Praktikumsaufgabe „Rechnergestützter Entwurf digitaler Systeme“

Sergei Sawitzki

5. Juli 2022

Nachfolgend ist die Aufgabenstellung für das Praktikum „Rechnergestützter Entwurf digitaler Systeme“ im Bachelor-Studiengang Technische Informatik (bzw. Ergänzungsblock Master-Studiengang Informatik) beschrieben.

1 Lernziele und Bewertungsgrundlage

In der Vorlesung wurden verschiedene Entwurfsschritte beschrieben und einige Algorithmen zu deren Umsetzung vorgestellt. Die Betrachtung erfolgte dabei auf relativ abstrakter Ebene, bei der viele Probleme einer detaillierten softwaretechnischen Implementierung nicht behandelt werden können. Das Ziel des Praktikums ist es, einen ausgewählten Algorithmus zur Entwurfsautomatisierung praktisch zu implementieren und dabei solche Probleme kennen und lösen zu lernen. Voraussetzungen für eine Bewertung des Praktikums als „bestanden“ sind dabei wie folgt:

- Implementierter Algorithmus muss einen kreativen Anteil enthalten, d. h. keine 1:1 Umsetzung eines aus der Vorlesung oder aus der Literatur bekannten Verfahrens (Modifikationen oder Erweiterungen der bekannten Verfahren sind erlaubt).
- Ein- und Ausgaben müssen in einem vorgegebenen Format vorliegen, d. h. in einen vorgegebenen Entwurfsfluss einbettbar sein (wird später präzisiert).
- Ausgabe muss gültig sein (z. B. eine legale Platzierung ohne Überlappungen oder eine Verdrahtung ohne Überschreitung der Kanalkapazitäten usw.)

- Das Ergebnis darf, **muss aber nicht besser sein**, als Ergebnisse anderer bekannter Werkzeuge bzw. Verfahren.
- Es muss eine schriftliche Dokumentation vorliegen (in Form einer Fachveröffentlichung, ca. 10–20 Seiten Umfang), die folgende Schwerpunkte widerspiegelt:
  - Präzisierung der Aufgabenstellung (gewählter Entwurfsschritt, Strategie, Zielstellung)
  - Beschreibung des gewählten Ansatzes (eigene „kreative Komponente“, Ansätze und Heuristiken)
  - Benchmarking
  - Analyse und Bewertung der Ergebnisse nach ingenieurtechnisch-wissenschaftlichen Kriterien.
- Die Ergebnisse müssen am Rechner vorgeführt werden, der Quellcode ist ein Bestandteil der Dokumentation (z. B. in Form eines beigelegten Datenträgers).

Das Bewältigen der Praktikumsaufgabe ist eine **Einzelleistung**, eine beratende Einbeziehung der anderen Teilnehmer sowie des Dozenten ist jedoch erlaubt. Bearbeitung verschiedener Entwurfsschritte mit gemeinsamer Schnittstelle unter expliziter Einbeziehung der Ergebnisse eines anderen Praktikumsteilnehmers ist erlaubt, solange die Einzelleistung klar erkennbar und dokumentiert ist. Die Endnote setzt sich aus den Bewertungen folgender Teilleistungen zusammen:

- Schriftliche Dokumentation
- Vorführung der Software
- Kolloquium (unter Einbeziehung anderer, im Projekt nicht behandelter Themen)

2 Vorbereitende Schritte

Vor der Bearbeitung der eigentlichen Aufgabe muss das VPR-Programmpaket heruntergeladen und installiert werden:

`https://www.eecg.utoronto.ca/~vaughn/vpr/vpr.html`

Aus Übersichtlichkeitsgründen soll die ältere Version 4.30 verwendet werden, da diese besonders einfach zu installieren ist und gut lesbare sowie einfach strukturierte Ein- und Ausgabeformate benutzt. In der dem VPR-Paket beiliegenden Dokumentation mit dem Dateinamen `manual_430.pdf` sind einzelne Werkzeuge und Dateiformate dokumentiert. Das Paket besteht aus den Komponenten `t-vpack` für das Clustering und `vpr` für die Platzierung und Verdrahtung. Die Lauffähigkeit beider Komponenten soll mit beiliegenden Beispieldateien überprüft werden. Zusätzlich können die auf dem Handout-Server unter

