# Laborübung 1

**Bearbeitungshinweise**

Die Laborübung ist in Intel x86 Assembler (32-bit Programmiermodell) für den NASM Assembler unter dem Betriebssystem Linux (z.B. Ubuntu 14.04) zu erstellen. Andere Programmiersprachen sowie C-Bibliotheksfunktionen in Assemblerprogrammen dürfen nicht verwendet werden. Verwenden Sie für die folgenden Aufgaben das Archiv labor1.tar.gz (siehe Kurshomepage) als Grundlage. Darin enthalten ist eine Musterlösung für Aufgabe 1.

Die Laborübung wird gruppenweise erstellt. Die Namen der Gruppenmitglieder sind im Quellcode zu vermerken. Außerdem ist zu vermerken, welches Gruppenmitglied welche Aufgabenteile schwerpunktmäßig bearbeitet hat.

Wenn Sie Programmfragmente aus der Literatur, dem Internet oder von anderen Quellen verwenden, ist die Quelle als Kommentar kenntlich zu machen.

**Abgabe**

Die Laborübung ist als „gezipptes“ Archiv im tar-Format bis spätestens

**Donnerstag, 20. November 2014, 23:59 CET**

per Email an kontakt@ralfreutemann.name zu schicken.

Der Name der abzugebenden Archivdatei ist labor1\_x.tar.gz, wobei x die Ihrer Gruppe zugewiesene Nummer entspricht. Dieses Archiv enthält ein Verzeichnis labor1, in dem der vollständige Quellcode, das Build-Skript bzw. Makefile sowie das ausführbare Programm für jede der folgenden Aufgaben enthalten ist.

In der Datei README.txt können und sollen Sie Bemerkungen zu den Aufgaben, wie z.B. fehlende Funktionalität, Probleme oder Fehler die nicht gelöst werden konnten usw., zusammenfassen.

**Hinweis:** Falls von Ihnen nicht anderweitig beschrieben, gehe ich bei der Bewertung davon aus, dass Sie die Aufgabenstellung vollständig implementiert haben und bei Tests keinerlei Fehler aufgetreten sind.

**Aufgaben**

1. **asmtime1**: Schreiben Sie ein Assembler-Programm asmtime1.asm, welches die aktuelle Systemuhrzeit in Stunden, Minuten und Sekunden berechnet.

Der Linux System Call[[1]](#footnote-1) time (siehe man 2 time) für das Ermitteln der Uhrzeit hat die Nummer 13 (0x0c). Der System Call liefert die Zeit in Sekunden seit 1. Jan. 1970, 0:00:00 GMT (sog. Unix Epoche). Das Ergebnis steht als 32-bit Integer im Register EAX.

Als optionales Argument kann man im Register EBX die Adresse eines 4 Byte großen Speicherblocks angeben, in dem der Rückgabewert zusätzlich gespeichert wird. Falls dies nicht gewünscht ist, muss das Register EBX den Wert Null (Null-Zeiger) enthalten.

Überlegen Sie zunächst, wie sie aus diesem Wert die Anzahl der am aktuellen Tag vergangenen Sekunden ermitteln können. Ermitteln Sie daraus die einzelnen Werte für Stunden, Minuten und Sekunden der aktuellen Uhrzeit und speichern Sie diese im bss Segment entsprechend Tabelle 1. Die verwendeten Register bei einer Division mit einem 32-Bit Divisor sind in Abbildung 1 dargestellt.

|  |  |  |
| --- | --- | --- |
| **Variable** | **Bytes** | **Beschreibung** |
| secs\_epoch | 4 | Sekunden seit der Unix Epoche |
| secs\_today | 4 | Sekunden des aktuellen Tags, d.h. seit 00:00 |
| hours | 1 | Stunden der aktuellen Uhrzeit **HH**:MM:SS |
| minutes | 1 | Minuten der aktuellen Uhrzeit HH:**MM**:SS |
| seconds | 1 | Sekunden der aktuellen Uhrzeit HH:MM:**SS** |

Tabelle : Variablen im bss Segment (asmtime1)

