Universität Karlsruhe (TH)
Institut für Industrielle Informationstechnik

Entwicklung eines VHDL-Codes zur Nachbildung eines Impulsstörerszenarios für Kfz Powerline Kanäle

Studienarbeit

von

cand. el. Jia Chen

Zeitraum:	01.05.04– 15.09.04
Hauptreferent:	Prof. Dr.-Ing. habil. Klaus Dostert
Betreuer:	Dipl.-Ing. Thorsten Huck [1]

[1]Robert Bosch GmbH - Schwieberdingen

Erklärung

Hiermit erkläre ich, dass die vorliegende Studienarbeit selbstständig und nur unter Verwendung der angegebenen Hilfsmittel und Quellen angefertigt wurde.

Karlsruhe, den 15.09.04

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Powerline Communication im Kraftfahrzeug

Da die Komplexität bzw. die Anzahl der Geräte im Kraftfahrzeug weiter zunimmt, steigt auch der Bedarf an Datenkommunikation. Es folgt daraus, dass zu viele Geräte mit Daten und Energie versorgt werden müssen. Zur Zeit sind im Kfz verschiedene Bussysteme für die Kommunikation der Komponenten im Einsatz: z.B. CAN, LIN sowie FlexRay.

- CAN (Controller Area Network): Hierbei handelt es sich um ein serielles Bussystem. Es können beliebige Geräte sowie Sensoren und Aktoren an den Bus angeschlossen werden, wenn sie die Spezifikationen erfüllen [Bos91].
- LIN (Local Interconnect Network): Dies ist ein kostengünstiges Bussystem, das als Subbus für eine Vernetzung unterhalb des CAN Busses geeignet ist. Der LIN Bus ermöglicht die kostengünstige Kommunikation zwischen kleinen Sensoren und Aktoren. Typische Anwendungsbeispiele sind die Vernetzung innerhalb einer Tür oder eines Sitzes [Wik04].
- FlexRay ist für zukünftige High-Speed Anwendungen gedacht und erlaubt eine Übertragungsdatenrate von bis zu 10 Mbits/s [Huc02].

Diese Bussysteme beruhen alle auf einer getrennten Übertragung von Daten und Energie. Weil die Anzahl an elektrischen Komponenten stetig ansteigt, und damit immer mehr Daten übertragen werden müssen, ist der Kabelbaum solcher Systeme immer komplizierter geworden. Es gibt also ein großes Interesse an einem einheitlichen und standardisierten Buskonzept für die Kommunikation der einzelnen Fahrzeugkomponenten, die sogenannte PLC (Power Line Communication), einer Datenübertragung über die Versorgungsleitungen.

Betrachtet man zur Veranschaulichung den Umfang eines heutigen Kabelbaums, so ergeben sich bei einem High-End Fahrzeug ca. 2000 Leitungen mit einer Gesamtlänge von ca. 3000m und ca. 3800 Steckverbindern [P.B01].

Durch die Einführung der Powerline Communication geht man davon aus, dass 15 bis 25 Prozent der Gesamtkabellänge in einem Auto eingespart werden können [P.B01].

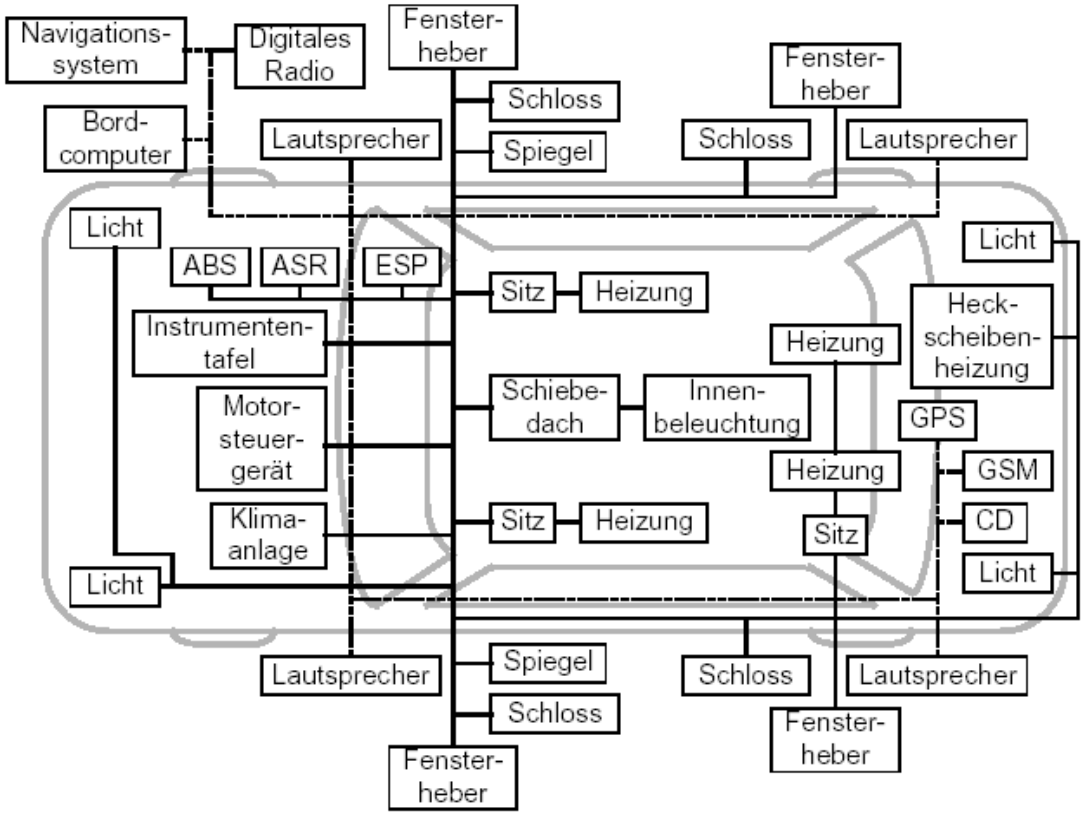


Abbildung 1.1: Schematischer Vernetzungsplan eines Kraftfahrzeugs

Zudem kann der PLC-Bus noch mit relativ hoher Datenrate übertragen: 1, 2, 5 und 10 MBit/s werden angestrebt. Am PLC-Bus werden zunächst nicht sicherheitsrelevante Komponenten (Multimedia (Radio, Navigationssystem) und Komfortelemente (Sitzverstellung o.ä.)) angeschlossen, später soll eine Ausdehnung auf alle Geräte im Kfz stattfinden, d.h. es werden keine weiteren Bussysteme benötigt.

Ein PLC-Bussystem im Kfz existiert in zwei Varianten: als Übertragung über eine Eindraht oder Twisted-Pair Leitung. Eindraht-Kabelbäume bedeuten, dass die Karosserie den Rückleiter darstellt, weswegen man nur eine Leitung für die Energieversorgung benötigt. Bei Twisted-Pair nimmt man zwei verdrillte Leitungen, die an dem '+' und '-' Pol der Versorgung angeschlossen werden. Ein Twisted-Pair Bussystem hat bessere Eigenschaften bezüglich elektromagnetischer Störungen, hinsichtlich der Ein- und Abstrahlung. Die zwei Leitungen befinden sich nahezu an der gleichen Position in einem magnetischen Störfeld, weswegen auch eine gleich hohe Rauschspannung in ihnen induziert wird. Diese kompensiert sich bei der Differenzbildung der Spannung der zwei Leitungen.

Mit dem PLC-Bus werden neben den Versorgungskabeln keine weiteren Datenleitungen benötigt, was einer Kosten- und Gewichtsreduktion entspricht. Den kritischen Stimmen, dass bei einem Twisted-Pair PLC-Bus gegenüber einem herkömmlichen System nichts gespart wird, weil das heutige Bussystem die Karosserie als Rückleiter benutzt, ist entgegenzuhalten, dass die Karosserie immer mehr Kunststoff enthält, was die Nutzung der Karosserie als elektrischen

Rückleiter erschwert. Somit ist ein Twisted-Pair PLC-Bus wieder von wesentlichem Vorteil.

Der PLC-Bus spart nicht nur Kabel, sondern auch Steckverbindungen. Die Verschmutzung der Kontakte eines Steckverbinders verursacht Fehler bei der Übertragung. Diese Verschmutzung wird durch die im PLC-Bus fließende hohe Leistung reduziert, was bei einem reinen Datenbus nicht der Fall ist.

1.2 Ziele der Arbeit

Es gibt diverse Störungen für die Powerline Communication im Kraftfahrzeugbordnetz. Dieses sind zum einen die elektromagnetischen Einwirkungen von außen, für die ein Twisted-Pair Bus System weniger empfindlich ist. Zum zweiten die Störimpulse, die durch die einzelnen Verbraucher wie z. B Scheinwerfer, Klimaanlage, Hupe usw. entstehen. Störimpulse, die sich besonders auf dem Kraftfahrzeugbordnetz bemerkbar machen, entstehen durch die Zündfunken [Huc02].

Das Störimpulsmodell setzt sich aus einer Überlagerung eines periodischen und aperiodischen Anteils zusammen. Der Zündvorgang löst eine Folge von aperiodischen Impulsstörungen aus, während der Zündvorgang periodisch ist. Die Parameter der Störimpulse haben fast alle stochastische Eigenschaften. Man möchte auf Basis einer FPGA-Software beliebig verteilte stochastische Größen erzeugen, die das Störimpulsmodell in Echtzeit simulieren und so einen Kanalemulator auf FPGA Basis vervollständigen. Diese ermöglichen einen Test der PLC Transceiver im Labor unter realen Bedingungen.

Der Emulator für die Störimpulse ist somit ein wichtiger Baustein für die Entwicklung eines PLC Systems im Kraftfahrzeug.

1.3 Aufbau der Arbeit

Bevor die eigentliche Ausarbeitung der Studienarbeit beginnt, werden in Kapitel 2 einige grundlegende Informationen zum Thema Powerline Communication im Kfz dargestellt. Dabei wird die Struktur eines PLC Systems im Kfz, die Besonderheit eines PLC Kanals, sowie die Aufgabe eines Kanalemulators vorgestellt.

Der Hauptteil der Studienarbeit gliedert sich in vier Kapitel:

Kapitel 3 behandelt die zur Verfügung stehenden Messdaten, wie die Messdaten gewonnen und was für Annahmen dabei gemacht wurden. In Kapitel 4 erfolgt die Grundidee des Verfahrens, um die stochastischen Störimpulse nachzubilden. In Kapitel 5 wird die Aufbereitung der vorhandenen Messdaten in Matlab ausführlich erläutert. Kapitel 6 befasst sich mit der Realisierung des Verfahrens in VHDL.

Das abschließende Kapitel 7 enthält eine Zusammenfassung der Arbeit und gibt einen Ausblick auf künftige Aufgabenbereiche.

Der Anhang beschreibt die Funktionen der Matlab Files und gibt zwei Beispiele, wie diese Anwendung finden.

Kapitel 2

Automotive Powerline Communications

2.1 Struktur eines PLC Systems

In Abbildung 2.1 wird ein typisches PLC-System dargestellt. Am Anfang findet sich ein Kanalkodierer, der aufgrund der Störungen auf dem PLC-Kanal erforderlich ist. Bislang wurde jedoch noch kein konkretes Kanalkodierungsverfahren festgelegt.

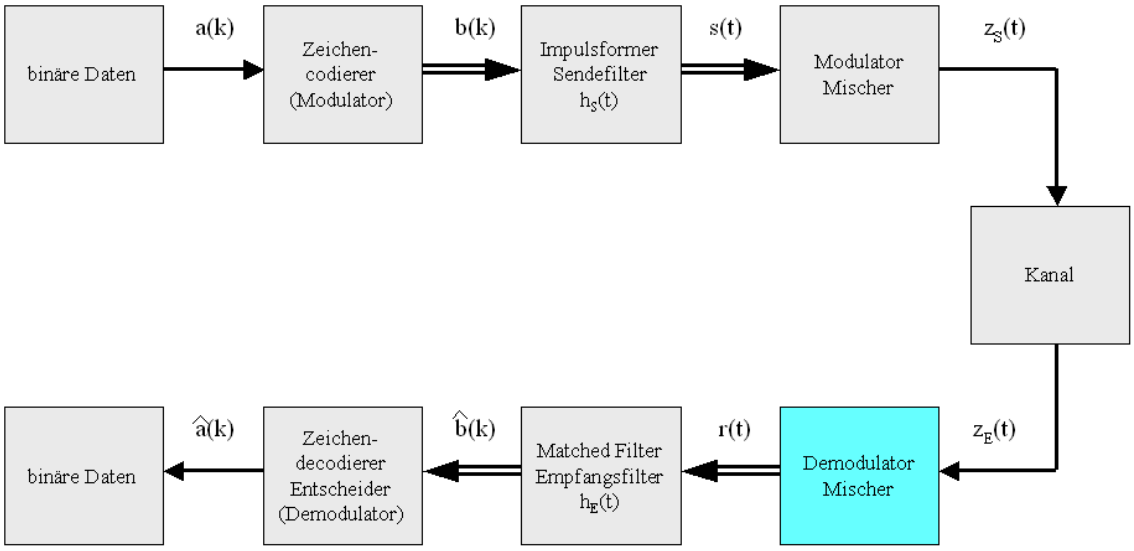
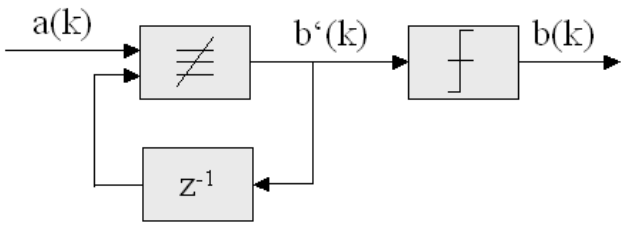


Abbildung 2.1: PLC System

Danach findet eine DBPSK Basisbandkodierung statt; DBPSK ist eine binäre PSK, mit dem Unterschied, dass die Information nicht in der absoluten Phase, sondern in der Phasendifferenz steckt. Daher braucht keine absolute Phaseninformation bei der Demodulation zur Verfügung zu stehen.

Das Sendefilter liefert Impulse mit gewünschter Bandbreite und vermeidet ISI (Inter Symbol Interferenz). Die Quadratur-Mischung mit einer Trägerfrequenz von 200 MHz führt das Signal auf einen höheren Frequenzbereich.



a(k)	b‘(k-1)	b‘(k)	b(k)
0	0	0	-1
0	1	1	1
1	0	1	1
1	1	0	-1

Abbildung 2.2: DBPSK-Modulator

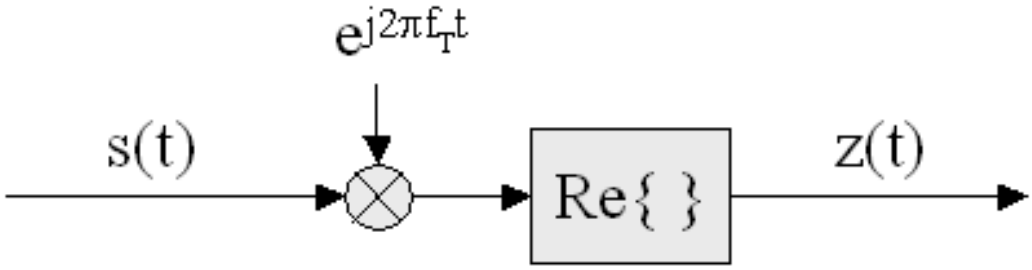


Abbildung 2.3: Quadratur-Mischer

Der Kanal ist ein wichtiger Bauteil zur Simulation des PLC Kommunikationssystems. Man möchte den Kanal im Labor simulieren, weil es zu viel Aufwand beim Testen mit einem realen Kanal inklusive aller Störquellen geben würde. Hierfür wäre der Einbau der zu testenden Transceiver in ein Kfz notwendig. Folglich ist der Einsatz eines Kanalemulators und eines Kanalmodells erforderlich.

Der Empfangsteil des PLC-Modems ist entsprechend dem Sendeteil konstruiert. Das Empfangsfilter ist das zum Sendefilter gehörende Matched-Filter, das mit Hilfe der Korrelation die Sendesymbole zurückgewinnt.

2.2 Kanalmodell

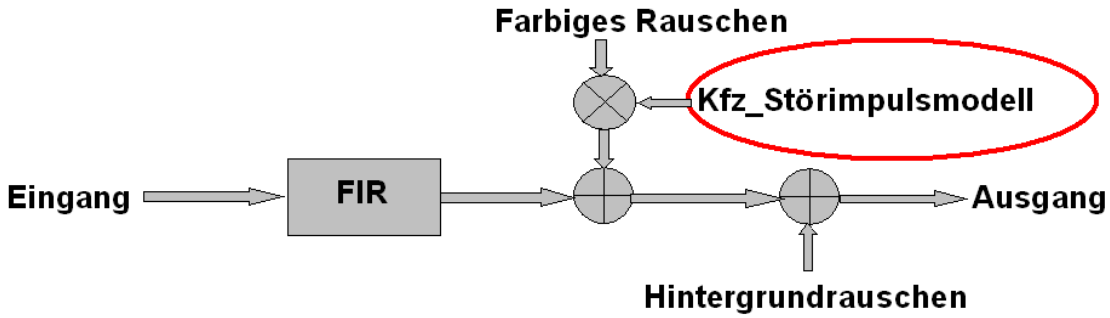


Abbildung 2.4: Kanalmodell für ein PLC System im Kfz

In Abbildung 2.4 ist das Kanalmodell für den PLC Kanal im Kfz dargestellt.

Das PLC Kanalmodell ist ähnlich zu dem AWGN Kanal. Es besteht aus einem Filter mit additivem Rauschen. Das Filter ist üblicherweise ein Tiefpass und liefert die frequenzabhängige Dämpfung des Kanals. Das Hintergrundrauschen ist entweder weißes oder farbiges Rauschen. Das Besondere an diesem Kanal sind die Impulsstörungen der Zündung. Diese sind, wie zuvor bereits erwähnt, Überlagerung eines periodischen und aperiodischen Impulsmodells. Mit der Nachbildung dieser Störimpulse beschäftigt sich diese Studienarbeit.

Der Kanalemulator (ohne das FIR-Kanalfilter) soll in Zukunft auf einem FPGA mit Taktfrequenz 200 MHz (EP1C12Q240C6 Cyclone I) realisiert werden. In der Studienarbeit wurde zur Simulation das FPGA EP1C12Q240C6 Cyclone I mit 100MHz verwendet. Sollte das Programm nicht auf Anhieb mit 200 MHz funktionieren, siehe Abschnitt 6.5.

Kapitel 3

Messdaten

3.1 Impulsdefinition

Ein Impuls definiert sich über zwei Schwellwerte: S_1, S_2 und eine Sperrzeit t_{sperr}. Wenn die gemessene Spannung betragsmäßig den ersten Schwellwert S_1 überschreitet, ist dies der Anfang eines Impulses. Bleibt die Spannung für die Dauer der Sperrzeit unter der zweiten Schwelle S_2 gilt der Impuls als beendet.

Für alle Auswertungen wurden $S_1 = 50\text{mV}$, $S_2 = 20\text{mV}$ und $t_{sperr} = 0{,}2\mu\text{s}$ gesetzt. (Zur besseren Veranschaulichung ist in Abbildung 3.1 ein zweiter Schwellwert in Höhe von 30mV dargestellt.) [Lin01]

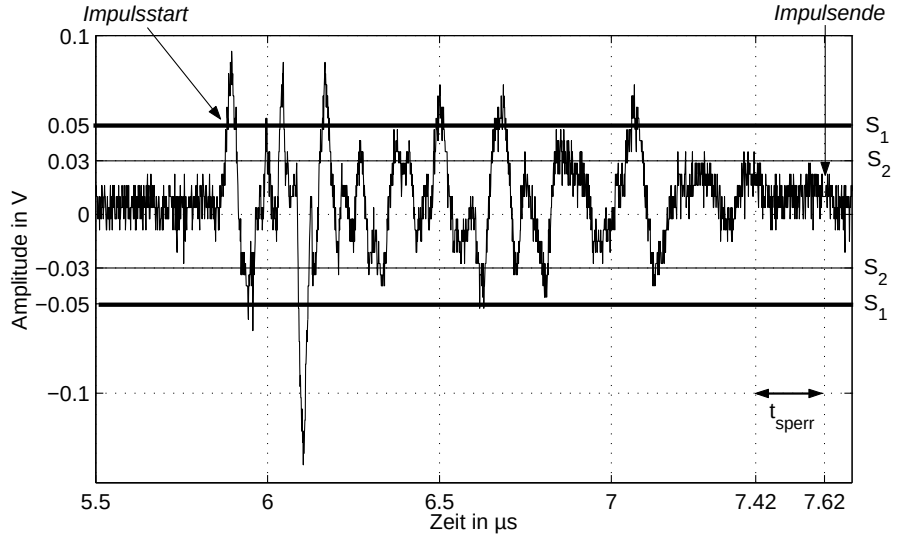


Abbildung 3.1: Messaufnahme eines Impulsstörers (Mercedes Benz C-Klasse)

In Abbildung 3.1, einer Messaufnahme eines Störimpulses, ist die Überlagerung aus einem Störimpuls und Hintergrundrauschen zu erkennen.

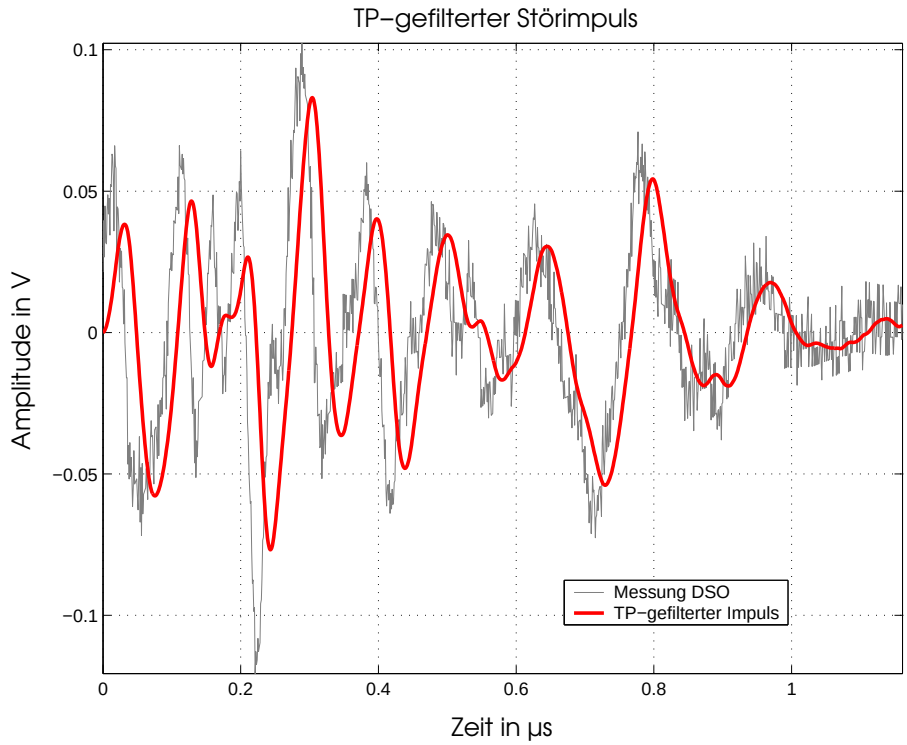


Abbildung 3.2: Impuls nach TP-Filterung

Ein Tiefpass-Filter eliminiert einen großen Teil des Hintergrundrauschens. Das Ziel ist es, die Rechteckeinhüllende des gefilterten Impulses nachzubilden.

3.2 Parameter des verwendeten Störimpulsmodells

Zum Zwecke der Modellierung wird sich der Annahme bedient, dass die Ursache für das Auftreten der Impulsstörer hauptsächlich Motorzündungen sind.

Der Impulsstörer ist durch fünf Parameter charakterisiert: Impulsdauer, Impulsamplitude, Impulsabstand, Impulsmusterdauer, Impulsmusterabstand. Es folgt eine detaillierte Beschreibung dieser Messgrößen:

- Impulsdauer:

  Zeitspanne von Impulsbeginn bis zum Impulsende (nach obiger Definition).

- Impulsamplitude:

  Diejenige Amplitude, die den Betrag der höchsten Spannungsspitze in dem Impuls besitzt. (Negative Werte wurden auch gemessen.)

- Impulsabstand:

  Zeitdifferenz der Startzeiten zweier detektierter Impulse.

- Impulsmusterdauer:

  Der Zeitraum nach dem Zündpunkt, in dem die aperiodischen Störimpulse auftreten.

- Impulsmusterabstand:

  Der Zeitabstand zwischen zwei benachbarten Zündpunkten.

Impulsdauer, Impulsamplitude, Impulsabstand, Impulsmusterdauer sind alles stochastische Größen. Der Impulsmusterabstand ist abhängig von der Drehzahl und damit deterministisch. Für diesen wird ein für das Szenario typischer konstanter Wert angenommen, weswegen auch keine Messdaten des Impulsmusterabstandes vorliegen.

Das Störimpulsmodell ist eine Überlagerung von einem periodischen und aperiodischen Störimpulsmodell: ein Zündpunkt löst eine Reihe von aperiodischen Störimpulsen aus, während das Auftreten der Zündpunkte periodisch ist.

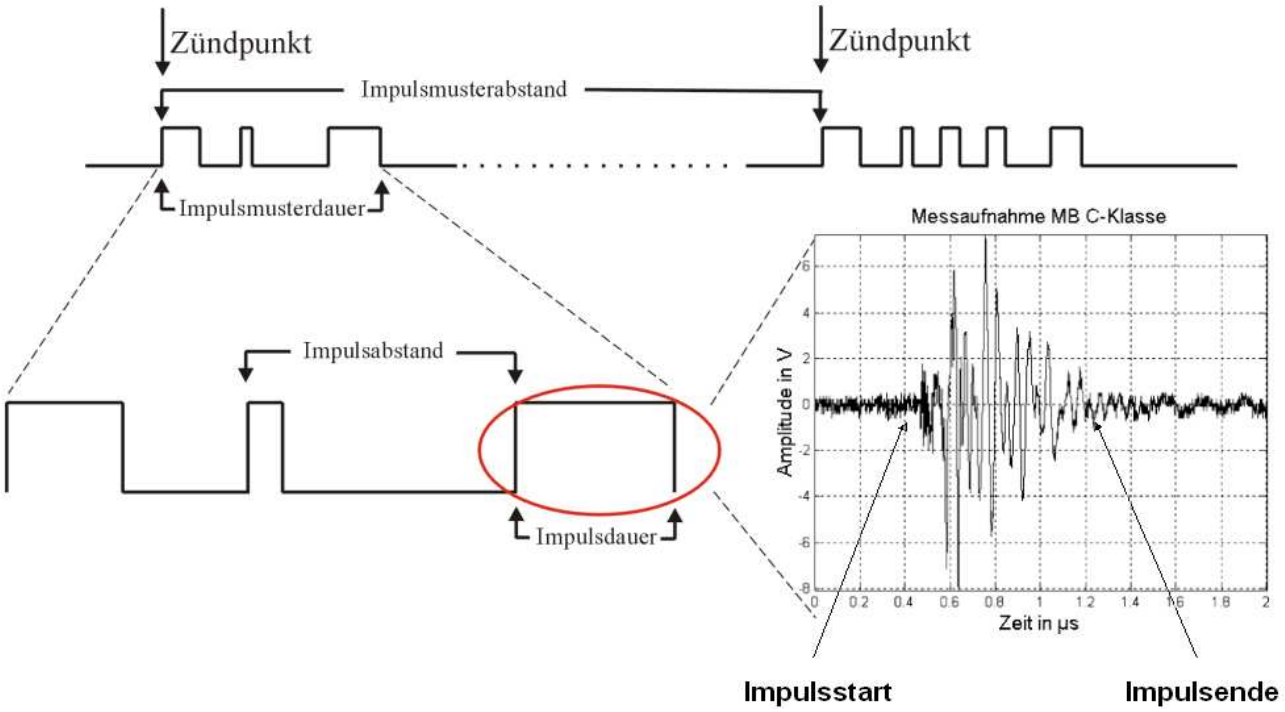


Abbildung 3.3: Die Parameter eines binären Störimpulsmodells

3.3 Vorhandene Messdaten

Für die Messungen wurde ein digitales Speicheroszilloskop des Typs LC547A der Firma LeCroy fest im Kofferraum eines Fahrzeugs des Typs Mercedes C-Klasse, installiert [Lin01]. Bei Messfahrten wurden so die Störungen auf der Versorgungsleitung gemessen. Diese Messungen

stellen einen Worst-Case dar, da es Störungen auf einem Eindrahtkabelbaum und nicht auf Twisted-Pair Leitungen sind.

Da der Frequenzbereich unter 500kHz für eine Datenübertragung uninteressant ist, wurde das Messsignal sowohl im Fahrzeug (analoger Hochpass), als auch mit einem digitalen Cauer-Filter gefiltert (Tiefpass).

In den nachfolgenden Abschnitten werden die erzielten Ergebnisse für die Szenarien Stadt, Landstraße sowie Autobahn vorgestellt. Für die Messung der Impulsdauer und Impulsamplitude wurde eine Abtastrate von 1 GHz und 4 GHz gewählt, was eine Aufzeichnung des Spannungsverlaufs über eine Dauer von 10 μs oder 5 μs erlaubt [Lin01]. Für die Messungen des Impulsabstandes und des Impulsmusterabstandes wurden Abtastfrequenzen von 10 MHz, 25 MHz und 50 MHz verwendet, was einem Messintervall von 5 ms, 2 ms und 1 ms entspricht.

Die Impulsmusterabstände liegen etwa im Bereich von 4 - 6 Millisekunden und müssten eigentlich mit Aufzeichnungen über einen Zeitraum, der zwei Zündungen abdeckt, überprüft werden. Leider stehen solche Messungen nicht zur Verfügung. Es ist daher sinnvoll aus einer typischen Motordrehzahl einen konstanten Impulsmusterabstand zu errechnen.

Beispiel: Für den Landstraßenverkehr ist 3000 $\frac{1}{min}$ eine typische Drehzahl. Da das Messfahrzeug durch einen 6-zylindrigen Ottomotor angetrieben wird, ergibt sich eine Zündfunkenfrequenz von

$$f_D = \frac{3000\frac{1}{min}}{60\frac{s}{min}} \cdot \frac{6}{2} = 150Hz \tag{3.1}$$

woraus sich ein Impulsmusterabstand von

$$T_D = \frac{1}{f_D} = 6.67ms \tag{3.2}$$

errechnen lässt [Lin01].