`Sawitzki/EDA/Praktikum/Benchmarks`

liegenden Beispiel-Netzlisten zu Testzwecken verwendet werden. Die mit `t-vpack` bzw. `vpr` für verschiedene Benchmarks erzielten Ergebnisse sollen in der schriftlichen Ausarbeitung mit Ergebnissen der eigenen Implementierung verglichen werden.

Es ist sinnvoll, sich bereits im Vorfeld mit den durch VPR verwendeten Dateiformaten vertraut zu machen:

`*.blif` Allgemeine Netzliste z. B. nach der *High-level*-Synthese, Logikoptimierung und Technologieabbildung.

`*.arch` Architekturbeschreibung des FPGA-Bausteins, für den VPR die Platzierung und Verdrahtung durchführen soll (u. a. die Größe der Logikblöcke, Anzahl der Logikblöcke pro Cluster usw.)

`*.net` Technologiespezifische Netzliste nach Technologieabbildung (die entsprechend den Vorgaben der Architekturbeschreibung durchgeführt werden sollte).

`*.place` Technologiespezifische Netzliste nach Platzierung durch VPR.

`*.route` Platzierte Netzliste nach Verdrahtung durch VPR.

Diese Dateiformate sind im bereits erwähnten Dokument `manual_430.pdf` sowie der Datei `blif.pdf` unter

`Sawitzki/EDA/Praktikum/Benchmarks/blif/doc`

beschrieben.

3 Praktikumsaufgaben

Es ist jeweils **einer** der folgenden drei Algorithmen zu implementieren.

3.1 Clustering

Die unter

`Sawitzki/EDA/Praktikum/Benchmarks/blif/`

abgelegten Netzlisten bestehen ausschließlich aus 4:1 LUT *(look-up-tabellen)* und Flipflops. Abhängig von der gewählten FPGA-Architektur müssen diese zu Clustern der Größe n zusammengefasst werden, so dass jeweils maximal n LUT und n Flipflops einem Cluster zugeordnet sind. Die Ausgabe ist eine Netzliste im `*.net`-Format. Die Ergebnisse sind für alle 20 Benchmarks mit der Ausgabe von `t-vpack` bezüglich folgender Kriterien zu vergleichen:

- Anzahl der erzeugten Cluster
- Durchschnittliche Auslastung der Cluster
- Schaltungstiefe (in Clustern)
- Laufzeit des Algorithmus

Weiterhin sind die erzeugten Netzlisten durch VPR zu platzieren und zu verdrahten und mit Ergebnissen der Platzierung und Verdrahtung der durch `t-vpack` erzeugten Netzlisten anhand folgender Kriterien zu vergleichen.

- Kritischer Pfad bzw. Taktfrequenz
- Minimale Breite der Verdrahtungskanäle

Die Ergebnisse der Vergleiche sind zu analysieren und zu bewerten.

3.2 Platzierung

Die unter

`Sawitzki/EDA/Praktikum/Benchmarks/net/`

abgelegten Netzlisten bestehen ausschließlich aus 4:1 LUT *(look-up-tabellen)* und Flipflops. Dabei sind, wenn möglich, eine LUT und ein Flipflop zu einem BLE *(basic logic element)* zusammengefasst. Es gibt aber entsprechend BLE, die nur aus einer LUT oder nur einem Flipflop bestehen. Die BLE-Netzlisten im `*.net`-Format sind zu platzieren. Die Ausgabe erfolgt im `*.place`-Format. Diese Ausgabe ist in VPR einzulesen und zu verdrahten. Die Ergebnisse sind für alle 20 Benchmarks mit der Ausgabe von `VPR`, die sich im Unterverzeichnis `vpr_placements` befindet, bezüglich folgender Kriterien zu vergleichen:

- Kostenfunktion der erzeugten Platzierung (wird von VPR nach einlesen der Platzierung berechnet und ausgegeben)
- Laufzeit des Algorithmus
- Kritischer Pfad bzw. Taktfrequenz (nach der Verdrahtung durch VPR, die entsprechenden Referenz-Benchmarks zum Vergleich befinden sich im Unterverzeichnis `vpr_routings`)
- Minimale Breite der Verdrahtungskanäle (nach der Verdrahtung durch VPR, die entsprechenden Referenz-Benchmarks zum Vergleich befinden sich im Unterverzeichnis `vpr_routings`)

Die Ergebnisse der Vergleiche sind zu analysieren und zu bewerten. Die vorgegebenen BLE bestehen aus maximal einer LUT und einem Flipflop. Bei Wunsch können mit `t-vpack` aus `*.blif`-Dateien Netzlisten mit komplexeren Clustern erzeugt werden, z. B. 4 LUT und 4 Flifplops sowie 10 verfügbaren Logikeingängen pro Cluster. In diesem Fall müssen die Referenz-Benchmarks durch VPR ebenfalls neu generiert werden, die in den Unterverzeichnissen `vpr_placements` und `vpr_routings` hinterlegten Ergebnisse haben keine Cluster. Es ist ausreichend, die Aufgabe für eine Architektur (entweder mit oder ohne Cluster) umzusetzen.

3.3 Verdrahtung

Die unter

`Sawitzki/EDA/Praktikum/Benchmarks/vpr_placements/`

abgelegten Netzlisten im `*.place`-Format sind mit VPR platziert. Diese Netzlisten sind zu verdrahten. Die Ausgabe erfolgt im `*.route`-Format. Die Ergebnisse sind für alle 20 Benchmarks mit der Ausgabe von VPR, die sich im Unterverzeichnis `vpr_routings` befindet, bezüglich folgender Kriterien zu vergleichen:

- Laufzeit des Algorithmus
- Kritischer Pfad bzw. Taktfrequenz
- Minimale Breite der Verdrahtungskanäle

Die Ergebnisse der Vergleiche sind zu analysieren und zu bewerten. Die vorgegebenen Netzlisten einhalten BLE mit maximal einer LUT und einem Flipflop. Bei Wunsch können mit `t-vpack` aus `*.blif`-Dateien Netzliste mit komplexeren Clustern erzeugt werden, z. B. 4 LUT und 4 Flipflops sowie 10 verfügbaren Logikeingängen pro Cluster. Es ist ausreichend, die Aufgabe für eine Architektur

(entweder mit oder ohne Cluster) umzusetzen. Die mit VPR verdrahteten Netzlisten wurden auf minimale Breite der Verdrahtungskanäle optimiert (minimale Anzahl der Leitungen pro Kanal, mit denen die Netzliste noch verdrahtet werden kann). Bei Wunsch und Interesse können Analysen für den sogenannten Fall von *„relaxed routing“* durchgeführt werden, d. h. für eine Verdrahtung mit 20 % mehr Leitungen als die minimal benötigte Anzahl (verbessert in der Regel das Timing). In diesem Fall müssen die unter `vpr_routings` abgelegten Referenzverdrahtungen von VPR natürlich neu generiert werden. Es ist ausreichend, die Aufgabe für einen Fall (entweder *„relaxed“* oder minimal) umzusetzen.

3.4 Alternative Aufgabenstellungen

Bei Interesse können auch weitere Aufgabenstellungn aus den Bereichen Logiksynthese, Partitionierung, Technologie-Abbildung, Floorplaning usw. diskutiert werden, es besteht jedoch kein Anspruch darauf. Als Anregung seien folgende Projekte empfohlen:

`https://github.com/verilog-to-routing/vtr-verilog-to-routing`
`http://opencircuitdesign.com/verilog/index.html`

Die Randbedingungen sind **vorab** mit dem Praktikumsleiter zu besprechen.