High-Speed Optical Transmission Systems*
Modulationsverfahren

18. Februar 2011

In dieser Übung sollen Aspekte des Senders genauer untersucht werden. Als externer elektro-optischer Modulator wird ein Mach-Zehnder-Interferometer verwendet. Die Aufgaben konzentrieren sich auf die Charakterisierung des Modulators, die elektrische Ansteuerung und auf die Generierung intensitäts- und phasenmodulierter Signale.

1 Kennlinie des Mach-Zehnder Modulators

In dieser Aufgabe soll zunächst die statische Kennlinie des Mach-Zehnder Modulators (MZM) aufgenommen werden. Dazu wird ein Simulationsaufbau gemäß Abb. 1 verwendet. Ein Laser emittiert Licht konstanter Leistung. Ein nachfolgender MZM wird in sogenannter *push-pull* Konfiguration betrieben, d.h. eine Elektrode wird mit dem invertierten Signal der anderen Elektrode angesteuert. Die Vorspannung soll in dieser Aufgabe 0 V betragen. Nach dem MZM wird die Leistung bestimmt und über einen XY-Schreiber ausgegeben.

Benötigte Module:

- Optical Sources/Laser CW
- OpticalModulators/ModulatorDiffMZ_DSM
- Electrical Sources/DC_Source
- Electrical Functions/InverterEl
- Instrumentation/Powermeter
- Analyzers/Numerical Analyzer2D

*Diese Übungsblätter basieren auf der ersten Ausarbeitung von Johannes Fischer, mit dem zusammen ich diese Veranstaltung entwickelt habe.

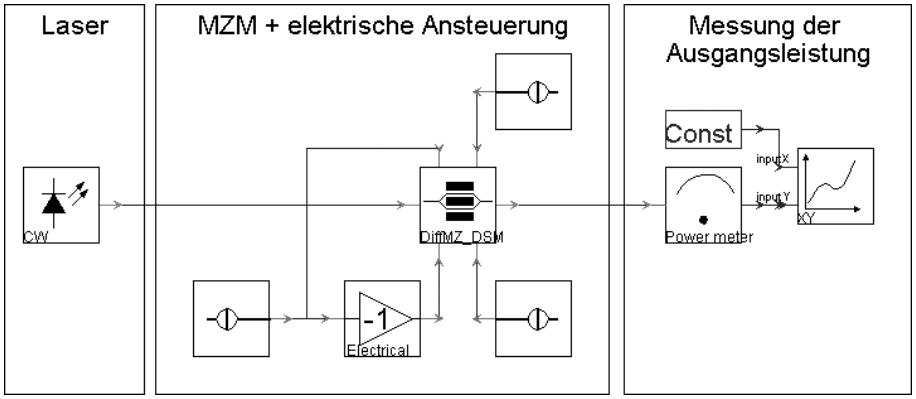

Abb. 1: Simulationsaufbau. Auf der rechten Seite des MZM befinden sich die beiden Spannungsquellen für die Vorspannung. Die Spannungsquelle auf der linken Seite des MZM bestimmt die Aussteuerung des MZM.

- Math Functions/Const

Einstellungen:

ModulatorDiffMZ_DSM	VpiDC	1
	VpiRF	1
	InsertionLoss	0
	LowerArmPhaseSense	POSITIVE
DC_Source (Vorspannung)	Amplitude	0
Numerical Analzer2D	Options	-x “Steuerspannung V/Vpi“ -y “Leistung [W]“

Aufgabe 1:

Nehmen Sie die Kennlinie des MZM für Steuerspannungen von 0 V bis $2 \cdot V_\pi$ auf![1]

2 Intensitätsmodulation mit einem MZM

In dieser Aufgabe soll ein in *push-pull* Konfiguration betriebener MZM als Intensitätsmodulator verwendet und damit ein NRZ-Sender aufgebaut werden. Dazu wird der Simulationsaufbau aus der letzten Aufgabe vom Vortag gemäß Abb. 2 erweitert.

[1] Dazu sollten Sie einen „Sweep” machen: Sie stellen die Spannung ein, indem Sie zuerst einen globalen Parameter für die Spannung definieren. Danach müssen Sie nur noch in den globalen Parametern einen Sweep für die Spannung definieren.