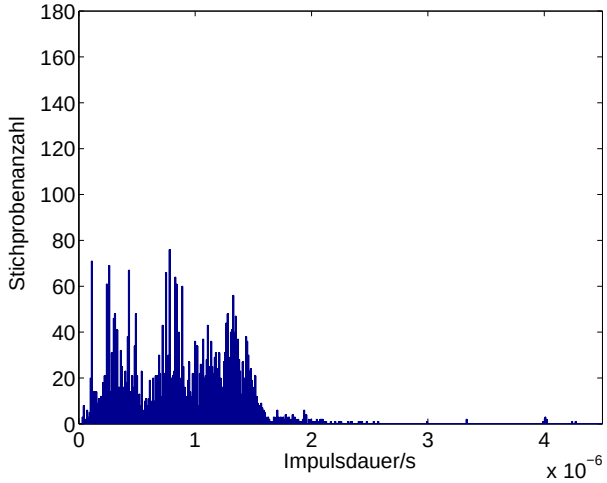
3.4 Messdatenvergleich

Vor der Messwertaufnahme wurden drei charakteristische Fahr-Szenarien festgelegt. Diese sind 'Stadt', 'Landstraße' und 'Autobahn'. Es sind Messdaten zu allen drei Szenarien vorhanden, die im Folgenden bewertet werden.

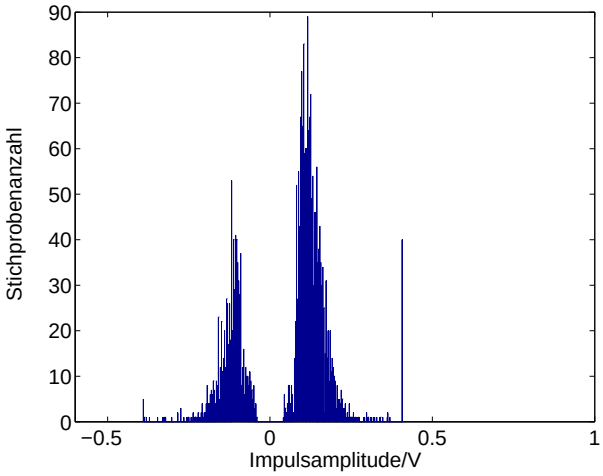
Hierzu sind die Histogramme von Impulsdauer, Impulsamplitude, Impulsabstand und Impulsmusterdauer aus den drei Szenarien dargestellt, um sie anschaulich miteinander zu vergleichen.

3.4.1 Impulsdauer und Impulsamplitude

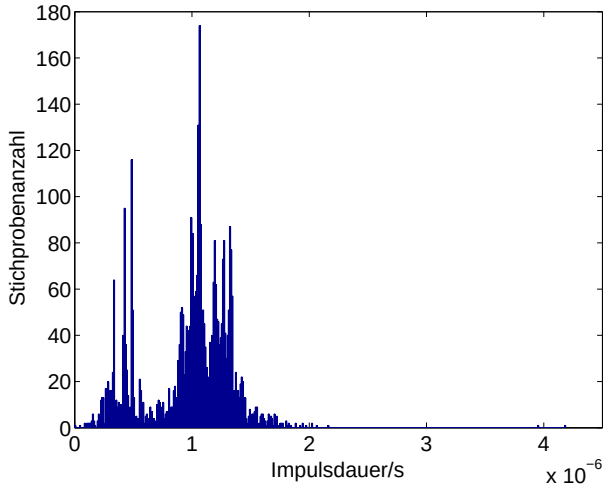
In Abbildung 3.4 sind die Histogramme der Impulsdauer und der Impulsamplitude dargestellt. Eine Korrelation zwischen Impulsamplitude und Impulsdauer ist vorhanden und wurde auch gemessen, wird jedoch aus Gründen der Einfachheit hier noch nicht beachtet.



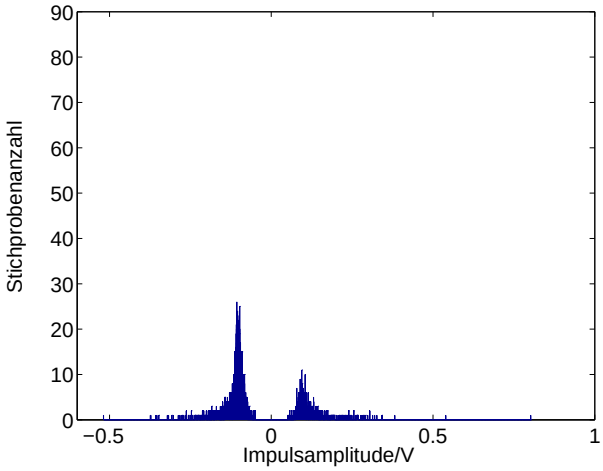
(a) Impulsdauer Stadt (3324 Samples)



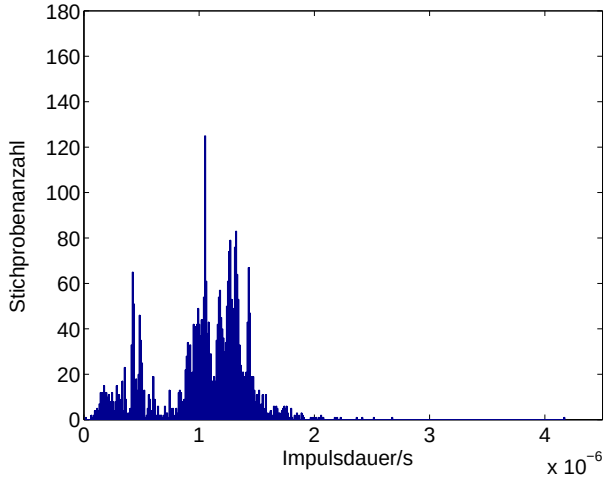
(b) Impulsamplitude Stadt (3324 Samples)



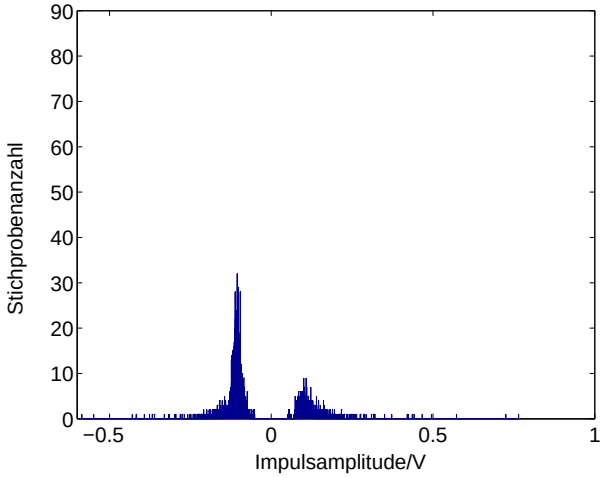
(c) Impulsdauer Landstraße (3870 Samples)



(d) Impulsamplitude Landstraße (3870 Samples)



(e) Impulsdauer Autobahn (3386 Samples)



(f) Impulsamplitude Autobahn (3386 Samples)

Abbildung 3.4: Histogramme der Messdaten

Es ist erkennbar, dass die Impulsdauer und die Impulsamplitude bei allen drei Szenarien in etwa der gleichen Größenordnung liegen.

Die Histogramme der Zeitdauern haben alle eine Klassenbreite von 10ns. Dies ist die Taktzeit des FPGAs und somit die kleinste Zeitdauer, die mit dem FPGA noch nachgebildet werden kann. Die Histogramme für die Impulsamplituden besitzen alle die Klassenbreite von 0.122mV. Die Referenzspannung des DA-Wandlers ist so gewählt, dass eine Quantisierungsstufe 0.122mV breit ist, was also ebenfalls der maximalen Auflösung der Impulsamplitude entspricht.

3.4.2 Impulsabstand und Impulsmusterdauer

Bei den Messwerten des Impulsabstandes und der Impulsmusterdauer sind große Unterschiede erkennbar, wie an den Histogrammen in Abbildung 3.5 deutlich wird.

Die größten Unterschiede zeigen sich bei den Merkmalen Impulsmusterdauer und Impulsabstand zwischen den Szenarien. Bei 'Stadt' und 'Landstraße' (Abbildung 3.5(a) und Abbildung 3.5(c)) sind die Impulsabstände sehr groß, während bei 'Autobahn' (Abbildung 3.5(e)) mehr als 10 mal so kleine Impulsabstände auftreten. Die Impulsmusterdauern liegen bei allen drei Szenarien etwa in der gleichen Größenordnung. Hieraus ist also die wesentlich größere Impulshäufigkeit im Falle der Autobahnmessung hervorzuheben.

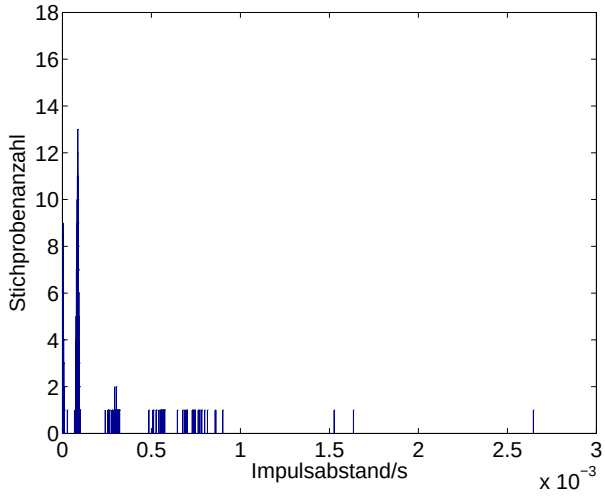
3.4.3 Impulsmusterabstand

Mittels der zuvor vorgestellten Methode ist es möglich aus der Motordrehzahl den Impulsmusterabstand zu berechnen [Lin01].

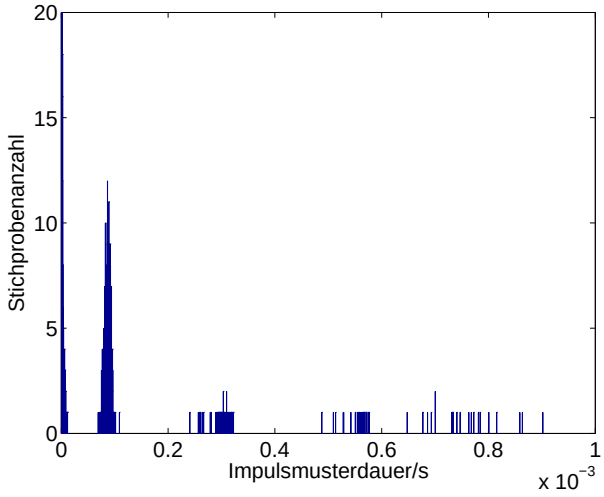
- Landstraße: Aus einer typischen Drehzahl von 3000 $\frac{1}{min}$, folgt ein Impulsmusterabstand von 6,67 ms.
- Autobahn: Eine höchste Drehzahl von 5000 $\frac{1}{min}$ für das Szenario Autobahn ergibt einen Impulsmusterabstand von 4 ms.
- Stadt: Das Szenario Stadt hat Drehzahlen von 3000 $\frac{1}{min}$ bis zu 5000 $\frac{1}{min}$. Was dazu führt, dass ein Impulsmusterabstand von 4 ms bis zu 6,67 ms möglich ist.

Das Autobahn Szenario besitzt wegen der konstant hohen Drehzahl den kleinsten Impulsmusterabstand. Bei diesem Szenario treten Störimpulse häufig auf, weil der Impulsmusterabstand klein ist. Das ist sehr günstig für die Simulation, da die Störimpulse gut beobachtbar sind. Dieses Szenario ist zudem sehr kritisch bezüglich der Datenübertragung, da es den Worst-Case mit der größten Störimpulsanzahl pro Zeit darstellt.

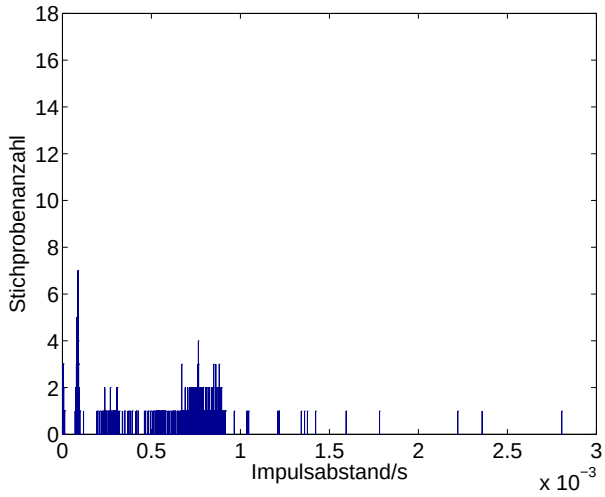
Alle Störungen wurden auf einer Eindrahtleitung gemessen, was wieder einem Worst-Case entspricht. Bei Twisted-Pair Leitungen sind weniger Störungen zu erwarten, weil diese, wie schon gesagt, eine größere Resistenz gegen magnetische Einflüsse haben.



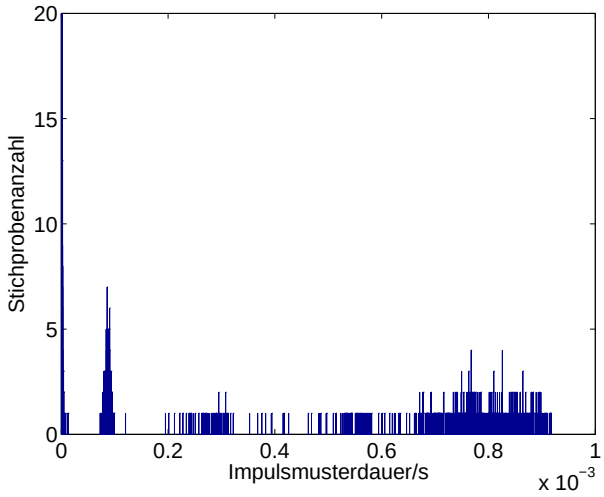
(a) Impulsabstand Stadt (1163 Samples)



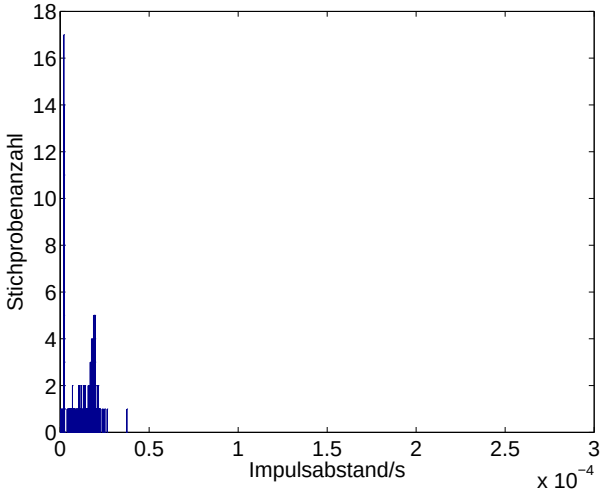
(b) Impulsmusterdauer Stadt (1806 Samples)



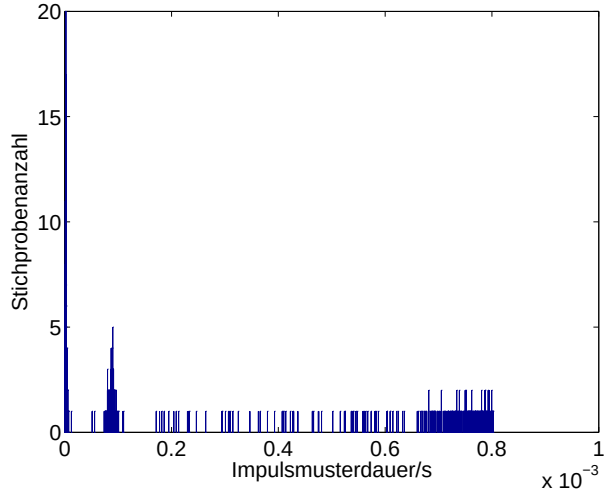
(c) Impulsabstand Landstraße (1189 Samples)



(d) Impulsmusterdauer Landstraße (1866 Samples)



(e) Impulsabstand Autobahn (917 Samples)



(f) Impulsmusterdauer Autobahn (1170 Samples)

Abbildung 3.5: Histogramme der Messdaten

In der weiteren Simulation werden daher die Autobahnmessdaten zur Nachbildung ausgewählt. Im Prinzip ist es aber durch die vorgestellten Algorithmen durchaus möglich beliebige Messdaten zu verwenden.

3.5 Dichtefunktion der Autobahnmessdaten

Das Histogramm ist ein einfacher Dichtefunktionsschätzer. Durch die Histogramme der Autobahnmessdaten (siehe vorheriger Abschnitt), ist zu erkennen, dass die Dichtefunktionen von der Impulsdauer, Impulsamplitude, Impulsabstand, Impulsmusterdauer keiner Standard-Dichte ähnlich sind. (Normalverteilung, Gleichverteilung, Exponentialverteilung etc...)

Die Frage ist nun, wie so verteilte Zufallszahlen erzeugt werden können. Mit diesen ist es möglich durch Abzählen von Taktzyklen Verweildauern zu erzeugen.

Kapitel 4

Vorstellung des Verfahrens

Struktur des Störsignals

Zusammenfassend kann festgehalten werden: Aus den Messdaten des Szenarios Autobahn wurden die Daten der Impulsdauer mit Impulsamplitude, Impulsabstand und Impulsmusterdauer extrahiert. Dabei gelten für die Messdaten folgende Annahmen:

- Es gibt keine Korrelation zwischen der Impulsmusterdauer und den darinliegenden aperiodischen Störimpulsen. D.h. Die Impulsabstände und Impulsdauern charakterisieren das aperiodische Störsignal unabhängig von den Impulsmusterdauern.
- Die Impulsmuster werden durch Zündfunken verursacht. Die Zündpunkte bzw. Impulsmusterabstände wurden nicht gemessen. Die Annahme ist, dass sich am Anfang des Impulsmusters immer ein Zündfunke und Impulsstörer befinden.

Die Erzeugung des Kfz Störimpulssignals verläuft in folgenden Schritten:

1. Erzeugung eines binären Signals für die aperiodischen Störimpulse (die Impulse in einem Impulsmuster), wobei die Impulsdauer und der Impulsabstand entsprechend den Messdaten verteilt sind.
2. Multiplikation der binären aperiodischen Störimpulse mit zugehörigen zufälligen Amplituden.
3. Generation eines binären Signals für das Störimpulsmuster, wobei die Impulsmusterdauer so verteilt sein soll wie in den Messdaten, und der Impulsmusterabstand drehzahlabhängig ist.
4. Multiplikation der Signale 2) und 3) um das gesamte Kfz Störimpulssignal zu erhalten. Anschaulich 'schaltet' Signal 3) die aperiodischen Impulse an und aus.

Randbedingungen

Dabei gelten folgende Randbedingungen:

- Die Impulsamplitude ist korreliert mit der Impulsdauer.
- Signal 1) ist nicht unabhängig von Signal 3): Ein Impulsmuster muss immer mit einem Einzelimpuls beginnen.
- Das Impulsmuster muss auch mit einem Einzelimpuls enden. Da aber bei der Messung keine Korrelation zwischen Impulsmuster und der darin liegenden Impulsdauer und Impulsabstand bestimmt wurde, ist dies unmöglich zu realisieren und gleichzeitig die Impulsmusterdauer mit der richtigen Verteilung zu erzeugen.

Möglichkeiten Zufallszahlen zu Erzeugen

Anhand der Messdaten und der dazugehörigen geschätzten Dichtefunktion können die Zufallszahlen, die zur Erzeugung des gesamten Störimpulssignals benötigt werden, mit verschiedenen Methoden erzeugt werden. Grundsätzlich verläuft eine Methode immer in zwei Schritten:

Messdaten → Dichte → Zufallszahlen.

Der erste Schritt bestimmt aus den Messdaten eine Dichtefunktion zu der im zweiten Schritt Zufallszahlen erzeugt werden. Die Dichtefunktion kann auf verschiedene Arten bestimmt oder gewählt werden. (direkt oder indirekt)

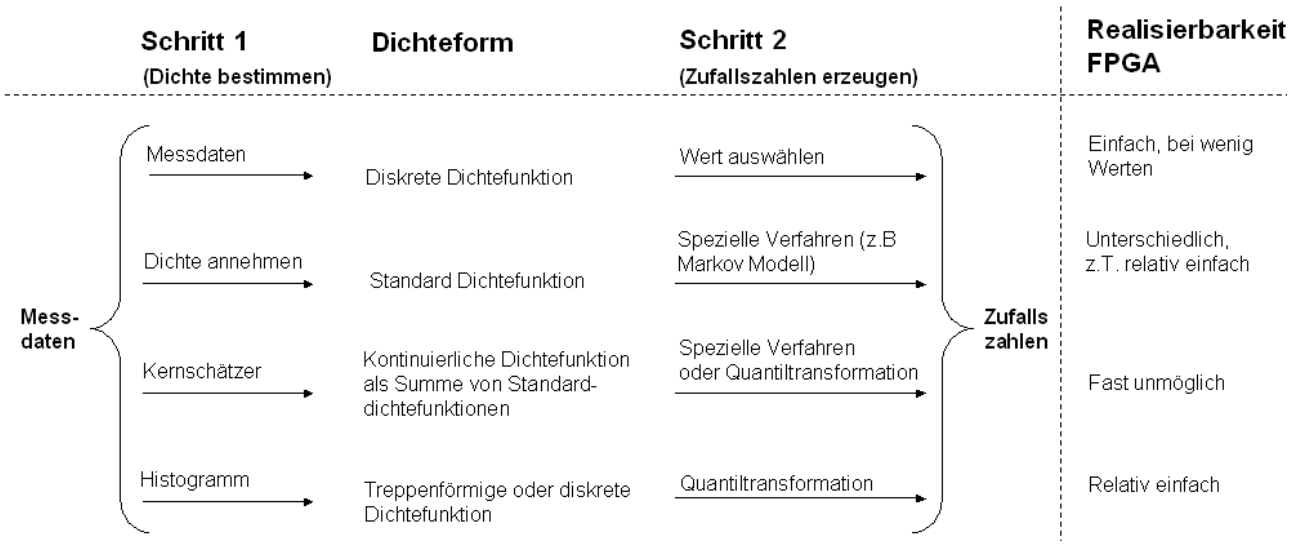


Abbildung 4.1: Verschiedene Methoden zur Erzeugung von Zufallszahlen mit gleicher Verteilung wie vorgegebene Messdaten

Beschreibung der Verfahren

1. **Messdaten speichern:** Entspricht der Vorgabe einer 'diskreten' Dichtefunktion. Die Möglichen Werte die als Zufallszahlen zu einer Dichte auftreten können, werden im folgenden als 'Ereignisse' bezeichnet. Alle Messdaten sind somit im vorliegenden Fall Ereignisse. Die Wahrscheinlichkeit für jedes Ereignis, hängt davon ab, wie oft ein bestimmter Messwert in der Messung auftritt.

   Zufallszahl-Erzeugung: Die Messdaten werden direkt in ein VHDL Programm eingegeben, wobei dieses zufällig einen Wert auswählt. Es bildet die Messdaten genau nach, benötigt nur eine sehr einfache Logik, dafür aber sehr viel Speicherplatz bzw. Logik-Zellen.

2. **Dichte Anpassung:** Man nimmt an, dass die Messdaten eine Standard-Verteilung haben, und passt diese durch Veränderung von verschiedenen Parametern der Standard-Verteilung an die Messdaten an. Standard-Verteilungen sind z.B. die Normalverteilung, Weibullverteilung, Gleichverteilung etc... Verschiedene Anpassungstests werden zur Verifikation verwendet, wie z.B. der χ^2-Anpassungstest.

   Zufallszahl-Erzeugung: (nur einige Beispiele)

   - *Gleichverteilte Zufallszahlen* mit einem linear rückgekoppelten Schieberegister. Dies ist sehr einfach in einem FPGA zu realisieren. Wichtig hierbei ist zu wissen, dass die Zahlen nur Pseudozufallszahlen sind. Die Bitbreite N des Schieberegisters muss groß genug gewählt sein, damit sich die Zufallszahlen in dem relevanten Zeitraum nicht wiederholen. Die Periodenlänge beträgt bei richtiger Wahl der Feed-backs $2^N - 1$.
   - *Exponentialverteilte Zufallszahlen* mit einem Markov-Modell. Dieses Modell ist ebenfalls relativ einfach in einem FPGA realisierbar. Das Markov-Modell erzeugt keine reinen Zufallszahlen, sondern Zustandsverweildauern, was für diesen Anwendungszweck keinen Unterschied machen würde. (siehe Abschnitt 4.1.1)
   - *Binomialverteilte Zufallszahlen* kann man als Summe von gleichverteilten Zufallszahlen erzeugen.

3. **Kernschätzer:** Ein Kernschätzer bildet die wahre kontinuierliche Dichtefunktion sehr genau nach. Eine mit einem Kernschätzer geschätzte Dichtefunktion ist eine Summe von verschiedenen Standard Dichtefunktionen (Kernen).

   Zufallszahl-Erzeugung: Es gibt spezielle Methoden um Zufallszahlen zu den Kern- Dichtefunktionen zu erzeugen. Diese sind leider sehr schwierig in einem FPGA zu realisieren. Es ist auch möglich mit der von dem Kernschätzer geschätzten kontinuierlichen Dichtefunktion eine Quantiltransformation durchzuführen, um Zufallszahlen zu erzeugen (siehe Abschnitt 4.3.1.2). Aber in einem FPGA ist die kontinuierliche Quantiltransformation ebenfalls sehr schwer zu programmieren, und benötigt in der Regel verschiedene Algorithmen für verschiedene Dichtefunktionen (siehe: Abschnitt 4.2.1).

4. **Histogramm:** Das Histogramm ist ein einfacher Dichteschätzer. Mit unterschiedlicher Klassenanzahl schätzt das Histogramm die Dichtefunktion unterschiedlich genau. Die

Schätzung ist treppenförmig also kontinuierlich. Wenn nur der Mittelwert eines Intervalls als mögliche Zufallszahl zugelassen ist, entspricht dies einer diskreten Dichteschätzung.

Zufallszahl-Erzeugung: In einem FPGA kann eine Quantiltransformation für eine diskrete Dichte sehr einfach programmiert werden. Die Quantiltransformation für eine stufenförmige Dichtefunktion ist ebenfalls einfach in einem FPGA realisierbar. Dazu erweitert man das Programm für die Quantiltransformation für eine diskrete Dichte, indem man zu den erzeugten Zufallszahlen gleichverteilte Zufallszahlen hinzu addiert (siehe Abschnitt 4.3).

Da die Dichtefunktionen von der Impulsdauer, Impulsmusterdauer, und dem Impulsabstand keine Standard-Dichtefunktionen sind (siehe: Abschnitt 3.4), ist es besonders ungünstig, Zufallszahlen mit Verfahren 2) zu erzeugen. Die Datenmenge ist ebenfalls zu groß (1000-3000 Samples), um Verfahren 1) anzuwenden. Wenn man eine kontinuierliche Dichteschätzung mit einem Kernschätzer macht und die Quantiltransformation bildet oder ein spezielles Verfahren benutzt, wird die Bearbeitung ziemlich aufwendig und es ist fast unmöglich dies in einem FPGA zu realisieren (siehe 4.2.1).

Das Verfahren, das in dieser Arbeit benutzt wird, stammt aus dem vierten Verfahren. Um eine Quantiltransformation auszuführen, muss man F^{-1}, die Umkehrfunktion der Verteilungsfunktion F berechnen. Um F^{-1} einfach auf einem FPGA berechnen zu können, muss F^{-1} eine Funktion sein, die in einer Tabelle gespeichert werden kann. Es liegt nahe eine Bearbeitung und Quantisierung der Messdaten mittels Histogramm vorzunehmen. Der Vorteil liegt darin, dass für alle Fälle keine komplizierten Rechnungen gebraucht werden, eine relativ kleine Datenmenge zu speichern ist, und trotzdem beliebig verteile Zufallszahlen erzeugt werden können. Deswegen ist dieses hier besonders geeignet, Zufallszahlen mit den vorher gezeigten Dichtefunktionen von Impulsdauer/amplitude, Impulsabstand und Impulsmusterdauer in einem FPGA zu erzeugen. (siehe Abschnitt 4.3)

Jedoch benötigen die Dichtefunktionen eine sehr geschickte Quantisierung (in X und Y-Richtung), damit die Genauigkeit nicht verloren geht.

4.1 Spezielle Verfahren für die Erzeugung von Zufallszahlen

4.1.1 Markov-Modell

Das Markov-Modell ist durch eine Wahrscheinlichkeitsmatrix definiert, welche die Wahrscheinlichkeit für ein Verbleiben im jeweiligen Zustand sowie für den Übergang in die übrigen Zustände enthält. Es werden Zustände im Zeitraster t_a anhand dem Markov-Modell generiert, wobei Pseudozufallszahlen mit den Wahrscheinlichkeiten in der Matrix verglichen werden, um zu bestimmen, in welchen Zustand das Modell wechseln wird. Die Gruppierung von gestörten und ungestörten Zuständen, ist verantwortlich für das Ein- und Ausschalten der Störungen.

Die Pseudozufallszahlen werden durch ein Schieberegister erzeugt. Hierbei muss beachtet werden, dass die Zahlen eine genügend hohe Auflösung haben, um sehr kleine Wahrscheinlichkeiten nachbilden zu können [Göt04].

Die Markov Theorie ist eine wichtige mathematische Methode zur Beschreibung der Dynamik stochastischer Prozesse.

Verteilung der Verweildauer

Alle Verteilungsfunktionen, die das Verhalten der zum System gehörenden Elemente beschreiben, sind Exponentialverteilungen. Die Wahrscheinlichkeitsdichte der Verweilzeit, nach deren Ablauf das System den gerade betrachteten Zustand verlässt, ist eine Exponentialverteilung [Kie97].

Deswegen ist das Markov-Modell auf Exponentialverteilungen, genauer gesagt, auf eine Überlagerung mehrerer Exponentialverteilungen beschränkt.

Bei genauerem Eingehen auf das geforderte Störimpulsmodell, ist deutlich zu erkennen, dass das Markov-Modell dafür nicht geeignet ist.

