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

Hardwarepraktikum
Wintersemester 21/22

Übungsblatt 7 – 30 Punkte

(Block B3 – insgesamt 70 Punkte)

Bearbeiten ab Samstag, 4. Dezember 2021.
Abgabe bis spätestens Freitag, 10. Dezember 2021, 23:59 Uhr.

7.1 Multiplizierwerk in VHDL (30 Punkte)

Das Multiplizieren ist eine der häufigsten Rechenoperationen die wir in unserem täglichen Leben durchführen. Ist die Grundregel des Multiplizierens einmal verstanden, kann das Produkt von zwei beliebigen Zahlen einfach errechnet werden. Abbildung 1 zeigt beispielhaft eine Multiplikation im Dezimalsystem mit Multiplikator 13 und Multiplikand 12 und dem Ergebnis 156.

```
    13
x   12
-------
    26
+ 13
-------
   156
```

Abbildung 1: Multiplikation im Dezimalsystem.

```
    1101
x   1100
---------
    0000
   0000
  1101
+1101
---------
10011100
```

Abbildung 2: Multiplikation im Binärsystem.

Da Computer jedoch im Binärsystem rechnen, müssen diese auch im Binärsystem multiplizieren können. Nur zur Darstellung für den Menschen wird zwischen dem Binär- und Dezimalsystem konvertiert. In Abbildung 2 ist die entsprechende binäre Multiplikation mit Multiplikator 1101 und Multiplikand 1100 dargestellt. Zu erkennen ist auch hier, dass in jeder Zeile eine Stelle des Multiplikanden mit dem vollständigen Multiplikator multipliziert wird. Anschließend werden die, je nach Stelle verschobenen, Teilprodukte aufsummiert um das Ergebnis zu berechnen. Abbildung 3 zeigt das allgemeine Vorgehen der Multiplikation von zwei 4-Bit Zahlen $x_3x_2x_1x_0$ und $y_3y_2y_1y_0$ mit dem Produkt z_7-z_0.

$$
\begin{array}{cccccccc}
 & & & x_3 & x_2 & x_1 & x_0 \\
 & & & \times\ y_3 & y_2 & y_1 & y_0 \\
\hline
 & & & x_3y_0 & x_2y_0 & x_1y_0 & x_0y_0 \\
+ & & x_3y_1 & x_2y_1 & x_1y_1 & x_0y_1 & & \\
+ & x_3y_2 & x_2y_2 & x_1y_2 & x_0y_2 & & & \\
+ & x_3y_3 & x_2y_3 & x_1y_3 & x_0y_3 & & & \\
\hline
z_7 & z_6 & z_5 & z_4 & z_3 & z_2 & z_1 & z_0
\end{array}
$$

Abbildung 3: Beispiel einer 4-Bit Multiplikation.

Aufgaben: Der parallel Multiplizier ist einer der häufigsten Realisierungen. In Abbildung 4 ist das Blockschaltdiagramm für 4-Bit Multiplikator $x_3x_2x_1x_0$, 4-Bit Multiplikand $y_3y_2y_1y_0$ und 8-Bit Ergebnis z_7-z_0 angegeben.

a. (4 Punkte) Untersucht die Schaltung, ergründet die Funktionsweise der einzelnen Blöcke und vervollständigt die Blöcke mit Euch bekannten Gattern oder Kombinationen von Gattern.