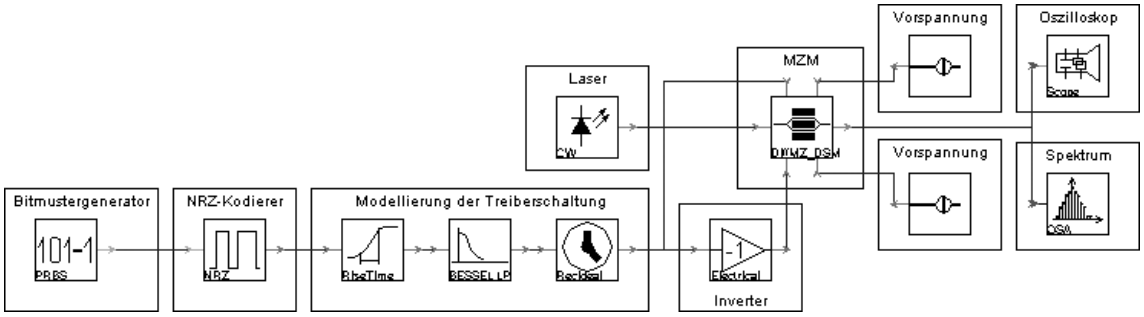

Abb. 2: Simulationsaufbau

Benötigte Module:

- Analzers/ViOSA
- Coder_NRZ
- PRBS
- Rise Time
- Filter_El
- Timing & Sampling/Clock RecoveryIdeal

Einstellungen:

LaserCW	Linewidth	0
ModulatorDiffMZ_DSM	VpiDC	1
	VpiRF	2
	InsertionLoss	0
	LowerArmPhaseSense	POSITIVE
DC_Source (oben)	Amplitude	0.5
DC_Source (unten)	Amplitude	-0.5

Aufgabe 2a:

Vergleichen Sie Spektrum und Augendiagramm der erzeugten NRZ-Signale für

1. Anstiegszeit $\Delta t = 0.25R$ und $f_{3dB} = 1.5R$, sowie
2. Anstiegszeit $\Delta t = 0.5R$ und $f_{3dB} = 0.7R$.

Hierbei stellt R die Bitrate dar.

Aufgabe 2b:

Löschen Sie die Module *ViScope* und *ViOSA*. Fügen Sie das Modul *Wiring Tools/output* ein und verbinden Sie dieses mit dem Ausgang des MZM. Speichern Sie den Aufbau als Galaxy (Dateiendung .vtmg) mit dem Namen *TX_NRZ*. Im Parametereditor eines Moduls kann durch Rechtsklick auf einen Parameter ein globaler Parameter erzeugt werden (Create Schematic Parameter). Erstellen Sie auf diesem Wege die folgenden Parameter:

Modul:	LaserCW
Name:	EmissionFrequency
Default Value:	193.1e12
Modul:	CoderNRZ_El
Name:	ChannelLabel
Default Value:	Ch1
Modul:	PRBS
Name:	PRBS_Order
Type:	Integer
Default Value:	7
Name:	RandomNumberSeed
Type:	Integer
Default Value:	1
Modul:	RiseTimeAdjust
Name:	RiseTime
Default Value:	0.25/BitRateDefault
Modul:	FilterBesselLP_El
Name:	CutOffFrequency
Default Value:	1.5*BitRateDefault

Die Standardwerte (*Default Value*) müssen gegebenenfalls noch über den globalen Parametereditor angepasst werden (über den Schalter *Parameter properties*).

Weisen Sie den globalen Parameter *ChannelLabel* auch dem Parameter *ChannelLabel* des Moduls *ClockRecoveryIdeal* zu (in geschweiften Klammern, da es sich um einen Parameter vom Typ *string* handelt).

3 Pulsformung mit einem MZM

Für die Generierung von optischen RZ-Signalen müssen zunächst optische Pulse mit einem Tastverhältnis $T_{FWHM}/T_{bit} < 1$ erzeugt werden. Dies kann ebenso wie die Intensitätsmodulation in Aufgabe 2 durch einen MZM in *push-pull* Konfiguration erreicht werden. Im Gegensatz zu Aufgabe 2 wird dieser jedoch mit einem sinusförmigen Signal angesteuert. Die Abb. 3 zeigt drei mögliche Steuerspannungen des MZM. Auf diese Weise können Tastverhältnisse von 0.33, 0.5 oder 0.67 erzeugt werden. Die Parameter der Steuerspannung für die verschiedenen Tastverhältnisse sind

in Tabelle 1 aufgeführt.