Das Markov-Modell kann nicht

- die Impulsdauer, den Impulsabstand bzw. die Impulsmusterdauer mit den vorher gezeigten Dichtefunktionen erzeugen.
- die Korrelation von Impulsdauer und Impulsamplitude realisieren.
- die Impulsmuster mit einem konstanten Impulsmusterabstand einfach erzeugen.
- die Korrelation zwischen Impulsmuster und den aperiodischen Impulsen einfach realisieren. (d.h. jedes Impulsmuster beginnt mit einem Einzelimpuls)

4.1.2 Schieberegister

Gleichverteilte Zufallszahlen können mit einem linear rückgekoppelten Schieberegister erzeugt werden. Die dadurch erzeugten Zahlen sind Pseudozufallszahlen. Eine maximale Periodenlänge der Zufallszahlen kann erreicht werden, indem bestimmte Positionen der rückgekoppelten Pins ausgewählt werden (M-Sequenz). Die maximale Periodenlänge ist $2^n - 1$, wobei '0' nicht erzeugt wird (n: Länge des Schieberegisters). Durch Erhöhung der Länge des Schieberegisters wird eine größere Periodenlänge erreicht. Dies ist insbesondere wichtig wenn nur Zufallszahlen benötigt werden, die weniger als n Bit haben. Dazu werden nur die benötigten unteren Bits des Schieberegisters genommen und die oberen verworfen. Bei diesen Zufallszahlen kann '0' wieder auftreten, sie tritt nur dann nicht auf, wenn eben alle Bits des Schieberegisters die Zufallszahl bilden.

4.2 Vorherige Methode der Dichtenachbildung

4.2.1 Kern Schätzer

Das Ziel bei der Kern-Schätzung besteht darin, eine stetige - keine stufige - Funktion zu erhalten. Diese Methode nähert die tatsächliche Dichte der Stichproben (x_1, x_2..., x_n) durch Funktionen der Form

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K(\frac{x - x_i}{h}) \tag{4.1}$$

an.

K ist eine Wahrscheinlichkeitsdichtefunktion, wobei die Bedingungen

1. $K(x) \geq 0$
2. $\int_{-\infty}^{+\infty} K(x)dx = 1$

erfüllt sein müssen. K wird als Kern bezeichnet. An der Form ist ersichtlich, dass dieses Verfahren versucht, die Dichtefunktion als eine Summe von vielen Standard-Dichtefunktionen zu beschreiben.

Eine Kern-Dichteschätzung kann mit hinreichender Genauigkeit an den Datensatz angepasst werden. Aber um Zufallszahlen zu erzeugen, werden normalverteilte Zufallszahlen und sehr komplizierte Rechnungen, wie Multiplizieren und Wurzelziehen benötigt [Lin01]. Daher ist der Kern-Schätzer schwierig auf einer FPGA-Hardware zu realisieren.

4.3 Verteilungstabelle

Die Verfahren, die bis jetzt vorgestellt wurden, können nicht mit kleinem Aufwand beliebige Verteilungen erzeugen und in einem FPGA realisiert werden.

Das Verteilungstabelle Verfahren kann dieses Problem mit hinreichender Genauigkeit lösen. Hierzu ist der grundsätzliche Ablauf der Erzeugung von Zufallszahlen in Abbildung 4.2 anschaulich dargestellt.

4.3.1 Mathematische Grundlagen

4.3.1.1 Dichtefunktion und Verteilungsfunktion

Dichtefunktion und Verteilungsfunktion sind zwei wichtige Begriffe in dieser Arbeit. Hier werden sie nocheinmal kurz wiederholt.

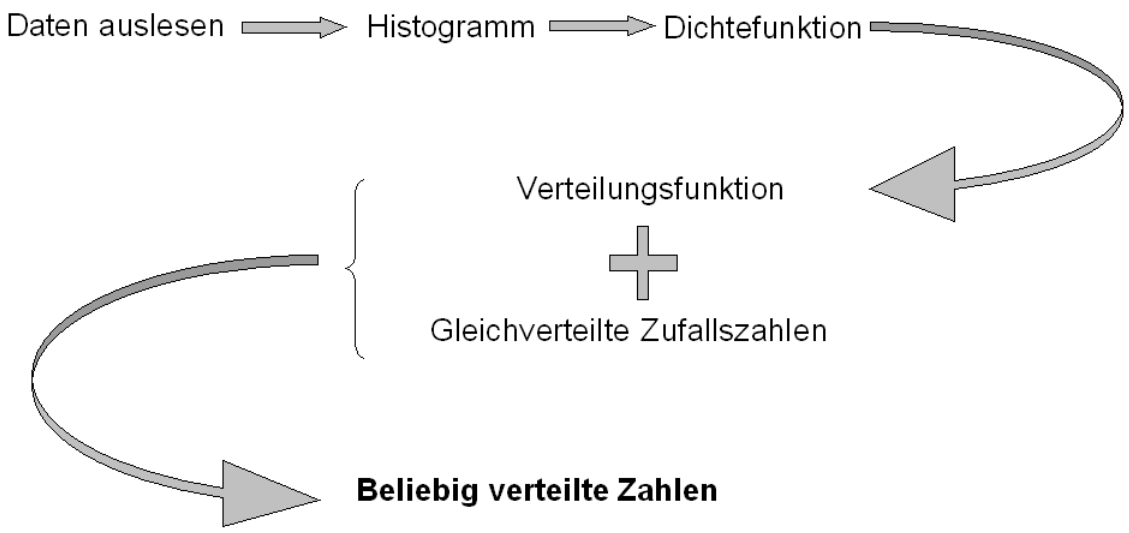


Abbildung 4.2: Ablauf der Erzeugung von Zufallszahlen mit einer Verteilungstabelle

Dichtefunktion

Die Dichtefunktion zeigt, in welchen Teilen des Definitionsbereiches einer Zufallsvariablen wie häufig Werte auftreten.

Die Wahrscheinlichkeit, dass der Wert der Zufallsvariablen innerhalb eines bestimmten Intervalls liegt, entspricht der Fläche unterhalb der Dichtefunktion zwischen den Grenzen des Intervalls. Die Gesamtfläche unterhalb einer Dichtefunktion hat somit immer den Wert 1.

- Diskrete Dichtefunktion:

  Eine Zufallsvariable X heißt diskret (diskret verteilt), wenn sie nur (höchstens) abzählbar viele Werte mit Wahrscheinlichkeit größer Null annimmt. Die Funktion

$$f(x) := \begin{cases} P(x = x_i) & \text{falls } x = x_i, \text{ i= 1, 2, 3...} \\ 0 & \text{sonst} \end{cases} \tag{4.2}$$

  heißt diskrete Dichtefunktion der diskreten Zufallsvariablen X mit den Werten x_1, x_2, x_3... . Für die Verteilungsfunktion einer diskreten Zufallsvariablen gilt:

  $F(x) = \sum_{x_i \leq x} f(x_i)$

- Stetige Dichtefunktion:

  Eine Zufallsvariable X heißt stetig (stetig verteilt), wenn

  $F(x) = \int_{-\infty}^{x} f(z)dz$

für eine sogenannte stetige Dichtefunktion f gilt.

- Stufenförmig Dichtefunktion:

  Eine mit einem Histogramm geschätzte Dichtefunktion ist stufenförmig, worauf in Kapitel 5 genauer eingegangen wird.

Verteilungsfunktion

Die Verteilungsfunktion ergibt sich durch Integration der Dichtefunktion:

$$F(x) = \int_{-\infty}^{x} f(x)dx \tag{4.3}$$

Also gilt: $P(a \leq X \leq b) = P(x \leq b) - P(x \leq a) = F(b) - F(a)$

4.3.1.2 Quantiltransformation

Bezeichnet F die Verteilungsfunktion einer Zufallsvariablen X, dann wird für $0 < \text{u} < 1$ durch

$$F^{-1}(u) := \inf\{x \in R \mid F(x) \geq u\} \tag{4.4}$$

die *Quantilfunktion* definiert.

'inf' ist die Abkürzung des Infimums. Es bezeichnet die größte untere Schranke einer Teilmenge von R. Wenn die Teilmenge endlich ist, ist das Infimum gleich dem kleinsten Element der Menge.

Anschaulich sucht man hier das kleinste Ereignis x, dessen Y-Wert der Verteilungsfunktion größer oder gleich einer auf (0..1) verteilten Zahl u ist.

Zunächst sei an dieser Stelle darauf hingewiesen, dass die Quantilfunktion genau der Umkehrfunktion von $F(x)$ entspricht, falls $F(x)$ streng monoton wachsend ist. Sie ist aber auch dann wohldefiniert, wenn $F(x)$ nicht überall streng monoton wächst. D.h. die Quantilfunktion ist eine verallgemeinerte Inverse von $F(x)$, also auch wenn die Verteilungsfunktion $F(x)$ diskret (impulsförmig) oder treppenförmig ist.

Satz 4.5 (Quantiltransformation)

Ist nun U eine auf (0..1) gleichverteilte Zufallsvariable und F(x) die Verteilungsfunktion zu einer beliebigen Zufallsvariablen X, dann gilt:

$F^{-1}(U) \sim X$

Man kann also durch Bildung der Umkehrfunktion (Quantilfunktion), aus gleichverteilten Zufallszahlen beliebig verteilte Zufallszahlen erzeugen. Hierzu muss die Dichte $f(x)$ der Zahlen, die erzeugt werden sollen, bekannt sein. Daraus lässt sich die Verteilungsfunktion $F(x)$ und deren verallgemeinerte Umkehrfunktion $F^{-1}(x)$ berechnen. Dies ist nicht immer sehr einfach, da beispielsweise für eine Normalverteilung $F(x)$ schon nicht in einer geschlossenen Form darstellbar ist.

Schematischer Beweis

Bekanntlich ist die Verteilung einer Zufallsvariablen eindeutig durch die zugehörige Verteilungsfunktion definiert. Wir müssen also hier zeigen, dass die Verteilungsfunktion von $F^{-1}(U)$ identisch mit F ist.

Zunächst kann aus der Definition der Quantilfunktion hergeleitet werden, dass für alle $x \in R$ und $0 < u < 1$

$F^{-1}(u) \leq x \Longleftrightarrow u \leq F(x)$

gilt (hier ohne Beweis). Dies benutzt man, um den Satz zu beweisen: Ist x ∈ R gilt:

$P(F^{-1}(U) \leq x) = P(U \leq F(x)) =_{(*)} F(x) = P(X \leq x)$

(*): weil U gleichverteilt auf [0..1], gilt $P(U \leq u) = u$

Damit besitzen X und F^{-1}(U) die gleichen Verteilungsfunktionen. Für weitere mathematische Eigenschaften und Sätze über die Quantiltransformation siehe z.B. [Düm00]

4.3.2 Anschauliche Erklärung

In Abbildung 4.3 ist eine Verteilungsfunktion abgedruckt. Bei jedem Funktionssprung befindet sich ein mögliches Ereignis. In diesem Beispiel gibt es 4 Ereignisse: 1, 3, 4, 7. Die zugehörige Dichtefunktion besteht hier aus 4 Impulsen mit unterschiedlicher Höhe, die die Wahrscheinlichkeit angibt. Die Verteilungsfunktion ist das Integral über die Dichtefunktion, weswegen die Wahrscheinlichkeit jedes Ereignisses durch die Blocklänge (Y-Achse) dargestellt wird. (Ereignis 4 hat die Wahrscheinlichkeit 0,4). Es sollen nun aus gleichverteilten Zufallszahlen, Zufallszahlen mit der vorgegebenen Dichte erzeugt werden. Dazu wird die gleichverteilte Zufallsvariable U mit den Y-Werten der Verteilungsfunktion verglichen. Anschaulich gesagt: Es wird untersucht in welchem Bereich der Verteilungsfunktion U liegt. Damit wird das Ereignis (X-Wert), das zu diesem Bereich gehört, gefunden. Dies ist gleichbedeutend mit der Bildung einer Umkehrfunktion bzw. Quantilfunktion nach der Definition durch Formel 4.4.

Ein Beispiel: Das Schieberegister erzeugt eine gleichverteilte Zufallszahl RN='0,536'. Diese liegt im Intervall (0,3 0,7] (Y-Achse) der Verteilungsfunktion, weswegen das entsprechende Ergebnis '4' ist.

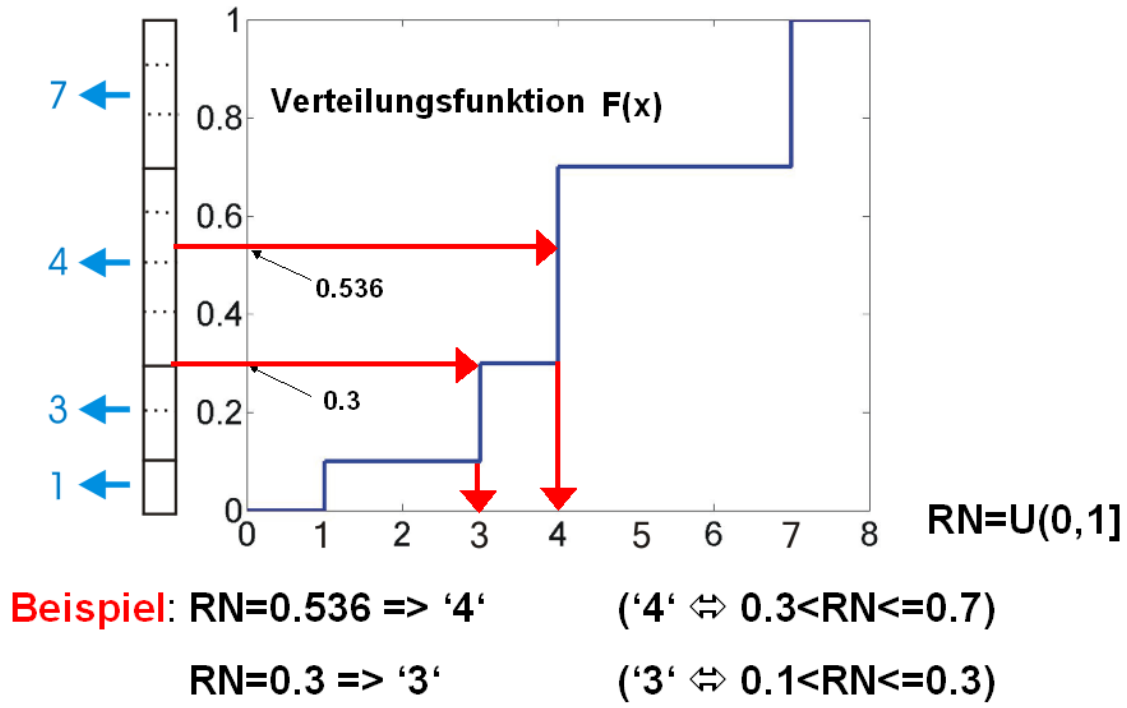


Abbildung 4.3: Anschauliche Erklärung der Verteilungstabelle

Die so erzielten Zahlen sind gerade mit der Dichte verteilt, die vorgegeben wurde. Dies liegt daran, dass die Breite der Y-Bereiche gerade gleich der Wahrscheinlichkeit ist. So liegen die auf (0..1) verteilten Zufallszahlen entsprechend häufig in einem Bereich und es ergeben sich die Ereignisse mit den gewünschten Wahrscheinlichkeiten.

Kapitel 5

Aufbereitung der vorhandenen Daten

Bevor man die vorhandenen Daten bearbeitet, muss zuerst klar sein, was nachgebildet werden soll.

- *Verfahren I:* Falls die Messdaten nachgebildet werden sollen, ist die Güte der Nachbildung direkt mit der Zeitauflösung verbunden. Eine Daten-Nachbildung entspricht einer diskreten Dichtefunktion der erzeugten Zufallszahlen. Diskret heißt, dass die auftretenden Ereignisse nur bestimmte Zufallszahlen sind. Ein Histogramm wird benutzt, um die Messdaten zusammenzufassen, also zu reduzieren. Die Messwerte jeder Klasse werden durch den Mittelwert der Klasse ersetzt. Aus der Dichtefunktion können durch Quantiltransformation Zufallszahlen erzeugt werden, wobei es -wie schon gesagt- auch sehr einfach ist, die Quantiltransformation für eine diskrete Dichte in einem FPGA zu programmieren.
- *Verfahren II:* Falls die kontinuierliche Dichtefunktion des hinterliegenden Prozesses nachgebildet werden soll, ist es wichtig die richtige Dichtefunktion zu schätzen. Dies ist mit einem Histogramm möglich. Die Schätzung ist stufenförmig, weswegen die Ergebnisse nicht nur bestimmte Zufallszahlen beinhalten, sondern auch Zwischenwerte ermöglichen. Die Quantiltransformation für eine stufenförmige Dichte ist ebenfalls in einem FPGA programmierbar, wodurch Zufallszahlen erzeugt werden können. Dies geschieht indem zu den Zahlen, die durch die Quantiltransformation für eine diskrete Dichte erzeugt werden (erstes Verfahren), gleichverteilte Zufallszahlen hinzu addiert werden (siehe Abschnitt 5.2.4)

Verfahren I ist für eine kleine bis mittlere Anzahl an Stichproben geeignet. Hingegen ist Verfahren II auch für eine größere Stichprobenanzahl geeignet (siehe Tabelle 5.3), wobei auch beachtet werden sollte, dass eine genaue Dichteschätzung nur mit einer großen Stichprobenanzahl erfolgen kann, da genügend Information über den Prozess gesammelt werden muss.

Der Realisierungsaufwand für beide Verfahren in VHDL weicht nicht wesentlich voneinander ab.

Beide Verfahren verwenden das Histogramm und die Quantiltransformation (siehe Abschnitt 4.3).

5.1 Verfahren I: Nachbildung der Messwerte

Bei der Nachbildung der Messdaten, ist die einfachste Möglichkeit, alle Daten in das Programm einzugeben und das Programm davon eine Zahl zufällig auswählen und ausgeben zu lassen. Wie schon gesehen, ist dieses Verfahren jedoch nicht für den vorliegenden Fall geeignet.

Deswegen ist es nötig, die Messdaten geschickt zusammenzufassen und zu reduzieren. Eine Zusammenfassung der Messdaten lässt sich mit einem Histogramm realisieren. Zusätzlich können selten auftretende Ereignisse weggefiltert werden, um die Datenmenge zusätzlich zu reduzieren. Ein solcher Schwellwert ist typischerweise eine Zahl von 0 bis 4. 0 bedeutet, dass keine Ereignisse weggelassen werden, eine Schwelle von 1 bedeutet, dass alle Klassen mit nur einer Stichprobe entfernt werden.

Es erfolgt ein kleiner Überblick über das Verfahren für die Datenbearbeitung:

- Laden der Messdaten in Matlab.
- Nutzung des Histogramms, um die Daten zusammenzufassen.(siehe Abschnitt 5.1.1)
- Prüfen der Randbedingungen und bei Verletzen dieser gegebenenfalls Anpassung der Dichte durchführen. (siehe Abschnitt 5.1.1)
- Einsatz eines Schwellwertes, um die Werteanzahl zu reduzieren. (siehe Abschnitt 5.1.2)
- Quantisierung der Y-Werte, und Umrechnung der X-Werte in Taktzyklen. (siehe Abschnitt 5.1.6)

5.1.1 Das Histogramm

Bedeutung und Zweck

Das Histogramm wird hier nicht benutzt, um die richtige Dichtefunktion zu schätzen, sondern die Messdaten zusammenzufassen. Dazu wird der Wertebereich der Stichproben in gleich große Intervalle unterteilt, und die Werte in einem Intervall zu einer Klasse zusammengefasst. Diese Werte werden dann durch den Mittelwert der Klasse ersetzt. Dabei ist die Werteanzahl in einer Klasse ein Maß für die Wahrscheinlichkeit des Mittelwertes.

Randbedingungen für die Klassenanzahl

Die Klassenanzahl des Histogramms ist beschränkt durch

1. Taktzeit des FPGAs.

   Die Taktzeit des FPGAs ist 10ns. Dies ist auch die kleinste Klassenbreite, die für das Histogramm sinnvoll ist. Es dürfen nicht zu viele Klassen ausgewählt werden, weil das FPGA die Genauigkeit der erzeugten Daten beschränkt.

2. Der Speicherplatz des FPGAs.

   Die Datenanzahl, die übrig bleiben soll, ist beschränkt durch den effektiven Speicherplatz des FPGAs. Wenn zu viele Daten in das VHDL Programm eingegeben werden, benötigt dies entweder mehr Logikzellen als das FPGA besitzt, oder es tritt ein Timing-Problem auf.

3. Die kleinste Berechnungsdauer.

   Die Erzeugung der Zufallszahlen auf dem FPGA benötigt eine gewisse (Takt-)Zeit. Diese Zeit setzt sich aus der Vergleichszeit und der Pipelineverzögerung zusammen. Das Programm verwendet einen Komparator mit drei Pipelinestufen, um eine lineare Suche zu realisieren. (siehe Abschnitt 6.2.1.2) D.h. die Anzahl der Ereignisse ist gleich der Vergleichszeit. Deswegen muss die Werteanzahl, die nach der Datenreduktion übrig bleibt, so gewählt werden, dass diese kleiner als der kleinste Abstand zwischen der Erzeugung von zwei Zufallszahlen minus 3 ist.

   Allgemein:

   $$\text{Max. Dauer Zufallszahlerzeugung} < \text{Min. Abstand zw. Erzeugung zweier Zufallszahlen}$$
   $$\Leftrightarrow$$
   $$\text{Werteanzahl 'Impulsdauer'} + 3 < \text{Kleinster Wert 'Impulsabstand'} \tag{5.1}$$
   $$\text{Werteanzahl 'Impulsabstand'} + 3 < \text{Kleinster Wert 'Impulsabstand'} \tag{5.2}$$
   $$\text{Werteanzahl 'Impulsmusterdauer'} + 3 < \text{Kleinster Wert 'Impulsmusterabstand'} \tag{5.3}$$

   Alle Dauern sind in Taktzyklen gemessen.

   Erkennbar ist, dass beide Seiten der Gleichung 5.2 gegenseitig voneinander abhängen. Weil der Impulsmusterabstand konstant ist, ist bei Gleichung 5.3 die rechte Seite eine Konstante.

Daraus folgt die Schlussfolgerung:

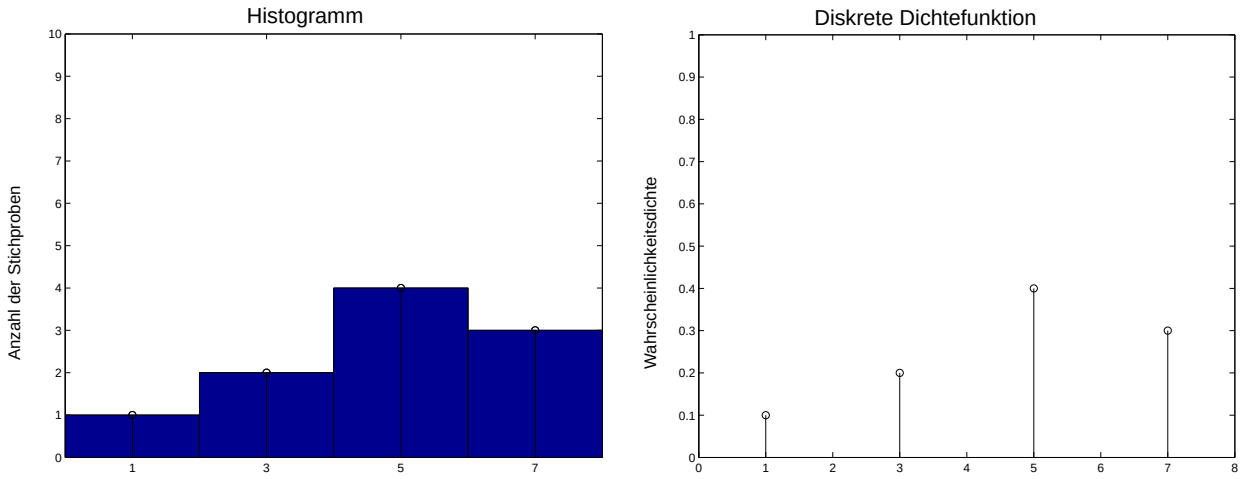
Die Klassenanzahl darf nicht zu groß gewählt werden, weil sonst nicht genügend Speicherplatz vorhanden ist (Randbedingung 2). Die Berechnungsdauer für die Zufallszahlen darf auch nicht zu groß sein, da sonst Randbedingung 3 verletzt wird. Zusätzlich wird die Klassenbreite des Histogramms möglicherweise Randbedingung 1 nicht erfüllen.

Andererseits kann die Klassenanzahl auch nicht zu klein gewählt werden, weil sich sonst eine große zeitliche Abweichung ergibt. Die zeitliche Abweichung kommt dadurch zustande, dass die Messdaten durch den Mittelwert einer Klasse ersetzt werden. Wird die Klassenanzahl zu klein gewählt, ist die Klassenbreite sehr groß, und die zeitliche Abweichung entsprechend groß. Dies ist sehr ungeeignet für die Nachbildung der Messdaten, weil die Genauigkeit der Messdaten nicht verloren gehen darf.

Die Auswahl der Klassenanzahl, wird in Abschnitt 5.1.4 genau diskutiert.

Die diskrete Dichtefunktion

Wenn die Messdaten genau nachgebildet werden sollen, wird angenommen, dass dieser stochastische Prozess nur die Werte, die in der Messfahrt auftreten, enthält. Dies entspricht einer diskreten Dichtefunktion. Das Histogramm ist keine diskrete Dichtefunktion, weswegen ein Wert aus einem Histogrammklassenintervall als diskreter Wert gewählt wird. Hier werden die Messdaten in einer Klasse durch den Mittelwert X, der durch den Matlab Befehl *[N,X]=hist(Datenarray, Klassenanzahl)* berechnet wird, ersetzt.



(a) Histogramm (b) Zugehörige Dichte

Abbildung 5.1: Beispiel zur Dichtebildung aus einem Histogramm

In Abbildung 5.1 ist ein Histogramm dargestellt. Die Mittelwerte sind die Werte der diskreten Dichtefunktion. Die Stichprobenanzahl in einer Klasse dividiert durch die gesamte Stichprobenanzahl entspricht der Wahrscheinlichkeit der diskreten Werte.

Die Verteilungsfunktion eines Histogramms ist dreieckförmig, während die Verteilungsfunktion von einer diskreten Dichtefunktion treppenförmig ist. Dieser Zusammenhang ist auf dem vorherigen Bild dargestellt.

Wenn die Quantiltransformation (→ Umkehrfunktion) mit diesen Verteilungsfunktionen gebildet wird, ergeben sich bei der diskreten Dichtefunktion nur diskrete Ereignisse; bei dem Histogramm ergeben sich zudem auch Zwischenwerte.

Fehler durch das Histogramm

Je größer die Klassenbreite des Histogramms ist, desto größer ist die zeitliche Abweichung.

Der absolute zeitliche Fehler ist auf die halbe Klassenbreite des Histogramms beschränkt (siehe Abbildung 5.3)

Weil später der zeitliche Fehler mit einem Dichtefehler zu einem Gesamtfehler addiert wird, müssen beide Fehler zuerst normiert werden. Es bietet sich an, einen relativen Fehler zu berechnen.

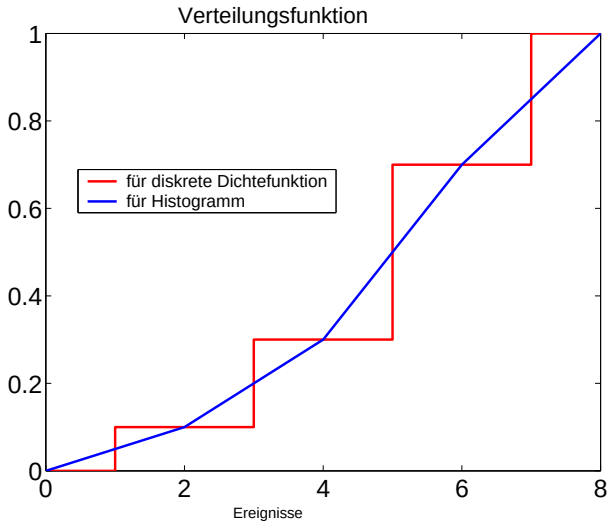


Abbildung 5.2: Verteilungsfunktionen des Histogramms und der diskreten Dichtefunktion aus Abbildung 5.1

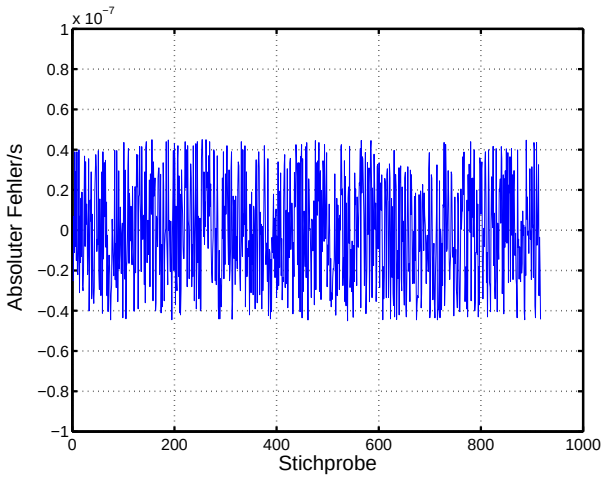


Abbildung 5.3: Absoluter zeitlicher Fehler von Impulsdauer bei 417 Klassen

Berechnung des relativen zeitlichen Fehlers

Es gibt zwei Möglichkeiten, einen relativen zeitlichen Fehler zu berechnen.

- Berechnung durch

$$\sum_i \left(\frac{Daten(i) - Klassenmittelpunkt}{Daten(i)}\right)^2 = \sum_i \left(\frac{Zeitabweichung(i)}{Daten(i)}\right)^2 \qquad (5.4)$$

  Für jedes Messdatenelement wird eine relative zeitliche Abweichung bezüglich sich selbst angegeben.

- Berechnung durch

$$\sum_i \left(\frac{Daten(i) - Klassenmittelpunkt}{max(Daten)}\right)^2 = \sum_i \left(\frac{Zeitabweichung(i)}{max(Daten)}\right)^2 \tag{5.5}$$

  Für jedes Messdatenelement wird die zeitliche Abweichung relativ zum größten Messwert angegeben.

Da versucht wird die Messdaten genau nachzubilden, ist es wichtig, dass der relative Fehler auf das Messdatenelement selbst bezogen wird, um zu überprüfen, wie gut die Messdaten angenähert sind. Wenn der relative Fehler auf das Maximum bezogen wird, werden große Abweichungen bei kleinen Messdaten nicht entsprechend stark gewichtet.

5.1.2 Die Schwelle

Ein Schwellwert wird benutzt, um die durch das Histogramm zusammengefassten Messwerte nochmals zu filtern. Die Ereignisse, die selten auftreten, werden dadurch weggelassen.

