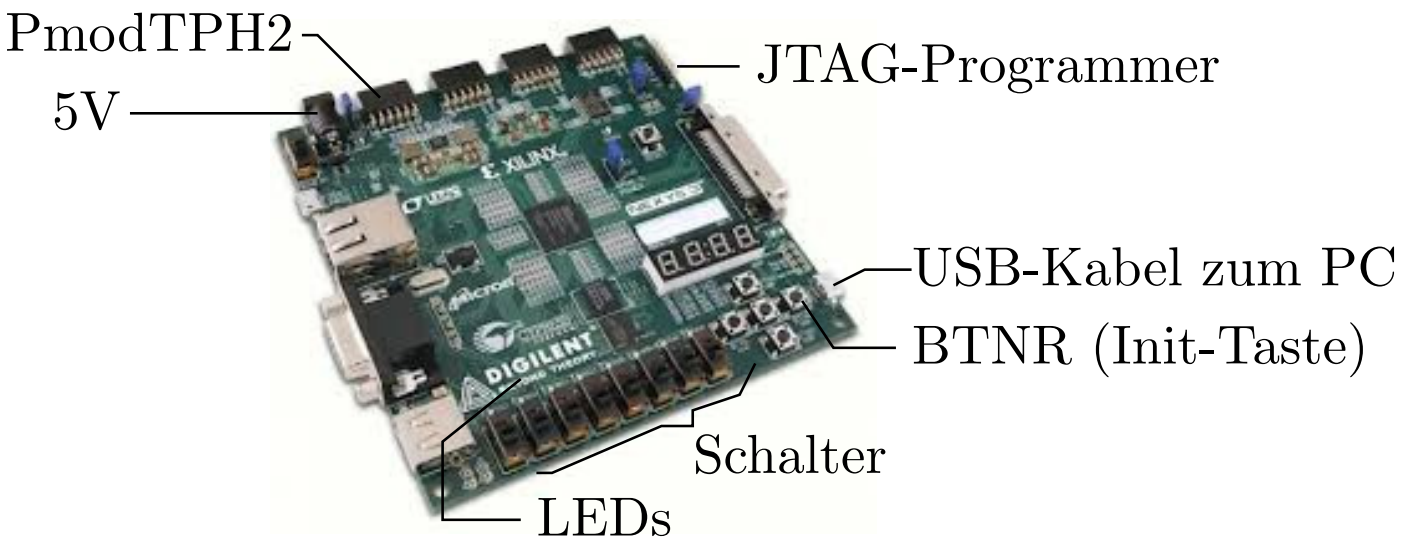
Grundlagen der Digitaltechnik
Große Übung 5
Test einer UART

Prof. G. Kemnitz, Dr. C. Giesemann

Institut für Informatik, Technische Universität Clausthal

14. April 2021

Versuchsvorbereitung



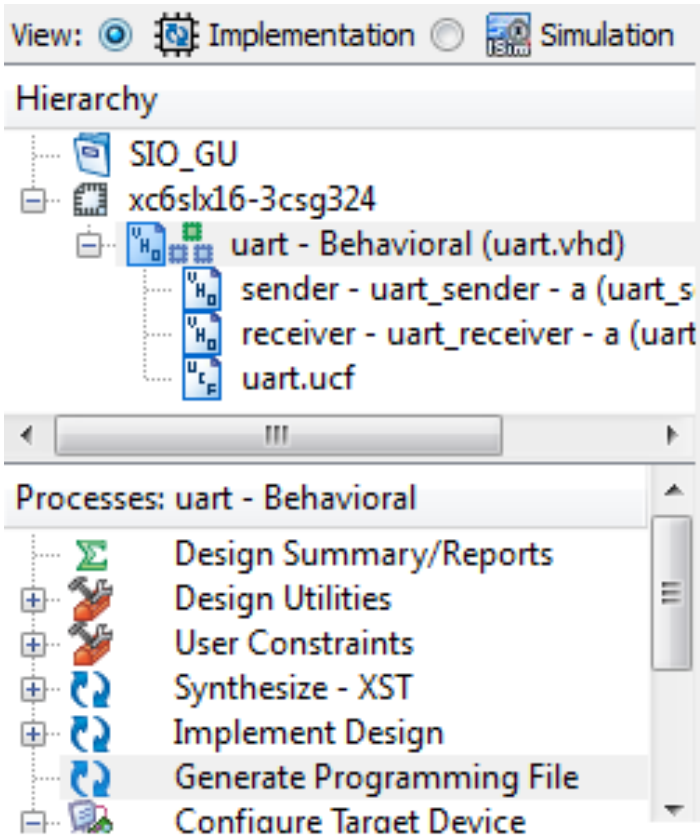
An das Versuchsboard sind anzuschließen:

- Spannungsversorgung 5 V,
- an J7 der JTAG-Programmer,
- an die USB-Buchse »UART« ein USB-Kabel zum PC,
- an den Stecker JA1 ein PmodTPH2 mit Logikanalysatoranschlüssen.

Genutzt werden weiterhin die Schalter, ein Taster und die LEDs.

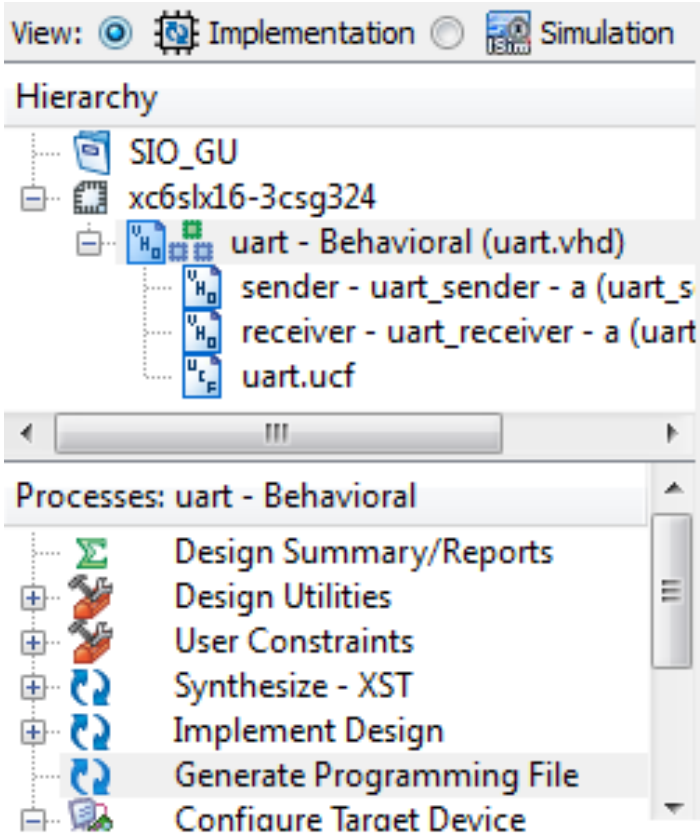
Projekt laden

- »uart.zip« von der Web-Seite im Arbeitsverzeichnis entpacken.
- Das Entwurfssystem mit Doppelklick auf »uart.xise« starten.




```
 1 library IEEE;
 2 use IEEE.STD_LOGIC_1164.ALL;
 3
 4 entity uart is
 5  port (
 6    GCLK : in std_logic;
 7    RXD : in std_logic;
 8    TXD : out std_logic;
 9    SW : in std_logic_vector(7 downto 0);
10    LD : out std_logic_vector(7 downto 0);
11    BTNR : in std_logic;
12    LA : out std_logic_vector(3 downto 0));
13 end uart;
14
15 architecture Behavioral of uart is
16
17 signal clk_16x_baud, clk_baud : std_logic;
18 signal start, busy_sender, busy_receiver,
```

Bit-File erzeugen




```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3
4  entity uart is
5   port (
6     GCLK : in std_logic;
7     RXD : in std_logic;
8     TXD : out std_logic;
9     SW : in std_logic_vector(7 downto 0);
10    LD : out std_logic_vector(7 downto 0);
11    BTNR : in std_logic;
12    LA : out std_logic_vector(3 downto 0));
13 end uart;
14
15 architecture Behavioral of uart is
16
17 signal clk_16x_baud, clk_baud : std_logic;
18 signal start, busy_sender, busy_receiver,
```

