DI - Praktikum 1

Fehlersichere Übertragung und Speicherung

Florian Tietjen Eric Antosch

2020

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Fehlersichere Übertragung und Speicherung von Daten

Aufgabe 1.0

In diesem Praktikumsversuch geht es darum, die Übertragung und Speicherung einfacher Daten innerhalb von System zu untersuchen und zu verstehen. Dabei wollen wir nicht nur eine theoretische Betrachtung eines Paritätsgenerators betrachten, sondern diesen auch innerhalb von Xilinx Vivado beschreiben und simulieren.

1.1 Vorbereitung: Schaltungaufbau

1.1.1 Sender

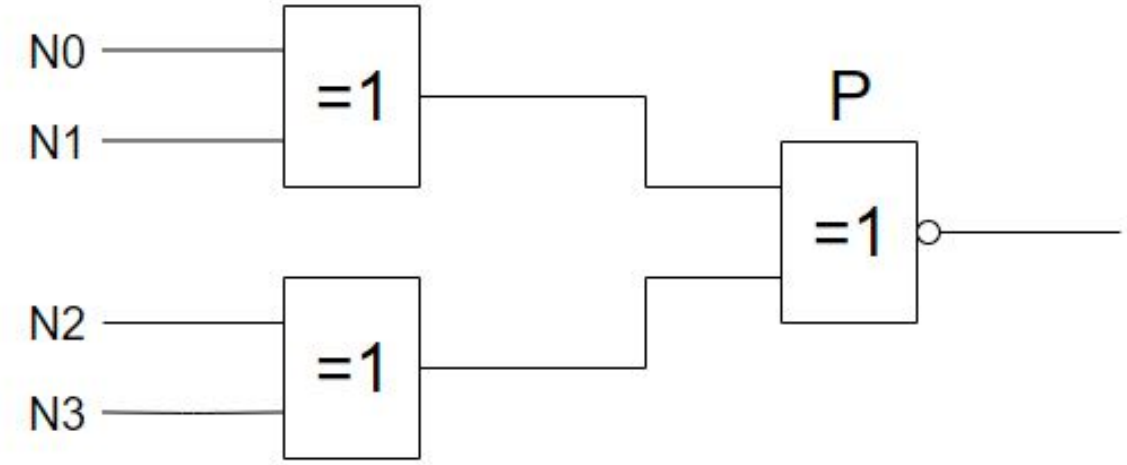

Abbildung 1: Schaltbild des Senders

Wir erstellen zunächst ein Schaltbild eines Senders, welche nur ungerade Paritätsbits zulässt. Dabei gilt zu beachten, dass, im Gegensatz zu einem Sender, der gerade Paritätsbits zulässt, das letzte XOR-Gatter negiert ist und nun zu einem XNOR-Gatter wird.

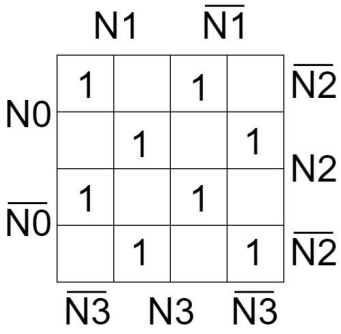

Abbildung 2: KV-Diagramm des Senders

$N0$	$N1$	$N2$	$N3$	P
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Tabelle 1: Wahrheitstabelle vom ungeraden Paritätsbit

Unter der Betrachtung der Wahrheitstabelle und des KV-Diagramms entsteht folgende bool'sche Formel:

$$P = (\overline{N0} \wedge \overline{N1} \wedge \overline{N2} \wedge \overline{N3}) \vee (\overline{N0} \wedge \overline{N1} \wedge N2 \wedge N3) \vee (\overline{N0} \wedge N1 \wedge \overline{N2} \wedge N3) \vee (\overline{N0} \wedge N1 \wedge N2 \wedge \overline{N3})$$
$$\vee (N0 \wedge \overline{N1} \wedge \overline{N2} \wedge N3) \vee (N0 \wedge \overline{N1} \wedge N2 \wedge \overline{N3}) \vee (N0 \wedge N1 \wedge \overline{N2} \wedge \overline{N3}) \vee (N0 \wedge N1 \wedge N2 \wedge N3)$$

Wir wollen diese nun einmal leicht vereinfachen, sodass wir diese auch leichter in unser Programm einbauen können:

$$P = \overline{(N0 \oplus N1) \oplus (N2 \oplus N3)}, \tag{1}$$

wobei es zu erwähnen gilt, dass $\oplus$ hier für die XOR-Verknüpfung gilt. (Diese wird auf dem Schaltplan mit $= 1$ dargestellt).

