Name der Vorlesung
Aufgaben zu Einführung in die ARM Assembler Sprache

1 Aufgabe 1 - Verständniss

1.1 Allgemeines

- Welches Speichermodell verwenden die ARM Architekturen ?
- Welche Wortbreite besitzt eine Instruktion ?
- Welche der ARM Instruktionen verwenden Erweiterungsworte ?
- Wie werden Instruktionen verarbeitet ?
- Angenommen zum Zeitpunkt n wird eine ADD Instruktion, an der Adresse `0xAFF6`, tatsächlich ausgeführt, an welcher Adresse befindet sich der `PC`, wenn sich der Prozessor im ARM Modus befindet?
- Was ist die Tool Chain? Geben pro Programmierwerkzeug eine kurze Beschreibung.
- Aus welchen Sektionen besteht eine Assembler Quelldatei?
- Zeichnen Sie das typische Speicher-Layout eines Prozesses unter Linux.

1.2 Bits and Bytes

- Welche Größe (in Bit) besitzen: `BYTE, HALF WORD, WORD, DOUBLE WORD`?
- Welche Endianness wird unter ARM genutzt?
- Beschreiben Sie `Little Endian` in einem Satz.

1.3 Register

- Was sind `Callee Save Register`?
- Welches Register (rX) stellt den Program Counter dar?
- Was ist der Zweck des Link Registers? Was ist in dieser Hinsicht der Unterschied zu x86?
- Nennen Sie die Flags, die für bedingte Instruktionen zur verfügung stehen.
- Nennen Sie zwei Prozessor Modi und beschreiben Sie diese kurz.

2 Data Processing Instructions

Für die nachfolgenden Aufgaben wird ein Arm Computer, wie z.B. Raspberry Pi, oder ein Emulator, wie z.B. Quemu, benötigt, sowie ein Linux Betriebssystem.

Die nachfolgenden Beschreibungen beziehen sich auf eine ARM Entwicklungsumgebung bestehend aus eine `Raspberry Pi 3` auf dem `Raspbian` installiert wurde.

2.1 Division

```
def gcd(a,b):
    if b==0:
        return a
    else:
        return gcd(b, a%b)

a = 5125
b = 6200

print "gcd(%d,%d)=%d" % (a,b,gcd(a,b))
```