Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences
Department Electrical and Information Engineering
Laboratory Processor Systems

Exercise 2e: Serial Communication

Semester/Group/ Team:	**Day of Protocol Submission:**	**Chairperson:**
Day of Exercise:		**Participants:**
Professor:		
Professor's Comments:		

1 Protocol

The protocol has to contain at least:

- A short description of the task
- A detailed description of the solution, of the algorithm and/or the measurement methode
- All more complex parts of the program has to be explained with graphical methods (Nassi-Shneiderman-Diagram or similar)
- Well commented listing of the program sources
- Numbered text sections with headlines, numbered figures and tables with captions
- A list of devices and all references
- A final discussion of the results
- For all oscilloscope measurements a distinctive labelling of signals with names is necessary:
  - A description of each measurement, including the contact point of the probe (marking and labelling of the probe in a circuit diagram)
  - A corresponding labelling of all drawn / printed signalcurves in oscilloscope diagrams

2 Introduction

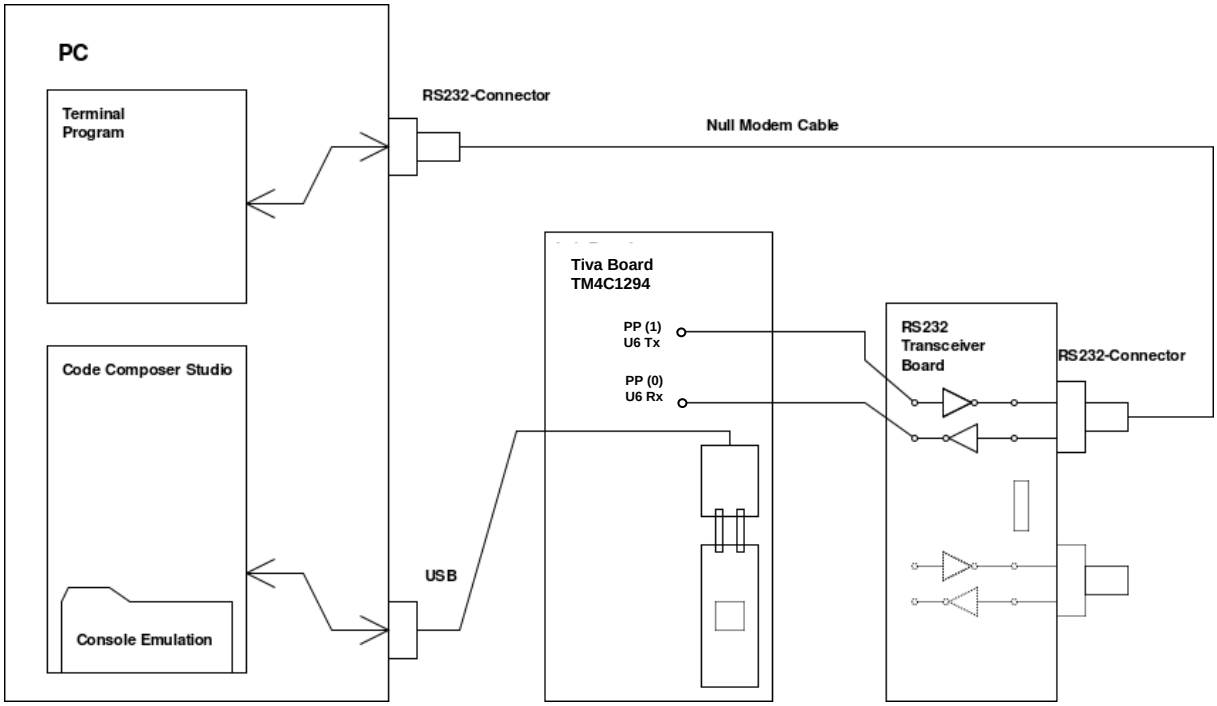

Figure 1: Overview of the Lab-Configuration Serial Communication

All exercises are concerned with the serial communication of the controller TM4C1294 with the peripheral devices UART(Universal Asynchronous Receivers/Transmitters). Figure 1 shows the PC with the development system Code Composer Studio, the serial terminal-program, the controller board and the additional RS232 transceiver board.

Unless otherwise stated, exercises 3, 4, and 5 have to be programmed and tested.

3 Serial Output with the UART

Write a C-program, which generate periodical UART-outputs of the same symbol in a loop.

Four different bit rates and/or data formats have to be programmed.

Display and draw the signals using an oscilloscope. Use a oscilloscope channel for TTL/CMOS-Levels and another channel for RS232-Level. Program an additional port pinoutput for triggering before controller starts serial output.

Label each bit (data bits, startbit, paritybit and the stopbit) in the signal curves in your protocol.

Measure the RS232 frame length for each of four programs. Recalculate the bit rate. Compare the measured bitrate with the configured bit rate in your program.

4 Transmission of text

Develop and test a transmission program in C with the following output to the terminalprogram:

```
Exercise Serial Communication Date: <Today's Date>.

Participants:

<Names>
```

Use in the prototocol: 9600 bit/s, seven databits, even parity, one stopbit (9600 bit/s, 7E1).

5 Receiving of data with and without echo

Develop and test a receiving program in C , which reads input characters from the terminal program. The received characters are stored in the controller memory.

Here is an interrupt handler not necessary, the following features are required:

- Typing errors may be corrected with backspace (BS, 0x08). After receiving backspace, the previous (wrong) character has to be replaced by the following (correct) character. Take into consideration the special case when backspace is typed as the first character.
- Input is finished by the character code 'end of file' (EOF, 0x04, 'Ctrl d' key). After that the received string has to be sent to the terminal for verification.
- An additional switch enables the non-echo-mode or as alternative the echo-mode. In the echo-mode each received character are send immediately to the terminal-program. The switch are continuously polled from the controller program.

6 Receiving of data with and without echo II

Develop and test a program with the features of experiment before, now using an interrupt handler for receiving characters.

7 Transmission Errors

Write a receive program for characters typed at the terminal. Transmission errors have to be detected and displayed by LEDs connected to the port PM(0) - PM(2).

At the beginning the receive program has to use the following transfer parameters: 9600 bits/s, seven data bits, parity even, two stop bits. Later the parameter could be changed. An interrupt handler may be used.

In the case of disturbed reception several errors may occur simultaneously. The errors may e.g. be generated by the following manipulations:

- Frame Error (LED at pin PM(0): select eight data bits at the terminal..
- Parity Error (LED at pin PM(1): select parity odd at the terminal.
- Overrun Error (LED at pin PM(2): add a wait loop to your program.

Generate all these Errors. Use a protocol table with three columns:

- Configuration in the controller program
- Configuration in the terminal program
- Errors shown with LED

8 Data output interrupted by a port interrupt

Your program transmits continuously the decimal numbers 0 to 9 to the terminal program. The following output is generated:

```
 0
 1
...
 9
 0
...
```

Using a external key switch an external interrupt is requested. The switch is connected to an port pin. The corresponding interrupt handler generates a short message to the the terminal program:

```
Interrupt for output of the date: <Today's date>.
```

After that the program continues with the number output.