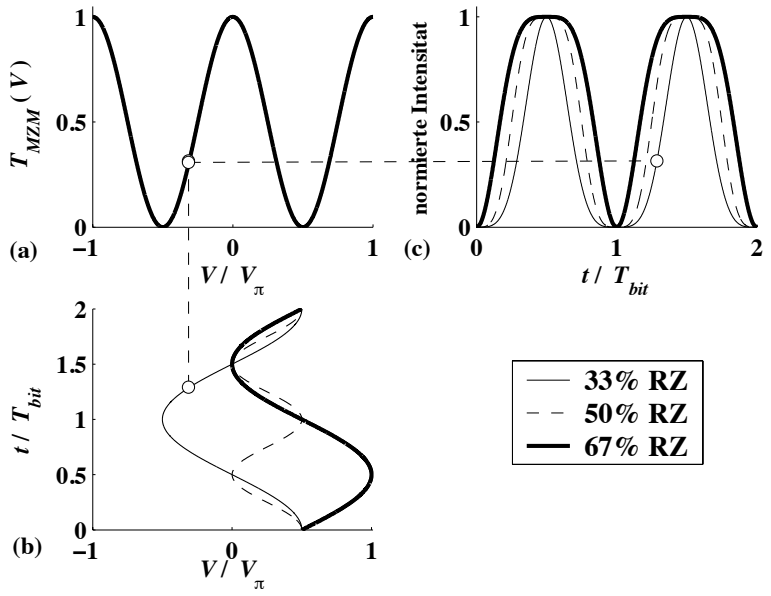

Abb. 3: (a) Transmission T_{MZM} des MZM in Abhängigkeit von der normierten Steuerspannung V/V_π, (b) Steuerspannung für verschiedene Tastverhältnisse, (c) Pulsform bei verschiedenen Tastverhältnissen. Beispielhaft ist ein korrespondierender Punkt in den drei Graphen eingezeichnet.

Tastverhältnis	33%	50%	67%
Vorspannung (V_{BIAS})	0	$0.25V_\pi$	$0.5V_\pi$
Amplitude	$0.5V_\pi$	$0.25V_\pi$	$0.5V_\pi$
Frequenz	$R/2$	R	$R/2$
Phase	$\pi/2$	$\pi/2$	0

Tabelle 1: Parameter der Steuerspannung für einen MZM in *push-pull* Konfiguration zur Generierung verschiedener Tastverhältnisse.

Benötigte Module:

- Electrical Sources/FuncSineEl

Einstellungen:

ModulatorDiffMZ_DSM	VpiDC	1
	VpiRF	1
	InsertionLoss	0
	LowerArmPhaseSense	POSITIVE

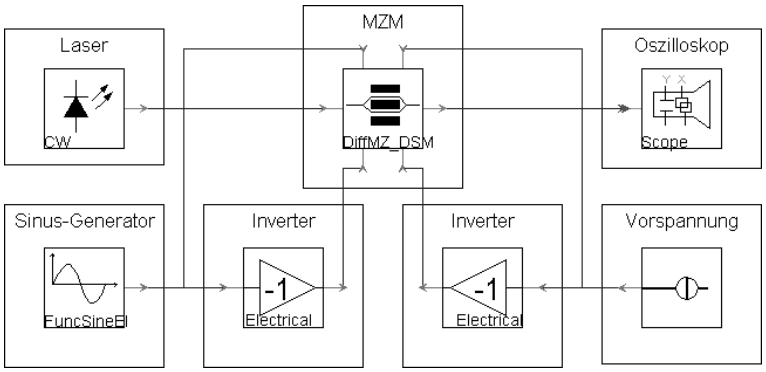

Abb. 4: Simulationsaufbau

Aufgabe 3a:

Speichern Sie den Aufbau aus Aufgabe 2a unter anderem Namen und ändern Sie den Simulationsaufbau gemäß Abb. 4. Die Simulation von vier bit sollte in dieser Aufgabe ausreichen. Passen Sie die Parameter des Sinus-Generators und der Vorspannung gemäß Tabelle 1 an und generieren Sie Pulse mit 33%, 50% und 67% Tastverhältnis. Stellen Sie durch Bestimmung der Halbwertsbreite T_{FWHM} sicher, dass die Parameter der Steuerspannung korrekt sind.