Um zu verdeutlichen was für Fehler durch den Schwellwert entstehen können, folgt ein Beispiel anhand der Messdaten der Impulsdauer: Wenn die Dichtefunktion der Impulsdauer betrachtet wird (hier zunächst ohne den Zusammenhang zu den Impulsamplituden), und aus den 3386 Stichproben ein Histogramm mit 10 ns Klassenbreite gebildet wird, ergeben sich 417 Klassen. Das ist die maximale Anzahl an Klassen, die mit dem FPGA sinnvoll ist. Wird eine Schwelle = 3 benutzt, werden die Klassen, die 3 Stichproben oder weniger enthalten, weggelassen. Einen Vergleich der Dichtefunktion ohne Schwelle und der Dichtefunktion mit Schwelle ist in Abbildung 5.4 gezeigt.

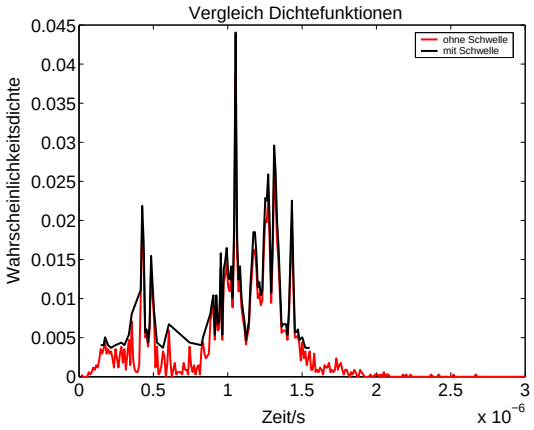


Abbildung 5.4: Dichtefunktionen von Impulsdauer mit und ohne Schwelle

Die Differenz der beiden Dichtefunktionen mit und ohne Schwelle ist eine Fehlerfunktion. (Abbildung 5.5)

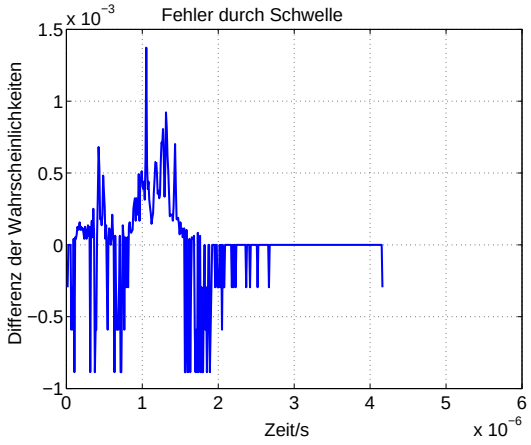


Abbildung 5.5: Fehler durch den Einsatz eines Schwellwertes

Wie in der Fehlerfunktion zu erkennen ist, entsteht sowohl ein positiver als auch ein negativer Fehler. Der negative Fehler entsteht durch das Weglassen der seltenen Ereignisse.

Wenn Ereignisse weggelassen werden, sind weniger Ereignisse als zuvor vorhanden, weswegen die verbleibenden Ereignisse wahrscheinlicher als vorher sind. Daher ergibt sich auch ein positiver Dichtefehler.

Für den Fall, dass keine Ereignisse weggelassen werden, entsteht auch kein Dichtefehler.

In Abbildung 5.6 sind Beispiele des Impulsdauer-Parameters abgedruckt, um zu zeigen, inwiefern verschiedene Schwellwerte Fehler in der Dichtefunktion und Verteilungsfunktion verursachen. Hierzu sind die höchsten Werte der Dichtefunktion markiert, um die Wahrscheinlichkeitserhöhung (positiver Dichtefehler) des zugehörigen Ereignisses durch die Schwelle zu verdeutlichen.

Ein größerer Schwellwert filtert mehr Ereignisse heraus, was sowohl den Offset der Dichtefunktion als auch die Abweichung der Verteilungsfunktion erhöht.

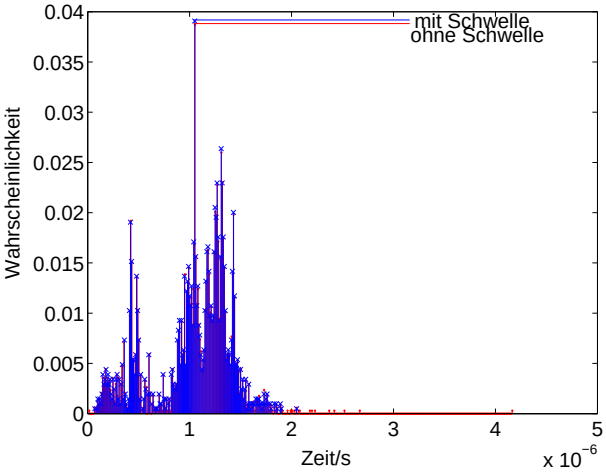
Es stellt sich die Frage nach der Wahl der Kriterien, um den Schwellwert zu bestimmen.

5.1.3 Reduktion der Ereignisse zwecks Berechnungsdauer

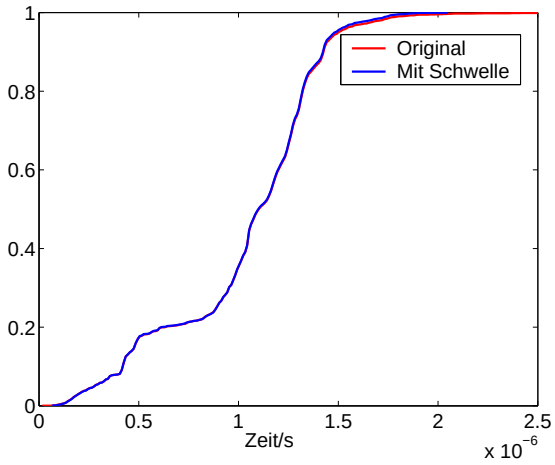
Die Erzeugung von Zufallszahlen benötigt eine konstante Zeit. Wie schon gesagt, ist diese Zeit abhängig von der Vergleichszeit und der Pipelineverzögerung. Hier ist die Pipeline-Stufenanzahl zu 3 gewählt, und die Vergleichszeit in Taktzyklen ist gleich der Ereignisanzahl der Dichtefunktion.

- Aperiodische Störimpulse:

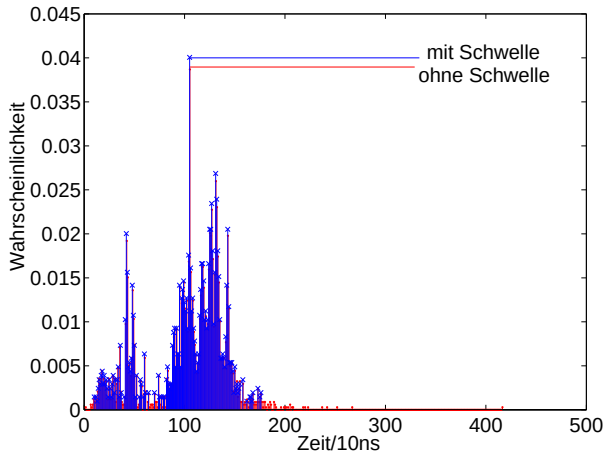
  Innerhalb der Verweildauer des Impulsabstandes, muss eine neue Zufallszahl für die Impulsdauer, Impulsamplitude und einen neuen Impulsabstand erzeugt werden. Daraus folgt, dass der kleinstmögliche Impulsabstand in Taktzyklen größer als die Ereignisanzahl von Impulsdauer/Impulsamplitude (siehe Abschnitt 5.1.4) und Impulsabstand plus



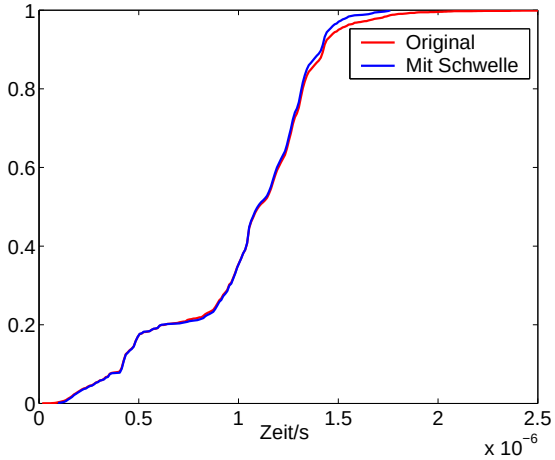
(a) Dichtefunktion, Schwelle = 1, 170 Ereignisse



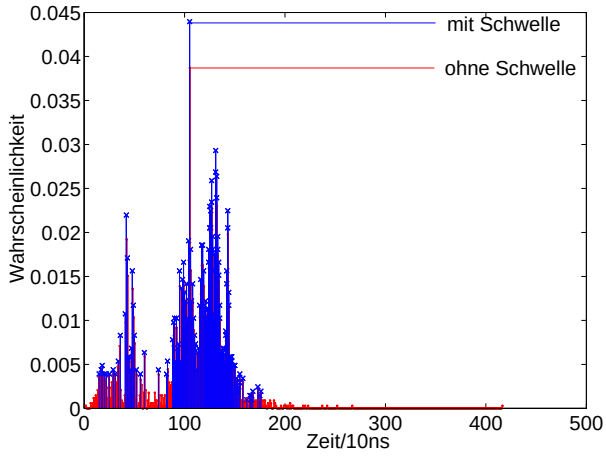
(b) Verteilungsfunktion, Schwelle = 1, 170 Ereignisse



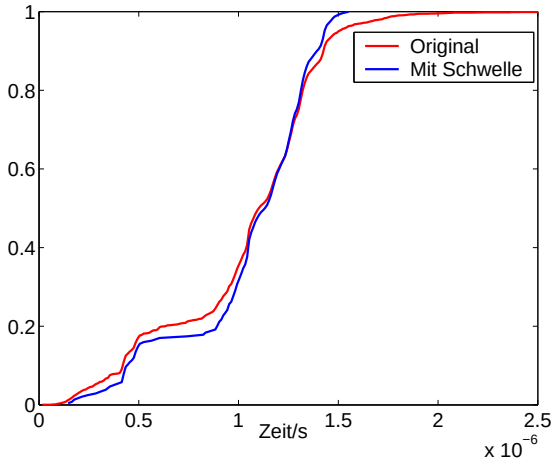
(c) Dichtefunktion, Schwelle = 3, 135 Ereignisse



(d) Verteilungsfunktion, Schwelle = 3, 135 Ereignisse



(e) Dichtefunktion, Schwelle = 10, 90 Ereignisse



(f) Verteilungsfunktion, Schwelle = 10, 90 Ereignisse

Abbildung 5.6: Impulsdauer mit verschiedenen Schwellwerten, bei 417 Histogrammklassen

3 (Pipeline-Stufenanzahl) sein muss. Wäre das nicht erfüllt, könnte es passieren, dass die Zeit für einen Impulsabstand schon abgelaufen ist, die VHDL Logik aber noch keine neue Zufallszahl berechnet hat.

Es ist also klar, dass die Daten des Impulsabstandes die maximal mögliche Werteanzahl für Impulsdauer/-amplitude vorgeben. Praktisch ist es so, dass die kleinen Ereignisse des Impulsabstandes, die mit der Erzeugungszeit in Konflikt stehen, weggelassen werden. Dies wird allerdings bei der Fehlerbetrachtung berücksichtigt. Es ist durchaus sinnvoll einige (seltene) Ereignisse wegzulassen, um dafür die zeitliche Auflösung (d.h. Ereignisanzahl) zu erhöhen.

- Impulsmuster:

  Innerhalb der Verweildauer des Impulsmusterabstandes, muss eine neue Zufallszahl für die Impulsmusterdauer erzeugt werden. Der Impulsmusterabstand ist in der Simulation konstant gewählt und auch sehr groß. Normalerweise liegt dieser bei 4 ms bis 6.67 ms, was 400.000 bis 667.000 Taktzyklen entspricht. Die maximale darstellbare Klassenanzahl für die Impulsmusterdauer ist also $\frac{max(Impulsmusterdauer)}{10ns} \approx 80256$. Deswegen ist die Ereignisanzahl von der Impulsmusterdauer praktisch nicht durch die Berechnungszeit der VHDL Logik begrenzt.

Beispiel für das Weglassen von Ereignissen

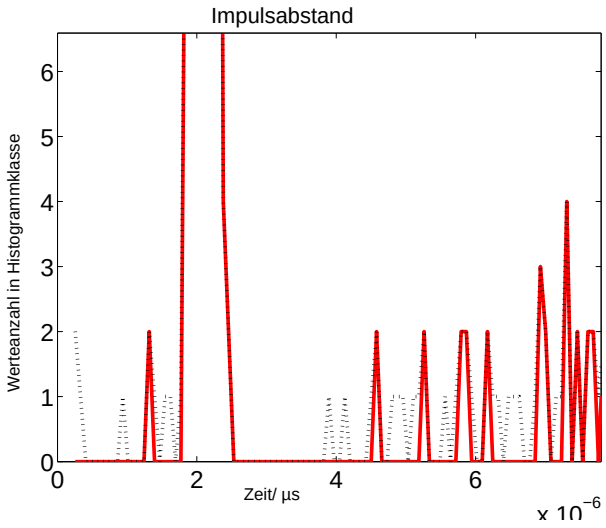


Abbildung 5.7: Die Ereignisse des Impulsabstandes, die kleiner als die Erzeugungszeit sind, werden weglassen

Es wird ein vergrößerter Ausschnitt eines Plots der Klassenmittelpunkte des Histogramms von Impulsabstand gezeigt, um dies anschaulich zu erklären.

Die schwarze gestrichelte Linie beschreibt die Werteanzahl in einer Klasse vor dem Weglassen der Ereignisse, die rote Linie beschreibt die Werteanzahl nach dem Weglassen.

Gründe Ereignisse wegzulassen sind:

- Durch die Schwelle: (siehe Abschnitt 5.1.2)

  Es wurde ein Histogramm mit 492 Klassen gebildet. Davon waren 187 Klassen nicht leer. Die Schwelle wurde zu '1' gewählt, also die Klassen, die nur eine Stichprobe enthalten werden weggelassen. Es verbleiben 104 Klassen/Ereignisse. Die höchste Stichprobenanzahl in einer Klasse ist 81.

- Durch Beschränkung der Berechnungszeit: (siehe 5.1.3)

  Die 104 Ereignisse, die sich ergeben haben, entsprechen einer maximalen Berechnungszeit von 107 Takten. Die kleinsten Ereignisse sind in Tabelle 5.1 aufgelistet.

ns	Takte	Stichprobenanzahl	Grund des Weglassens
255	25	2	Berechnungszeit
331	33	1	Schwelle & Berechnungszeit
937	94	1	Schwelle & Berechnungszeit
1317	132	2	-
1393	139	1	Schwelle
1544	154	1	Schwelle
1620	162	1	Schwelle
1772	177	1	Schwelle
1848	185	12	-
1923	192	25	-
1999	200	48	-

Tabelle 5.1: Kleinste Ereignisse der Messdaten der Impulsdauer

  Die Impulsabstände, die kleiner als 107 Takte sind, werden weggelassen. Die Klassen mit nur einer Stichprobe werden aufgrund des Schwellwertes weggelassen. Hinzu kommt somit nur noch das erste Ereignis (255ns). Wäre die Schwelle = 0, müssten die ersten 9 Ereignisse durch die Berechnungszeit weggelassen werden (kleiner als 190 Takte).

Berechnung des relativen Dichtefehlers

Der Dichtefehler wird durch das Weglassen von Werten verursacht. Es gibt wiederum zwei Arten der Berechnung, einen relativen Dichtefehler zu bestimmen.

- Es wird zum einen

$$\sum_i \left(\frac{Dichte\_nachher(i) - Dichte\_vorher(i)}{Dichte\_vorher(i)}\right)^2 = \sum_i \left(\frac{Dichteabweichung(i)}{Dichte(i)}\right)^2. \quad (5.6)$$

berechnet.

Für jedes Messdatenelement wird eine relative Dichteabweichung zu sich selbst angegeben. Das Problem hierbei ist, wenn eine Klasse weggelassen wird, die nur eine Stichprobe enthält, dass der relative Fehler um '1' erhöht wird. D.h. es gibt in der Summe des relativen Fehlers soviel Summanden des Wertes '1', wie man Werte weggelassen hat, obwohl der Fehler im ganzen gesehen gar nicht so groß ist.

- Es wird daher zum anderen

$$\sum_i \left(\frac{Dichte\_nachher(i) - Dichte\_vorher(i)}{max(Dichte)}\right)^2 = \sum_i \left(\frac{Dichteabweichung(i)}{max(Dichte)}\right)^2. \quad (5.7)$$

berechnet. Im Vergleich zu der vorherigen Berechnung des relativen Fehlers, ist diese Berechnung objektiver. Sie vergleicht die Wahrscheinlichkeit der Ereignisse, die weggelassen werden, mit der Wahrscheinlichkeit des Ereignisses, das am wahrscheinlichsten ist.

Bei der Datenaufbereitung wird die zweite Berechnung des relativen Dichtefehlers angewandt, da sie vernünftige Ergebnisse liefert.

5.1.4 Das Optimalitätskriterium für die Histogrammklassenanzahl und den optimalen Schwellwert

Die Histogrammklassenanzahl und der Schwellwert ist empirisch ermittelbar. Der Fehler, der durch das Histogramm und die Schwelle verursacht wird, ist nach dem empirischen Optimieren nicht sehr groß.

Es wurde jedoch ein Kriterium und ein Algorithmus entwickelt, der die optimale Klassenanzahl und den optimalen Schwellwert bestimmt. Mit dem Kriterium und dem Algorithmus können fast alle Arten von Messdaten bearbeitet werden.

Der Gesamtfehler definiert sich aus der Summe von Gleichung 5.7 und 5.4 zu:

$$Gesamtfehler = \sum_i \left(\frac{Dichteabweichung(i)}{max(Dichte)}\right)^2 + \sum_i \left(\frac{Zeitabweichung(i)}{Zeitwert}\right)^2 \quad (5.8)$$

Die Klassenanzahl und der Schwellwert soll so bestimmt werden, dass der Gesamtfehler minimiert wird. Dafür wird im ersten Schritt eine maximale Ereignisanzahl 'maxW' gewählt, die nach der Datenreduktion übrig bleiben soll. Die maximale Ereignisanzahl wird also vorgegeben und der Algorithmus kann dazu benutzt werden, eine optimale Histogrammklassenanzahl und eine optimale Schwelle zu bestimmen, so dass höchstens 'maxW' Ereignisse übrig bleiben.

Es muss allerdings nicht nur ein 'maxW', sondern auch eine Größe 'maxX' vorgegeben werden. Dies ist das kleinste Ereignis, welches noch erlaubt ist. Dies gilt wegen der Beschränkung durch die Berechnungsdauer der Zufallszahlen. (siehe Abschnitt 5.1.3). Praktisch lässt der Algorithmus alle Ereignisse kleiner als 'maxX' weg, und zählt sie zum Gesamtfehler hinzu.

'maxX' wurde bei dem Impulsabstand zu 'maxW'+3 gewählt. Bei der Impulsdauer, der Impulsamplitude sowie der Impulsmusterdauer wird 'maxX' zu 0 gesetzt, weil diese drei Zufallsgrößen keinen Takt für die Erzeugung zweier Zufallszahlen liefern und so auch nicht die Zeit zwischen der Erzeugung von zwei Zufallszahlen bestimmen. Jedoch ist bei der Impulsdauer und der Impulsamplitude 'maxW' als genauso groß wie bei dem Impulsabstand vorgegeben. Bei der Impulsmusterdauer ist 'maxW' theoretisch durch den konstanten Impulsmusterabstand bestimmt. Dieser ist jedoch so groß, dass praktisch keine Begrenzung für 'maxW' durch die Impulsmusterdauer existiert.

Die Klassenanzahl ist begrenzt, weil eine Histogrammklassenbreite von weniger als 10ns nicht mehr sinnvoll ist.

$$\text{Maximale Klassenanzahl } = \frac{\max(\text{Messdaten})}{\text{Taktzeit FPGAs}} \tag{5.9}$$

Die Schwelle ist dafür da, die Ereignisanzahl zum Ende unter 'maxW' Werte zu bekommen. Zusätzlich werden bei den Impulsabstandereignissen, die Werte, welche die Randbedingung 3 nicht erfüllt haben (kleiner als 'maxX'), weggelassen.

Die Logik des Algorithmus, der im ersten Schritt beschrieben wurde, ist in Ablaufdiagramm Abbildung 5.8 dargestellt.

Der zweite Schritt des Algorithmus wiederholt den ersten Schritt, um festzustellen, welche 'maxW' (maximale Werte Anzahl) den kleinsten Gesamtfehler auslöst. In Abbildung 5.9 wird der minimale Gesamtfehler für den Impulsabstand in Abhängigkeit von 'maxW' dargestellt. Grundsätzlich ist es so, dass der Fehler mit zunehmendem 'maxW' sinkt. Der steile Anstieg des Fehlers bei hohen 'maxW' Werten ist auf Randbedingung 3) zurückzuführen: Bei einer großen Werteanzahl müssen viele Ereignisse aufgrund der erhöhten Berechnungszeit weggelassen werden.

Bei 'maxW'=180, sieht man, dass es den kleinsten minimalen Gesamtfehler gibt. Die getrennten Gesamtfehler in Abhängigkeit von der Klassenanzahl bei 'maxW'=180 werden in Abbildung 5.10 dargestellt.

Die rote Kurve ist die Summe des relativen zeitlichen Fehlerquadrats, die blaue Kurve bezeichnet die Summe des relativen Dichtefehlerquadrats.

Der zeitliche Fehler nimmt ab, wenn die Klassenanzahl erhöht wird.

Der Dichtefehler steigt mit der Klassenanzahl an. Bei vielen Klassen, ist ein großer 'Aufwand' nötig, um die Werteanzahl unter die vorgegebene maximale Werteanzahl ('maxW') zu bekommen. Dazu wird ein immer größerer Schwellwert gebraucht, was eine Zunahme des Dichtefehlers verursacht.

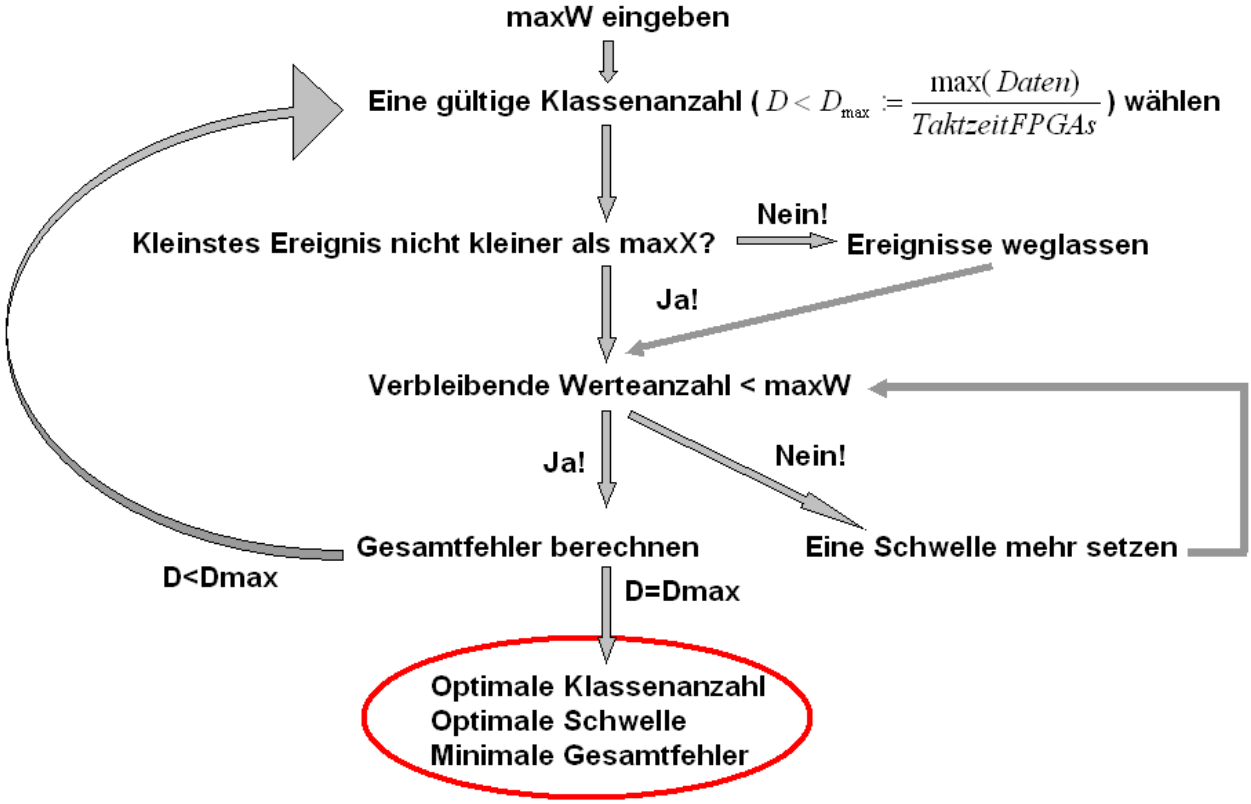


Abbildung 5.8: Logik für den ersten Schritt

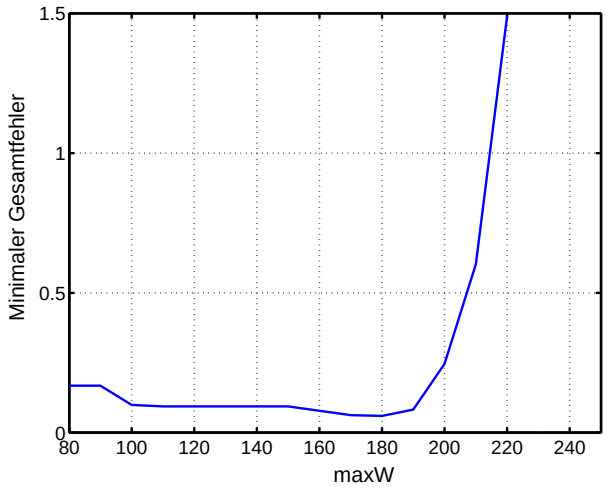


Abbildung 5.9: Minimaler Gesamtfehler für den Impulsabstand in Abhängigkeit von 'maxW'

Die optimale Klassenanzahl 492 mit einem Schwellwert von '0' bei dem Impulsabstand verursacht einen minimalen Gesamtfehler. (siehe Abbildung 5.11)

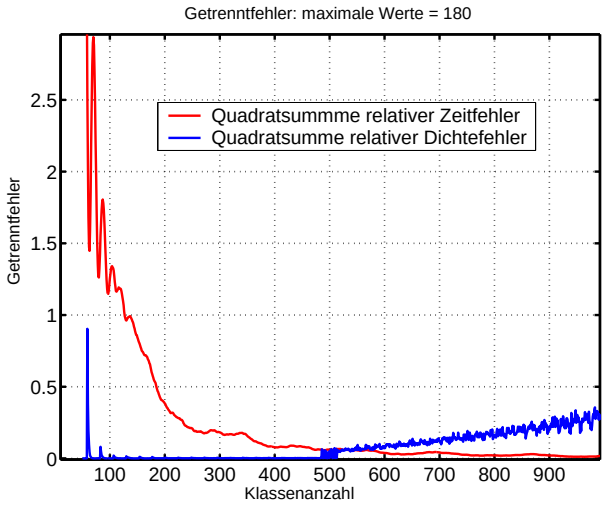


Abbildung 5.10: Getrennte Fehler für den Impulsabstand in Abhängigkeit der Klassenanzahl bei 'maxW' = 180

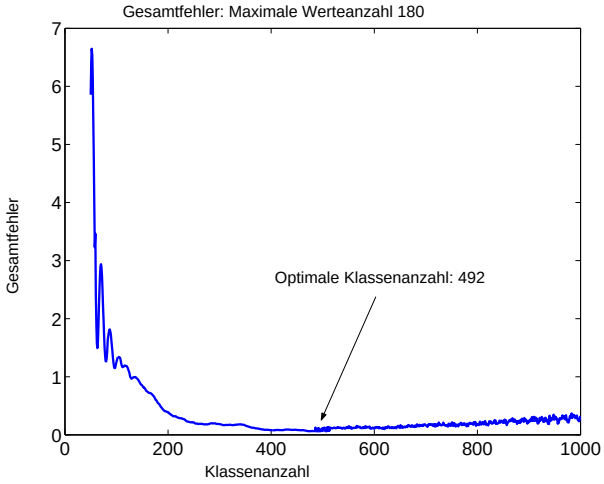


Abbildung 5.11: Gesamtfehler für Impulsabstand in Abhängigkeit der Klassenanzahl bei maxW=180

Zweidimensionale Dichtefunktion

Prüfen der Korreliertheit

Es gibt korrelierte stochastische Größen, d.h. die Größen sind voneinander abhängig. Um die Korreliertheit zu prüfen, wird getestet, ob Gleichung 5.10 nicht erfüllt ist.

$$f(x,y) = f(x)f(y) \tag{5.10}$$

$$f(x) = \int_{-\infty}^{+\infty} f(x,y)dy \tag{5.11}$$

$$f(y) = \int_{-\infty}^{+\infty} f(x, y)dx \tag{5.12}$$

Zweidimensionales optimales Kriterium

Die Impulsdauer und die Impulsamplitude sind nach der Überprüfung des Kriteriums korreliert. Daher muss eine zweidimensionale Dichtefunktion mit gleichem Optimalitätskriterium bearbeitet werden.

Bei der Entscheidung der Histogrammklassenanzahl, müssen in diesem Fall auch die drei vorigen Randbedingungen beachtet werden. (siehe 5.1.1)

Die Klassenanzahl des 2D Histogramms ist beschränkt durch:

- Taktzeit des FPGAs und der Referenzspannung des DA-Wandlers.

  Die maximale Zeitauflösung der Impulsdauer, ist wie beim Impulsabstand und Impulsmusterdauer, von der Taktzeit des FPGAs abhängig. Die maximale Amplitudenauflösung hängt von der Referenzspannung des DA-Wandlers ab. Das FPGA wird an einen DA-Wandler angeschlossen. Die Spannung, der ein Bit entspricht, ist 0.122mV. Deswegen wurde in der Simulation die Amplitudenskalierung auf 0.122mV gesetzt. Es ist nicht möglich genauer Digital-Analog zu wandeln.