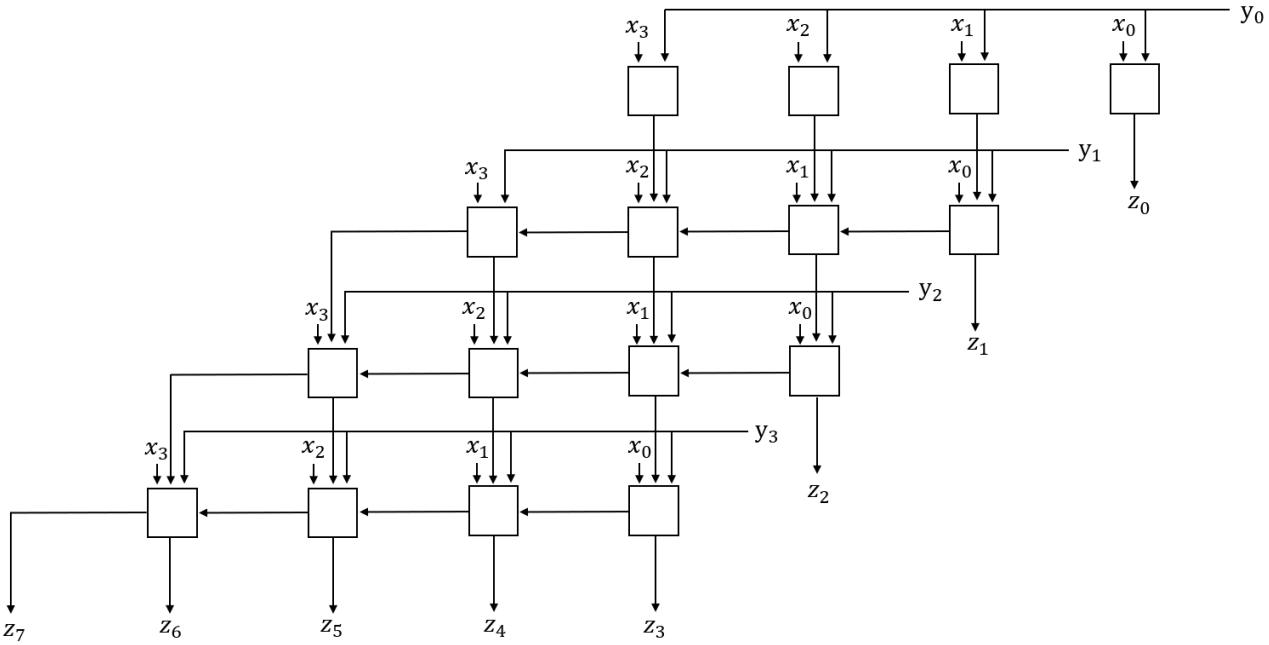

Abbildung 4: Ein parallel Multiplizierer.

b. (8 Punkte) Implementiere den vollständigen 4-Bit parallel Multiplizierer und teste diesen mit den Eingaben $x_3x_2x_1x_0 \in \{1010, 1001\}$ und $y_3y_2y_1y_0 \in \{1101, 0111\}$.

Eine andere Realisierung des Multiplizierwerks ist die *add shift* Methode, welches Schieberegister, Addierwerke und endlichen Automaten nutzt. Das Schaltbild und das Flussdiagramm des Multiplizierwerks sind in Abbildung 5 bzw. Abbildung 6 dargestellt.

Zuerst müsst Ihr das LSB (least-significant bit) des Multiplikators überprüfen. Wenn LSB=1 gilt, addiert den jetzigen Inhalt des Multiplikanden mit dem derzeitigen Produkt und schiebt danach den Multiplikanden eine Stelle nach links und den Multiplikator eine Stelle nach rechts. Wenn LSB=0 gilt, müsst Ihr nichts auf das Produkt addieren, sondern schiebt direkt den Multiplikanden eine Stelle nach links und den Multiplikator eine Stelle nach rechts.

c. (8 Punkte) Implementiere das Multiplizierwerk in VHDL mithilfe des Schieberegisters, eines Addierwerks und eines endlichen Automaten und teste die Schaltung erneut mit den Eingaben $x_3x_2x_1x_0 \in \{1010, 1001\}$ und $y_3y_2y_1y_0 \in \{1101, 0111\}$.

d. (2 Punkte) Vergleiche den *add shift* Multiplizierer aus Abbildung 5 mit dem parallel Multiplizierer aus Abbildung 4. Gebe die Vor- und Nachteile der jeweiligen Realisierung an.

e. (6 Punkte) Die Implementierung des Multiplikators in Abbildung 5 ist noch ineffizient, könnt ihr versuchen, diesen Multiplikator zu optimieren? Zeichnet eure Entwürfe und implementiert es in VHDL.

f. (2 Punkte) Teste eure Implementierung mit unterschiedlicher Eingaben.