Aufgabe 3b:

Ersetzen Sie das *ViScope* durch das Modul *Wiring Tools/output* und speichern Sie das Setup als Galaxy mit dem Namen *PulseSource* ab. Erstellen Sie folgende globale Parameter:

Modul:	LaserCW
Name:	EmissionFrequency
Default Value:	193.1e12

Modul:	FuncSineEl
Name:	Amplitude
Default Value:	0.5
Name:	Frequency
Default Value:	BitRateDefault/2
Name:	Phase
Default Value:	90.0

Modul:	DC_Source
Name:	VBias
Default Value:	0.0

Aufgabe 3c:

Speichern Sie den Simulationsaufbau aus Aufgabe 2a unter neuem Namen und ersetzen Sie das Modul *LaserCW* durch Ihre erstellte Galaxy *PulseSource*. Vergleichen Sie Augendiagramme und Spektren von 33%, 50% und 67% RZ (mit $\Delta t = 0.25 T_{bit}$, $f_{3dB} = 1.5R$).

Aufgabe 3d:

Öffnen Sie Ihre Galaxy *TX_NRZ* und speichern Sie den Aufbau unter dem Namen *TX_RZ*. Ersetzen Sie das Modul *LaserCW* durch Ihre Galaxy *PulseSource*. Erstellen Sie folgende globale Parameter:

Modul:	PulseSource
Name:	VBias
Default Value:	0.0
Name:	Amplitude
Default Value:	0.5
Name:	Frequency
Default Value:	BitRateDefault/2
Name:	Phase
Default Value:	90.0

Die Trägerfrequenz ist durch den globalen Parameter *EmissionFrequency* bereits definiert und muss nur noch dem entsprechenden Parameter des Moduls *PulseSource* zugewiesen werden.

4 Phasenmodulation mit einem MZM

Ein MZM kann auch zur Modulation der Phase verwendet werden. Je nach Art der Ansteuerung wirkt er dabei wie ein Phasenmodulator, wobei die Phase kontinuierlich verändert wird, oder wie

ein Phasenschalter, bei dem die Phase um 180° springt.

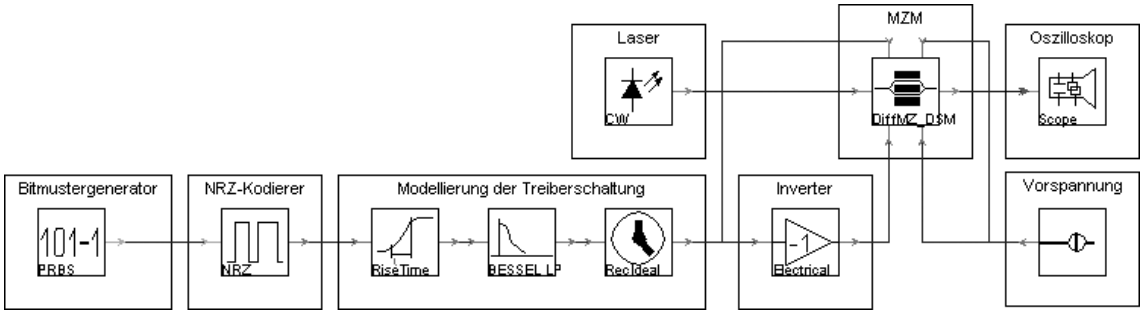

Abb. 5: Simulationsaufbau

Einstellungen:

PRBS	PRBS_Type	Alternate
	PreSpaces	0
	PostSpaces	0
LaserCW	LineWidth	0
ModulatorDiffMZ_DSM	VpiDC	1
	VpiRF	1
	InsertionLoss	0
	LowerArmPhaseSense	POSITIVE
DC_Source	Amplitude	0

Aufgabe 4a:

Speichern Sie den Simulationsaufbau aus Aufgabe 2a unter neuem Namen und ändern Sie ihn gemäß Abb. 5. In dieser Aufgabe sollte die Simulation von acht bit ausreichen. Die Vorspannung beträgt 0 V. Um den MZM als Phasenmodulator zu betreiben wird zunächst das Modul *InverterEl* ausgeschaltet. Dadurch wird der MZM nicht mehr differentiell angesteuert. Vergleichen Sie den Phasenverlauf und Chirp des generierten Signals für

1. Anstiegszeit $\Delta t = 0.25 T_{bit}$ und $f_{3dB} = 1.5R$, sowie
2. Anstiegszeit $\Delta t = 0.5 T_{bit}$ und $f_{3dB} = 0.7R$.

