

LEHRSTUHL FÜR RECHNERARCHITEKTUR UND PARALLELE SYSTEME

**Praktikum Rechnerarchitektur**

LU-Zerlegung (A328)

Projektaufgabe – Aufgabenbereich Algorithmik

**1 Organisatorisches**

Auf den folgenden Seiten finden Sie die Aufgabenstellung zu Ihrer Projektaufgabe für das Praktikum. Die Rahmenbedingungen für die Bearbeitung werden in der Praktikumsordnung festgesetzt, die Sie über die Praktikumshomepage<sup>1</sup> aufrufen können.

Wie in der Praktikumsordnung beschrieben, sind die Aufgaben relativ offen gestellt. Besprechen Sie diese innerhalb Ihrer Gruppe und konkretisieren Sie die Aufgabenstellung. Die Teile der Aufgabe, in denen Assembler-Code anzufertigen ist, sind für die 64-Bit x86-Architektur (x86-64) unter Verwendung der SSE-Erweiterungen zu schreiben; alle anderen Bestandteile der Hauptimplementierung sind in C nach dem C11-Standard anzufertigen.

Der **Abgabetermin** ist der **16. Juli 2021, 11:59 Uhr (MESZ)**. Die Abgabe erfolgt per Git in das für Ihre Gruppe eingerichtete Projektrepository. Bitte beachten Sie die in der README.md angegebene Liste von abzugebenden Dateien.

Die **Abschlusspräsentationen** finden in der Zeit vom **09.08.2021 – 03.09.2021** statt. Weitere Informationen werden noch bekannt gegeben. Beachten Sie, dass die Folien für die Präsentation am obigen Abgabetermin im PDF-Format abzugeben sind.

Sofern die Rahmenbedingungen (Hygiene-, Abstandsregeln etc.) für eine Präsenzprüfung nicht vorliegen, kann gemäß §13a APSO die geplante Prüfungsform auf eine virtuelle Präsentation (Videokonferenz) oder eine kurze schriftliche Fernprüfung umgestellt werden. Die Entscheidung über diesen Wechsel wird möglichst zeitnah, spätestens jedoch 14 Tage vor dem Prüfungstermin nach Abstimmung mit dem zuständigen Prüfungsausschuss bekannt gegeben.

Bei Fragen/Unklarheiten in Bezug auf den Ablauf und die Aufgabenstellung wenden Sie sich bitte an Ihren Tutor.

Wir wünschen Ihnen viel Erfolg und Freude bei der Bearbeitung Ihrer Aufgabe!

Mit freundlichen Grüßen  
Die Praktikumsleitung

---

<sup>1</sup><https://gepasp.in.tum.de/eraweb/>

---

## 2 LU-Zerlegung

### 2.1 Überblick

Die Algorithmik ist ein Teilgebiet der Theoretischen Informatik, welches wir hier unter verschiedenen praktischen Aspekten beleuchten: Meist geht es um eine konkrete Frage- oder Problemstellung, welche durch mathematische Methoden beantwortet oder gelöst werden kann. Sie werden im Zuge Ihrer Projektaufgabe ein Problem lösen und die Güte Ihrer Lösung wissenschaftlich bewerten.

### 2.2 Funktionsweise

Für eine Matrix  $A \in \mathbb{R}^{n \times n}$  bezeichnet die *LU-Zerlegung* eine Faktorisierung von  $A$  in zwei Dreiecksmatrizen  $L$ , wo alle die Einträge überhalb der Diagonale 0 sind, und  $U$ , wo alle Einträge unterhalb der Diagonale 0 sind, sodass  $A = LU$  gilt. Diese Zerlegung wird beispielsweise verwendet, um lineare Gleichungssysteme zu lösen. Ihre Projektaufgabe ist die Implementierung der LU-Zerlegung.

