georg.von-der-brueggen [☺] tu-dortmund.de
harun.teper [☺] tu-dortmund.de

Hardwarepraktikum
Wintersemester 21/22

Übungsblatt 1 – 10 Punkte

(Block 1 – insgesamt 60 Punkte)

Bearbeiten ab Samstag, 16. Oktober 2021.
Abgabe bis spätestens Freitag, 22. Oktober 2021, 23:59 Uhr.

In diesem Praktikum werdet Ihr digitale Schaltungen implementieren, die Ihr in Rechnerstrukturen kennengelernt habt. Dazu zählen Gatter, Multiplexer, Codierer und andere einfache Komponenten, sowie komplexere Schaltungen wie Addierer, Multiplizierer, Flip-Flops, Speicher und endliche Automaten. Zum Abschluss des Praktikums werdet Ihr einen MIPS-Prozessor implementieren und erweitern, wobei wir euch die wesentlichen Entwurfsschritte als Hilfestellung vorgeben werden. Nach dem Praktikum werdet Ihr in der Lage sein, euren eigenen Prozessor zu entwickeln. Für die Implementierung aller Schaltungen werden wir eine Hardwarebeschreibungssprache verwenden.

Bei Fragen oder Unklarheiten könnt Ihr gerne im Forum ein Thema erstellen, euch an eure Übungsleitung wenden, oder am Online-Helpdesk teilnehmen.

1.1 VHDL und GHDL + GTKWave (10 Punkte)

VHDL (VHSIC (Very High Speed Integrated Circuit) Hardware Description Language) ist eine Hardwarebeschreibungssprache, mit der unabhängig vom Stand der Technik digitale Systeme auf unterschiedlichen Abstraktionsebenen modelliert und simuliert werden können. Man kann also mit VHDL Modelle von digitalen Schaltungen als Code aufstellen und sie dann ausführen, wobei der Code maschinen- und anwenderlesbar ist.

Um ein digitales System in VHDL zu entwerfen, spezifiziert ein Designer zuerst die benötigten Komponenten als Black Box, indem nur die In- und Outputs angegeben werden. Danach erfolgt eine Implementierung der Funktionalität und die Verbindung der einzelnen Komponenten. Im nächsten Schritt wird ein Simulationsprogramm genutzt, um das Modell auf korrekte Funktionalität zu prüfen. Nach ausgiebigen Tests kann dann der VHDL-code synthetisiert werden. Das heißt, echte Hardware kann direkt aus dem VHDL-Code erstellt werden. In der Online-Variante des HaPras werden wir uns vorerst nicht mit der Synthese beschäftigen. Dafür bräuchte man FPGAs (Field Programmable Gate Array). Wir werden in diesem Praktikum jedoch mit Simulatoren arbeiten.

GHDL [1] ist ein Open Source Simulator für die VHDL Sprache, mit dem VHDL Programme auf dem Heim- oder Arbeitsrechner kompiliert und ausführt werden können, ganz ohne zusätzliche Hardware. Damit die Funktionalität der entworfenen Schaltungen selbst nachvollziehen und verifizieren zu können, wird noch Software benötigt um die Signale zu plotten. Dafür nutzen wir in diesem Praktikum GTKWave. GHDL und GTKWave laufen bei uns auf Windows, Linux-Distributionen und OS X ohne größere Schwierigkeiten.

Im Folgenden beschreiben wir, wie Ihr GHDL und GTKWave auf eurem Rechner installieren und ausführen könnt. GHDL und GTKWave kann auf Linux mit dem folgenden Befehl installiert werden:

```
$ sudo apt-get install ghdl gtkwave
```

Schaut bitte für andere Betriebssysteme auf den entsprechenden Websites nach [3, 2]. Falls mal eine Installation an dem Rechner an dem gearbeitet wird nicht möglich ist, dann kann im Notfall auch das Online-Tool EDA Playground [4] verwendet werden.