  Deswegen kann die Klassenanzahl der einzelnen Achsen (X-Achse für die Impulsdauer, Y-Achse für die Impulsamplitude) nicht beliebig hoch aufgelöst werden.

  Die gesamte Klassenanzahl des 2D-Histogramms ist das Produkt der Klassenanzahlen der einzelnen Achsen.

- Speicherplatz des FPGAs

  Alle verbleibenden Ereignisse, d.h. Kombinationen von der Impulsdauer mit der Impulsamplitude werden in den Logikzellen oder Speicherzellen des FPGAs abgelegt. Da die sich die Gesamtklassenanzahl quadratisch zu der Klassenanzahl der Achsen verhält, ergibt sich sehr schnell das Problem, dass zu viele Werte vorhanden sind. Diese kann man nicht alle im FPGA speichern. Außerdem müssen anstatt eines Wertes, zwei Werte als Kombination gespeichert werden. Der Bitverbrauch verdoppelt sich dadurch in etwa. Das Programm benötigt somit etwa doppelt so viel Logikzellen im FPGA, womit sich auch sehr schnell ein Timing Problem ergeben kann.

- Berechnungsdauer

  Es muss weiter berücksichtigt werden, dass die Anzahl der Kombinationen kleiner als der kleinste Abstand zwischen der Erzeugung von zwei Zufallszahlen - Pipelineverzögerung ist.

Bei der 2D-Dichtefunktion ist es also noch wichtiger, nicht zu viele Klassen zu erzeugen. Andererseits, wenn man zu wenig Klassen bereitstellt, verlieren die Zufallszahlen die Ähnlichkeit zu den Messdaten.

Um die Werteanzahl zu reduzieren, ohne an Genauigkeit bei der Zeit und der Amplitude zu verlieren, muss man die Klassenanzahl von der Amplitude und der Impulsdauer, sowie die Schwelle im Histogramm entsprechend koordinieren.

Der 2D-Algorithmus ist fast gleich zu dem 1D-Algorithmus (siehe Abschnitt 5.1.4). Nach der Vorgabe von 'maxW' bestimmt der Algorithmus wieder die optimale Schwelle sowie die optimalen Histogrammklassenanzahlen für die Impulsdauer und die Impulsamplitude. Man erhält eine vergleichsweise hohe Schwelle, weil sehr viele Kombinationen der Messdaten für die Impulsdauer und die Impulsamplitude vorhanden sind. Bei dem Algorithmus handelt es sich um einen rekursiven Algorithmus: der Schwellwert wird solange erhöht, bis die Anzahl der verbleibenden Werte unter 'maxW' ist.

Abbildung 5.12 zeigt eine zweidimensionale Dichtefunktion mit einer Schwelle. Die optimale Schwelle ist '4'.

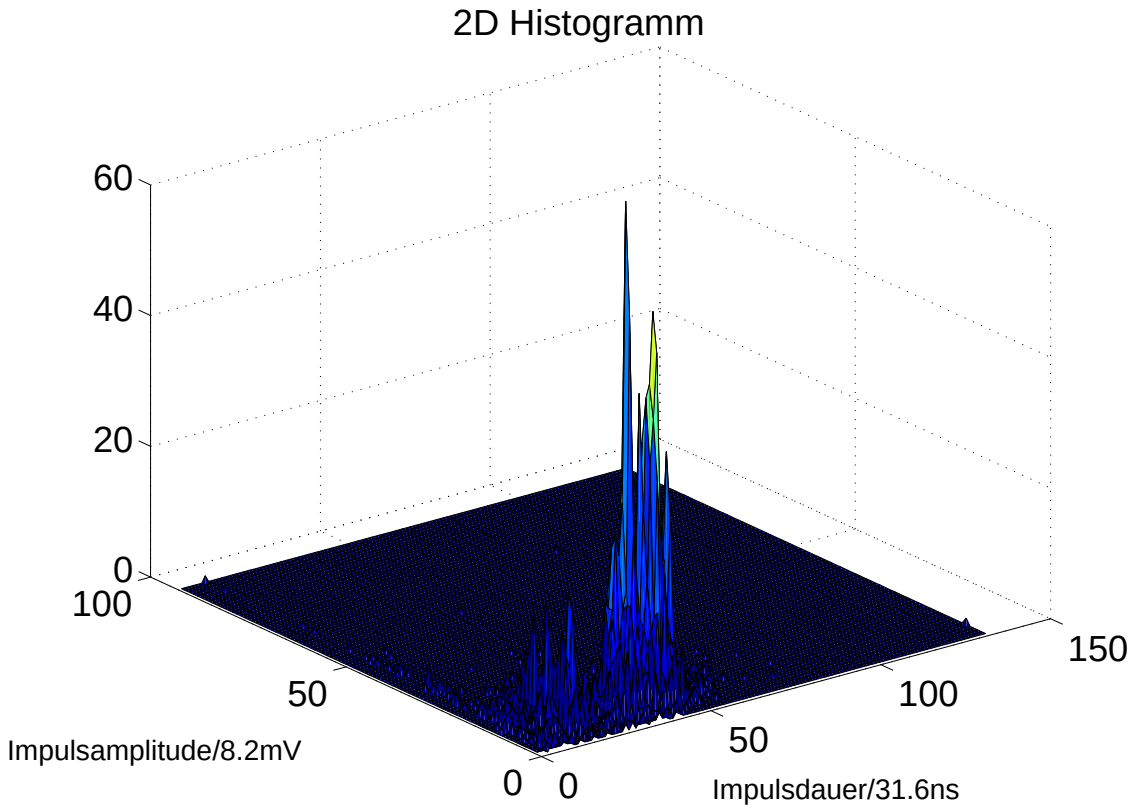


Abbildung 5.12: 2D-Dichtefunktion der Impulsamplituden und Impulsdauer

5.1.5 Ergebnis

In Tabelle 5.2 sind nur die Ergebnisse von Verfahren I dargestellt, für einen Vergleich mit den Ergebnissen aus Verfahren II siehe Tabelle 5.3.

Datensatz	Anzahl Werte	Optimale Schwelle	Optimale Klassenzahl
Impulsdauer/Impulsamplitude	170	4	132/93
Impulsabstand	179	0	492
Impulsmusterdauer	166	1	9000

Tabelle 5.2: Zusammenfassung der bestimmten optimalen Werte für Szenario Autobahn

5.1.6 Quantisierungsbits

Nach der Optimierung und Reduktion der X- und Y-Werte der Verteilungsfunktion, sollen die Y-Werte quantisiert werden, damit sie an das VHDL-Programm angepasst werden können.

Es müssen mindestens so viele Quantisierungsbits verwendet werden, dass sich die Y-Werte nach der Quantisierung immer noch gerade um 1 Bit unterscheiden. Wenn dagegen mit zu vielen Bits quantisiert wird, ist dies eine Verschwendung des Speicherplatzes.

Dazu wird der 'Schwellwert+1' als 1 Bit gewählt. Nach der Anwendung einer Schwelle, haben alle Klassen eine Stichprobenanzahl größer als der Schwellwert. Durch Bildung des Integrals, d.h. die Y-Werte der Verteilungsfunktion werden addiert, haben diese jeweils einen minimalen Unterschied gleich dem 'Schwellwert+1'. Die gesamte Stichprobenanzahl wird in Intervalle von 'Schwellwert+1' unterteilt, woraus sich die höchste Anzahl an Y-Werten der Verteilungsfunktion ergibt.

$$2^{Quantisierungsbits} \geq \frac{Stichprobenanzahl}{(Schwellwert + 1)} \tag{5.13}$$

Deswegen werden mindestens

$$\log_2 \frac{Stichprobenanzahl}{(Schwellwert + 1)} \tag{5.14}$$

Bits zum Quantisieren benötigt.

Mit dieser Methode ergibt sich für die Quantisierungsbits

- Impulsdauer (ohne Impulsamplitude): 11 Bits

- Impulsabstand: 10 Bits
- Impulsmusterdauer: 10 Bits
- Impulsdauer und Impulsamplituden (2D): 10 Bits

Obwohl die 2D-Dichtefunktion mehr Klassenmöglichkeiten hat, besteht sie insgesamt nur aus 3386 Stichproben, weswegen für die Quantisierung 10 Bits genügen (Schwelle = 4).

5.1.7 Erzeugung von Zufallszahlen

In den vorigen Abschnitten ist durch den Algorithmus die optimale Klassenanzahl, die optimale Schwelle bzw. die Anzahl der Quantisierungsbits bestimmt worden. Die einzelnen Schritte sind hier nun noch einmal kurz dargestellt:

1. Daten in Matlab laden
2. Mit dem Histogramm die Messdaten zusammenfassen und evtl. einen Schwellwert benutzen.
3. Der Mittelwert eines Intervalls zusammen mit dem dazu gehörigen Wert für die Wahrscheinlichkeit (Anzahl Stichproben in der Klasse / Gesamtzahl Stichproben) bildet die diskrete Dichtefunktion.
4. Aus der diskreten Dichtefunktion wird die Verteilungsfunktion gebildet. Die X- und Y-Werte der Verteilungsfunktion werden quantisiert, und in das VHDL Programm eingegeben.
5. Mit der in VHDL programmierten Quantiltransformation die Zufallszahlen erzeugen.

Hierbei werden die ersten 4 Schritte in Matlab durchgeführt.

5.2 Verfahren II: Kontinuierliche Dichtefunktionsschätzung mit einem Histogramm

Sind genügend Messdaten vorhanden, ist es möglich die richtige Dichtefunktion des hinterliegenden Prozesses der Messdaten zu schätzen. Das Histogramm ist ein einfacher Schätzer für die Dichtefunktion. Es ist schwierig zu entscheiden, wie viele Klassen benötigt werden, um die richtige Dichtefunktion möglichst genau zu schätzen. Dies ist jetzt ein vollständig anderes Gütekriterium als zuvor! Es werden nun kontinuierliche nicht mehr diskrete (impulsförmige) Dichtefunktionen verglichen. Werden Zufallszahlen zu einer kontinuierlichen Dichte erzeugt, können alle Werte als Ereignis auftreten, nicht nur einige bestimmte (diskrete) Ereignisse.

Werden zu viele Klassen angenommen, dann liegen zu wenige Stichproben in einer Klasse und die Höhe einer Klasse nähert die wahre Dichte zu ungenau an. Gibt es zu wenig Klassen, ist die Schätzung ebenfalls zu ungenau. Im Extremfall bei nur einer Klasse, ist die Schätzung eine Gleichverteilung. Dies führt offensichtlich auf ein falsches Ergebnis.

Das hier vorgestellte Verfahren II ist nur für eindimensionale Dichtefunktionen geeignet, weswegen nur ein binäres Modell oder ein Modell ohne Beachtung der Korreliertheit der Impulsdauer mit der Impulsamplitude realisiert werden kann.

Es gibt viele Verfahren, die sich mit dem Problem der optimalen Klassenanzahl beschäftigen, von denen im folgenden zwei vorgestellt werden.

5.2.1 Sturges Verfahren

'Sturges Verfahren' gibt es schon seit relativ langer Zeit. Die Regel lautet:

$$k = 1 + \log_2 n \tag{5.15}$$

wobei n die Stichprobenanzahl ist, und k die resultierende Klassenanzahl bezeichnet. Die optimale Klassenanzahl ist nur von der Stichprobenanzahl abhängig, nicht von den Messdaten selbst. D.h. die gleiche Menge der Stichproben ergibt immer die gleiche Anzahl an Klassen, somit kann es nur für eine bestimmte Verteilung optimal sein.

Die Regel ist speziell nur für Dichtefunktionen, die Normalverteilungscharakter haben geeignet, wobei die Varianz mitunter falsch geschätzt wird. Sind die Messdaten nicht normalverteilt, muss die Klassenanzahl erhöht werden [Hyn95]. Außerdem funktioniert diese Regel nur, wenn die Stichprobenanzahl kleiner als 200 ist. 'Sturges Verfahren' führt zu einer 'oversmoothed' Dichteschätzung, besonders für große n [Hyn95].

5.2.2 Verfahren von Birgé und Rozenholc

Neben umfangreich vorhandenen alten Verfahren, beschäftigt sich auch ein relativ neues Verfahren mit dem Gebiet der Histogramm Dichteschätzung.

$X_1, X_2, ..., X_n$ sind die n Stichproben einer unbekannten Dichtefunktion mit Dichte f. Die Histogramm Dichteschätzung von f, basierend auf D gleich breiten Klassen, ist gegeben durch:

$$\hat{f}_D(x) = \hat{f}_D(X_1, X_2, ..., X_n)(x) = \frac{D}{n} \sum_{j=1}^{D} N_j 1_{I_j}(x) \tag{5.16}$$

wobei $N_j = \sum_{i=1}^{n} 1_{Ij}(X_i)$ die Stichprobenanzahl in der j-ten Klasse bei Klassenbreite D ist. $1_{I_j}(x)$ ist die Indikatorfunktion, die als $1_{I_j}(x) = \begin{cases} 1 & \text{für } x \in I_j \\ 0 & \text{sonst} \end{cases}$ definiert ist.

$\hat{f}_D$ ist die geschätzte Dichtefunktion. Um die Qualität der Histogrammschätzer zu testen, ist eine Verlustfunktion definiert. Hier wird die spezielle Verlustfunktion (Quadratischer Fehler L_2) verwendet:

$$l(f, \hat{f}_D(X_1, ..., X_n)) = \|f - \hat{f}_D\|^2 = \int_{-\infty}^{+\infty} (f(y) - \hat{f}_D(y))^2 dy \tag{5.17}$$

Der Erwartungswert dieser Verlustfunktion

$$R_n(f, \hat{f}_D, l) = E_f[l(f, \hat{f}_D(X_1, ..., X_n))] \tag{5.18}$$

wird bezüglich D minimiert. Die optimale Klassenanzahl D^{opt} des Histogramms ist abhängig von der Stichprobenanzahl n und auch der richtigen Dichtefunktion f, also

$$D^{opt} = D^{opt}(f, n) \tag{5.19}$$

D^{opt} kann also theoretisch dadurch berechnet werden, dass der kleinste Erwartungswert der Verlustfunktion gebildet wird.

$$R_n(f, \hat{f}_{D^{opt}}, l) = inf_{D \geq 1} R_n(f, \hat{f}_D, l) \tag{5.20}$$

Leider ist es unmöglich das exakte $D^{opt}(f, n)$ zu bekommen, wenn die Dichtefunktion f gerade geschätzt werden soll.

Penalized Maximum Likelihood Estimator

Der hier vorgestellte Dichteschätzer basiert auf einem Penalized Maximum Likelihood Schätzer. Diese maximieren eine Gleichung der Form:

$$L_n(D) - pen(D) \tag{5.21}$$

wobei

$$L_n(D) = \sum_{j=1}^{D} N_j \log(DN_j/n) \tag{5.22}$$

die Likelihood Funktion und

$N_j = \sum_{i=1}^{n} 1_{I_j}(X_i)$ wieder die Stichprobenanzahl in einer Klasse ist.

Die Penalty-Funktion $pen(D)$ muss nach der Theorie der Penalized Maximum Likelihood Schätzer eine bestimmte Form haben und kann in gewissen Grenzen frei gewählt werden. In [BR02] ist die $pen(D)$ Funktion zu

$$pen(D) = D - 1 + (\log D)^{2.5} \text{ für } 1 \leq D \leq \frac{n}{\log n} \tag{5.23}$$

bestimmt worden.

$L_n(D)$ und $pen(D)$ sind beide abhängig von der Klassenanzahl D. Zusätzlich ist die Funktion $L_n(D)$ noch abhängig von N_j, also der Stichprobenanzahl in einer Klasse. Die Klassenanzahl D, die $L_n(D) - pen(D)$ maximiert, führt zur Lösung.

Bewertung des Verfahrens

Um das Verfahren zu testen, wurde in [BR02] der Dichtefehler für einige vorgegebene Dichtefunktionen berechnet. Um zu vergleichen, wurden 15 altbekannte Schätzer: f_1, f_2,...f_{15} der verschiedenen Verfahren zur Histogramm Dichteschätzung eingesetzt: Sturges Rule, Verfahren von Daly, Verfahren von He und Meeden, Verfahren von Devroye und Lugosi, Verfahren das auf Wavelet Thresholding basiert....

Es wurde eine Reihe verschiedenartiger Dichtefunktionen gewählt, und mit den Schätzern aus zu den Dichten zugehörigen Zufallszahlen geschätzt.

Durch empirische Beobachtung, ist erkennbar, dass dieser neue Schätzer der Beste von allen anderen Schätzern ist. Er ist besonders für die Schätzung von unregelmässigen Dichtefunktionen geeignet, wie sie hier vorliegen [BR02].

5.2.3 Ergebnisse

Dieses Verfahren wurde in Matlab programmiert. Die Funktion $ln\_pen(Daten, D)$ ist dafür zuständig.

Durch Variation von D und Berechnung $L_n(D) - pen(D)$ wird eine optimale Klassenanzahl D ermittelt.

Hieraus ergeben sich folgende Ergebnisse:

- Impulsabstand: Aus den 917 Stichproben ergeben sich 69 Klassen. Davon haben 44 Klassen mehr als '0' Stichproben.
- Impulsmusterdauer: Aus den 1170 Stichproben ergeben sich 142 Klassen. Davon haben 82 Klassen mehr als '0' Stichproben.
- Impulsdauer: Aus den 3386 Stichproben ergeben sich 125 Klassen. Davon haben 69 Klassen mehr als '0' Stichproben.

Für einen Vergleich mit den Ergebnissen aus Verfahren I siehe Tabelle 5.3.

Solche Ergebnisse widerlegen die Vorhersage, dass Schätzungen mit vielen Klassen generell besser sind. Dies ist anschaulich in der folgenden Abbildung dargestellt:

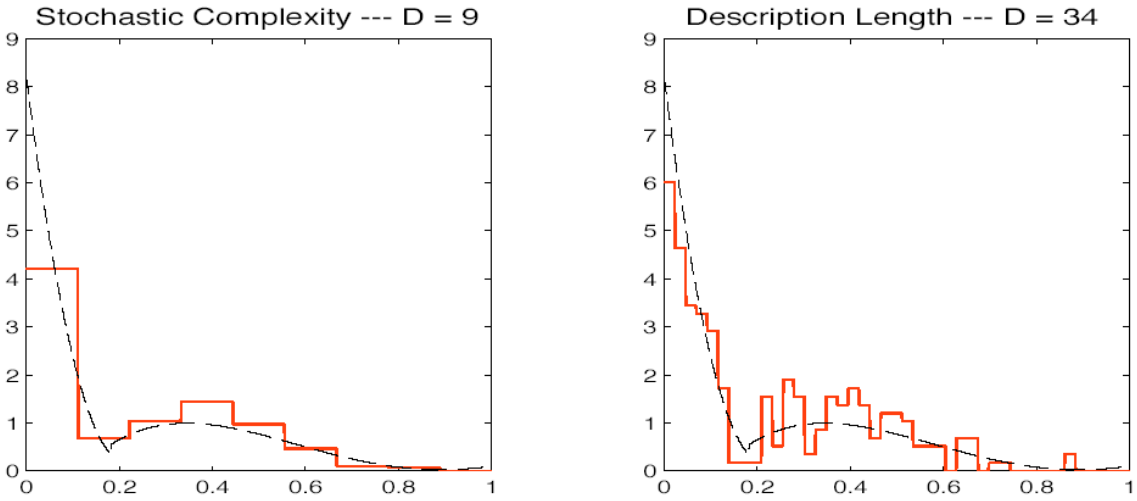


Abbildung 5.13: Histogramm mit verschiedenen Klassenanzahlen um die Dichtefunktion zu schätzen

In Abbildung 5.13 ist zu erkennen, dass es bei D=9 weniger Fehler als bei D=34 gibt. Es kommt darauf an, wie die richtige kontinuierliche Dichtefunktion aussieht. Wenn sich die Kurve langsam ändert, sind weniger Klassen durchaus besser geeignet.

Das Wählen zu vieler Klassen ist sicherlich für die Dichteschätzung nicht geeignet. Es werden ausreichend Stichproben insgesamt, und genug Stichproben in einer Klasse benötigt, um die Dichtefunktion zu schätzen. Wenn beispielsweise jede Klasse nur wenige Stichproben enthält, ist das Histogramm eine sehr schlechte Schätzung der richtigen Dichtefunktion.

Verteilungsfunktion und Umkehrfunktion

Da sich die Verteilungsfunktion als das Integral der Dichtefunktion berechnet, ist die Verteilungsfunktion des Histogramms dreieckförmig. Daraus folgt, dass die Umkehrfunktion (Quantilfunktion) ebenfalls dreieckförmig ist.

5.2.4 Erzeugung von Zufallszahlen

Das Histogramm wird benutzt, um die Dichte zu schätzen. Bei einer Histogramm- Dichteschätzung haben alle Werte aus dem Intervall einer Klasse die gleiche Wahrscheinlichkeit wie der Mittelwert des Intervalls. Deswegen wird im ersten Schritt nur der Mittelwert als diskretes Ereignis genommen,(*) und daraus eine treppenförmige Verteilungsfunktion gebildet (diskrete Dichte). Mit der Quantiltransformation einer diskreten Dichtefunktion, die in VHDL programmiert ist, ergeben sich diskrete Zufallszahlen (die Mittelwerte eines Intervalls). Dazu werden dann Zufallszahlen einer gleichverteilten Zufallsvariablen auf (-D/2, D/2] addiert, wobei D die Klassenbreite ist. Dieser Schritt ist die Umkehrung von Schritt (*)

Der zusätzlich zu Verfahren I nötige Aufwand im FPGA ist nicht besonders hoch. Es wird höchstens eine Multiplikation mit einer Konstanten benötigt, sollte D keine Zweierpotenz sein (siehe Abschnitt 6.4).

Zusammenfassend kann man den Ablauf festhalten:

1. Messdaten laden
2. Histogramm mit dem Verfahren von Birgé und Rozenholc bestimmen, um die Dichtefunktion zu schätzen.
3. Aus den Mittelwerten der Klassenintervalle die entsprechende diskrete Dichtefunktion bilden.
4. Aus der diskreten Dichtefunktion ergibt sich eine treppenförmige Verteilungsfunktion. Die X- und Y-Werte der Verteilungsfunktion werden im VHDL Programm gespeichert.
5. In VHDL die Quantiltransformation bilden, und Zufallszahlen erzeugen.
6. Auf [-D/2, D/2] gleichverteilte Zufallszahlen addieren.

Die Schritte 1 bis 4 werden mit Matlab abgearbeitet, Schritt 5 und 6 in einem echtzeitfähigen VHDL Programm. Nur Schritt 2) und 6) unterscheiden sich von dem ersten Verfahren, das die Messwerte mit einer diskreten Dichte nachbildet.

5.3 Vergleich von Verfahren I und II

Gehen wir noch mal zurück. Es gibt zwei Verfahren, um Zufallszahlen für die Impulsdauer, den Impulsabstand und die Impulsmusterdauer in dem Störimpulsmodell zu gewinnen: Erstens ist es möglich die Messdaten aus den Fahrten nachzubilden, oder zweitens die hinterstehende Dichtefunktion der Messdaten mit einem Histogramm zu schätzen um daraus Zufallszahlen zu erzeugen.

Beide brauchen das Histogramm, aber für verschiedene Zwecke:

'Messdaten Nachbilden' benötigt das Histogramm, um die Daten zusammenzufassen. Es wird nur der Mittelwert eines Intervalls als Ereignis gespeichert.

Die Dichteschätzung benötigt das Histogramm, um die hinterstehende Dichtefunktion richtig zu schätzen. Die Ereignisse sind nicht nur die Mittelwerte eines Intervalls, sondern es sind auch Zwischenwerte möglich.

Beide haben unterschiedliche Optimalitätskriterien für das Histogramm. Die sich daraus ergebende optimale Klassenanzahl ist auch stark unterschiedlich. Wie gesehen, ist die optimale Klassenanzahl bei Verfahren II viel kleiner als bei Verfahren I, was daran liegt, dass auch Zwischenwerte erzeugt werden.

Das zweite Verfahren ist besonders für Messungen mit sehr vielen Stichproben geeignet, also wenn Verfahren I nicht mehr angewendet werden kann. Die Klassenanzahl, die Verfahren II liefert, ist viel geringer und eine Dichteschätzung ist zudem genauer, falls ein größerer Umfang an Stichproben vorhanden ist. Allerdings ist Verfahren II, wie schon erwähnt, in der vorliegenden Form nicht für die Nachbildung einer zweidimensionalen Dichtefunktion geeignet. Korrelationen zwischen Zufallsgrößen können also nicht beachtet werden.

5.4 Zusammenfassung der Ergebnisse

Zufallszahlen	Verfahren	Matlab-Array	Klassen-anzahl[a]	Schwelle[b]	Werte-anzahl[c]	maxW[d]	maxX
Impulsdauer	I (binär)	ab100_dau	417	1	170	180[e]	0
	II (binär)		125	0	69	-	0
Impulsdauer/ amplitude	I	ab100_dau/ ab100_amp	132/93	4	170	180[e]	0
Impulsabstand	I	ab_haeuf_abs	492	0	179	180[f]	182[g]
	II		69	0	43	-	72[h]
Impulsmuster-dauer	I	ab_haeuf_pl	9000	1	166	180[i]	0
	II		142	0	82	-	0

[a] Verfahren I: Durch Vorgabe von 'maxW' optimal bestimmt.
Verfahren II: Durch Minimieren von Formel 5.21 mit Formel 5.23 direkt gewonnen

[b] Verfahren I: Durch Vorgabe von 'maxW' optimal bestimmt.
Verfahren II: Ist immer 0, da das Verfahren keine Schwellwerte benutzt.

[c] Ergibt sich aus der Histogrammklassenanzahl und der Schwelle. Bei Verfahren I ist die Werteanzahl grundsätzlich $\leq$ 'maxW'.

[d] Verfahren I: Durch Optimalitätskriterium bestimmt oder vorgegeben.
Verfahren II: Nicht vorhanden.

[e] Wird wie 'maxW' bei Impulsabstand gewählt.

[f] Durch Optimalitätskriterium bestimmt.

[g] Ist gleich Werteanzahl + 3.

[h] Ist gleich *max(Werteanzahl Impulsdauer, Werteanzahl Impulsabstand)* + 3.

[i] Vorgegeben.

Tabelle 5.3: Zusammenfassung der durch die Verfahren ermittelten Parameter

Kapitel 6

Programmierung des vorgestellten Verfahrens

Das Matlab Programm quantisiert und reduziert die X- und Y-Werte der Dichtefunktion bzw. Verteilungsfunktion. Die Ergebnisse bilden die Parameter der Entity von 'Verteilung_Tabelle' im VHDL Code, die mit der Zufallszahl-Erzeugung beschäftigt ist. Das VHDL Programm erzeugt mittels Quantiltransformation die Zufallszahlen, mit denen dann das Störimpulssignal gebildet wird.

6.1 Struktur eines VHDL Codes

Die Störimpulse auf dem Kfz-Bordnetz sind durch Zufallszahlen der Impulsdauer, Impulsamplitude, Impulsabstand, Impulsmusterdauer bestimmt. Die Schritte zur Gewinnung der Parameter, sind in Kapitel 4 beschrieben.

Mit verschiedenen VHDL Strukturen werden Störimpulse in verschiedener Weise erzeugt.

Das VHDL Programm besteht aus drei Entities (Einheiten der VHDL Struktur), die im folgenden Abschnitt genau erklärt sind:

- Die Entity Verteilung_Tabelle stellt die Verbindung zwischen den in Matlab bearbeiteten Daten und der VHDL-Programmierung dar. Diese Entity liefert die Zufallszahlen gemäß der jeweils geforderten Dichtefunktion.
- Die 'As_Impulse' Entity formt eine Reihe von aperiodischen Störimpulsen mit Hilfe von zwei Verteilung_Tabelle Instanzen für die Impulsdauer und den Impulsabstand.
- Die 'Kfz_Störimpulse' Entity beeinflusst die aperiodischen Störimpulse mit zwei Zeitbeschränkungen: Die Impulsmusterdauer, die durch eine Verteilung_Tabelle bestimmt ist und dem Impulsmusterabstand, einer Konstanten.

Es sind zwei Möglichkeiten denkbar, wie die Kfz_Störimpulse Entity die As_Impulse Entity kontrollieren kann. Entsprechend gibt es auch zwei Möglichkeiten einer VHDL-Struktur.

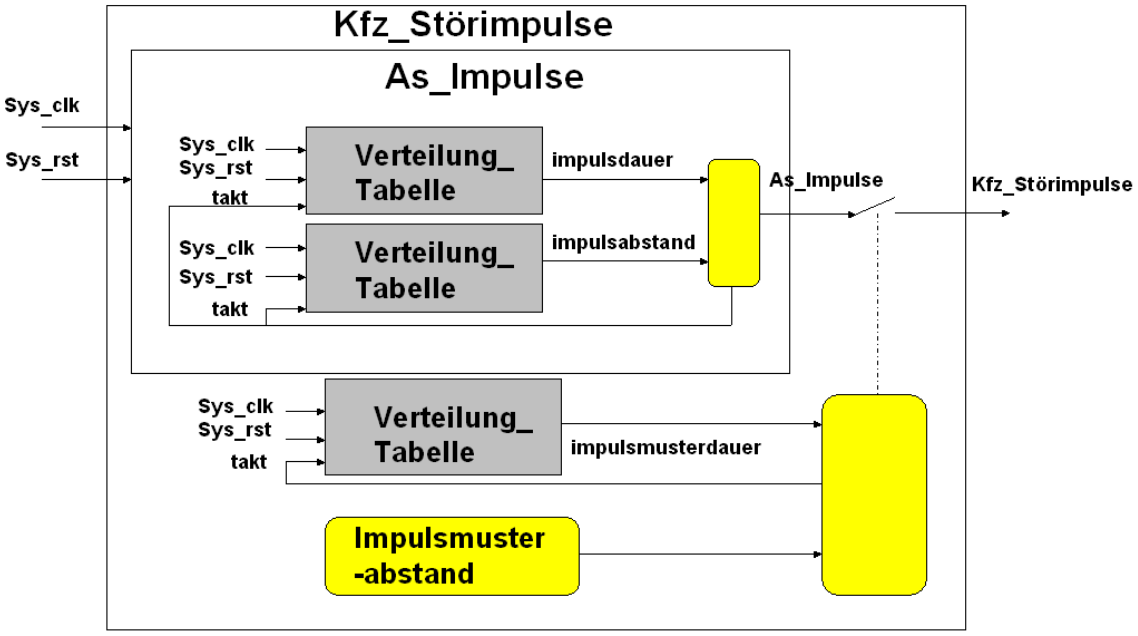


Abbildung 6.1: VHDL Struktur I

6.1.1 Binäres Störimpulsmodell

6.1.1.1 Struktur I

In Abbildung 6.1 sind drei Entities als rechteckige Boxen dargestellt: Verteilung_Tabelle, As_Impulse und Kfz_Störimpulse.