Aufgabe 4b:

Nun soll der MZM als Phasenschalter betrieben werden. Schalten Sie dazu das Modul *InverterEl* wieder ein, so dass der MZM differentiell angesteuert wird. Vergleichen Sie auch hier den Phasenverlauf und Chirp des generierten Signals für

1. Anstiegszeit $\Delta t = 0.25 T_{bit}$ und $f_{3dB} = 1.5R$, sowie
2. Anstiegszeit $\Delta t = 0.5 T_{bit}$ und $f_{3dB} = 0.7R$.

5 Differentielle Phasenumtastung (DPSK)

In dieser Aufgabe soll ein Sender für DPSK (*differential phase-shift keying*) aufgebaut werden. Bei diesem Modulationsverfahren ist die Information in eine Phasendifferenz kodiert. Wird eine logische Eins gesendet, so ändert sich die Phase um π, bei einer logischen Null bleibt sie konstant. Das elektrische Steuersignal muss daher zunächst vorkodiert werden. Dies geschieht mit einer rückgekoppelten XOR-Verknüpfung (Abb. 6).

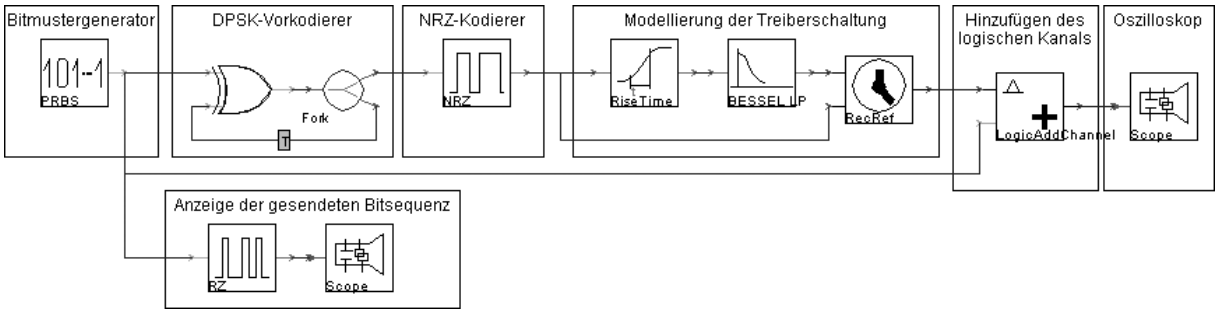

Abb. 6: Simulationsaufbau

Benötigte Module:

- Information Sources & Coding/CoderRZ_El
- Signal Processing/Logic/XOR
- Wiring Tools/Fork_2
- ClockRecoveryModules/ClockRecoveryIdealRef
- Signal Representation Conversion/LogicAddChannel

Einstellungen:

PRBS	PreSpaces	1
	PostSpaces	0
	PRBS_Type	PRBS_N
	PRBS_Order	4
	RandomNumberSeed	1
CoderNRZ_El	ChannelLabel	Ch1
RiseTimeAdjust	RiseTime	0.25/BitRateDefault
FilterBesselLP_El	CutOffFrequency	1.5*BitRateDefault
LogicAddChannel	LabelSetMode	LeaveUnchanged
	LogicalChannelMode	Modify
	ChannelLabel	Ch1
	CarrierFrequency	0

Aufgabe 5a:

Speichern Sie den Simulationsaufbau aus der letzten Aufgabe vom Vortag unter neuem Namen und fügen Sie gemäß Abb. 6 den DPSK-Vorkodierer ein. Die Rückkoppelschleife muss eine Verzögerung aufweisen. Durch einen Doppelklick auf die Verbindung zwischen dem Ausgang des Moduls *Fork_2* und dem Modul *XOR* öffnen Sie die Parameter der Verbindung. Weisen Sie dem Parameter *Delay Value* den Wert 1 zu. Der Vorkodierer verändert die originale Bitfolge. Diese muss dem logischen Kanal daher gesondert eingeschrieben werden. Mit dem Modul *LogicAddChannel* können einem Signal beliebige logische Kanäle zugeordnet werden. Zur Anzeige der originalen Daten dient ein RZ-Kodierer mit angeschlossenem *ViScope*. Vergleichen Sie das vorkodierte Signal mit dem ursprünglichen Signal und stellen Sie sicher, dass jede logische Eins zu einem Pegelwechsel des vorkodierten Signals führt. Die Simulation von 16 bit ist hier ausreichend.

Aufgabe 5b:

Speichern Sie den Simulationsaufbau aus Aufgabe 4 unter neuem Namen. Ersetzen Sie die Module zur Generierung des elektrischen Steuersignals durch den Aufbau aus Aufgabe 5a. Betreiben Sie den MZM einmal als Phasenmodulator (Modul *InverterEl* ausgeschaltet) und einmal als Phasenschalter (Modul *InverterEl* eingeschaltet) und vergleichen Sie die generierten optischen DPSK-Signale. Die Simulation von 16 bit ist hier ausreichend.

Aufgabe 5c:

Ersetzen Sie das Modul *ViScope* durch das Modul *Wiring Tools/output* und speichern Sie den Aufbau unter dem Namen *TX_DPSK* als Galaxy ab. Erstellen Sie folgende Parameter:

Modul:	LaserCW
Name:	EmissionFrequency
Default Value:	193.1e12

Modul:	LogicAddChannel
Name:	ChannelLabel
Type:	String
Default Value:	Ch1

Modul:	PRBS
Name:	PRBS_Order
Type:	Integer
Default Value:	7
Name:	RandomNumberSeed
Type:	Integer
Default Value:	1

Modul:	RiseTimeAdjust
Name:	RiseTime
Default Value:	0.25/BitRateDefault

Modul:	FilterBesselLP_El
Name:	CutOffFrequency
Default Value:	1.5*BitRateDefault

Erstellen Sie im globalen Parametereditor folgende Parameter

Name:	InverterActive
Type:	Enumeration
Default Value:	On
Values:	On, Off
Name:	ViScopeActive
Type:	Enumeration
Default Value:	On
Values:	On, Off

Der Parameter *InverterActive* wird dem Parameter *Active* des Moduls *InverterEl* zugewiesen. Dadurch kann dieser ein- und ausgeschaltet werden. Ebenso wird der Parameter *ViScopeActive* dem Parameter Active des Moduls *ViScope* (nach dem Modul CoderRZ_El) zugewiesen, so dass später die gesendete Bitsequenz angezeigt werden kann. Schließlich muss noch der globale Parameter *ChannelLabel* an das Modul *CoderNRZ_El* übergeben werden (bei der Übergabe der drei Parameter die geschweiften Klammern nicht vergessen).

Aufgabe 5d:

Speichern Sie den Simulationsaufbau aus Aufgabe 5b unter neuem Namen. Ersetzen Sie das Modul *LaserCW* durch Ihre in Aufgabe 3b aufgebaute Pulsquelle (Modul *PulseSource*). Stellen Sie die Pulsquelle so ein, dass sie Pulse mit einem Tastverhältnis von 0.33 generiert. Simulieren Sie 128 bit (ändern Sie auch den Parameter *PRBS_Order* auf den Wert 7) mit eingeschaltetem Modul *InverterEl* und analysieren Sie das generierte optische RZ-DPSK Signal im Zeit- und Frequenzbereich.

Aufgabe 5e:

Speichern Sie Ihre Galaxy *TX_DPSK* unter dem Namen *TX_RZDPSK*. Ersetzen Sie das Modul *LaserCW* durch Ihre Galaxy *PulseSource*. Folgende Parameter sollen erstellt bzw. zugewiesen werden:

Modul:	PulseSource
Name:	EmissionFrequency
Default Value:	193.1e12
Name:	VBias
Default Value:	0.0
Name:	Amplitude
Default Value:	0.5
Name:	Frequency
Default Value:	BitRateDefault/2
Name:	Phase
Default Value:	90.0