlich der Parameter-Register nach Rückkehr zum Hauptprogramm dieselben Werte wie vor dem Aufruf des Unterprogramms haben. Das einzige nach außen (außerhalb des Unterprogramms) veränderte Register ist das `Result` Register.

Simulieren Sie das Verhalten mit dem **ITImega16-Simulator** und verfolgen Sie die einzelnen Schritte im Speicher.