Die As_Impulse Entity enthält zwei Verteilung_Tabelle Instanzen, die mittels Quantiltransformation Zufallszahlen für die Impulsdauer und die Impulsabstand gemäß der richtigen Dichtefunktion erzeugt.

Nach dem Zeitablauf eines Impulsabstandes, erzeugt eine Logik einen Taktimpuls, durch den die beiden Verteilungstabellen für die Impulsdauer und den Impulsabstand neue Zufallszahlen liefern. Die Zufallszahlen werden immer im letzten Takt der Verteilungstabelle erzeugt. Bei jeder Taktinformation wird die schon erzeugte Zufallszahl ausgelesen und eine Berechnung einer neuen Zufallszahl gestartet, die dann beim nächsten Takt (also nach Ablauf eines Impulsabstandes/Impulsmusterabstandes) ausgelesen wird.

Die Impulsdauer und der Impulsabstand definieren die Breite des Impulses Bzw. den Abstand zum nächsten Impuls. Mit Zufallszahlen des Impulsabstandes und der Impulsdauer bildet die As_Impulse Entity eine Folge von binären aperiodischen Impulsen.

In der Kfz_Störimpulse Entity befindet sich eine Verteilung_Tabelle und eine As_Impulse Instanz. Diese Verteilung_Tabelle erzeugt die Zufallszahlen für die Impulsmusterdauer. Der Impulsmusterabstand wird als Konstante im Programm gespeichert. In einer späteren Version sollte hier eine drehzahlabhängige Variable vorgesehen werden. Nach jedem Zeitablauf des Impulsmusterabstandes erzeugt eine Logik einen Takt mit der die Verteilung_Tabelle wieder eine

neue Impulsmusterdauer ausgibt. Die Impulsmusterdauer und der konstante Impulsmusterabstand erzeugen zusammen ein binäres 'Impuls_Set' Signal, das bei dem Impulsmuster Anfang auf '1' und nach der Zeit der Impulsmusterdauer auf '0' gesetzt wird.

In der Kfz_Störimpulse Entity schaltet das Impuls_Set Signal das Ausgangssignal von As_Impulse ein und aus, und bildet so die aperiodischen Impulse mit Impulsmuster.

Die Flow-Summary des Kompiliervorgangs der Struktur I ist in Tabelle 6.1(a) (binäres Modell) und in Tabelle 6.1(b) (mit Amplitudeninformation, siehe Abschnitt 6.1.2 dargestellt).

(a) Binäres Modell

Top-level Entity Name	kfz_stoerimpulse
Family	Cyclone
Device	EP1C12Q240C6
Total logic elements	890 / 12,060 (7 %)
Total pins	5 / 173 (2 %)
Total memory bits	4,096 / 239,616 (1 %)
Total PLLs	0 / 2 (0 %)

(b) Mit Amplitudeninformation

Top-level Entity Name	kfz_stoerimpulse
Family	Cyclone
Device	EP1C12Q240C6
Total logic elements	917 / 12,060 (7 %)
Total pins	21 / 173 (12 %)
Total memory bits	6,400 / 239,616 (2 %)
Total PLLs	0 / 2 (0 %)

Tabelle 6.1: Flow-Summary zu Struktur I

6.1.1.2 Struktur II

Nachdem im vorherigen Abschnitt die einfache Struktur I vorgestellt wurde, folgt nun Struktur II, die zwei Nachteile der vorherigen Struktur I überwindet.

1. Es kann nicht garantiert werden, dass aperiodische Impulse im Zeitraum der Impuls_Set Zeit überhaupt auftreten, da kein Impuls am Anfang des Impulsmusterdauer Signals erzeugt wird. Die Definition der Impulsmusterdauer ist der Zeitraum vom ersten bis zum letzten Impuls nach dem Zündpunkt (siehe: Kapitel 4). Daher ist es auch nicht realistisch, wenn kein Impuls der aperiodischen Impulse am Anfang des Impulsmusters generiert wird.

2. Das Programm erzeugt das As_Impulse Signal dauerhaft, obwohl außerhalb des Zeitraums der Impulsmusterdauer gar keine Störimpulse benötigt werden. Dies liefert zwar keine falschen Ergebnisse, verlangsamt aber die Simulation des FPGAs in Quartus. Bei

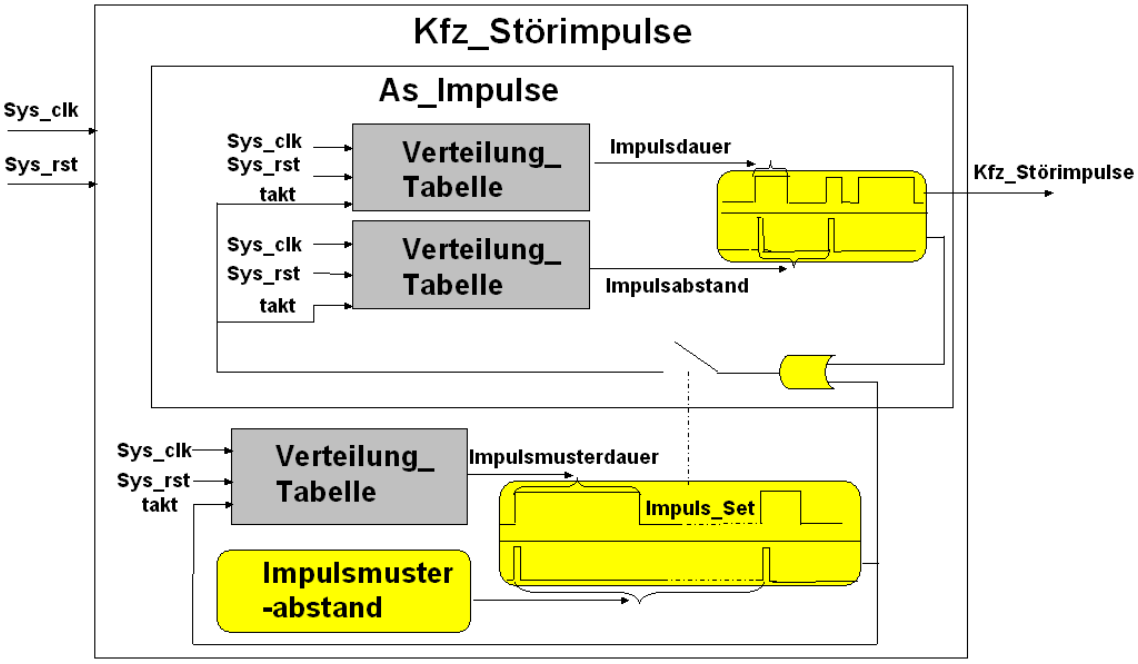


Abbildung 6.2: VHDL Struktur II

der Simulation wird die Schaltung nicht parallel realisiert wie bei einem realen FPGA, weswegen die Simulation sehr viel Zeit benötigt.

Dies ist nun dadurch zu verbessern, dass das Programm statt des As_Impulse Ausgangssignals die Taktinformation für den Impulsabstand und die Impulsdauer ein- und ausschaltet (siehe Abbildung 6.2).

Es ist ersichtlich, dass die Verteilung_Tabellen in der As_Impulse Entity und Kfz_Störimpulse Entity genauso wie vorher funktioniert. Die erzeugte Taktinformation besteht hingegen aus zwei Komponenten: Nach dem Zeitablauf eines Impulsabstandes oder dem Zeitablauf eines Impulsmusterabstandes entsteht ein Taktimpuls. Dies garantiert, dass die aperiodischen Störimpulse immer genau mit dem Anfang einer Impulsmusterdauer (einem Zündpunkt) beginnen. Zusätzlich werden die Taktinformationen zu den Verteilung_Tabellen durch das Impuls_Set Signal geschaltet. Wenn Impuls_Set gleich '0' ist, erzeugen die Verteilung_Tabellen keine Zufallsfolgen.

Die Simulation des Programms ist somit wesentlich schneller als die der ersten Struktur, und es wird zu jedem Impulsmusteranfang ein Störimpuls erzeugt.

Die Flow-Summary des Kompiliervorgangs der Struktur II mit Amplitudeninformation (siehe Abschnitt 6.1.2) ist in Tabelle 6.2 dargestellt. Auf eine Simulation des binären Modells mit Struktur II wurde hier zur Einfachheit verzichtet.

Top-level Entity Name	kfz_stoerimpulse
Family	Cyclone
Device	EP1C12Q240C6
Total logic elements	452 / 12,060 (3 %)
Total pins	14 / 173 (8 %)
Total memory bits	16,896 / 239,616 (7 %)
Total PLLs	0 / 2 (0 %)

Tabelle 6.2: Flow-Summary zu Struktur II mit Amplitude

6.1.2 Störimpulsmodell mit Amplitudeninformation

Es ist relativ einfach, die Störimpulse mit einer Amplitudeninformation zu versehen. Dazu wird die Impulsdauer und die Amplitudeninformation in einer Binärzahl zusammengefasst: z.B. Bit 0 bis Bit 7 für die Impulsdauer und Bit 8 bis Bit 16 für die Amplitude.

Impulsamplitude

$$\overbrace{101001100}\underbrace{101001100}$$

Impulsdauer

Diese Binärzahlen definieren über die Generic Konstanten der Verteilung_Tabelle die X-Werte der Verteilungsfunktion. Die Y-Werte der Verteilungsfunktion entsprechen dem Integral über der 2D Dichtefunktion.

6.2 VHDL Programmeinheit

6.2.1 Verteilung_Tabelle

Entity Deklaration

In der Entity Deklaration wird die Schnittstelle des Moduls mit der Außenwelt bzw. mit anderen Modulen beschrieben und mit einem Namen versehen.

Die Entity Deklaration enthält Port und Generic Definitionen. Mittels der PORT Anweisung werden die Eingangs- bzw. Ausgangssignale und deren Datentypen festgelegt. Mit der GENERIC Anweisung werden Steuerungskonstanten zugewiesen. Durch GENERIC MAP bei der Entity Instanzierung, werden die Zahlen oder Array-Konstanten für die Generics definiert.

In Abbildung 6.3 ist die Entity Verteilung_Tabelle dargestellt. Es gibt drei Eingangsports:

- sys_clk : Clock Signal, 10ns Taktzeit
- sys_rst : Reset Signal, aktiv 'low'
- Takt : Taktsignal für die Ausgabe der Zufallszahl.

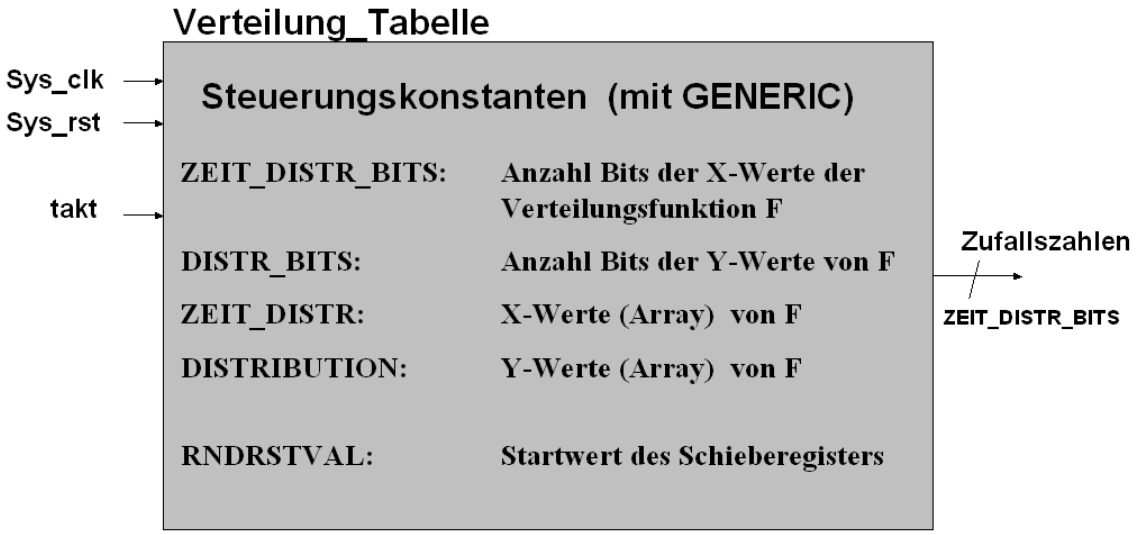


Abbildung 6.3: Die Entity Verteilung_Tabelle

Das Ausgangssignal liefert die Zufallszahlen mit der geforderten Dichtefunktion.

In der GENERIC Definition sind 5 Konstanten definiert:

- ZEIT_DISTR_BITS: Anzahl der Bits der X-Werte der Verteilungsfunktion F.
- DISTR_BITS : Anzahl der Bits der Y-Werte der Verteilungsfunktion F (diese sind durch den Schwellwert bestimmt).
- ZEIT_DISTR : X-Werte (Array) der Verteilungsfunktion, skaliert in Taktzyklen.
- DISTRIBUTION : Y-Werte (Array) der Verteilungsfunktion, die mit DISTR_BITS quantisiert sind. Die Binäre '00...0' entspricht der Zahl 0 und die binäre '11...1' der Zahl 1,0.
- RNDRSTVAL : Der Startwert des Schieberegisters. Diese Zahl legt auch die Bitbreite des Schieberegisters fest.

In der Entity, insbesondere bei den Generics von dem VHDL Programm Verteilung_Tabelle ist sehr einfach die Verbindung zwischen Matlab und VHDL erkennbar. Die in Matlab bearbeiteten Daten werden direkt in VHDL eingegeben.

Funktion

In der Verteilungstabelle wird die Quantiltransformation durchgeführt. Dabei werden die folgenden Schritte realisiert:

1. Das Schieberegister erzeugt gleichverteilte Pseudozufallszahlen. Die Bitbreite des Schieberegisters muss sehr groß sein (ca. 40Bits), damit die gleichverteilte Zufallszahlen sich

nicht zu schnell wiederholen. Außerdem müssen die Rückführungen so gewählt sein, dass die Periodenlänge maximal ist.

2. Die X- und Y-Werte der Verteilungsfunktion bzw. deren Bitbreite sind durch die Generics definiert. (siehe obiger Abschnitt)

3. Die unteren Bits aus dem Schieberegister mit der gleichen Bitbreite, wie die der Y-Werte der Verteilungsfunktion, werden mit den Y-Werten der Verteilungsfunktion verglichen. So wird das Intervall bestimmt, in dem die Pseudozufallszahl des Schieberegisters liegt.

4. Der zu dem richtigen Intervall gehörende X-Wert wird ausgegeben.

6.2.1.1 Erzeugung von gleichverteilten Zufallszahlen

Gleichverteilte Pseudozufallszahlen sind mit einem LFSR (linear feed-back shift register) einfach erzeugbar. Ein LFSR ist besonders für eine Realisierung in Hardware geeignet.

M-Sequenz

Ein LFSR mit n Bits, kann höchstens $2^n - 1$ gleichverteilte Zufallszahlen erzeugen. Diese Zufallszahlenfolge wird als M-Sequenz bezeichnet. Dazu müssen an dem Schieberegister bestimmte Feed-back angeschlossen werden. Hierbei wird das letzte Bit immer angeschlossen.

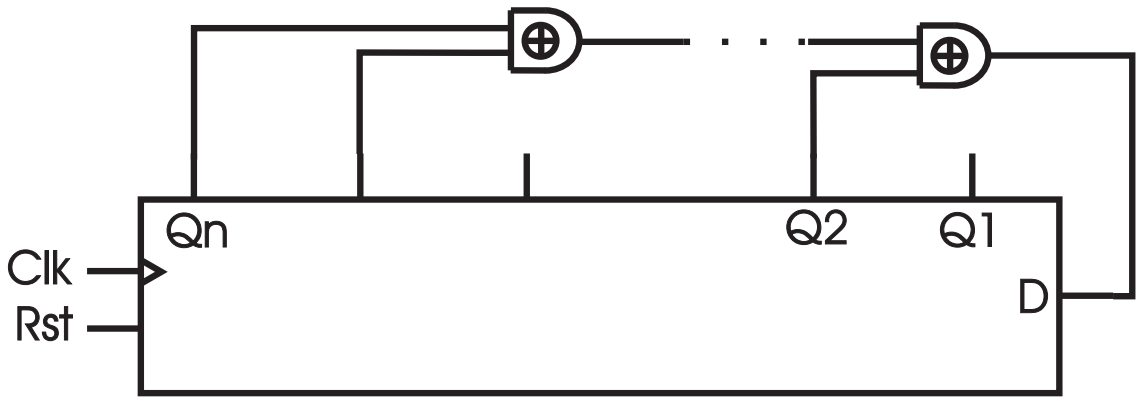


Abbildung 6.4: Erzeugung einer M-Sequenz

Die LFSRs unterschiedlicher Breite benötigen verschiedene Feed-back Positionen, um eine M-Sequenz zu erzeugen. Da die Breite des Schieberegisters durch eine Generic der Verteilung_Tabelle bestimmt ist, ist es wichtig, die richtige Position der Rückführung in dem Programm automatisch zu wählen.

Konstante Matrix für Feed-back

Es wird eine Matrix 'FEMATRIX' definiert. Die Zeilenzahl dieser Matrix entspricht der maximalen Breite des Schieberegisters - 1, weil das LFSR hier mindestens 2 Bits hat. Die Zahlen

jeder Zeile entsprechen den Positionen der Rückführung. Zu diesen kommt noch das letzte Bit hinzu, welches hier nicht abgespeichert wird, da es immer angeschlossen werden muss. Die Matrix ermöglicht Schieberegister Breiten von 2 bis 53 Bit. Die richtigen Feed-back Positionen stammen aus [Mat02].

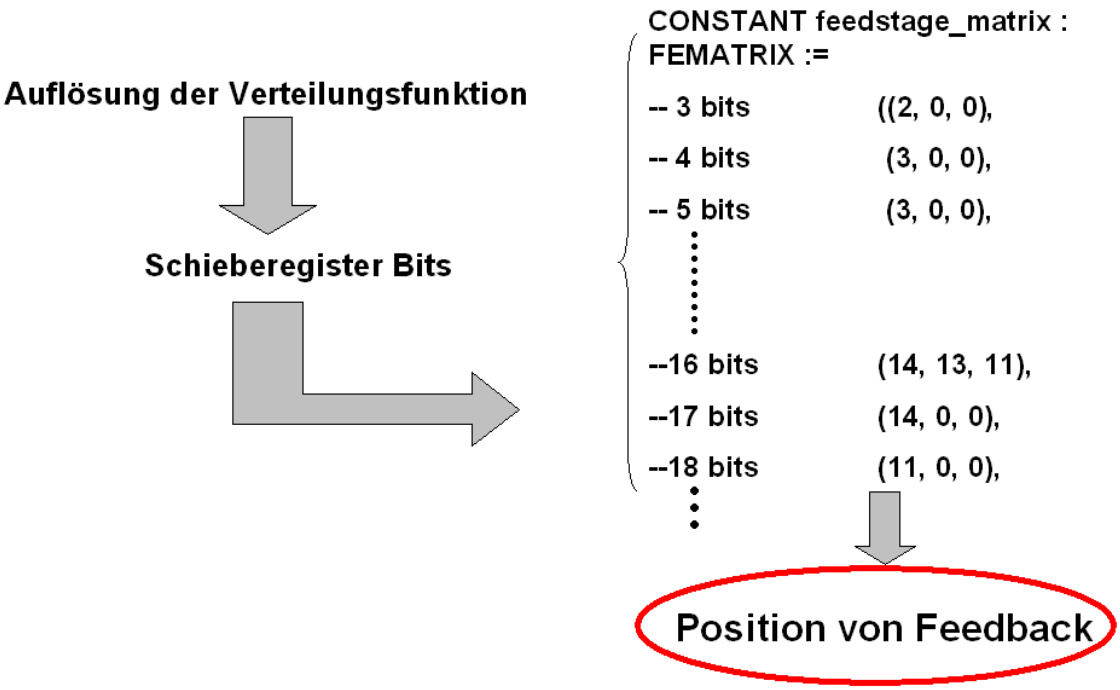


Abbildung 6.5: Darstellung der M-Sequenz Bildung

Die gleichverteilten Zufallszahlen werden aus den ersten 'DISTR_BITS' der M-Sequenz gebildet. Damit die gleichverteilten Zufallszahlen sich nicht zu schnell wiederholen, wird normalerweise eine sehr hohe Breite für die LFSR definiert, hier z.B. 40 Bits. Die Schieberegister werden nicht angehalten, sondern permanent mit dem Systemtakt getaktet. Immer wenn es nötig ist, wird eine Zufallszahl ausgelesen. Da dies an zufälligen Zeitpunkten geschieht (nach Ablauf eines Impulsabstandes oder Impulsmusterabstandes) wiederholen sich die Zufallszahlen nicht mit der gleichen Periode wie die der Sequenz des Schieberegisters. Die Periodenlänge ist nicht genau bestimmbar, aber in der Regel viel größer als $2^N - 1$ Taktzyklen. Um eine Periodenlänge von etwa $2^N - 1$ zu erreichen, müsste, wenn sich die Schieberegister Zahlen wiederholen, die gleiche Zahl wie am Anfang aus dem Schieberegister gezogen werden. Die Wahrscheinlichkeit hierfür ist jedoch sehr klein.

Die Wiederholzeit der Schieberegister Zahlen bei Bitbreite N und 100MHz Taktfrequenz ist in Tabelle 6.3 dargestellt.

N	$2^N - 1$	T
10	1023	10.2 μs
20	1.048.575	10.5 ms
30	1.073.741.823	10.7 s
40	1.099.511.627.775	3.1 h
50	1.125.899.906.842.623	130.3 d

Tabelle 6.3: Periodendauer des Schieberegisters bei Bitbreite N und Taktfrequenz 100MHz

6.2.1.2 Komparator und lineare Suche

Komparator mit Pipelining

In Quartus gibt es einen vordefinierten Komparator: Lpm_Compare. Dieser unterstützt Pipelining, d.h. die Ergebnisse stehen n Takte später am Ausgang zur Verfügung. ('n' ist die Anzahl der Pipelinestufen). Pipelining kann nötig sein, um eine gewisse Taktfrequenz zu erreichen.

Funktionsweise der linearen Suche

Das Programm hat zu einer gegebenen gleichverteilten Zufallszahl den Bereich der Y-Werte der Verteilungsfunktion, in dem diese Zahl liegt, festzustellen. Lineare Suche bedeutet, dass in der Verteilung_Tabelle von dem ersten Y-Wert bis zum letzten Wert der Verteilungsfunktion nacheinander alle mit der gleichverteilten Zufallszahl verglichen werden. Das Programm benötigt dafür so viele Vergleiche wie die Verteilungsfunktion Werte besitzt.

Dabei gibt es folgende Varianten:

- Sollen die Y-Werte der Verteilungsfunktion von oben nach unten oder von unten nach oben mit der gleichverteilten Zufallszahl verglichen werden?
- Welcher Vergleichsoperator ist der Richtige? '$<$', '$\leq$', '$>$' oder '$\geq$'.

Abbildung 6.6 verdeutlicht die einzelnen Fälle.

Es gibt sechs Ereignisse X: 1, 2, 3, 4, 5 und 6. Die Wahrscheinlichkeit für jedes Ereignis ist P(X). Die Y-Werte der Verteilungsfunktion sind die Zahlen F(X). Die X-Werte der Verteilungsfunktion sind die Ereignisse X.

Es sind drei gleichverteilte Zufallszahlen 0.2, 0.5, 1 vorgegeben. '1' ist ein Extremfall des ganzen Bereiches, '0.5' liegt im Inneren eines Intervalls und '0.2' liegt genau an einer Grenze von drei Intervallen. Das Intervall zum Ereignis 3 besitzt die Breite Null, da das Ereignis unmöglich ist (P(3) = 0). Die Zufallszahlen sind so gewählt, um den Operator und die Vergleichsweise für alle Fälle zu untersuchen.

Das Intervall zum Ereignis 1 definiert sich zu (0;0,1] und zum Ereignis 2 zu (0,1;0,2] usw.

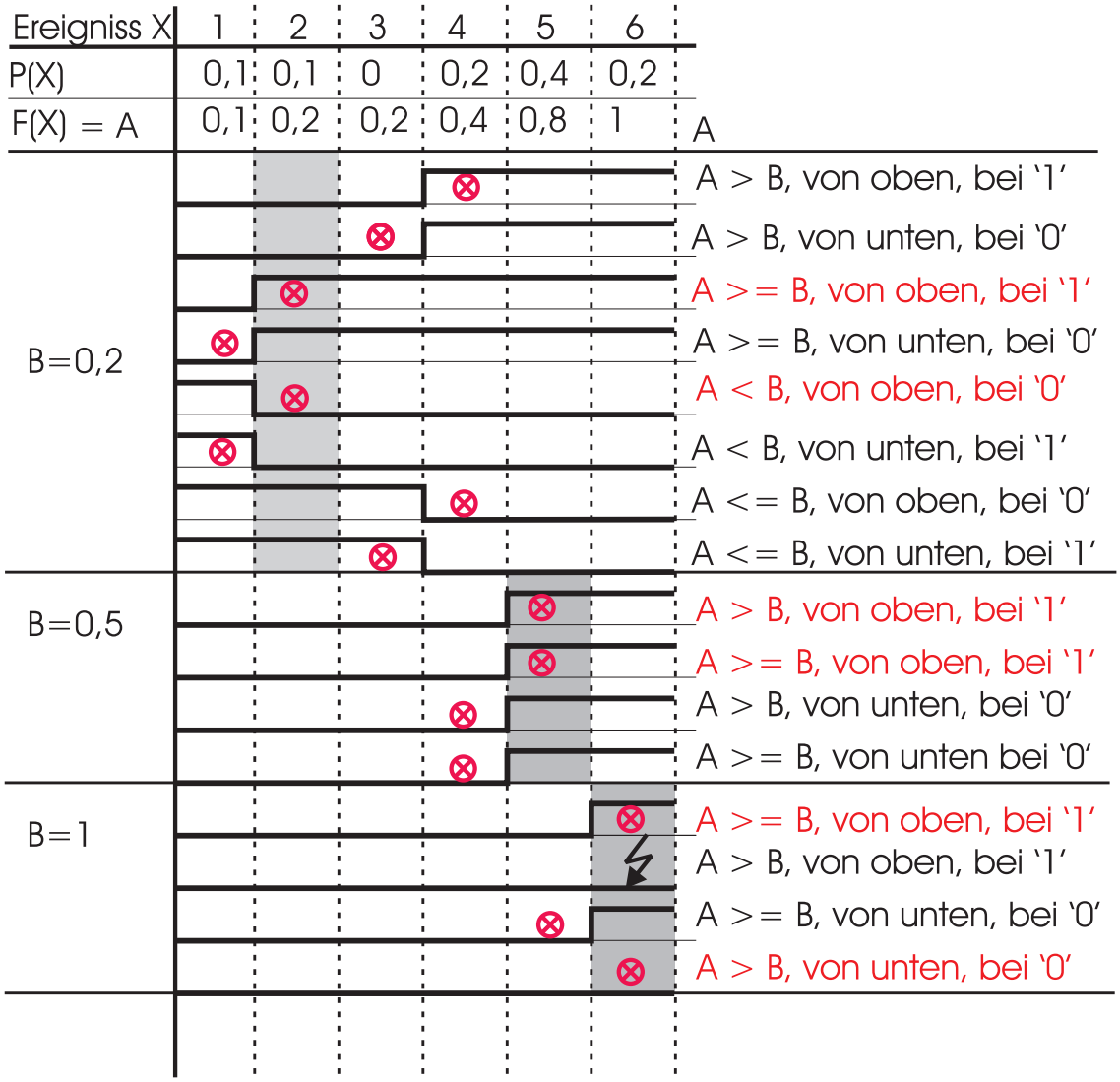


Abbildung 6.6: Beispiel für lineare Suche

Im Fall B=0,2 ist erkennbar, dass zwei Möglichkeiten jeweils äquivalent sind, also das gleiche Ergebnis liefern. Sie sind daher in den zwei anderen Fällen nicht dargestellt.

In jeder Zeile ist das Ausgangssignal des Komparators ohne Pipelining dargestellt. Dahinter ist der Vergleichsoperator und die Vergleichsrichtung angegeben. Das rote Kreuz zeigt das Endergebnis, welches das Verfahren liefern würde. Es liefert die Zufallszahl, welche die Verteilung_Tabelle erzeugen würde. Das Endergebnis ist das letzte Ereignis, vor dem Übergang von 1 nach 0 oder 0 nach 1. Letzteres hängt natürlich von der Vergleichsrichtung ab. Die graue Spalte zeigt das richtige Ergebnis zu der gegebenen Zufallszahl, wobei ein Verfahren, dass das richtige Ergebnis liefert in Rot gekennzeichnet ist. Ein Blitz bedeutet, dass kein Ereignis gefunden wird, das Programm wäre in einem undefinierten Zustand.

Alle drei Beispiele haben belegt, dass eine Vergleichsrichtung von oben nach unten mit dem Operator $\geq$ die richtigen Ergebnisse liefert. Hierbei macht '$\geq$' bei '1' oder '$<$' bei '0' keinen Unterschied, diese Verfahren sind äquivalent.

Extremfälle: Gleichverteilte Zufallszahl ist '0' oder '1'

Die Quantiltransformation untersagt die beiden Grenzen '0' und '1' für die gleichverteilte Zufallszahl. Hier in diesem Fall ist die Quantiltransformation erweitert. Da die X-Werte der Verteilungstabelle endlich sind, besitzt '1' als gleichverteilte Zufallszahl nun ebenfalls Gültigkeit.

'0' ist nicht erlaubt, da den Intervallen nicht zwei Ränder zugeordnet sind. Theoretisch tritt '0' bei einer mit einem Schieberegister erzeugten M-Sequenz nicht auf. Eine Folge die nur aus unteren Bits eines Schieberegisters besteht, kann sehr wohl eine '0' enthalten. Vermeidbar ist das dadurch, dass die binäre Zahl '00...00' zur binären Zahl '00...01' geändert wird, was der kleinsten positiven Festkommazahl entspricht. So wird die '0' einem definierten Intervall zugeordnet. Dies verändert die Wahrscheinlichkeiten nur unwesentlich.

