# Laborübung 1

**Bearbeitungshinweise**

Die Laborübung ist in Intel x86 Assembler (64-bit Programmiermodell) für den NASM Assembler unter dem Betriebssystem Linux (z.B. Ubuntu 14.04) zu erstellen. Verwenden Sie für die folgenden Aufgaben das folgende Repository als Grundlage: [github.com/rdrcode/sysprog](http://github.com/rdrcode/sysprog)

Im Verzeichnis asmtime sind Musterlösungen für die ersten beiden Aufgaben sowie ein Makefile enthalten.

**Aufgaben**

1. **asmtime1**: Schreiben Sie ein 32-bit 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 (0x0d). 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)

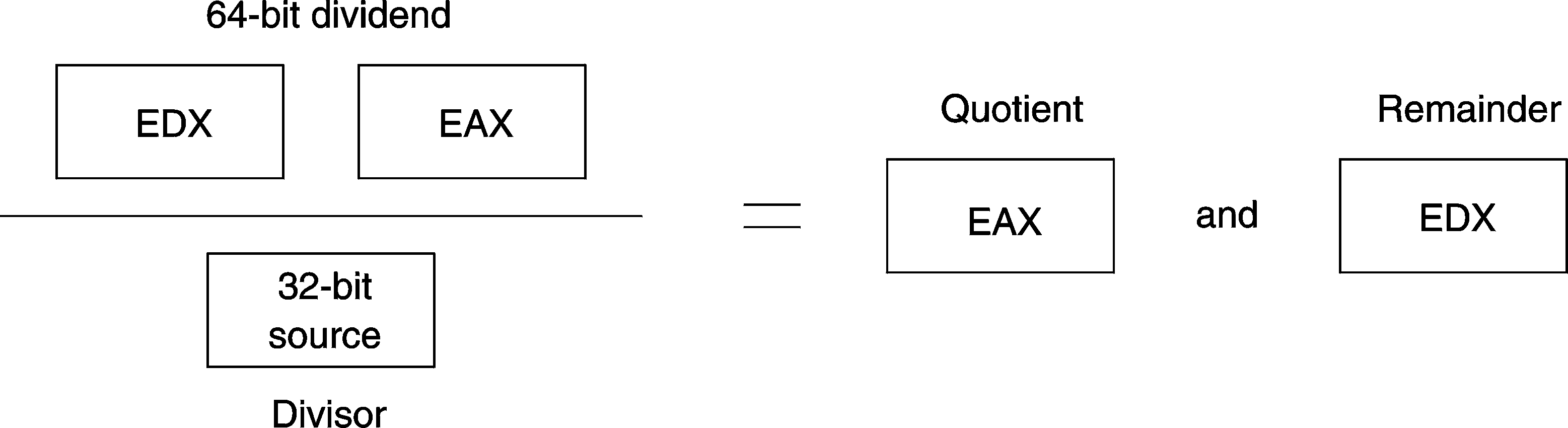


Abbildung : Registerbelegung bei einer Division mit 32-Bit Divisor

1. **asmtime2**: Ändern Sie das Programm aus Aufgabe 1 in das x86 64-bit Programmiermodell und verwenden Sie den Linux System Call gettimeofday (siehe man 2 gettimeofday)
2. **asmtime3**: 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 2 die 32-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* | *4* | *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 (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.

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

**Beispielausgabe (englisch):**  
$ ./asmtime3  
Mon 20:28:31 GMT

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

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

1. **cpuid64**: Erweitern Sie das bereits vorhandene 64-bit Assembler-Programm cpuid64.asm im Verzeichnis cpuid, so dass das Programm zusätzlich zur Prozessorkennung die Bitbreite der physikalischen und linearen Adressen ausgibt (siehe Beschreibung des cpuid Befehls im Intel oder AMD Processor User Manual, Extended Function 0x80000008).

Link: <http://support.amd.com/TechDocs/25481.pdf>

1. Die Aufrufsemantik von Linux System Calls behandeln wir in einem späteren Teil der Vorlesung. [↑](#footnote-ref-1)