1.1.2 Empfänger

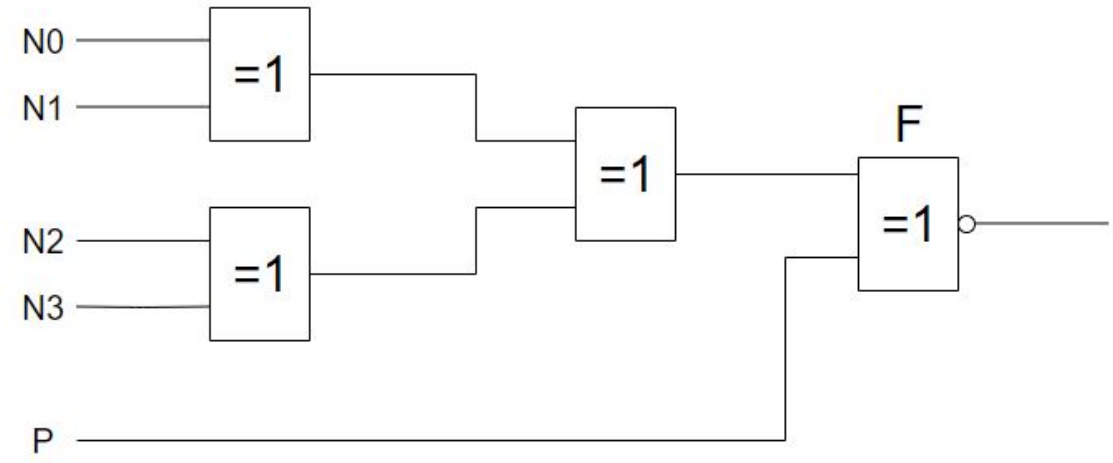

Abbildung 3: Schaltbild des Empfängersystems

Im Empfängersystem wird nun überprüft, ob es eine gerade oder ungerade Anzahl von Einsen gibt. Im Falle, dass sich eine ungerade Anzahl an Einsen ergibt, entsteht kein Fehlersignal F (bzw. $F = 0$). Sollte es jedoch zu einem Fehler kommen, der durch verschiedene Einflüsse auf die Geräte kommen kann, und es sind nun eine gerade Anzahl an Einsen vorhanden, so entsteht ein Fehlersignal am Ausgang ($F = 1$).

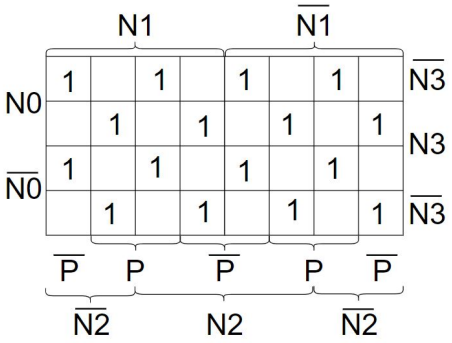

Abbildung 4: KV-Diagramm des Empfängers

$N0$	$N1$	$N2$	$N3$	P	F
0	0	0	0	1	0
0	1	1	0	1	0
0	1	0	0	1	1
1	0	1	0	0	1

Tabelle 2: Wahrheitstabelle vom ungeraden Paritätsbit

Analog zum Sendersystem wollen wir hier auch einmal betrachten, wie sich aus dem KV-Diagramm die bool'sche Formel für den Wahrheitswert von F ergibt. (Eine vollständige Wahrheitstabelle ist im Anhang zu finden (Seite: 12)).