Der Index bei linearer Suche

Jeder X-Wert bzw. der entsprechende Y-Werte der Verteilungsfunktion hat einen Index. Mit dem Index wird einfach der Vergleichsprozess gestartet und beendet. Nach jeder Taktinformation am Eingang der Verteilung_Tabelle ('takt_ran' nicht 'sys_clk'), wird der Index zu DISTRIBUTION'High (der höchste Index des Arrays DISTRIBUTION) gesetzt, und das Programm beginnt, die aus dem LFSR kommende gleichverteilte Zufallszahl mit dem letzten Y-Wert der Verteilungsfunktion zu vergleichen. Wenn der Index einen Wert von DISTRIBUTION'Low (der niedrigste Index des Arrays DISTRIBUTION) erhält, hört der Vergleichsprozess auf. Nach jedem Systemtakt, wird der Index um eins erniedrigt, und die gleichverteilte Zufallszahl wird mit dem nächsten Y-Wert der Verteilungsfunktion verglichen. Wenn das richtige Intervall zu der gleichverteilten Zufallszahl gefunden ist, springt das Ausgangssignal des Komparators auf einen anderen Wahrheitswert.

Der Pipeline Effekt wird durch den Index einfach korrigiert. Dazu wird ein Signal 'Index_pip' als Lesezeiger für das Ereignis verwendet. Aufgrund der Pipeline Einstellung des Komparators, ist Index_pip gegenüber Index um 3 Takte verzögert. Das Ereignis, das zu Index_pip gehört, ist das Ereignis was ausgegeben wird.

Die Stufenanzahl der Pipeline des Komparators ist im VHDL-Programm über die Konstante NPIPE einstellbar. Bei kleinen Bitbreiten der Verteilungsfunktion könnte die Stufenanzahl reduziert werden, worauf hier jedoch aus Gründen der Einfachheit verzichtet wurde. NPIPE ist durchgehend auf 3 gesetzt.

6.2.2 As_Impulse

(Hier nur für Struktur II erklärt (siehe Abbildung 6.2)).

Die As_Impulse Entity enthält zwei Verteilung_Tabelle Instanzen. Dort werden die aperiodischen Störimpulse mit den zwei Informationen von der Impulsdauer und dem Impulsabstand erzeugt. Der Anfang eines Impulses ist durch den Zeitablauf eines Impulsmusterabstandes oder Impulsabstandes festgelegt. Zusätzlich wird die Erzeugung von aperiodischen Störimpulsen durch das Impuls_Set Signal (Impulsmuster) ein- und ausgeschaltet.

Dabei entsteht folgende Problematik: Da die Verteilungstabelle die Zufallszahlen im Zeitraum vor dem Auslesen erzeugt, ist nach einem Reset noch keine Berechnung durchgeführt.

Das Programm sieht nun folgende Lösung vor: Beim ersten Takt nach einem Reset, wird der kleinste Impulsabstand und die kleinste Impulsdauer aus der Tabelle der Verteilungsfunktion gelesen. Dies ist kein Ergebnis der Quantiltransformation, sondern lediglich so definiert, weil beim Start des Programms kein vorher berechnetes Ergebnis vorliegt.

6.2.3 Kfz_Störimpulse

Diese Entity erzeugt Impuls_Set, das Muster des Kfz_Störimpuls-Signals mit den zwei Informationen der Impulsmusterdauer (stochastisch) und des Impulsmusterabstandes (konstant). Die Impulsmusterdauer ist durch eine Verteilung_Tabelle bestimmt. Der Anfang eines Impuls_Set ist durch den Zeitablauf eines Impulsmusterabstandes festgelegt, und das Signal schaltet die As_Impulse an und aus.

6.2.4 KfzNoisePack

In diesem Package werden alle Konstanten eingegeben: Die X-, Y-Werte der Verteilungsfunktionen, die Bitbreite der Werte, die Breite der Klassen sowie die Anfangswerte der Schieberegister.

6.3 Ergebnisse der Simulation

Hier sind die Simulationsergebnisse der verschiedenen VHDL Strukturen und Programme dargestellt.

Struktur I, As_Impulse Entity: nur aperiodische binäre Störimpulse

Simulation mit Quartus 4.0
FPGA: Stratix I, EP1S10F484C5
Taktfrequenz: 100 MHz
Simulationstyp: Timing

Die Signale in Abbildung 6.7 sind:

- asimpulse: Die aperiodischen Störimpulse ohne Impulsmuster.
- takt_ran: Die Taktimpulse, mit der ein aperiodischer Störimpuls anfängt. Mit diesem Takt werden die Zufallszahlen für die Impulsdauer und den Impulsabstand aus den Verteilung_Tabellen gelesen.

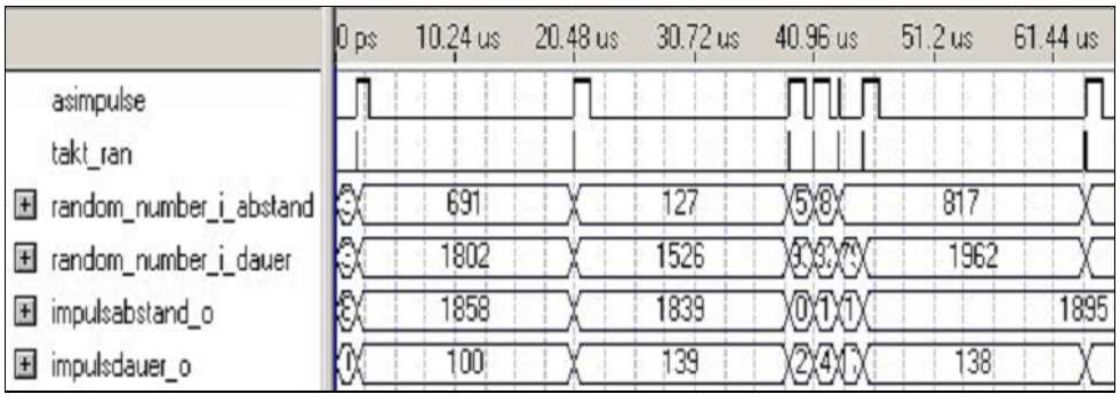


Abbildung 6.7: Simulationsergebnis der As_Impulse Entity, Struktur I

- random_number_i_dauer/_abstand: Die gleichverteilten Zufallszahlen, die nach der Taktinformation der Verteilung_Tabelle aus den Schieberegistern gelesen und mit den Y-Werten der Verteilungsfunktion verglichen werden. Dadurch werden die Zufallszahlen der Impulsdauer und des Impulsabstandes erzeugt.

- impulsdauer/impulsabstand_o: Die nach dem Vergleichen erzeugten Zufallszahlen. Diese beschreiben die Dauer und den Abstand der aperiodischen Störimpulse.

Struktur I, Kfz_Störimpulse Entity: vollständiges binäres Störimpulssignal

In Abbildung 6.8 sind die Simulationsergebnisse des Kfz_Störimpulse Entities der VHDL Struktur I dargestellt.

Simulation mit Quartus 4.0
FPGA: Stratix I, EP1S10F484C5
Taktfrequenz: 100 MHz
Simulationstyp: Timing

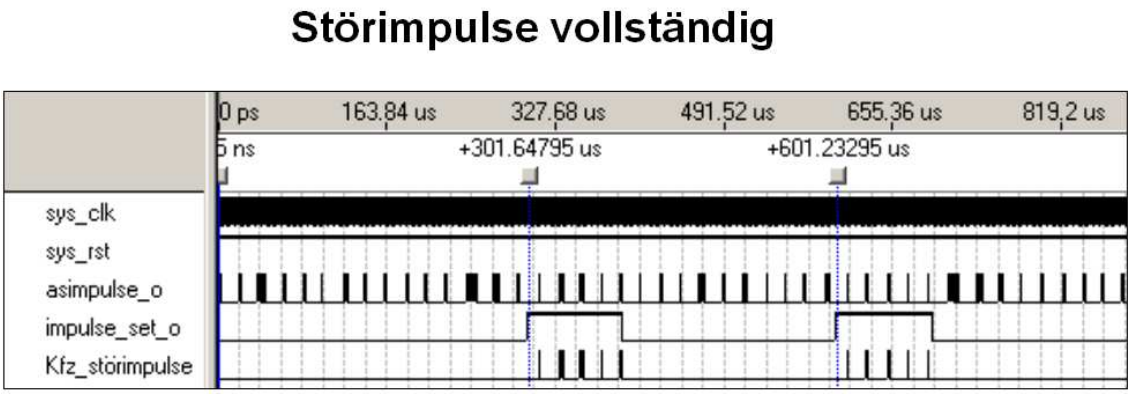


Abbildung 6.8: Simulationsergebnis der Kfz_Störimpulse Entity, Struktur I

Die Signale in Abbildung 6.8 sind:

- sys_clk: Taktsignal des Systems, die Periodendauer beträgt 10ns. Dieses Signal ist von dem Takt-Signal der Verteilung_Tabelle zu unterscheiden: das letztere ist der Takt, der den Anfang eines aperiodischen Störimpulses festlegt.
- sys_rst: Das Reset Signal des Systems. Für die Simulation wurde es zu Beginn auf '0' gesetzt, nach der Zeit von 5 sys_clk's ist es '1'.
- asimpulse_o: Die aperiodischen Störimpulse ohne Impulsmuster.
- impuls_set_o: Das Impulsmuster-Signal, mit konstantem Impulsmusterabstand und zufälliger Impulsmusterdauer.
- Kfz_störimpulse: Das Ausgangssignal des implementierten Programms.

Es ist sehr gut zu erkennen, dass die As_Impulse durch Impuls_Set an- und ausgeschaltet werden. Daher sind dies die aperiodischen Störimpulse mit dem Impulsmuster, also die Überlagerung des periodischen und aperiodischen Störimpulsmodells, was das Ausgangssignal der Kfz_Störimpulse Entity bildet. Hierbei ist auch zu erkennen, dass die Impulsmuster hier nicht immer mit einem Störimpuls beginnen.

Anmerkung: Hier ist aufgrund der Darstellung der Impulsmusterabstand künstlich kleiner gewählt.

Struktur II, Kfz_Störimpulse Entity: vollständiges Störimpulssignal mit Amplitudeninformation

Das Simulationsergebnis der Kfz_Störimpulse Entity der VHDL Struktur II (mit Amplitudeinformation) ist in Abbildung 6.9 dargestellt.

Simulation mit Quartus 4.0
FPGA: Cyclone, EP1C12Q240C6
Taktfrequenz: 100 MHz
Simulationstyp: Timing

Bei der Struktur II werden die aperiodischen Störimpulse nur erzeugt, wenn Impuls_Set gleich '1' ist. Deswegen ist das Ausgangssignal As_Impulse gleich dem Ausgangssignal der Kfz_Störimpulse Entity.

Die Werte in der Zeile Kfz_Störimpulse entsprechen den Amplitudenwerten für den D/A Wandler in 0,122mV Schritten. Die Impulsdauer geht aus der Zeitdauer der Intervalle hervor.

6.4 Besonderheit bei Verfahren II

Das VHDL Programm für das Verfahren I muss erweitert werden, um das Verfahren II zu realisieren:

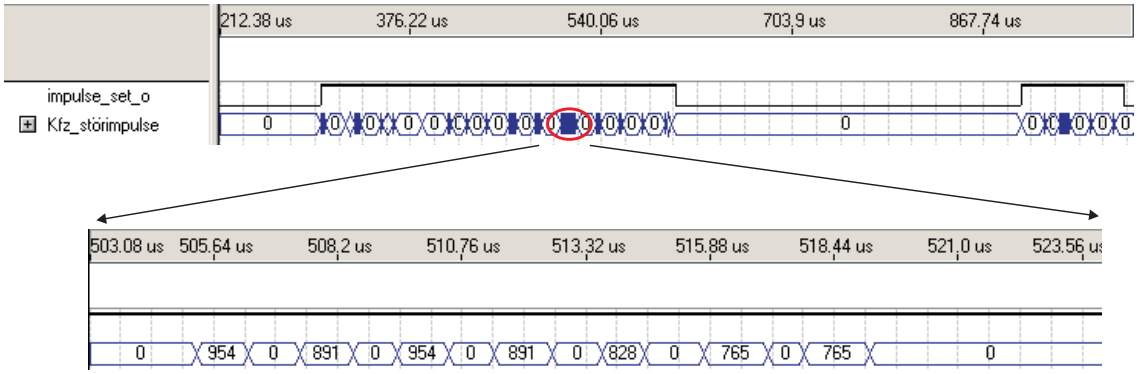


Abbildung 6.9: Simulationsergebnis der Kfz_Störimpulse Entity, Struktur II

Auf dem Bereich $(-\frac{D}{2}, \frac{D}{2}]$ gleichverteilte Zahlen müssen zu den erzeugten diskreten Zufallszahlen hinzu addiert werden. Genauer: falls D gerade ergibt sich der Bereich zu $[-\frac{D}{2}+1, \frac{D}{2}]$, falls D ungerade zu $[-\frac{D-1}{2}, \frac{D-1}{2}]$. Dies entspricht in beiden Fällen einer Breite von $D-1$. Um die Addition zu realisieren, wird die Entity Verteilung_Tabelle entsprechend modifiziert.

Mit einem Schieberegister können gleichverteilte Zufallszahlen nur im Bereich $[-2^{N-1}, 2^{N-1}-1]$ (signed) bzw. $[0, 2^N-1]$ (unsigned) erzeugt werden. Um Zufallszahlen Z im geforderten Bereich zu erzeugen, muss folgende Umrechnung erfolgen. (Die Breite des Bereiches der Schieberegister-Zufallszahlen ist 2^N-1). Hierbei wird '1' addiert, um die Grenzen korrekt zu bestimmen:

$$Z = RN\frac{D-1}{2^N-1}+1 = \frac{RN(D-1)+2^N-1}{2^N-1} \tag{6.1}$$

Ein Nachprüfen der Grenzen ergibt: (Die Division von Integerzahlen rundet nach $-\infty$)

$$\frac{-2^{N-1}(D-1)+2^N-1}{2^N-1} = -\frac{D-1}{2}\underbrace{-\frac{1}{2}\frac{D-1}{2^N-1}+1}_{\in(0,1)} = -\frac{D}{2}+1\underbrace{-\frac{1}{2}\frac{D-1}{2^N-1}+\frac{1}{2}}_{\in(0,1)} \tag{6.2}$$

$$\frac{(2^{N-1}-1)(D-1)+2^N-1}{2^N-1} = \frac{D-1}{2}\underbrace{-\frac{1}{2}\frac{D-1}{2^N-1}+1}_{\in(0,1)} = \frac{D}{2}\underbrace{-\frac{1}{2}\frac{D-1}{2^N-1}+\frac{1}{2}}_{\in(0,1)} \tag{6.3}$$

Es ist also ersichtlich, dass in den Fällen D gerade und D ungerade jeweils die richtigen Grenzen erzeugt werden.

Es ist vorteilhaft die Division mit 2^N-1 durch eine Division mit 2^N zu ersetzen. Diese Division wird mit einer Rechts-Schiebe Operation realisiert, wofür in VHDL eine Lpm_Clshift Entity zur Verfügung steht. Für die Multiplikation wird eine Lpm_Mult Entity benutzt.

$$\begin{aligned} Z &= \frac{RN(D-1)+2^N-1}{2^N-1} = \frac{(RN(D-1)+2^N-1)\frac{2^N}{2^N-1}}{2^N} \\ &= \frac{RN(D-1)\frac{2^N}{2^N-1}+2^N}{2^N} = \frac{RN(D-1)(1+\frac{1}{2^N-1})+2^N}{2^N} \end{aligned} \tag{6.4}$$

Das VHDL Programm berechnet nun:

$$Z' = \frac{RN(D-1)+2^N}{2^N} \tag{6.5}$$

Wenn die Approximation zu ungenau ist (bei kleinen N), könnte auch folgendes verwendet werden:

$$Z'' = \frac{RN(D-1)(1+\frac{1}{2^N})+2^N}{2^N} = \frac{RN(D-1)(2^N+1)+2^{2N}}{2^{2N}} \tag{6.6}$$

Formel (6.5) entspricht Formel (6.4) mit einer Klassenbreite $D'-1 = (D-1)(1+\frac{1}{2^N-1})$. Die Klassenbreite minus 1 ist also um den Faktor $(1+\frac{1}{2^N-1})$ zu klein. Somit kommt es dazu, dass die Randwerte des Bereiches weniger wahrscheinlich als die Werte dazwischen sind. Der Fehler wird um so geringer, je größer N ist. Eine große Breite des Schieberegisters ist also von Vorteil, wenn wirklich gleichverteilte Zufallszahlen erzeugt werden sollen. In der Realisierung des Programms wurde $N = 39$ gewählt, wobei die Breite $D = 565$ Taktzyklen bei der Impulsmusterdauer ist. ($565 \hat{=} 10$ Bit) Hierbei hat die Impulsmusterdauer die größte Klassenbreite von allen Parametern.

Folgendes muss noch zusätzlich beachtet werden: Die Werte von RN unterscheiden sich immer um mindestens 1. Damit für Z alle Zahlen erzeugt werden können und auch wirklich gleichverteilt sind, muss $\frac{D-1}{2^N} << 1$ gelten. Dies bedeutet, dass die Breite des Schieberegisters viel größer als die Bitanzahl der Klassenbreite sein muss.

Zusammenfassend kann man sagen, dass das Schieberegister so groß wie möglich sein sollte.

Da das Programm zur Multiplikation und Berechnung des Shifts so viele Taktzyklen zur Verfügung hat, wie die Vergleichszeit beträgt, ergibt sich im vorliegenden Fall auch kein Timing Problem. Die Vergleichszeit ist gleich der Werteanzahl der Verteilungsfunktion plus die Verzögerungszeit durch die Pipeline, und beträgt hier mindestens 45.

Die Entity der Verteilung_Tabelle des Verfahrens II ist in Abbildung 6.10 zu sehen:

Die Entity der Verteilung_Tabelle für das Verfahren I wird in Abschnitt 6.2.1 erklärt, hier werden nur die Variablen beschrieben, die in Verfahren II gegenüber denen in Verfahren I abweichen.

- KLASSENBREITE: Klassenbreite des Histogramms in Taktzyklen. Diese ergibt sich aus dem Maximum der Messdaten dividiert durch die Klassenanzahl.
- KLASSENBREITE_BITS: Anzahl der Bits der Klassenbreite. z.B. KLASSENBREITE $= 54 \Rightarrow$ KLASSENBREITE_BITS $= 6$.
- KLASSENBREITE_RNDRSTVAL: Dies legt neben dem Startwert des Schieberegisters auch die Bitbreite des Schieberegisters fest und somit die Breite der gleichverteilten Zufallszahlen RN. Diese muss nach der vorherigen Theorie möglichst groß sein. Hierbei ist die Breite von RN ein Bit kleiner als die Breite des Schieberegisters, damit für RN auch Null erzeugt werden kann.

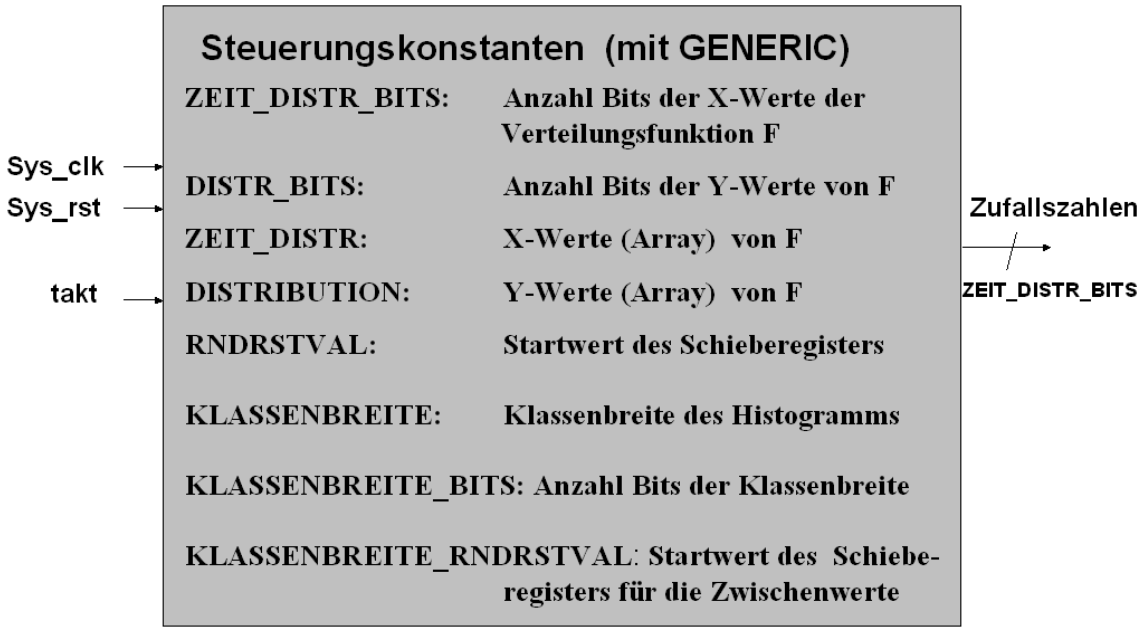


Abbildung 6.10: Die Entity Verteilung_Tabelle für das Verfahren II

Simulation mit Quartus 4.0
FPGA: Cyclone, EP1C12Q240C6
Taktfrequenz: 100 MHz
Simulationstyp: Timing

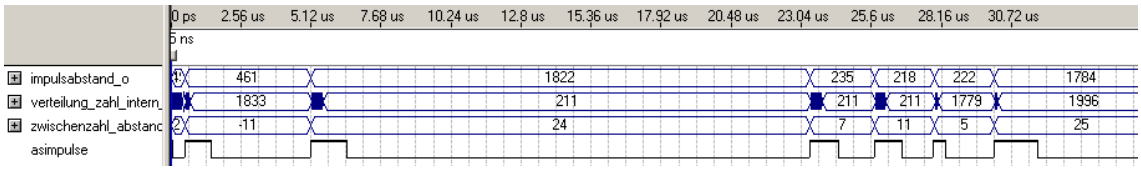


Abbildung 6.11: Simulationsergebnis der As_Impulse Entity, Verfahren II (binäres Modell)

Abbildung 6.12: Simulationsergebnis der Kfz_Störimpulse Entity, Verfahren II (binäres Modell)

Die Flow-Summary des Kompiliervorgangs des Programms für Verfahren II ist in Tabelle 6.4 dargestellt.

In Abbildung 6.11 ist ein Teilausschnitt des Signals der aperiodischen Störimpulse gezeigt. Um die Funktionsweise des Verfahrens zu verdeutlichen, sind zusätzlich einige Signale aus dem Inneren der Entity dargestellt. Die Klassenbreite bei dem Impulsabstand beträgt 54 Taktzyklen.

Top-level Entity Name	kfz_stoerimpulse
Family	Cyclone
Device	EP1C12Q240C6
Total logic elements	1,382 / 12,060 (11 %)
Total pins	22 / 173 (12 %)
Total memory bits	0 / 239,616 (0 %)
Total PLLs	0 / 2 (0 %)

Tabelle 6.4: Flow-Summary zum Programm zu Verfahren II

- impulsabstand_o: Der Abstand zwischen zwei Störimpulsen in Taktzyklen. Dies ist das Ausgangssignal der Verteilung_Tabelle in der As_Impulse Entity, der die Zufallszahlen für die Impulsabstände erzeugt. Dieser Wert ist gleich verteilung_zahl_intern_abstand + zwischenzahl_abstand. (Erläuterung siehe nachfolgend.)
- verteilung_zahl_intern_abstand: Dies ist das Signal, dass nur aus den Mittelpunkten der Histogrammklassen besteht (siehe Verfahren I).
- zwischenzahl_abstand: Die auf dem Intervall $[-26, 27]$ gleichverteilte Zufallszahl. Sie entspricht der Zahl Z' in der Formel 6.5 mit D = 54.

In Abbildung 6.12 ist das Simulationsergebnis der Kfz_Störimpulse Entity dargestellt. Dies ist die Überlagerung des aperiodischen mit dem periodischen Anteil (Impulsmuster) und bildet somit das Ausgangssignal des Störimpulsmodells.

6.5 Erhöhung der Taktfrequenz

Bei einer großen Werteanzahl kann es problematisch sein, die Taktfrequenz über ein bestimmte Grenze hinaus zu erhöhen. Es werden sehr viele Logikzellen zum Speichern der Werte benötigt, die alle in einem Takt durchlaufen werden müssen. Daher kann sich ein Timing-Problem ergeben.

Um diesem Abhilfe zu schaffen, ist es möglich den Systemtakt (sys_clk) in dem gesamten VHDL Programm zu teilen. Dazu müssen keine Werte der Verteilungsfunktionen neu berechnet werden, es ergibt sich aber keine höhere Zeitauflösung.

Alternativ ist es möglich, nur den Systemtakt in der Entity Verteilung_Tabelle zu halbieren. Dann werden die Vergleichsschritte entsprechend langsamer durchgeführt. Dazu müssen die Werte der Verteilungsfunktionen mit einer angepassten Randbedingung 3 (siehe Abschnitt 5.1.1) und geänderter Taktfrequenz neu berechnet werden. Die Vergleichszeit ist nun nicht mehr gleich der Werteanzahl, sondern doppelt/dreifach so groß (entsprechend dem Takt-Teilverhältnis).

Trotz der geänderten Randbedingung ergibt sich kein schlechteres Ergebnis als mit 100 MHz, da auch jeweils der 'kleinste Wert' angepasst wird, aufgrund der Skalierung in Taktzyklen. Es kann

hierbei nützlich sein, eine binäre Suche statt der linearen Suche zu verwenden (siehe Kapitel 7).

Kapitel 7

Zusammenfassung und Ausblick

Ziel dieser Arbeit ist die Nachbildung eines Störimpulsmodells (einer Überlagerung eines periodischen und aperiodischen Störimpulsmodells) für einen PLC Kanalemulator eines Kraftfahrzeugs. Dazu ergibt sich für die Messdaten (1000-3000 Stichproben) der Impulsdauer, des Impulsabstandes sowie der Impulsmusterdauer eine gewisse Beschränkung:

- Die Korrelation zwischen der Impulsmusterdauer und den darin enthaltenen aperiodischen Störimpulsen ist unbekannt.
- Der Impulsmusterabstand wurde nicht gemessen, stattdessen als konstanter Wert angenommen (in der Realität ist dieser drehzahlabhängig).

Trotz allem wird versucht, mittels vorhandener Messdaten, ein Störimpulsmodell nachzubilden. Fast alle Parameter des Störimpulsmodells sind alles stochastische Größen. Es musste folglich ein Verfahren entwickelt werden, das Zufallszahlen mit beliebiger Dichtefunktion erzeugen kann.

Es ist möglich alle Messdaten in das Programm einzugeben und das Programm eine Zahl davon zufällig auswählen zu lassen. Dies bildet die Messdaten zwar genau nach, benötigt aber viel Speicherplatz.

Die zweite Möglichkeit ist die Dichte der Zufallszahlen als eine Standard- Dichtefunktion anzunehmen und mit speziellen Verfahren die Standard- Dichtefunktion nachzubilden, z.B. mittels des Markov Modells, oder eines Schieberegisters. Dies ermöglicht aber nicht die Bildung beliebig verteilter Zufallszahlen wie sie hier benötigt werden.

Ein Kern-Schätzer kann nun die Dichtefunktion als Summe von verschiedenen Standard- Dichtefunktionen zerlegen. Dieses Verfahren ist zwar für beliebig verteilte Dichtefunktionen anwendbar, aber aufgrund der komplizierten Berechnung bei der Generation von Zufallszahlen ungeeignet für die echtzeitfähige Simulation in einem FPGA.

Das Verfahren, das in dieser Arbeit entwickelt wurde, basiert auf Histogrammen und der Quantiltransformation. Der Ablauf der verschiedenen Schritte von den Messdaten zur Erzeugung von Zufallszahlen ist in Abbildung 7.1 dargestellt.

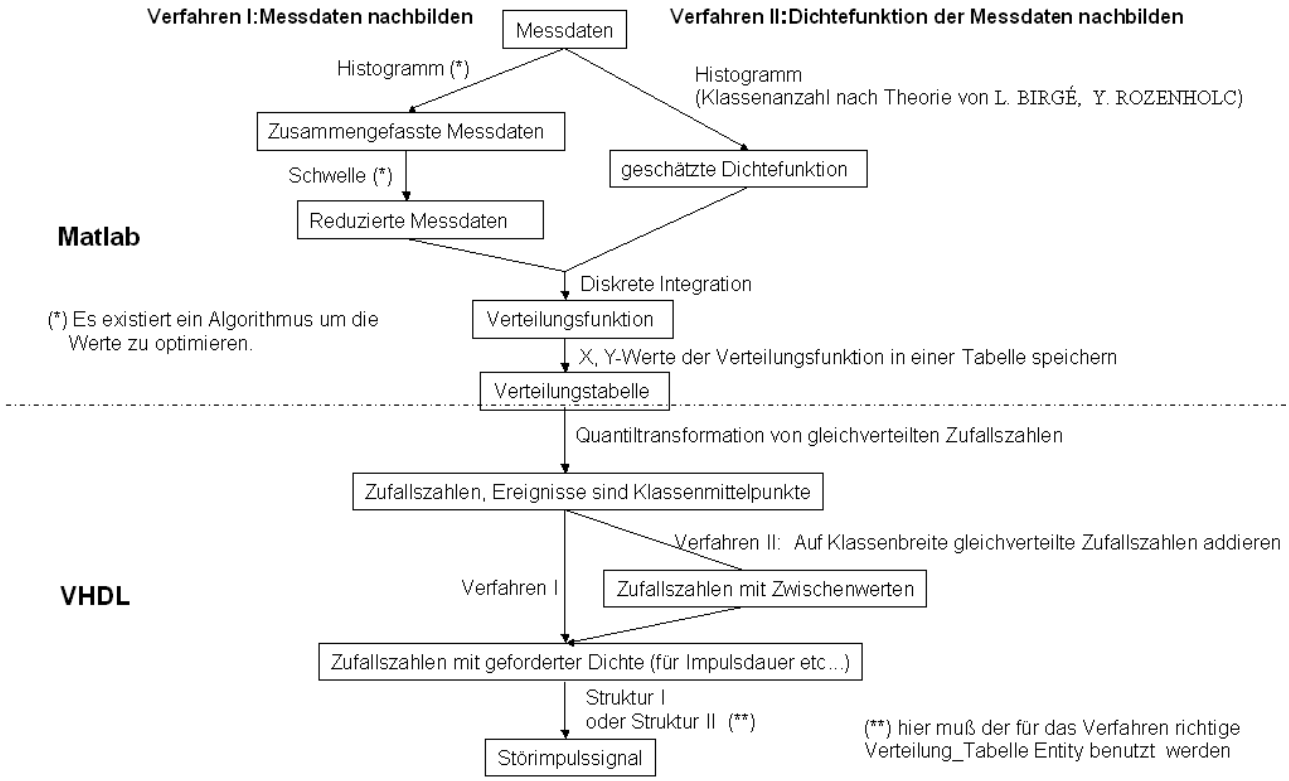


Abbildung 7.1: Überblick über Verfahren I und Verfahren II

Das Histogramm wird benutzt, um die Dichtefunktion zu bilden. Aus der Dichtefunktion berechnet sich die Verteilungsfunktion. Die X- und Y- Werte der Verteilungsfunktion werden in einer Tabelle gespeichert, in VHDL weiter verarbeitet und eine Quantiltransformation gebildet. Die daraus erzeugten Zufallszahlen sind eine Nachbildung der stochastischen Größen des Störimpulsmodells. Deswegen wurde dieses Verfahren Verteilungstabelle genannt.