Um VHDL-Code mit GHDL zu kompilieren und mit GTKWave anschauen zu können, können folgende Befehle nacheinander ausgeführt werden ("#"kennzeichnet den Beginn eines Kommentars):

```
$ ghdl -s test_file.vhdl # Syntax-Check
$ ghdl -a test_file.vhdl # Analyse
$ ghdl -e test_file # Build
$ ghdl -r test_file --vcd=testbench.vcd # VCD-Dump
$ gtkwave testbench.vcd # Startet GTKWave
```

Aufgaben:

a. (3 Punkte) Führt die fünf Befehle mit *test_file* aus und betrachtet die Wellenform in GTKWave. Schaut danach in die Dokumentation von GHDL hinein und sucht nach Erläuterungen zu den erwähnten Befehlen zur Simulation von Systemen in VHDL. Schreibt zu jedem Befehl eine kurze Zusammenfassung.

b. (3 Punkte) Entwerft ein Skript (z.B. ein Python-Skript oder ein Bash-Skript unter Linux) das die oben beschriebenen Befehle automatisch ausführt und als Parameter eine Liste von Quellcode-Dateien übergeben bekommt. Das Skript soll nach jedem Befehl entweder eine Fehlernachricht oder ein OK ausgeben. Bei erfolgreicher Ausführung z.B.: *Syntax-check OK, Analysis OK, Build OK, VCD-Dump OK, Starting GTKWave*. Und bei fehlerhafter Ausführung soll z.B. Folgendes ausgegeben werden: *Syntax-check OK, Analysis OK, Build Failed*.

   Hinweise: (1) Bitte kommentiert den Code hier (und bei allen anderen Abgaben) ausreichend.
   (2) GHDL akzeptiert auch eine Liste von Quellcode-Dateien, z.B. *test_file.vhdl* für eine Datei. Für den weiteren Verlauf der Übung soll es auch möglich sein, eine Liste mit mehreren Elementen, z.B. der Form *ha.vhdl fa.vdhl fa_tb.vhdl* anzugeben. Ihr könnt davon ausgehen, dass die Testbench-Datei immer als letztes übergeben wird.
   (3) Um die Kompilation und Ausführung mit mehreren Dateien zu testen, nutzt bitte die bereitgestellten Dateien *adder.vhdl* und *adder_tb.vhdl*. Mit dem Inhalt der beiden Dateien müsst Ihr euch vorerst nicht beschäftigen. Bei einem korrekten Ablauf der Befehle sollte *adder_tb.vhdl:54:5:@8ns:(assertion note): end of test* ausgegeben werden und in GTKWave sollten einige Wellenformen angezeigt werden.

c. (0 Punkte) Grundwissen über die Modellierung von Schaltungen in VHDL wird im weiteren Verlauf des Praktikums nötig sein. Arbeitet dazu im VHDL-Skript das erste Kapitel durch.

d. (4 Punkte) Fertigt eine Skizze der Schaltung an, die im unten stehenden VHDL-Code implementiert ist. Verwendet die IEC 60617-12 Norm für die Zeichnung von Gattern. Beschriftet eure Skizze mit den Bezeichnungen a-g und gebt die implementierte boolsche Funktion an.

```
entity unknown is
  port(
    a : in std_logic;
    b : in std_logic;
    c : in std_logic;
    d : in std_logic;
    e : out std_logic
  );
end unknown;

architecture behavior of unknown is
  signal f : std_logic;
  signal g : std_logic;
begin
  f <= a or b;
  g <= c or d;
  e <= f and g;
end unknown;
```

Literatur

[1] GHDL GitHub Repository.
https://github.com/ghdl/ghdl

[2] GTKWave Website.
http://gtkwave.sourceforge.net/

[3] GHDL Installation Instructions.
http://ghdl.free.fr/site/pmwiki.php?n=Main.Installation

[4] EDA Playground.
https://www.edaplayground.com/