Netzwerke und Datenkommunikation
B-LS-MI 004
Physical Layer

rolf.schmutz@fhnw.ch

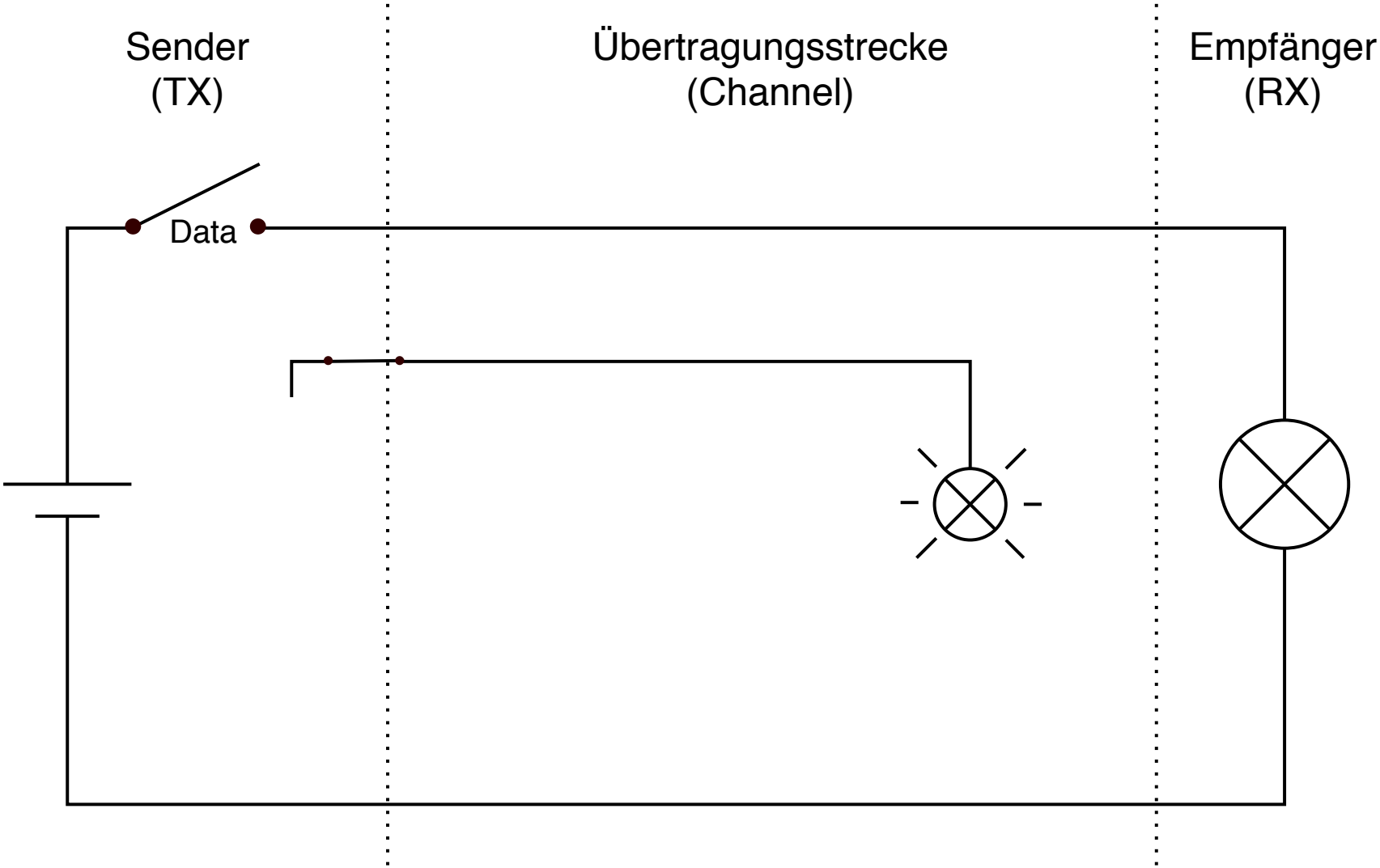
FHNW

22. September 2020

Ziele

- Repräsentation des Quellsignals auf elektromagnetischer Ebene
- Codierung des Quellsignals (Abgekürzt)
- Verfahren zur Leitungscodierung der Daten aus einem Quellenstrom
- Techniken in Bezug auf Basisband- und Breitband-Kommunikation (Modulation)
- Fehlererkennung und -Korrektur

Einfachste Bit-Serielle Datenübertragung



- "Hello World!" soll übertragen werden

Probleme

- wie wird “Hello World!” als Abfolge von Licht/kein-Licht (0, 1) dargestellt? (Quellcodierung)
- wann beginnt die Nachricht, einzelne Buchstaben, einzelne Bits, wann enden sie?
- wie können einzelne gleiche “bits” getrennt werden? z.B. “o”=01101111

Quellencodierung (source-coding) 1/2

Das ist die Repräsentierung von Informationen in binärer (numerischer) Form, also nicht Programm-Quellcode/sourcecode