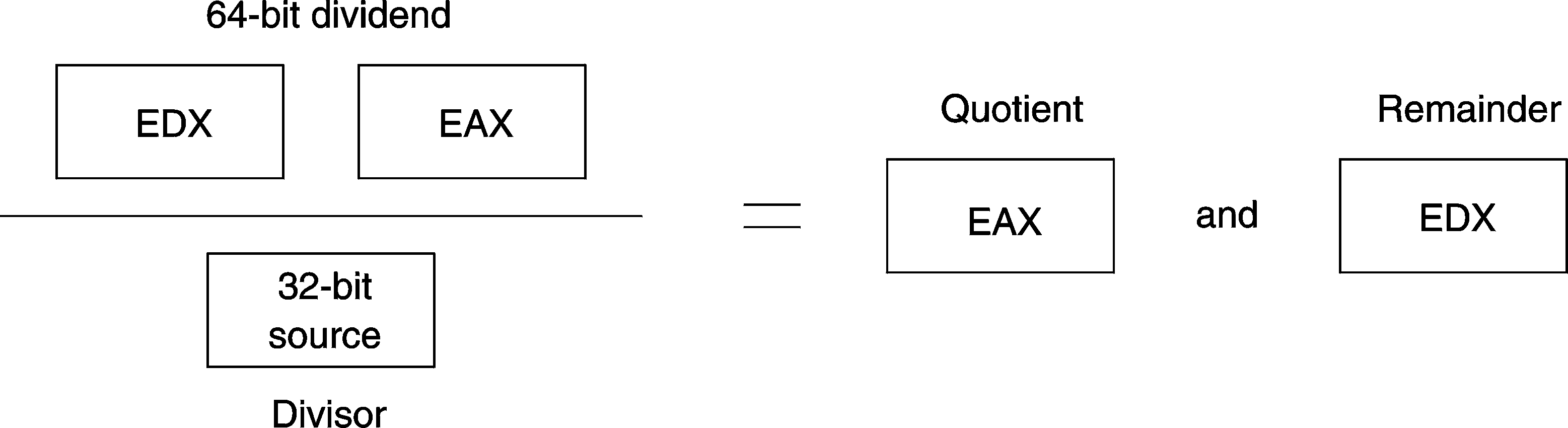


Abbildung : Registerbelegung bei einer Division mit 32-Bit Divisor

1. **asmtime2**: Erweitern Sie das Assembler-Programm aus der vorangegangenen Aufgabe, so dass das Programm die aktuelle Systemuhrzeit in Stunden, Minuten und Sekunden auf der Standardausgabe ausgibt. Hierfür wandeln Sie die einzelnen Integer-Werte in ihre entsprechende BCD Darstellung um und speichern das Ergebnis in einer Zeichenkette im data Segment. Die Ausgabe erfolgt mit einem *einzelnen* Aufruf des Linux System Calls write (siehe man 2 write).

Ergänzen Sie in der Datenstruktur aus Aufgabe 1 die 16-Bit Integer-Variable days\_epoch entsprechend Tabelle 2 und speichern Sie darin die Anzahl der Tage seit der Unix Epoche.

|  |  |  |
| --- | --- | --- |
| **Variable** | **Bytes** | **Beschreibung** |
| secs\_epoch | 4 | Sekunden seit der Unix Epoche |
| secs\_today | 4 | Sekunden des aktuellen Tags, d.h. seit 00:00 |
| *days\_epoch* | *2* | *Tage seit der Unix Epoche* |
| hours | 1 | Stunden der aktuellen Uhrzeit **HH**:MM:SS |
| minutes | 1 | Minuten der aktuellen Uhrzeit HH:**MM**:SS |
| seconds | 1 | Sekunden der aktuellen Uhrzeit HH:MM:**SS** |

Tabelle : Variablen im bss Segment (asmtime2)

Formatieren Sie diese Werte bei der Ausgabe jeweils als zweistellige Dezimalzahl mit führender Null (siehe folgendes Beispiel):

**Beispiel:**  
$ ./asmtime2  
20:28:31 GMT

1. **asmtime3**: Geben Sie zusätzlich zur Uhrzeit auch den aktuellen Wochentag auf zwei (deutsch) oder drei Buchstaben (englisch) abgekürzt aus und speichern Sie den numerischen Wert in der Variablen wday im rodata Segment, wobei „Sonntag“ mit 0, „Montag“ mit 1 usw. kodiert ist.

Außerdem soll das Programm den Rückgabewert des System Calls, d.h. die sog. Ticks, als vorzeichenlose Dezimalzahl[[2]](#footnote-2) mit 10 Stellen rechtsbündig ohne führende Nullen ausgeben.

**Beispiel:**  
$ ./asmtime3  
Mon 20:28:31 GMT 1385411311

Die korrekte Ausgabe Ihres Programms können Sie mit dem folgenden Befehl überprüfen (siehe man 1 date):

$ date --utc +"%a %H:%M:%S GMT %s"

1. Die Aufrufsemantik von Linux System Calls behandeln wir in einem späteren Teil der Vorlesung. [↑](#footnote-ref-1)
2. Zur Vereinfachung der Zahlendarstellung ignorieren wir hier, dass Unix Zeitstempel eigentlich vorzeichenbehaftet sind. [↑](#footnote-ref-2)