Christian-Albrechts-Universität zu Kiel
Institut für Informatik

Hagen Peters, Prof. Dr.-Ing. Norbert Luttenberger, Miro Spönemann

Institut für Informatik

4. Übung zur Vorlesung „Organisation und Architektur von Rechnern“

Data Lab

Abgabe am Montag, 09. Mai – 16:00

Das Ziel dieser Aufgaben ist die Vertiefung im Umgang mit Zahlen auf Bit-Ebene. Dies geschieht durch die Lösung einer Reihe von Programmierpuzzles, wovon einige recht künstlich sind, durch die Sie aber lernen werden Bits wesentlich besser zu verstehen.

Bearbeitung

Kopieren Sie die Datei datalab-handout.tar in ein Arbeitsverzeichnis und entpacken Sie diese mit dem Kommando `tar xvf datalab-handout.tar`. Die einzige der daraus entpackten Dateien, die Sie verändern und abgeben werden, ist `bits.c`. Die Datei `btest.c` erlaubt es die funktionale Korrektheit des Codes zu evaluieren. Weitere Dokumentation hierzu finden Sie in der Datei `README`. Verwenden Sie `make btest`, um den Test Code zu erzeugen, und führen Sie es mit `./btest` aus. Die `dlc-*` Dateien sind Programme für verschiedene Plattformen, die verwendet werden können, um die Lösungen auf Einhaltung der Vorgaben zu prüfen. Wählen Sie die für Ihre Plattform entsprechende Version aus. Die übrigen Dateien werden verwendet, um `btest` zu compilieren.

Bei Betrachtung der Datei `bits.c` wird Ihnen eine Datenstruktur `team` auffallen, in die Sie die Identifikationsdaten Ihrer Gruppe eintragen sollten. Tun Sie dies am besten sofort, damit Sie es nicht vergessen.

Die `bits.c` Datei enthält auch ein Gerüst für jedes Programmierpuzzle. Ihre Aufgabe besteht darin, jedes Funktionsgerüst zu vervollständigen, indem Sie nur gradlinigen Code (d.h. keine Schleifen oder Verzweigungen) und nur eine begrenzte Auswahl von arithmetischen und logischen Operatoren verwenden. Genauer dürfen Sie nur die folgenden acht Operatoren verwenden:

```
! ~ & ^ | + << >>
```

Einige Funktionen schränken diese Liste noch weiter ein. Sie dürfen außerdem keine Konstanten nutzen, die länger als 8 Bits sind. Weitere Hinweise zu den Regeln und dem gewünschten Lösungsstil finden Sie in der Datei `bits.c`.

Auswertung

Ihr Code wird automatisch compiliert und ausgewertet. Sie können bis zu 20 Punkte erreichen, was sich aus folgender Verteilung ergibt:

- 10 Punkte für die Korrektheit
- 8 Punkte für Effizienz, basierend auf der Anzahl von Operatoren, die Sie in den einzelnen Funktionen verwenden
- 2 Stil-Punkte, basierend auf der Qualität Ihrer Lösungen und Ihrer Kommentare

Die 12 Aufgaben, die Ihnen gestellt sind, haben eine Schwierigkeitswertung zwischen 1 und 4, so dass in der Summe 30 Einheiten zu erreichen sind; die 10 Punkte werden hieraus anteilig berechnet, d.h. für eine Korrektheits-Wertung von 15 erhalten Sie 5 Punkte. Die Korrektheit wird mit den Tests in `btest.c` ermittelt: eine Lösung zu einer Aufgabe wird mit der vollen Wertungszahl angerechnet, wenn sie alle Tests in `btest.c` besteht; wenn ein Test fehlschlägt, wird die halbe Wertungszahl angerechnet, und wenn mehr fehlschlagen, gibt es keine Punkte.

Was Effizienz angeht, geht es hier hauptsächlich darum, dass Sie eine richtige Lösung finden. Dennoch sollten Sie ein Gefühl dafür bekommen, die Dinge so kurz und einfach zu halten wie Sie können. Für manche Aufgaben gibt es eine naive Lösung, Sie sollten aber versuchen eine schlauere zu finden. Daher gibt es für jede Funktion eine maximale Anzahl von Operatoren. Diese Zahl ist sehr großzügig und soll ausgesprochen ineffiziente Lösungen abfangen. Sie bekommen zwei Punkte für jede Lösung, die unter der Operatorgrenze bleibt.

Schließlich gibt es 2 Punkte für eine subjektive Auswertung des Stils Ihrer Lösungen und Ihrer Kommentare. Ihre Lösung sollte so sauber und direkt wie möglich sein. Ihre Kommentare sollten informativ sein, aber nicht extensiv.

Hinweise

Hier die empfohlene Vorgehensweise zusammengefasst:

- datalab-handout.tar entpacken
- `bits.c` bearbeiten: Team-Daten eintragen und Aufgaben lösen
- Lösung validieren:

  ```
  $ dlc-linux bits.c
  ```

- Lösung compilieren:

  ```
  $ make
  ```

- Korrektheit testen:

  ```
  $ ./btest
  ```

Lösungen können nach dem Hochladen online getestet werden. Die Auswertungen der getesteten Lösungen erscheinen auf der “Beat the Prof” Contest Seite.

Das Lab funktioniert nur wenn die Dateien für 32 Bit compiliert werden. Auf 64 Bit Systemen muss beim `gcc` die Option `-m32` verwendet werden. Für Linux ist diese nur verfügbar, wenn das Paket `gcc-multilib` installiert ist.

Sollte das beigefügte Programm `dlc` auf Ihrer Architektur nicht funktionieren, empfehlen wir die entsprechende Variante auf dem Server `bilbo` (`dlc-sunsparc`) oder `elrond` (`dlc-sunx86`) zu verwenden.

Beachten Sie, dass wir uns vorbehalten keine oder nur wenige Punkte für eine Abgabe zu geben, die beim Online-Test nicht korrekt compiliert wird. Laden Sie also nur gültigen C-Code hoch.

Und zuletzt: Fangen Sie nicht erst kurz vor dem Abgabetermin mit der Bearbeitung an! Je nach Stand der Vorkenntnisse kann die Bearbeitung der Lab-Aufgaben recht aufwändig sein, Sie sollten also möglichst bald damit anfangen.

Aufgabe 1 - Data Lab

20 Punkte

Laden Sie Ihre bearbeitete Datei `bits.c` aus datalab-handout.tar hoch (*nicht* das ganze archiv, sondern nur die Quelldatei!). Anschließend können Sie Ihre Lösung online testen. Das Ergebnis erscheint auf der Contest Seite.