Hardwarenahe Softwareentwicklung
Frühlingssemester 2023
Roger Weber
Klasse E2a

Übung 7

Arithmetische und logische Instruktionen, Programmverzweigungen ARM V7M

Ziele dieser Übung:
Sie sollen in der Lage sein,

1) Arithmetische und logische Instruktionen eines Cortex-Mx Prozessors anzuwenden.
2) Sprungbefehle eines Cortex-Mx Prozessors anzuwenden.

Verwenden Sie für den Test Ihres Codes den Debugger und das Leguan-Board. Beachten Sie bei der Programmausführung die Register, die Statusbits und das Memory.

Aufgabe 7.1 Arithmetische und logische Instruktionen

Aufgabe 7.1a) ADD Instruktion

- Laden Sie die Werte 10 in r2 und 100 in r1
- r0 = r1 + r2
- r0 = r1 + (4 * r2)
- r0 = r1 + r2 + Carry

Aufgabe 7.1b) SUB Instruktion

- r0 = r1 – r2
- r0 = r1 – (2 * r2)
- r2 = 2
- r2 dekrementieren, welchen Wert haben die Statusbits?
- r2 nochmals dekrementieren, welchen Wert haben die Statusbits jetzt?

Aufgabe 7.1c) Multiplikations-Instruktionen

- r2 = 0x80000001 und r3 = 0x100
- r0 = r2 * r3, 32-Bit Multiplikation

Aufgabe 7.1d) Multiply-Accumulate Instruktionen

- r2 = 0x80000001, r3 = 0x100 und r4 = 0x55
- r0 = r2 * r3 + r4

Aufgabe 7.1e) Vergleichs-Instruktionen

- r0 = 10, r1 = 10
- Vergleichen Sie r0 mit r1 (CMP), wie werden die Statusbits gesetzt?
- Vergleichen Sie r0 mit 9, wie werden die Statusbits jetzt gesetzt?

Aufgabe 7.1f) Logische Instruktionen

- Laden Sie r1 mit 0x33333333
- Maskieren Sie das niederwertigste Byte von r1

- Führen Sie eine ODER-Verknüpfung von r1 mit 0x0C durch
- Setzen Sie Bit 12 von r1
- Führen Sie eine EXOR-Verknüpfung von r1 mir 0xF0 durch
- Toggeln Sie bit 7 von r1 (toggeln = Zustand wechseln)
- Löschen Sie die 4 tiefsten Bits von r1
- Löschen Sie Bit 6 von r1

Aufgabe 7.1g) Shift Instruktionen

- Laden Sie r1 mit 0xC0000001
- r0 = r1 logic-shift-left 2 , wie ändern sich die Statusbits?
- r0 = r1 logic-shift-right 2 , wie ändern sich die Statusbits?
- r0 = r1 arithmetic-shift-right 2, vergleichen Sie das Ergebnis mit logic-shift-right
- r0 = r1 rotated-right 2

Aufgabe 7.2 Programmverzweigungen

Aufgabe 7.2a) Unbedingte Verzweigung
Programmieren Sie eine Endlosschleife, in welcher r0 inkrementiert wird.

Aufgabe 7.2b) Loop
Programmieren Sie einen Loop, welcher 10 Mal durchlaufen wird. In jedem Durchgang wird r0 inkrementiert.

Aufgabe 7.2c) Summe von Speicherinhalten
Addieren Sie alle 8-Bit Werte, welche im Speicher zwischen den Adressen 0x20001000 und 0x20001007 abgelegt sind (Grenzen inklusive). Um einen sinnvollen Test zu ermöglichen können Sie im Memory-Fenster bei den entsprechenden Adressen geeignete Werte eingeben.

Aufgabe 7.2d) Skalarprodukt
Es soll das Skalarprodukt zweier Vektoren berechnet werden. Die Vektoren liegen auf den Adressen 0x20001000 und 0x20001100 und haben je 10 Komponenten.

Als Repetition für all diejenigen, die es vergessen haben (Quelle: Wikipedia): In der linearen Algebra ist das Skalarprodukt oder innere Produkt zweier Vektoren

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ . \\ x_n \end{pmatrix} \text{ und } \vec{y} = \begin{pmatrix} y_1 \\ y_2 \\ . \\ y_n \end{pmatrix}$$

des n-dimensionalen euklidischen Raumes als jene reelle Zahl definiert, die sich als Summe der Produkte der Komponenten der Vektoren ergibt:

$$\vec{x} * \vec{y} = \sum_{i=1}^{n} x_i y_i = x_1 y_1 + x_2 y_2 + \ldots + x_n y_n$$

Aufgabe 7.2e) Subroutinen
Eine Subroutine soll die Werte in r1 und r2 addieren und das Resultat in r0 zurückgeben. Rufen Sie die Subroutine mit verschiedenen Werten für r1 und r2 auf.