Laborübung 4

Die Bearbeitung der Aufgaben erfolgt in Gruppen zu je max zwei Teilnehmern. Alle Aufgaben sind schriftlich zu lösen und in einem Protokoll festzuhalten. Alle praktische Realisierungen sind dem Dozenten im Rahmen der Gruppenrücksprache zu präsentieren. Erfolgreiche Abnahmen werden auf dem Aufgabenblatt schriftlich quittiert. Eine erfolgreiche Laborteilnahme ist Voraussetzung für die Teilnahme an der Klausur am Ende des Semesters. Labore werden benotet und gehen zu 30% in die Endnote ein. Bitte bedenken Sie, dass eine gute Vorbereitung zu den Laborterminen unabdingbar ist.

Vorbereitung

Laden Sie sich das Vivado Tutorial und die dazugehörige Zipdatei *ug871-design-files.zip* von der Seite von Prof. Dr. Sven-Hendrik Voß herunter. Diese Dateien sind für die Vivado Version 2016.2, so wie sie auch auf den Laborcomputern installiert ist.

Falls Sie auf Ihrem Privatrechner selbst Vivado installiert haben, können Sie sich die beiden Dateien auf der Xilinx Homepage `https://www.xilinx.com/support/documentation-navigation/self-paced-tutorials/see-all-tutorials.html` herunterladen. Setzen Sie dort an der linken Seite den Haken bei *High Level Synthesis*, klicken Sie dann auf *See all Versions*. Dort finden Sie eine Auswahl für die verschiedenen Versionen. Zum Herunterladen ist allerdings ein Xilinx Account nötig, den man sich in der Regel in ein paar Minuten anlegen kann.

Hinweis für die Version 2016.2: Wenn Sie die Zip-Datei auspacken, wundern Sie sich nicht, dass der Ordner *2016.1* heißt.

Aufgabe 1

Öffnen Sie Vivado HLS und klicken Sie auf *Create New Project*, um ein neues Projekt anzulegen. Auf der ersten Seite des Wizards geben Sie einen Namen an.

Fügen Sie auf der nächsten Seite die Datei *fir.c* hinzu, drücken Sie dazu den *Add Files...* Button, die Datei finden Sie in der Zipdatei unter Introduction/lab1. Setzen Sie ebenfalls die Top Function mit dem Button *Browse...* auf die *fir* Funktion. Auf der nächsten Seite fügen Sie die *fir_test.c* und *out.gold.dat* hinzu. Tipp: Wenn Sie beim Klicken die *STRG*-Taste gedrückt halten, können Sie eine Mehrfachselektion ausführen.

Als nächstes klicken Sie bei *Part* den ...-Button, wählen Sie wieder bei *Boards* das Zedboard aus und klicken Sie *Finish*. Nun öffnet sich ein Fenster, eine Übersicht über dieses finden Sie auf Seite 17 im HLS Tutorial, *Figure 2-8*.

Gucken Sie sich zunächst erst einmal in der IDE um. Wenn Sie mit der Maus über

einen Button gehen, sehen Sie seine Funktion. Ändern Sie ruhig einmal die *Perspective* oben rechts. Lesen Sie sich nun den Code durch. Als nächstes drücken Sie den Button *Run C Simulation* und bestätigen Sie mit *OK*. Sie bekommen die Bestätigung der Funktion aus der Testbench in der Konsole angezeigt. Ihr Code ist nun bereit, synthetisiert zu werden. Klicken Sie dazu auf den Button *Run C Synthesis*. Wenn diese fertig ist, öffnet sich automatisch der Report der C Synthese. Lesen Sie sich diesen durch.

Wie viele Zyklen braucht der Code? Gucken Sie dazu unter Performance Estimates → Detail → Loop.

Wie viel Ressourcen werden benötigt? Gucken Sie dazu im Report unter *Utilization Estimates*. Auffällig: Wie man unter *Instance* sieht, werden für eine Multiplikation vier DSP48E benötigt. Warum?

Unter *Interface* sehen Sie, dass für Sie automatisch Signale angelegt werden, mit denen Ihr Block später mit seiner Umwelt kommunizieren kann. Gucken Sie sich jetzt die *Perspective Analyse* mal genauer an. In *Figure 2-25* auf Seite 39 des Tutorials finden Sie eine Übersicht dieser Ansicht. Erkunden Sie diese ein wenig.

In dem Tab *Performance* sehen Sie eine Tabelle mit allen Operationen. Nach rechts weg gehen die Zyklen der Bearbeitung, nach unten sehen Sie alle Operationen aufgelistet. Einfache Operationen werden in blau, Schleifen in gold und Unterfunktionen in grün dargestellt, die Breite der Kästchen gibt die Länge der Ausführungszeit an. Operationen, deren Kästchen untereinander beginnen, werden gleichzeitig ausgeführt.

Klappen Sie die Operationen aus, wenn vor ihnen ein Plus zu sehen ist und gucken Sie sich die Operationen mal genauer an. Mit einem Rechtsklick in die farbigen Blöcke finden Sie die Option *Goto Source*. Dort können Sie zu der Operation die dazugehörige Zeile im Quelltext sehen.

Welche Operationen benötigt das Design und wie lange braucht jede?

Als nächstes drücken sie den Button *Run C/RTL Cosimulation* und bestätigen wieder mit *OK*. Jetzt wird ihr RTL Code simuliert und durch die C Testbench getestet. Auch hier bekommen Sie wieder das Ergebnis in der Konsole angezeigt. Nun kann ihr Code als IP Core gepackt werden. Klicken Sie den Button *Export RTL*. Stellen Sie sicher, dass *IP Catalog* eingestellt ist und drücken Sie *OK*.

Damit ist der IP Core, welcher in C Code beschrieben wurde, fertig.

Aufgabe 2

Jetzt können Sie kreativ werden: Schreiben Sie Ihren eigenen Code oder bringen Sie selbst ein paar Algorithmen in C mit. Synthetisieren Sie den Code mit HLS und analysieren Sie den Ressourcenverbrauch und das Timing Ihres Codes.

Nur vom Dozenten nach erfolgreicher Abnahme auszufüllen!

Gruppenname

Abnahme

Aufgabe 1	Aufgabe 2

Beurteilung

Kriterium	Max.	Punkte
Themenbearbeitung entsprechend Zielstellung	15	
Fundierte Erarbeitung des Lösungsansatzes	20	
Qualität der technischen Umsetzung	15	
Funktionsfähigkeit der technischen Umsetzung	20	
Darstellung von Ergebnissen und Lösungsvorschlägen	15	
Bearbeitungszeit	15	
Gesamtpunkte für Beurteilungskriterium	100	
Note		

Index	95	90	85	80	75	70	65	60	55	50
Note	1,0	1,3	1,7	2,0	2,3	2,7	3,0	3,3	3,7	4,0