Wenn nicht genügend Messdaten vorhanden sind, um die dahinterliegende Dichtefunktion zu schätzen, können die Messdaten als solche nachgebildet werden.

Dafür wird ein Histogramm benötigt, um die Messdaten zusammenzufassen. Zusätzlich wird unter Umständen auch eine Schwelle hinzugezogen, um die Werteanzahl weiter zu reduzieren. Dazu steht ein Algorithmus zur Verfügung: Er bestimmt die optimale Klassenanzahl und den Schwellwert. Nach der Bildung des Histogramms und Anwendung der Schwelle wird der Mittelwert einer Klasse als Ereignis dieses Intervalls ausgewählt. Es ergibt sich eine diskrete Dichtefunktion bzw. treppenförmige Verteilungsfunktion. Im FPGA wird dann eine Quantiltransformation von gleichverteilten Zufallszahlen für eine diskrete Dichtefunktion realisiert.

Wenn eine hinreichende Anzahl an Messdaten vorhanden ist, ist es möglich daraus die richtige Dichtefunktion mit einem Histogramm zu schätzen. In diesem Fall ist das Verfahren I ungünstig, weil die Anzahl der Ereignisse sehr groß sein kann. Für die optimale Klassenanzahl einer Histogramm Dichteschätzung wurde das Kriterium von Birgé und Rozenholc vorgestellt. Die optimale Klassenanzahl ist viel kleiner im Vergleich zu der, des ersten Verfahrens. Werden die Mittelwerte einer Klasse als Ereignis einer diskreten Dichtefunktion gespeichert, ergibt sich wie-

der die Verteilungstabelle des ersten Verfahrens. Diese beinhaltet allerdings viel weniger Werte bei gleicher Messdatenanzahl. Jedoch ist Verfahren II in der hier gezeigten Form nur für ein binäres Störimpulsmodell geeignet. Durch die Quantiltransformation mit einer treppenförmigen Verteilungsfunktion berechnen sich die Zufallszahlen des Verfahren I. Um Zufallszahlen aus der Histogramm Dichteschätzung zu erhalten, werden zu den Zahlen des Verfahren I gleichverteilte Zufallszahlen auf dem Intervall $(-\frac{D}{2}, \frac{D}{2}]$ addiert (D: Klassenbreite des Histogramms). Somit besteht die Möglichkeit Zwischenwerte von den Ereignissen, die durch das Verfahren I geliefert werden, zu erzeugen. Dies ist der Grund, warum weniger Werte bei gleicher Messdatenanzahl in der Verteilungstabelle gespeichert werden müssen.

Die Quantiltransformation im FPGA verläuft wie folgt: Es werden die Y-Werte der Verteilungstabelle mit gleichverteilten Zufallszahlen verglichen. Dabei wird eine lineare Suche mit 3 Pipelinestufen des Komparators verwendet, wobei die Pipeline Stufenanzahl einstellbar ist. Ist das richtige Intervall gefunden, wird der X-Wert des Intervalls als Zufallszahl am Ausgang der Verteilungstabelle ausgegeben. Mit den so erhaltenen Zufallszahlen werden die Signale für die aperiodischen Störimpulse, sowie für das Impulsmustersignal erzeugt. Daraus resultiert das Kfz_Störimpulse-Signal am Ausgang des FPGAs.

Ausblick

Die Störimpulse, die hier im FPGA erzeugt werden, sollen anschließend in einen Bereich zwischen 170-230MHz hochgemischt werden. Aufgrund der begrenzten Taktfrequenz eines FPGAs erfolgt dies im analogen Teil der Schaltung.

Des weiteren wird im FPGA ein weißes Rauschen mit variablem SNR (Signal-to-Noise-Ratio) erzeugt. Ein FIR-Filter zur Kanalnachbildung ist nur vereinfacht nötig, da der Kanal in diesem Bereich recht konstant ist, es genügt folglich ein einfaches Dämpfungsglied.

Fortsetzung der VHDL-Beschreibung

Zur Zeit wird eine lineare Suche für den Vergleich eingesetzt, d.h. es werden so viele Vergleichsschritte, wie Werte in der Verteilungstabelle sind, benötigt.

Alternativ ist es möglich eine binäre Suche zu verwenden: Dazu wird die Datenmenge (Y-Werte der Verteilungsfunktion) immer in zwei Hälften unterteilt. Die gleichverteilte Zufallszahl wird mit dem mittleren Wert der Datenmenge verglichen: ist die Zahl kleiner als der mittlere Wert, dann wird sie der unteren Hälfte zugeteilt und weiter mit dem mittleren Wert der unteren Hälfte verglichen. Wenn die gleichverteilte Zahl größer als der Mittelwert der Datenmenge ist, entsprechend umgekehrt.

Mit diesem Verfahren werden zum Vergleichen theoretisch nur $\log_2 n$ Schritte benötigt. (n ist die Werteanzahl der Verteilungsfunktion)

Die Berechnungsdauer der Zufallszahlen entsteht durch die Vergleichszeit und die Pipeline. Wenn die Vergleichszeit niedriger wird, wird auch die Erzeugungszeit der Zufallszahlen reduziert. Die Grenze für den kleinsten Zeitabstand zwischen der Erzeugung von zwei Zufallszahlen

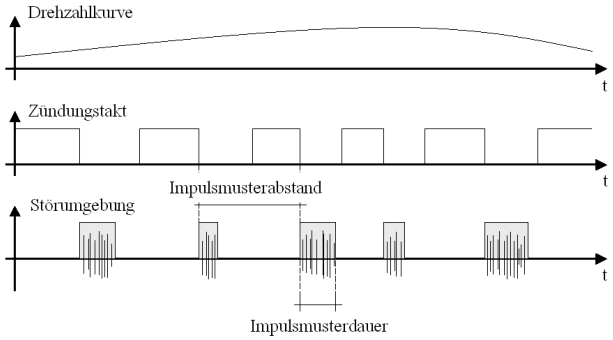


Abbildung 7.2: Variabler Impulsmusterabstand

wird sich nach unten schieben, und weniger Ereignisse müssen eliminiert werden. (siehe Abschnitt 5.1.3) Dies ermöglicht es viel mehr Werte im FPGA zu speichern, und so die Qualität der Nachbildung zu erhöhen.

In dieser Arbeit wird der Impulsmusterabstand als konstant angenommen. Dies ist nicht realistisch, da sich die Drehzahl des Motors mit der Zeit ändert. In Abbildung 7.2 veranschaulicht dies in einem kleinen Beispiel.

Weiterentwicklung des Verfahrens

Mit dem Verteilungstabelle-Verfahren ist es möglich beliebige bekannte Dichtefunktionen nachzubilden. Es müsste nicht immer der Umweg über eine Dichteschätzung mit Histogramm gegangen werden, insbesondere dann nicht, wenn die Dichte schon bekannt ist. Dazu muss die Dichtefunktion stufenförmig umgeformt werden (Bildung einer Histogramm-Einhüllenden). Zufallszahlen werden dann mit dem zweiten Verfahren der Verteilungstabelle gebildet.

Weiter wäre es sinnvoll, Verfahren II auf zweidimensionale Dichtefunktionen zu erweitern, um korrelierte Zufallsgrößen behandeln zu können.

Im allgemeinen gesehen, ist das Verfahren Verteilungstabelle ein sehr flexibles Verfahren: Es überwindet den Nachteil des Markov-Modells und der Kern-Schätz Methode. Weiterhin ist es in Echtzeit in einem FPGA realisierbar. Die erzeugten Störimpulse werden in einem Kanalemulator implementiert, um PLC-Transceiver zu testen: Es ist möglich herauszufinden, was für Kanalkodierungen benötigt werden, und ob DBPSK als Modulationsverfahren hier wirklich geeignet ist.

Powerline Kommunikation im Kraftfahrzeug ist ein sehr interessantes Thema, dass durch Emulation des Kanals, die Anpassung der Transceiver an das Bordnetz ermöglicht. Das Thema dieser Studienarbeit, ist einer der Grundbausteine des Kanalemulators. Dabei handelt es sich auch um ein neues Verfahren für die Generation beliebig verteilter stochastischer Ereignisse auf Basis einer Verteilungstabelle in echtzeitfähiger Hardware.

Anhang A

Anhang A

Die Matlabfunktionen

A.1 Beispiele

A.1.1 Verfahren I: Bestimmen der Parameter für Impulsabstand

1. **Workspace messdaten.mat in Matlab öffnen.**

   Die folgenden Variablen erscheinen:

   ab100_dau : Messdaten zu Impulsdauer

   ab100_amp : Messdaten zu Impulsamplitude (gehören zu 'ab100_dau')

   ab_haeuf_abs : Messdaten zu Impulsabstand

   ab_haeuf_pl : Messdaten zu Impulsmusterdauer

2. ***1D_Dichte/maximumW.m* öffnen und folgende Zeilen einstellen**:

daten = ab_haeuf_abs/10e-9;	In Taktzyklen umrechnen
maxW = [150 160 170 180 190];	Hier den Bereich einstellen, zuerst grob und dann feinere Auswahl der Zahlen. Ist nur eine Zahl vorgegeben, hier nur ein Element einfügen.
D=[1:1000];	1 bis max(daten) (Auflösung kann gröber gewählt werden, wenn der Bereich groß ist)
maxX = maxW+3;	3 ist Pipelinestufenanzahl. Bei den Daten außer für Impulsabstand ist *maxX = zeros(length(maxW));*

3. ***1D_Dichte/maximumW.m* ausführen**

   Es ist sinnvoll, das Array 'maxW' nicht zu groß zu machen, da die Berechnung sonst zu lange dauert. Besser die Auflösung in 'maxW' zuerst grob wählen und nach einem ersten Durchlauf, also wenn eine Tendenz erkennbar ist, 'maxW' entsprechend variieren.

4. **Optional: Durch den erscheinenden Plot des Gesamtfehlers (-> Zoom) den neuen Bereich für 'maxW' für einen zweiten (genaueren) Durchlauf bestimmen.**

5. **Das optimale 'maxW' wird so bestimmt: Durch Eingabe im Matlab Command Window können die gesuchten Parameter ausgegeben werden:**

   [m,I] = min(Sumerror_min);
   maxWopt = maxW(I)

KlassenanzahlOpt = Dopt(I)
SchwellwertOpt = schwelleOpt(I)

6. ***Werte_ bestimmen/impulsabstand.m* öffnen und folgende Zeilen hinzufügen:**

   maxX= maxWopt+3; — Dies wird nur bei dem Impulsabstand benötigt, bei allen anderen wird 'maxX' = 0 gesetzt. 3 ist die Anzahl Pipelinestufen.

   klassenanzahl = KlassenanzahlOpt;
   schwelle = SchwellwertOpt;

   taktzeit = 10e-9;
   [bitaufloesung, zeit_distr, distribution]= verteilung_tabelle(daten, maxX, klassenanzahl, schwelle, taktzeit);

7. **Auf dem Bildschirm werden die X- und Y-Werte der Verteilungsfunktion ausgegeben, und können in das VHDL Programm in das Package KfzNoisePack eingefügt werden.**

A.1.2 Verfahren II: Bestimmen der Parameter für Impulsabstand

1. **Workspace messdaten.mat in Matlab öffnen.**

2. ***histo/D.m* öffnen und folgende Zeilen einstellen:**

   daten = ab_haeuf_abs/10e-9; — In Taktzyklen umrechnen

3. ***histo/D.m* ausführen**

   Die optimale Histogrammklassenanzahl 'Dopt' und die sich daraus ergebende Werteanzahl 'WZ' werden auf dem Bildschirm ausgegeben.

4. ***Werte_ bestimmen/impulsabstand.m* öffnen und folgende Zeilen hinzufügen:**

   maxX= WZ+3; — Dies wird nur bei Impulsabstand benötigt, bei allen anderen wird maxX = 0 gesetzt. 3 ist die Anzahl Pipelinestufen.

   klassenanzahl = Dopt;
   schwelle = 0; — Ist bei Verfahren II immer 0.
   taktzeit = 10e-9;
   [bitaufloesung, zeit_distr, distribution]= verteilung_tabelle(daten, maxX, klassenanzahl, schwelle, taktzeit);

5. **Auf dem Bildschirm werden die X- und Y-Werte der Verteilungsfunktion ausgegeben, und können in das VHDL Programm in das Package KfzNoisePack eingefügt werden.**

A.2 Matlab m-Files

Die Matlab Files sind in verschiedenen Unterverzeichnissen:

1D-Dichte: Die m-Files zur Berechnung der optimalen Histogrammklassenanzahl und optimale Schwelle für Verfahren I ('Messwerte nachbilden') mit einer eindimensionalen Dichtefunktion.

maximumW.m: Das Hauptprogramm, um das optimale 'maxW', den dazugehörigen optimalen Schwellwert und optimale Histogrammklassenanzahl zu bestimmen. Siehe Abschnitt A.1.1 für ein Beispiel.

2D-Dichte: wie oben, nur für eine zweidimensionale Dichtefunktion.

zwei_maximumW.m: Analog zu maximumW.m. Bestimmt jedoch zwei Histogrammklassenanzahlen: Eine für die Amplitude und eine für die Dauer.

histo: Bestimmung der optimalen Histogrammklassenanzahl für Verfahren II (Histogramm Dichteschätzung).

ln_pen.m: Funktion zur Berechnung von Formel 5.21 mit Formeln 5.23, 5.22 in Abhängigkeit der Messdaten und einer Histogrammklassenanzahl D.

D.m: Das Hauptprogramm: Wiederholtes Aufrufen von 'ln_pen.m' um die Histogrammklassenanzahl zu bestimmen, die Formel 5.21 minimiert.

Werte_bestimmen: m-Files zur Quantisierung und Ausgabe der Werte als Array, das mittels Kopieren in den VHDL Code eingefügt werden kann. Hierzu muss der vorher berechnete optimale Schwellwert bzw. die Histogrammklassenanzahl angegeben werden. Für Beispiele siehe Abschnitt A.1.

A.2.1 Verfahren I, eindimensionale Dichtefunktion

Die folgenden m-Files befinden sich im Verzeichnis *1D-Dichte*.

Berechnung des zeitlichen Fehlers

- *Zeit_abweichung.m:*

  Funktion: Die Summe des relativen zeitlichen Fehlerquadrats wird in Abhängigkeit der Klassenanzahl berechnet.

  Typ: Function

  Eingabe: daten: Messdaten
  D: Histogrammklassenanzahl

Ausgabe: Dif: Die Summe des relativen zeitlichen Fehlerquadrats

M: Der maximale Wert der Daten

- *zerror.m:*

Funktion: Wie Zeit_abweichung.m, aber für einen Vektor von Klassenanzahlen, nicht nur für einen einzelnen Wert.

Typ: Function

Eingabe: daten: Array Messdaten

D: Array mit Histogrammklassenanzahlen für die der zeitliche Fehler berechnet wird.

Ausgabe: Die Summe des relativen zeitlichen Fehler-Quadrats in Abhängigkeit der Klassenanzahlen in D.

Berechnung des Dichtefehlers

- *schwelle_ abweichung.m:*

Funktion: Dichtefehler in Abhängigkeit der Klassenanzahl und des Maximalwertes, der nach der Datenreduktion übrig bleiben soll. Wird nicht wie zwei_schwelle_abweichung.m rekursiv gemacht, weil eine Schwelle = 1 normalerweise ausreicht.

Typ: Function

Eingabe: daten: Messdaten

D: Histogrammklassenanzahl

maxW: Die maximale Werteanzahl, die übrig bleiben soll. Die Schwelle wird solange erhöht (bis maximal 2), bis die verbleibende Werteanzahl kleiner als 'maxW' ist.

maxX: Die kleinste Zufallszahl, die das Programm erzeugen soll. Ereignisse kleiner als dieser Wert werden weggelassen und zum Dichtefehler hinzugezählt.

Ausgabe: WZ: Werteanzahl, die übrig bleibt (ist kleiner gleich 'maxW').

Schwelle: benötigter Schwellwert

Serror0: Dichtefehler, nach dem Weglassen der Ereignisse, die kleiner als 'maxX' sind.

Serror1: Dichtefehler, nach dem Weglassen der Ereignisse kleiner als 'maxX' und der durch eine Schwelle von 1.

Serror2: Dichtefehler, nach dem Weglassen der Ereignisse kleiner als 'maxX' und der durch eine Schwelle von 2.

- *serror.m:*

Funktion: Berechnung der Summe des relativen Dichtefehler-Quadrats in Abhängigkeit der Klassenanzahl, 'maxW' und 'maxX'. Wie schwelle_abweichung.m aber Berechnung für mehrere Klassenanzahlen.

Typ: Function

Eingabe: daten: Messdaten

D: Array mit den Histogramm Klassenanzahlen

maxW: Die maximale Werteanzahl, die übrig bleiben soll.

maxX: Die kleinste Zufallszahl, die durch das Programm erzeugt werden soll.

Ausgabe: Werteanzahl: Array mit den Werteanzahlen, die übrig bleiben (die Elemente sind kleiner oder gleich 'maxW'), in Abhängigkeit der Klassenanzahlen in D.

schwelle: Array mit den Schwellwerten die nötig sind, die Ereignisanzahl auf kleiner 'maxW' zu reduzieren, in Abhängigkeit der Klassenanzahlen in D.

SerrorOUT: Array mit den relativen Dichtefehlern, das durch das Weglassen der Ereignisse entsteht. Das wird zu einen durch die Schwelle und zum anderen durch das Weglassen von Ereignissen kleiner 'maxX' verursacht.

Berechnung der optimalen Werte

- *sumerror.m*

Funktion: Die optimale Klassenanzahl und Schwelle bestimmen nach Vorgabe eines Wertes 'maxW' für die maximale Werteanzahl.

Typ: Function

Eingabe: daten: Messdaten

D: Array mit den Histogramm Klassenanzahlen, die ausprobiert werden.

maxW: Die maximale Werteanzahl, die übrig bleiben soll.

maxX: Die kleinste Zufallszahl, die durch das Programm erzeugt werden darf.

Dif: Die Summe des relativen zeitlichen Fehlerquadrats. Wird für die Anzeige eines Plots vom relativen zeitlichen Fehler und relativen Dichtefehler benötigt und zur Berechnung des Gesamtfehlers benötigt. Ist wie D ein Array und enthält den relativen zeitlichen Fehler für jede Klassenanzahl aus D. Wird normalerweise mit zerror.m berechnet.

zeige_plot: Wenn 1, zeigt ein Plot mit den beiden Fehlern.

Ausgabe: Dopt: Die optimale Klassenanzahl, die den kleinsten Gesamtfehler auslöst.

schwelleOpt: Der optimale Schwellwert, der zusammen mit der optimalen Histogrammklassenanzahl, unter der Voraussetzung der Werte für 'maxW' und 'maxX', den kleinsten Gesamtfehler verursacht.

Sumerror_min: Der kleinste Gesamtfehler.

- *maxW.m*

  Funktion: Programm zur Berechnung der optimalen Werte für Schwelle und Klassenanzahl bei Variation von 'maxW'. So kann man herausfinden, welches maxW das Beste ist (wenn es ein Bestes gibt). Das Beste maxW kann man durch Ablesen des minimalen Fehlers im Plot herausfinden. Alternativ ist es möglich alle optimalen Werte durch folgenden Programmtext zu bestimmen:
  [m,I] = min(Sumerror_min);
  maxWopt = maxW(I)
  KlassenanzahlOpt = Dopt(I)
  SchwellwertOpt = schwelleOpt(I)

  Typ: Script

  Eingabe: maxW: Der Bereich von 'maxW' (Array)
  D: Der Bereich der Klassenanzahlen (Array)
  daten: Das Array der Messdaten

  Ausgabe: Plots vom Gesamtfehler Klassenanzahl und in denen das optimale 'maxW' abgelesen werden kann.
  Dopt: (Array) optimale Klassenanzahlen bei dem vorgegebenen 'maxW'
  schwelleOpt: (Array) optimale Schwellwerte bei dem vorgegebenen 'maxW'
  Sumerror_min: (Array) Fehler bei der Klassenanzahl und Schwelle

  Uses: zerror.m, sumerror.m

A.2.2 Verfahren I, zweidimensionale Dichtefunktion

Die folgenden m-Files befinden sich im Verzeichnis *2D-Dichte*.

Berechnung des Dichtefehlers

- *zwei_schwelle_abweichung.m*

  Funktion: Die Summe des relativen Dichtefehlers berechnet in Abhängigkeit der Klassenanzahlen und 'maxW'. Es wird auch der nötige Schwellwert bestimmt, um die Werteanzahl unter 'maxW' zu bekommen.

  Typ: Function

  Eingabe: datend: (Array) Datenquelle1, hier für die Impulsdauer
  datena: (Array) Datenquelle2, hier für die Impulsamplitude
  D1: Klassenanzahl für die Datenquelle1
  D2: Klassenanzahl für die Datenquelle2

maxW: Maximale Werteanzahl, die nach der Datenreduktion übrig bleiben soll.

schwelle: Der Anfangs-Schwellwert (sollte 0 sein, wird nur intern benutzt)

Ausgabe: Serror: Die Summe des relativen Dichtefehlerquadrats, das durch durch die Schwelle erzeugt wurde.

SchwelleOpt: Der optimale Schwellwert, um die Werteanzahl unter maxW zu bekommen.

Berechnung der optimalen Werte

- *zweisumerror.m*

  Funktion: Berechnet die optimale Klassenanzahl und den optimalen Schwellwert für zweidimensionale Messdaten in Abhängigkeit von der Histogrammklassenanzahlen für die Daten.

  Typ: Function

  Eingabe: datend: (Array) Datenquelle1, hier für die Impulsdauer

  datena: (Array) Datenquelle2, hier für die Impulsamplitude

  D1: (Array) Klassenanzahlen für die Datenquelle1

  D2: (Array) Klassenanzahlen für die Datenquelle2

  maxW: maximale Werteanzahl, die nach der Datenreduktion übrig bleiben soll

  Ausgabe: D1opt: Die optimale Klassenanzahl für die Datenquelle1

  D2opt: Die optimale Klassenanzahl für die Datenquelle2

  Schwelleopt: Der zugehörige optimale Schwellwert

  Sumerror_min: Die zugehörige minimale Gesamtfehler

- *zweimaximumW.m*

  Funktion: Wie maximumW.m nur für den zweidimensionalen Fall

  Typ: Script

  Eingabe: maxW: (Array) Der Bereich von maxW

  datend: (Array) Messdaten von der Impulsdauer

  datena: (Array) Messdaten von der Impulsamplitude

  D1: (Array) Der Bereich der Histogrammklassenanzahl für die Impulsdauermessdaten: (von 1 bis max(daten)/Taktzeit FPGAs).

  D2: (Array) Der Bereich der Histogrammklassenanzahl für Impulsamplitudemessdaten: (von min(daten)/Referenzspannung des DA Wandlers bis max(daten)/Referenzspannung des DA Wandlers).

  Ausgabe: Plot mit minimalem Gesamtfehler und Histogrammklassenanzahl in Abhängigkeit von 'maxW' (Daraus kann man den optimalen Wert für 'maxW' ablesen).

SchwelleOpt: Optimale Schwellwerte in Abhängigkeit von 'maxW'

D1opt: Optimale Histogrammklassenanzahl für die Impulsdauermessdaten in Abhängigkeit von 'maxW'.

D2opt: Optimale Histogrammklassenanzahl für die Impulsamplitudemessdaten in Abhängigkeit von 'maxW'.

Sumerror_min: Minimale Summe des relativen Fehlerquadrats in Abhängigkeit von 'maxW'.

Uses: zwei_sumerror.m

A.2.3 Verfahren II

Die folgenden m-Files befinden sich im Verzeichnis *histo*.

- *ln_pen.m:*

  Funktion: Berechnung von $L_n(D) - pen(D)$, um durch Verfahren II die optimale Histogrammklassenanzahl zu bestimmen.

  Typ: Function

  Eingabe: daten: Array mit Messdaten

  D: Histogramm Klassenanzahl, für die $L_n(D) - pen(D)$ berechnet werden soll.

  Ausgabe: lnD: Wert von $L_n(D)$

  penD: Wert von $pen(D)$

  R: Die Differenz von $L_n(D$ und $pen(D)$

- *D.m:*

  Funktion: Aufrufen der ln_pen Funktion, um die optimale Histogrammklassenanzahl zu bestimmen.

  Typ: Script

A.2.4 Quantisierung und Ausgabe

Die folgenden m-Files befinden sich im Verzeichnis *Werte_bestimmen*.

- *verteilung_tabelle.m:*

  Funktion: Die X- und Y- Werte der Verteilungsfunktion berechnen, quantisieren und ausgeben. Für den eindimensionalen Fall.

  Typ: Function

Eingabe: daten: (Array) Die Quelldaten

maxX: Das kleinste Ereignis, das durch das Programm erzeugt werden soll. Ist in der Regel 0 oder 'maxW'+3. Ereignisse die in kleiner 'maxX' als sind, werden weggelassen.

schwelle: Der Schwellwert

Taktzeit: Die Taktzeit des FPGAs oder die Amplitudenauflösung des D/A Wandlers.

Ausgabe: Gibt die X- und Y- Werte der Verteilungsfunktion auf dem Bildschirm aus.

bitaufloesung: Die Bitauflösung für die Y-Werte der Verteilungsfunktion

zeit_distr: (Array) Die X-Werte der Verteilungsfunktion

distribution: (Array) Die Y-Werte der Verteilungsfunktion

- *zwei_dimensional_optimal.m:*

Funktion: Die X- und Y- Werte der Verteilungsfunktion berechnen, quantisieren und ausgeben. Für den zweidimensionalen Fall.

Typ: Function

Eingabe: datend: (Array) Datenquelle1, hier für die Impulsdauer

datena: (Array) Datenquelle2, hier für die Impulsamplitude

D1: Klassenanzahl für die Datenquelle1

D2: Klassenanzahl für die Datenquelle2

Schwelle: Der Schwellwert

Ausgabe: Gibt die X- und Y- Werte der Verteilungsfunktion auf dem Bildschirm aus. Die Y-Werte für Amplitude und Dauer werde separat und bitweise aneinandergehängt ausgegeben.

dau_distr: Die Bits, die für die Impulsdauer zuständig sind, von den X-Werten der Verteilungsfunktion.

amp_distr: Die Bits, die für die Impulsamplitude zuständig sind, von X-Werten der Verteilungsfunktion.

zeit_distr: Die X-Werte der Verteilungsfunktion (Bitweise dau_dist und amp_distr aneinandergehängt).

Distribution: Die Y-Werte der Verteilungsfunktion

- *Impulsdauer.m, Impulsabstand.m, Impulsmusterdauer.m:*

Funktion: Dies sind Skripte, welche die 'verteilung_tabelle' Funktion mit den richtigen Parametern aufrufen. Beim Aufrufen der Skripte werden die Werte für die Verteilungsfunktion am Bildschirm ausgegeben, diese können dann in das VHDL Programm eingefügt werden.
Die Parameter wurden mit den in der Studienarbeit beschriebenen Methoden

bestimmt. Sie sind für Verfahren I und Verfahren II gleichermaßen, man setzt dazu vor dem Ausführen die 'VerfahrenII' Variable in den m-Files auf 1 oder 0.

Typ: Script

- *Impulsdauer_ Impulsamplitude.m:*

  Funktion: Analog zu dem oben beschriebenen, jedoch für die zweidimensionale Verteilung von Impulsabstand und Impulsamplitude. Ruft 'zwei_dimensional_optimal' anstatt 'verteilung_tabelle' auf.

  Typ: Script

Anhang

Literaturverzeichnis

[Bos91] BOSCH, http://www.can.bosch.com/docu/can2spec.pdf: *CAN Specification*, September 1991.

[BR02] BIRGÉ and ROZENHOLC: *How many bins should be put in a regular histogram*, April 2002.

[Düm00] DÜMBGEN, LUTZ: *Empirische Prozesse*. Universität zu Lübeck, http://www.quantlet.com/mdstat/scripts/ep/EP.pdf, 2000.

[Göt04] GÖTZ, MATTHIAS: *Mikroelektronische, echtzeitfähige Emulation von Powerline-Kommunikationskanälen*. Mensch & Buch Verlag, ISBN 3-89820-672-6, 2004.

[Huc02] HUCK, THORSTEN: *Simulation der Datenübertragung im KFZ mittels Powerline Communications*. Diplomarbeit, Institut für Industrielle Informationstechnik (IIIT), Universität Karlsruhe, April 2002.

[Hyn95] HYNDMAN, ROB J: *The problem with sturges rule for constructing histograms*, July 1995.

[Kie97] KIENCKE, UWE: *Ereignisdiskrete Systeme*. R.Oldenbourg Verlag, ISBN 3-486-24150-8, München, 1997.

[Lin01] LINK, THOMAS: *Entwurf und Analyse neuer Bordnetzsysteme für Powerline-Kommunikation im Kfz*. Diplomarbeit, Institut für Industrielle Informationstechnik (IIIT), Universität Karlsruhe, 2001.

[Mat02] MATHWORKS, http://www.mathworks.com/access/helpdesk/help/toolbox/commblks/ref/pnsequencegenerator.html: *Matlab Hilfe: Communications Blockset/PN Sequence Generator*, June 2002.

[P.B01] P.BRAUN: *Powerline-Kommunikation im Auto*. Auto & Elektronik, 1:38–39, 2001.

[Wik04] WIKIPEDIA: *LIN-Bus*. http://de.wikipedia.org/wiki/LIN-Bus, 2004.

Anhang

Abbildungsverzeichnis