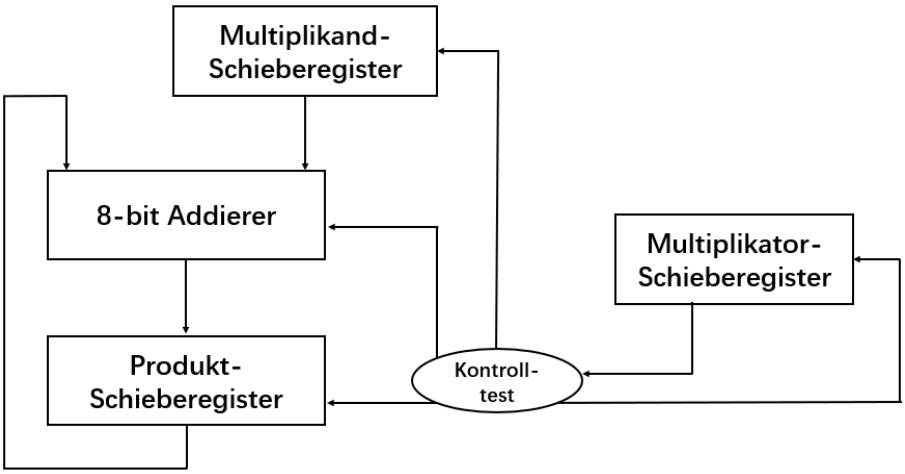

Abbildung 5: Schaltbild eines Multiplizierwerk nach der *add shift* Methode.

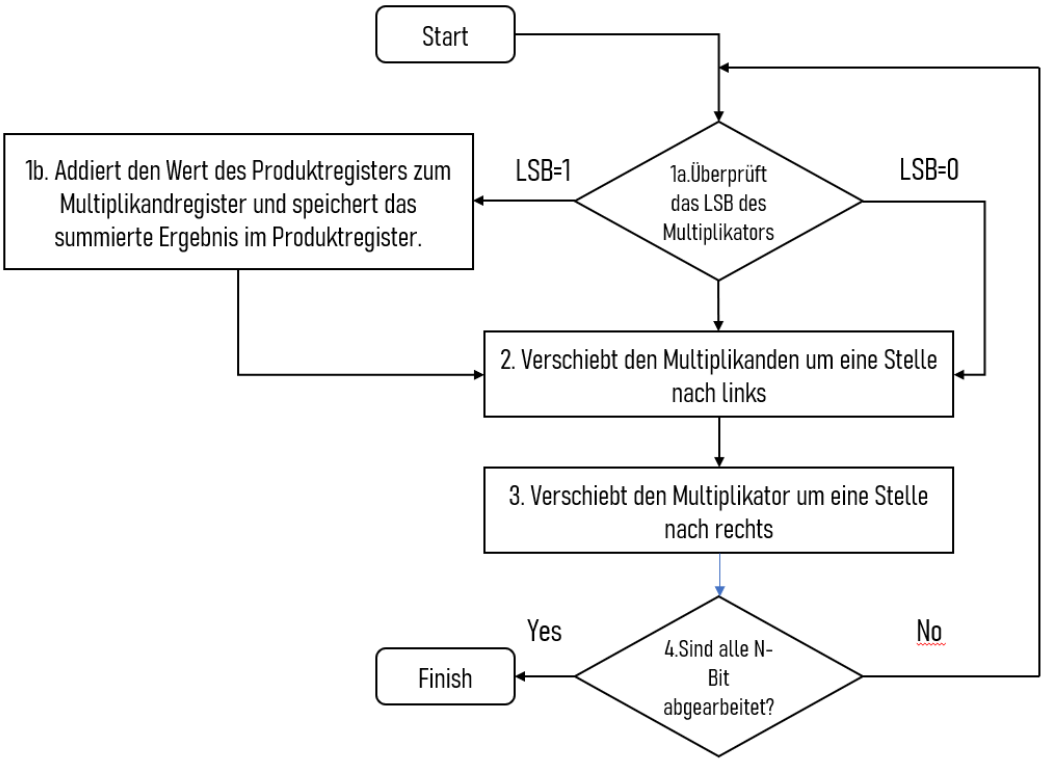

Abbildung 6: Flussdiagramm des Multiplizierwerk nach der *add shift* Methode.