$$
\begin{aligned}
F = (&\overline{N0} \wedge \overline{N1} \wedge \overline{N2} \wedge \overline{N3} \wedge \overline{P}) \vee (\overline{N0} \wedge \overline{N1} \wedge \overline{N2} \wedge N3 \wedge P) \vee (\overline{N0} \wedge \overline{N1} \wedge N2 \wedge \overline{N3} \wedge P) \\
&\vee (\overline{N0} \wedge \overline{N1} \wedge N2 \wedge N3 \wedge \overline{P}) \vee (\overline{N0} \wedge N1 \wedge \overline{N2} \wedge \overline{N3} \wedge P) \vee (\overline{N0} \wedge N1 \wedge \overline{N2} \wedge N3 \wedge \overline{P}) \\
&\vee (\overline{N0} \wedge N1 \wedge N2 \wedge \overline{N3} \wedge \overline{P}) \vee (\overline{N0} \wedge N1 \wedge N2 \wedge N3 \wedge P) \vee (N0 \wedge \overline{N1} \wedge \overline{N2} \wedge \overline{N3} \wedge P) \\
&\vee (N0 \wedge \overline{N1} \wedge \overline{N2} \wedge N3 \wedge \overline{P}) \vee (N0 \wedge \overline{N1} \wedge N2 \wedge \overline{N3} \wedge \overline{P}) \vee (N0 \wedge \overline{N1} \wedge N2 \wedge N3 \wedge P) \\
&\vee (N0 \wedge N1 \wedge \overline{N2} \wedge \overline{N3} \wedge \overline{P}) \vee (N0 \wedge N1 \wedge \overline{N2} \wedge N3 \wedge P) \vee (N0 \wedge N1 \wedge N2 \wedge \overline{N3} \wedge P) \\
&\vee (N0 \wedge N1 \wedge N2 \wedge N3 \wedge \overline{P})
\end{aligned}
$$

Auch aus dieser Formel ist es sinnvoll, eine einfachere Form zu bilden, um diese übersichtlich zu halten:

$$F = \overline{P \oplus ((N0 \oplus N1) \oplus (N2 \oplus N3))} \tag{2}$$

Mit dieser Vorbereitung steht nun der eigentlichen Beschreibung der Hardware mithilfe von VHDL und Xilinx Vivado nichts mehr im Weg.

1.2 Hardwarebeschreibung und -simulation mit Vivado

1.2.1 Source: Paritätsgenerator und Empfängersystem

Zunächst werden die beiden Sources, der Paritätsgenerator bzw. unser Sender, und das Empfängersystem mithilfe der Hardwarebeschreibungssprache VHDL in das Projekt selbst eingepflegt:

```
1  entity sender is
2      Port (N : in bit_vector(3 downto 0);
3            P : out bit);
4
5  end sender;
6
7  architecture senderBehav of sender is
8  begin
9      P <= (N(0) XOR N(1)) XNOR (N(2) XOR N(3));
10 end senderBehav;
```

Abbildung 5: Sender-VHDL-Beschreibung (Paritätsgenerator)

```
1  entity receiver is
2      Port (N : in bit_vector(3 downto 0);
3            P : in bit;
4            F : out bit);
5
6  end receiver;
7
8  architecture receiverBehav of receiver is
9  begin
10    F <= ((N(0) XOR N(1)) XOR (N(2) XOR N(3))) XNOR P;
11 end receiverBehav;
```

Abbildung 6: Empfänger-VHDL-Beschreibung

1.2.2 Testbench: Paritätsgenerator und Empfängersystem

Nachdem nun beide Bauteile erfolgreich mit VHDL beschrieben worden sind, wollen wir uns nun daran machen, die beiden Systeme mit einer entsprechenden Testbench zu simulieren. Dazu erstellen wir für jedes der Bauteile eine einzelne Datei, in der wir die Test, die später in der Simulation ablaufen sollen, beschreiben:

```
1  entity senderTB is
2
3  end senderTB;
4
5  architecture senderTBehav of senderTB is
6  component sender
7      Port (
8          N : in bit_vector(3 downto 0);
9          P : out bit);
10 end component;
11 signal NSi : bit_vector(3 downto 0);
12 signal PSi : bit;
13 begin
14     TEST: sender port map(N(3) => NSi(3), N(2) => NSi(2), N(1) => NSi(1),
15                           N(0) => NSi(0), P => PSi);
16
17 process
18 begin
19     NSi <= "0000";
20     wait for 100 ns;
21     NSi <= "0001";
22     wait for 100 ns;
23     NSi <= "0010";
24     wait for 100 ns;
25     NSi <= "0011";
26     wait for 100 ns;
27     NSi <= "0100";
28     wait;
29 end process;
30 end senderTBehav;
```

Abbildung 7: Sender-Testbench (Paritätsgenerator)

```
1  entity receiverTB is
2
3  end receiverTB;
4
5  architecture receiverTBehav of receiverTB is
6      component receiver
7          Port (
8              N : in bit_vector(3 downto 0);
9              P : in bit;
10             F : out bit);
11     end component;
12     signal NSi : bit_vector(4 downto 0);
13     signal FSi : bit;
14     begin
15         TEST: receiver port map(P => NSi(4), N(3) => NSi(3), N(2) => NSi(2), N(1) => NSi(1),
16                                 N(0) => NSi(0), F => FSi);
17
18     process
19     begin
20         NSi <= "10000";
21         wait for 100 ns;
22         NSi <= "00001";
23         wait for 100 ns;
24         NSi <= "00000";
25         wait for 100 ns;
26         NSi <= "11111";
27         wait for 100 ns;
28         NSi <= "10111";
29         wait;
30     end process;
31     end receiverTBehav;
```