### 2.3 Aufgabenstellungen

Ihre Aufgaben lassen sich in die Bereiche Konzeption (theoretisch) und Implementierung (praktisch) aufteilen. Sie können (müssen aber nicht) dies bei der Verteilung der Aufgaben innerhalb Ihrer Arbeitsgruppe ausnutzen. Antworten auf konzeptionelle Fragen sollten an den passenden Stellen in Ihrer Ausarbeitung in angemessenem Umfang erscheinen. Entscheiden Sie nach eigenem Ermessen, ob Sie im Rahmen Ihres Abschlussvortrags auch auf konzeptionelle Fragen eingehen. Die Antworten auf die Implementierungsaufgaben werden durch Ihrem Code reflektiert.

#### 2.3.1 Theoretischer Teil

- Recherchieren Sie anhand geeigneter Sekundärliteratur, wie die LU-Zerlegung berechnet werden kann.
- Überlegen Sie, wie Sie die Transformation mittels SIMD optimieren können.
- Spezifizieren Sie ein einfaches Eingabeformat für Matrizen.

#### 2.3.2 Praktischer Teil

- Implementieren Sie im Rahmenprogramm I/O-Operationen in C, mit welchen Sie den Benutzer eine Datei, die die Matrix  $A$  enthält, übergeben oder eine zufällige Matrix automatisch generieren lassen können.
- Implementieren Sie in der Datei mit dem Assemblercode die Funktion

```
void ludcomp(size_t n, const float* A, float* L, float* U);
```

welche eine Matrix  $A$  der Größe  $n \times n$  übergeben bekommt und diese in  $L$  und  $U$  faktorisiert. Die Einträge mit dem Wert 0 oberhalb bzw. unterhalb der Diagonale in  $L$  bzw.  $U$  können, müssen aber nicht gesetzt werden.

## 2.4 Allgemeine Bewertungshinweise

Die folgende Liste soll Ihnen als Gedächtnisstütze beim Bearbeiten der Aufgaben dienen. Beachten Sie ebenfalls die in der Praktikumsordnung angegebenen Hinweise.

- Stellen Sie unbedingt sicher, dass *sowohl* Ihre Implementierung *als auch* Ihre Ausarbeitung auf der Referenzplattform des Praktikums (1xhalle) kompilieren und vollständig korrekt bzw. funktionsfähig sind.
  - Die Implementierung soll mit GCC/GNU as kompilieren. Verwenden Sie keinen Inline-Assembler und keine x87-FPU- oder MMX-Instruktionen. Sie dürfen alle SSE-Erweiterungen bis SSE4.2 benutzen. Andere ISA-Erweiterungen (z.B. AVX, BMI1) dürfen Sie nur benutzen, sofern Ihre Implementierung auch auf Prozessoren ohne derartige Erweiterungen lauffähig ist.
  - Sie dürfen die angegebenen Funktionssignaturen (nur dann) ändern, wenn Sie dies (in Ihrer Ausarbeitung) begründen.
  - I/O-Operationen dürfen grundsätzlich in C implementiert werden.
  - Denken Sie daran, das Laufzeitverhalten Ihres Codes zu testen (Sichere Programmierung, Performanz) und behandeln Sie *alle möglichen Eingaben*, auch Randfälle. Ziehen Sie ggf. alternative Implementierungen als Vergleich heran.
  - Eingabedateien, welche Sie generieren, um Ihre Implementierungen zu testen, sollten mit abgegeben werden; größere Eingaben sollten stattdessen stark komprimiert oder (bevorzugt) über ein abgegebenes Skript generierbar sein.
  - Stellen Sie Performanz-Ergebnisse nach Möglichkeit grafisch dar.
  - Vermeiden Sie unscharfe Grafiken und Screenshots von Code.
  - Geben Sie die Folien für Ihre Abschlusspräsentation im PDF-Format ab. Achten Sie auf hinreichenden Kontrast (schwarzer Text auf weißem Grund!) und eine angemessene Schriftgröße. Verwenden Sie 16:9 als Folien-Format.
-