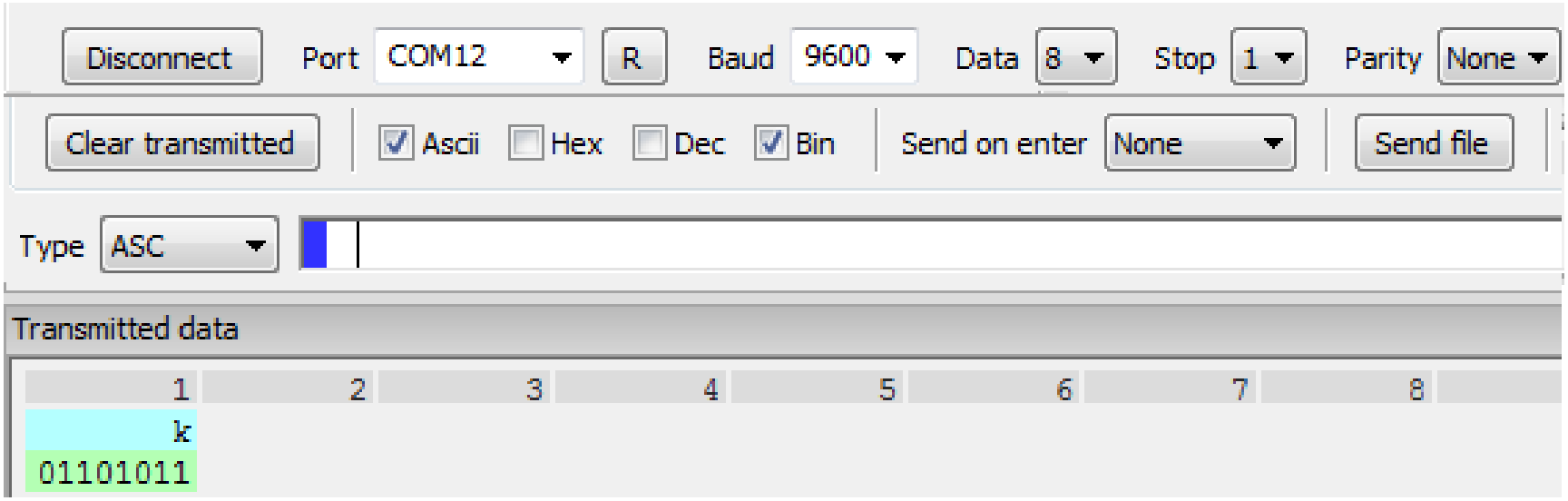
- Alle Dateien incl. ucf-Datei bereits im Projektbaum enthalten.
- »Generate Programming File« (Bit-File erzeugen).
- »Configure Target Device« (Programmiertool starten).

FPGA Programmieren

- Baugruppe einschalten.
- (iMPACT) Doppelklick auf »Boundary Scan«.
- (Boundary Scan) Rechtsklick auf (Right click ...) ▷ »Initialize Chain«. »Auto Assign ...« Yes.
- Für »xc6slx16« »uart.bit« auswählen; »Open«.
- »Attache SPI...« »No«. Nächstes Fenster »OK«.
- Rechtsklick auf den Chip »xc6slx16« ▷ »Program«.
- Nächstes Fenster »OK«.