Abbildung 8: Empfänger-Testbench

1.2.3 Testbench: Auswertung der Ergebnisse im Waveform-Viewer

Im letzten Abschnitt wollen wir nun die Auswertung der Ergebnisse der Simulation aus dem Waveform-Viewer von Vivado vornehmen. Dazu betrachten wir zunächst unseren Paritätsgenerator um festzustellen, ob wir diesen richtig implementiert haben.

Wir können gut erkennen, dass unser Paritätsgenerator das erwartete Verhalten zeigt. So wird zum Beispiel im Bereich von 0ns bis 100ns eine 1 auf das Paritätsbit (in unserem Fall PSi genannnt) gesetzt, da beide vor geschalteten XOR-Gatter eine 0 ausgeben.

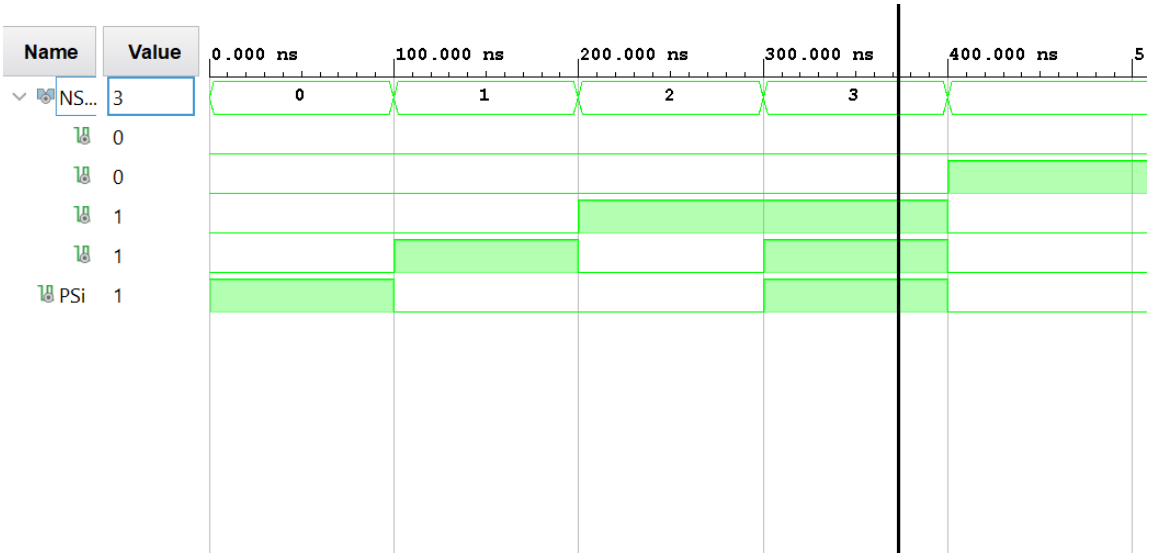

Abbildung 9: Waveform des Senders (Paritätsgenerator)

Nun wollen wir uns auch unseren Empfänger anschauen:

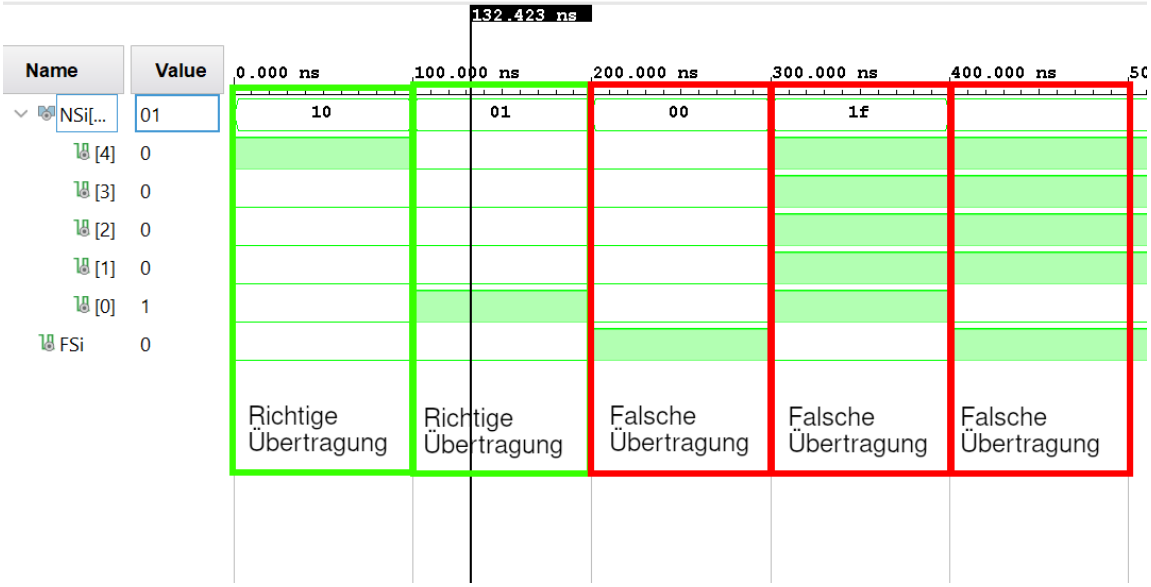

Abbildung 10: Waveform des Empfängers

Es wird deutlich, dass sich unser Sender-Empfängersystem unseren Vorstellungen entsprechend verhält. Zunächst werden zwei Übertragungen (0ns-100ns, 100ns-200ns) gemacht, die richtige Einstellungen haben. Im Bereich von 200ns-300ns liegt ein einfacher Fehler vor, der von dem Fehlersignal FSi richtig erkannt wird. Zwischen 300ns und 400ns sind jedoch zwei Fehler eingebaut, was von unserer Überprüfungsstrecke nicht korrekt erfasst wird. Im letzten Bereich ist unser Paritätssignal nicht korrekt, es entsteht erneut ein Fehler am Ausgang FSi.

1.3 Fehler

Es wird deutlich, dass eine einfach Paritätsüberprüfungsstrecke nicht in der Lage ist, Fehler mit gerader Anzahl zu ermitteln. Bei Fehlern mit ungerader Anzahl jedoch, schlägt das Signal aus und

der Fehler könnte von aus behoben werden.

1.4 Freiheiten der Verbesserung der Übertragungsstrecke

Es liegt nahe, einer wichtigen Übertragungsstrecke von Daten mehrere Sicherheitssysteme zu verleihen. So würde es sich beispielsweise anbieten, verschiedene Paritätssignale in eine Übertragungsstrecke einzubauen, um auch Fehler abzufangen, die dann in gerade Anzahl auftauchen. Dies verkompliziert den Aufbau einer solchen Schaltung auch nur im bedingten Maße.

2 Fazit

Es ist eindeutig zu erkennen, dass eine Paritätsprüfung zur Sicherstellung der Datensicherheit nur einen eingeschränkten gegen Fehler bietet. Es gilt sich daher zu überlegen, dieses System mithilfe von Erweiterungen gegen größere Bandbreite an Fehlern abzusichern. Während der Durchführung des Praktikums ist vor allen Dingen die steile Lernkurve sowohl von VHDL als auch Vivado aufgefallen. Es ist zu Beginn sehr viel Zeit in die Einarbeitung in die Programmierweise und das Programm selbst geflossen, sowie in das Beheben von Fehlern, die sich als Flüchtigkeits- oder Unwissenheitsfehler entpuppt haben. Vor Beginn des nächsten Praktikums bietet es sich daher an, mehr über VHDL als Sprache und das Programm Xilinx Vivado zu lernen.

3 Anhang

$N0$	$N1$	$N2$	$N3$	P	F
0	0	0	0	0	1
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	1
0	0	1	0	0	0
0	0	1	0	1	1
0	0	1	1	0	1
0	0	1	1	1	0
0	1	0	0	0	0
0	1	0	0	1	1
0	1	0	1	0	0
0	1	0	1	1	1
0	1	1	0	0	1
0	1	1	0	1	0
0	1	1	1	0	0
0	1	1	1	1	1
1	0	0	0	0	0
1	0	0	0	1	1
1	0	0	1	0	1
1	0	0	1	1	0
1	0	1	0	0	1
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	1
1	1	0	0	0	1
1	1	0	0	1	0
1	1	0	1	0	0
1	1	0	1	1	1
1	1	1	0	0	0
1	1	1	0	1	1
1	1	1	1	0	1
1	1	1	1	1	0

Tabelle 3: Wahrheitstabelle vom Empfänger