- es wird eine Übereinkunft/Tabelle benötigt, die die Information in numerischer Form (Bitmuster) festlegt (code-point)
- es gibt eine Vielzahl von Codierungen für verschiedene Datenformate

Die Codierung muss auf beiden Seiten bekannt sein und ist nicht gleich "Verschlüsselung"

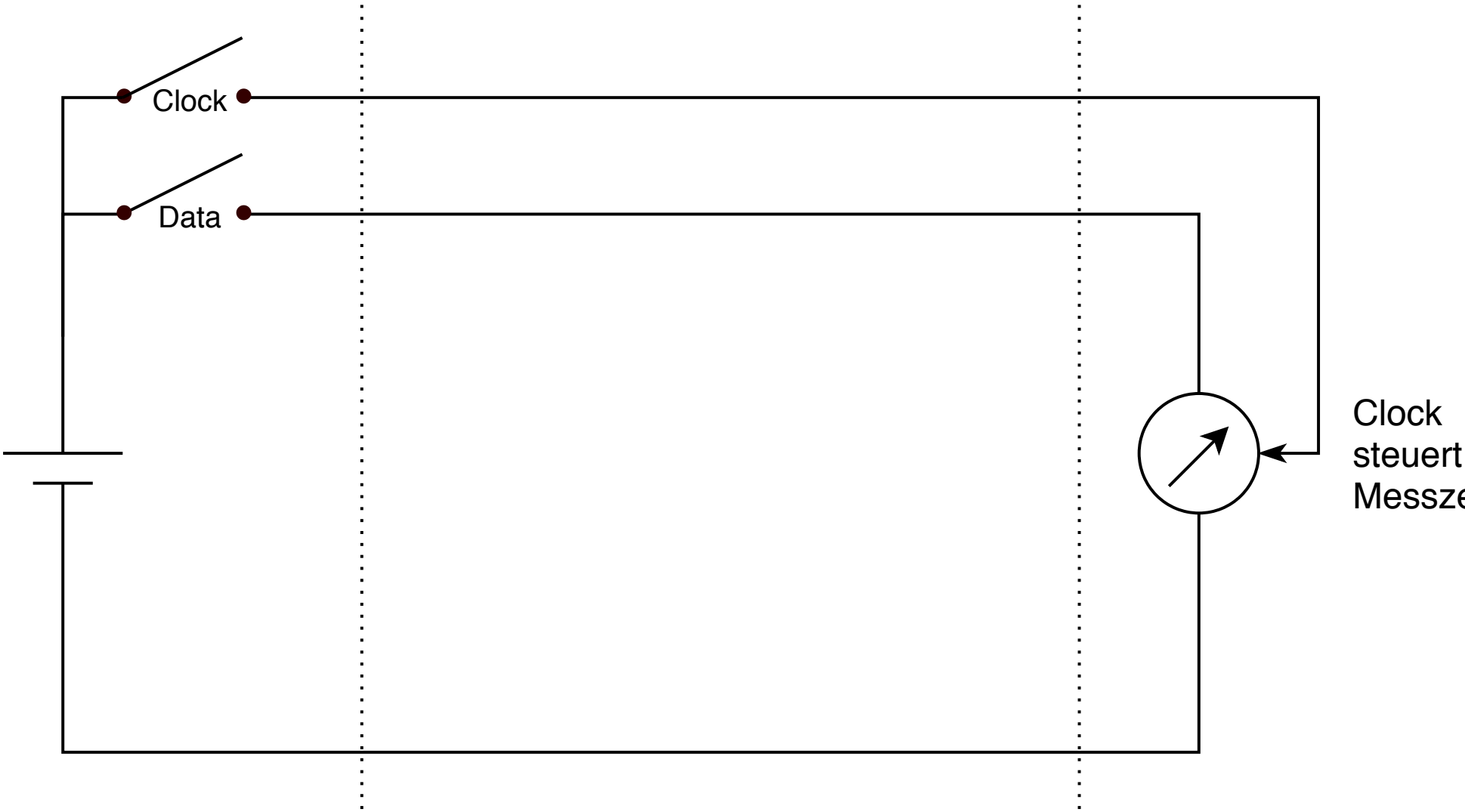
Quellencodierung (source-coding) 2/2

Für unsere Zwecke benutzen wir die alterwürdige ASCII-Codierungstabelle (ohne Kontrollzeichen):

```
   2 3 4 5 6 7 <- Hi-Nibble
  -------------
0:   0 @ P ‘ p
1: ! 1 A Q a q
2: " 2 B R b r
3: # 3 C S c s
4: $ 4 D T d t
5: % 5 E U e u
6: & 6 F V f v
7: ’ 7 G W g w
8: ( 8 H X h x
9: ) 9 I Y i y
A: * : J Z j z
B: + ; K [ k {
C: , < L \ l |
D: - = M ] m }
E: . > N ^ n ~
F: / ? O _ o DEL
```

z.B. “H”: $48_{16} = 0100'1000_2$

Bit-Synchronisation: Strobe/Clock/Sampling (1/2)