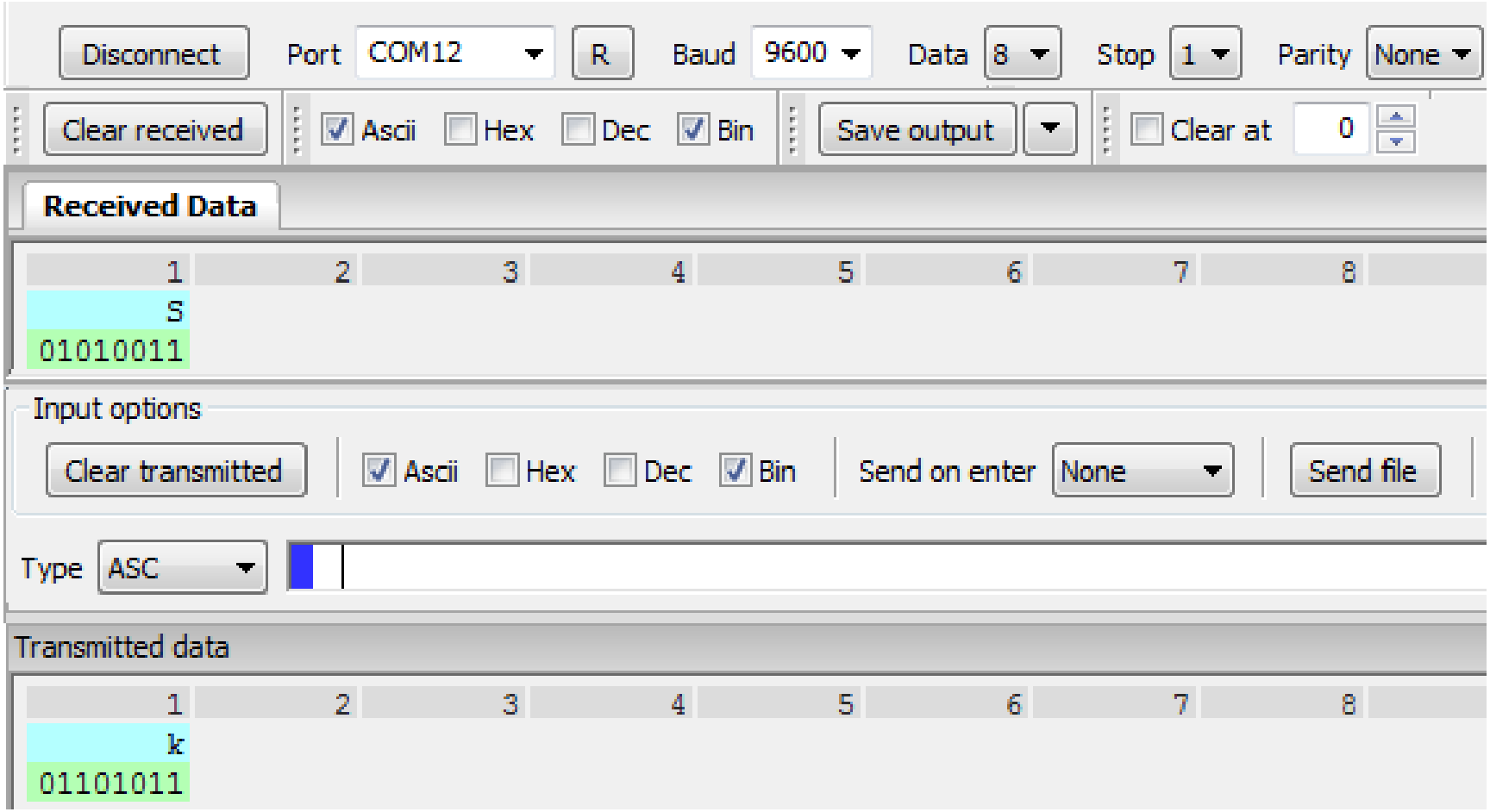
Verbindung mit HTerm herstellen

- Auf dem PC HTerm starten.
- COM-Port auswählen[1].
- 9600 Baud, 8 Daten-, 1 Stopp- und kein Paritätsbit einstellen.
- Verbindung herstellen (Connect).
- Testweise Zeichen versenden und auf Antwort warten.



[1] Die COM-Schnittstelle, die nach Anstecken des USB-Kabels vom Stecker »uart« und »R« (Refresh Comport List) als neuer Port erscheint.

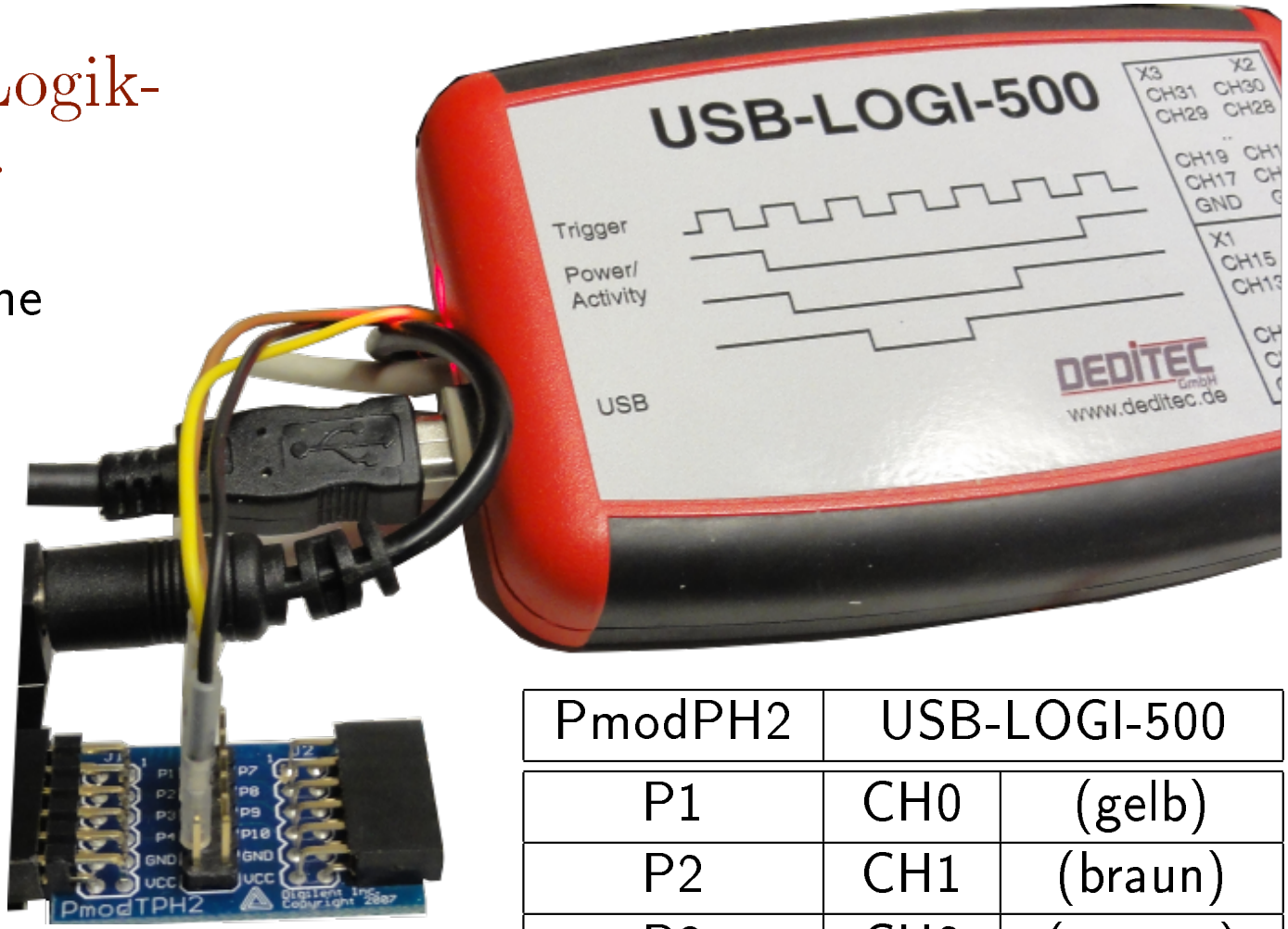
Die empfangenen und gesendeten Zeichen können als ASCII-Zeichen, Binär-, Hexadezimal- und/oder Dezimalwerte dargestellt werden.



Wo wird eingestellt, was für ein Zeichen der PC empfängt?

Test mit Logik-analysator

Anschluss siehe Tabelle:



PmodPH2	USB-LOGI-500	
P1	CH0	(gelb)
P2	CH1	(braun)
P3	CH2	(orange)
P4	CH3	(weiß)
GND	GND	(schwarz)

Konfiguration des USB-LOGI-500

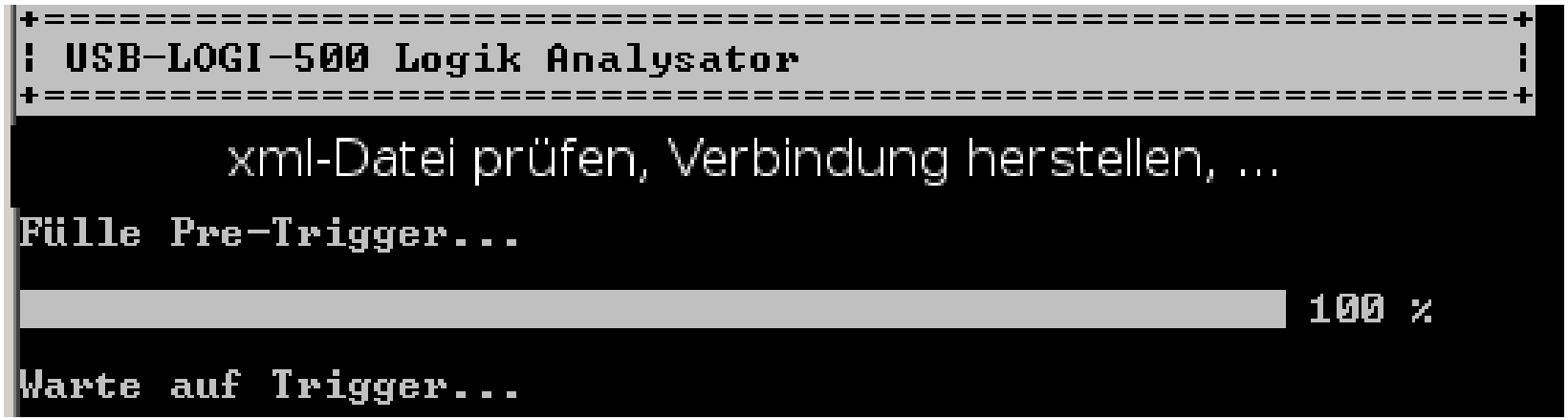
Der USB-LOGI wird bei uns mit einem xml-File konfiguriert, für das Beispiel »uart.xml« aus »uart.zip« (siehe Doku auf der Web-Seite):

```
<la>
 <samplerate>50000</samplerate> #50.000 Werte pro s
 #Trigger nach 1/8 der Gesamtaufzeichnungsdauer
 <pretrigger>1</pretrigger>
 <signals>
  <signal name="RxD"><ch>0</ch></signal>
  <signal name="busy_receiver"><ch>1</ch></signal>
  <signal name="TxD"><ch>2</ch> </signal>
  <signal name="busy_sender"><ch>3</ch></signal>
 </signals>
 #Aufzeichnungsbeginn mit fallender Flanke von RxD
 <trigger when="A">
  <A> <ch when="falling_edge">0</ch> </A>
 </trigger>
</la>
```

Messung durchführen

- Windows-Konsole (cmd.exe) starten.
- Zum Arbeitsverzeichnis »...\USBLOGI« wechseln und USB-Logi starten mit dem Kommando:

```
usb-logi uart.xml
```

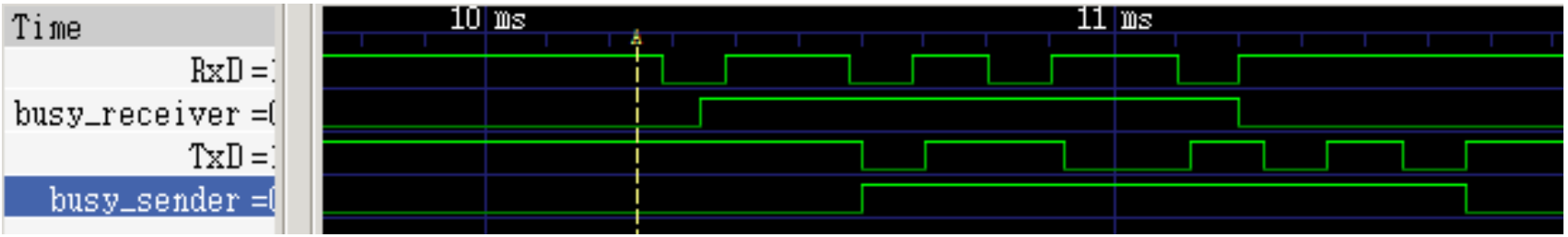


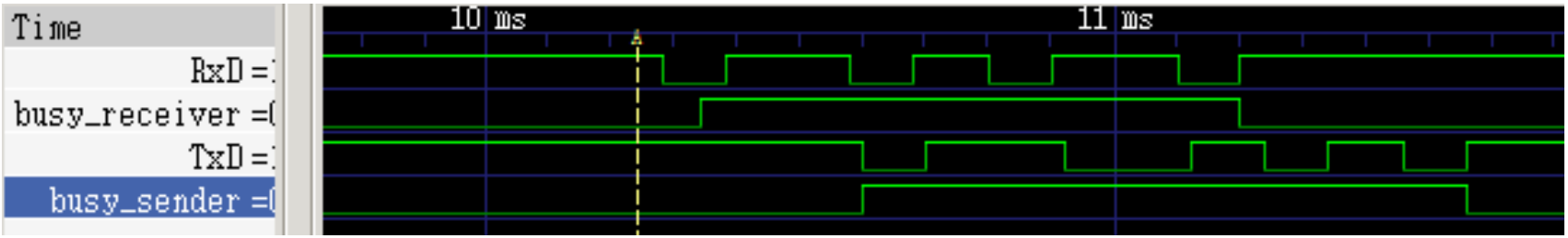
- Sobald der USB-LOGI auf sein Triggerereignis wartet, mit HTerm ein Zeichen senden.

- Aufzeichnungsbeginn bei nächster fallender RxD-Flanke:

```
Warte auf Trigger...

Aufzeichnung läuft...

                                                                100 %

+-----------------------------------------------------------------+
| (4) Erzeuge Waveform Dateien...                                 |
+-----------------------------------------------------------------+

LXT Datei ./uart.lxt erzeugt.

Bestehende SAV Datei ./uart.sav beibehalten.

+-----------------------------------------------------------------+
| (5) Starte GTKWave...                                           |
+-----------------------------------------------------------------+
```

- Im sich öffnenden GTKWave-Fenster mit Zoom und Scroll-Leisten Darstellung anpassen.





- Entspechend Einstellung werden 1/8 der Werte vor und 7/8 der Werte nach dem Trigger-Ereignis aufgezeichnet.

Programmieraufgabe 1

Ändern Sie das Rahmenprogramm so ab, dass

1. von den ankommenden Zeichen statt des Zeichenwertes die Anzahl der bisher empfangenen Zeichen auf die LEDs ausgegeben wird und
2. statt des Schalterwertes die Summe aus Schalterwert und Zeichenwert gefolgt vom Zeichencode für ein Leerzeichen (ASCII-Wert 0x20) zurück gesendet werden.

Die Antwort mit zwei Zeichen auf ein ankommendes Zeichen verlangt gleichfalls einen kleinen Automaten, der

- auf ein ankommendes Zeichen wartet,
- dann, sobald der Sender frei ist, die Summe sendet und
- danach, sobald der Sender das nächste mal frei ist, das Leerzeichen sendet.

Programmieraufgabe 2

Ändern Sie das Rahmenprogramm so ab, dass beim Drücken des Tasters »BTND« die Zeichenfolge »Hallo« an den PC gesendet wird.

Die Lösung der Aufgabe erfordert:

- Tasterabfrage mit Entprellung.
- Einen Automaten, der auf einen Tastendruck wartet und solange nicht alle Zeichen versendet sind, immer, wenn der Sender frei (nicht beschäftigt) ist, das nächste Zeichen versendet.