Bit-Synchronisation: Strobe/Clock/Sampling (2/2)

- mit der "Clock" Leitung wird dem Empfänger der korrekte Messzeitpunkt signalisiert
- Folgen von "gleichen" Bits (alles 0 oder alles 1) können problemlos getrennt werden

Synchrone Bitserielle Übertragung

Es werden mindestens drei Leitungen benötigt, dafür sind keine weiteren Massnahmen nötig.
Synchrone Datenübertragung wird vorallem im Nahbereich (im Computer, Embedded Systems $I^2C/SPI, HDMI, etc$)*eingesetzt*

- es kann auch zwischen "keine Daten" (Clock=0) und "0" Bits unterschieden werden

Asynchrone Serielle Übertragung (1/3)

Eine weitere Möglichkeit eine Synchronisierung[1] ist das "Framing" der Übertragung

- eine Startsequenz (Startbit oder Preamble) und eine optionale Endsequenz werden in den Datenstrom eingefügt[2]
- der Empfänger hat damit die Möglichkeit, sich *für die Dauer der Nachricht/Zeichens* mit dem Sender zu synchronisieren
- mit dem Framing kann beim Empfänger auch zwischen Daten/keine-Daten unterschieden werden (ausserhalb des Frames werden Daten ignoriert)

Asynchrone Bitserielle Übertragung

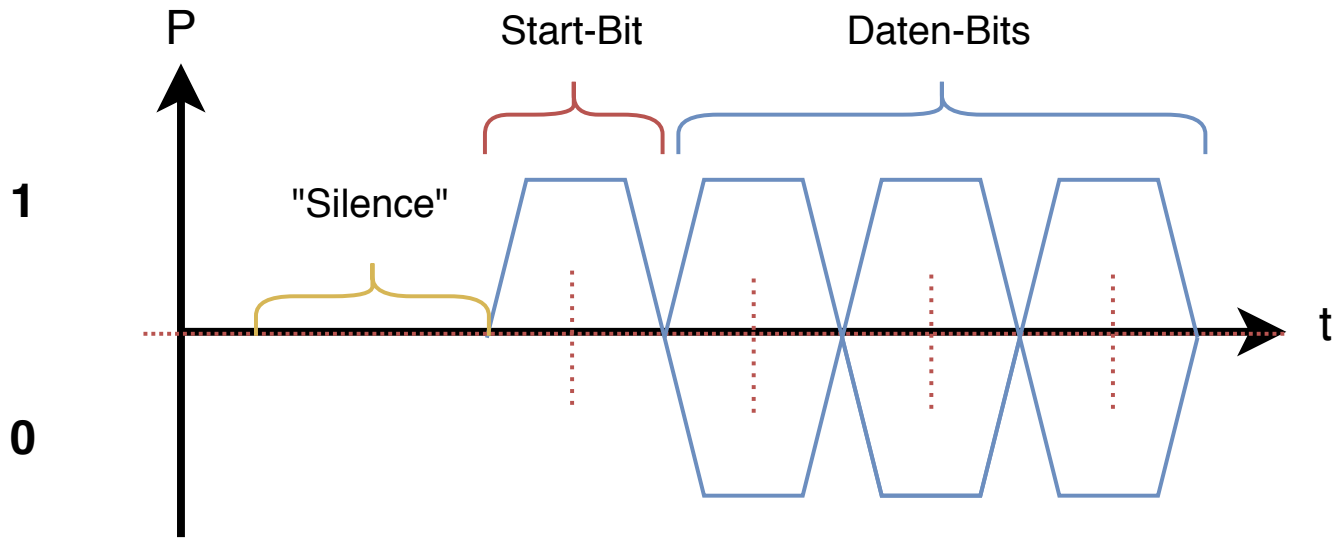
Es werden nur zwei Leitungen/ein Kanal benötigt. Dafür ist die Methode ein wenig aufwendiger zu implementieren.

[1] wenn auch im Titel "Asynchron"

[2] dies sind bereits keine "Nutzdaten" mehr sondern Teil des Protokolls

Asynchrone Serielle Übertragung (2/3)

Bei einfachen seriellen Schnittstellen (RS232 und äquivalent) wird ein Startbit (optional Stopbit) eingefügt:



- der Empfänger muss ungefähr die Transferrate/Bitzeit schon kennen und kann das Sampling nach dem Startbit einstellen
- die Möglichkeit einer Startsequenz "10" vereinfacht dies weiter
- moderne Implementationen buffern die Übertragung ein paar bits und können damit "autobaud" – selbständige Adaption an die Datenrate implementieren

Asynchrone Serielle Übertragung (3/3)

Bei "Ethernet" (der Quasi-Standard im Internet/IP-Netzwerken) wird mit einer Präambel gearbeitet

- 7 Bytes AA_{16} + 1 Byte AB_{16} (d.h. insgesamt 64 Bit)
- der Empfänger hat eine eigene Clock-Source mit der ungefähren Frequenz aber unbekannter Phase. Über eine PLL wird die korrekte Phase ermittelt: