I. RS232

Theory: Serial communication, RS232, operating an oscilloscope

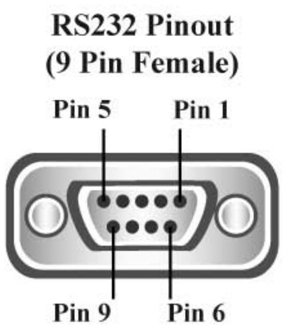
Exercises:

1. Familiarize yourself with Terminal v1.93b application.
2. Try out serial communication between two PCs with RS232.
3. Examine physical signals on the RS232 interface using an oscilloscope. Store selected representative waveforms on a flash drive.
4. Try decoding the transmitted symbol out of a displayed waveform.

Measurement guide:

1. Run the Terminal application and familiarize yourself with its settings and functions. For a successful RS232 communication a correct setting of baudrate, number of data bits and parity is necessary.
2. Connect two PCs serial ports using a serial cable. Set the same baudrate (115200), number of data bits (8) and parity on both PCs. Select the connected COM Port a click [Connect]. Try sending symbols or messages between interconnected PCs and check for a proper function of the interface. Change the parity and number of stop bits on just one of the PCs and send a symbol. Was it received properly? Turn the parity off, change the number of data bits (5) and send a symbol. Can it be sent/received properly? Change the number of data bits back to 8 and disconnect PCs. Connect TxD and RxD lines of the same PC. What happens if you send a symbol or a message?
3. Turn a periodic macro on in the terminal application: [Set Macros] -> write a character -> write a repeating interval (200ms) -> check [Auto Repeat]. Connect an oscilloscope probe to TxD line, display the transmitted waveform and store it (a USB flash drive with FAT32 file system is necessary). What voltage levels represent a logic 1 (mark) and logic 0 (space)? Try changing the baudrate and assess how the waveform is changing.

Pin 1	RI
Pin 2	TXD
Pin 3	RXD
Pin 4	DSR
Pin 5	GND
Pin 6	DTR
Pin 7	CTS
Pin 8	RTS
Pin 9	Power Input /DCD



4. Send binary codes of numbers 0 to 10 (macro prefix $, e. g. $7 sends 0x00000111b) and compare them. Where are start bit and stop bit located? On which positions are the LSB and MSB transmitted? Send an ASCII character and read its binary code from the waveform. Compare the transmitted code with ASCII table ([ASCII table] button in Terminal). How does the waveform change when a smaller number of data bits (5) is set? Why is the character sent incorrectly? Set parity to even and send various characters. Where is the parity bit located and what is its value? How does this value change with odd parity? What is the change in waveform when a different number of stop bits is set? Store selected waveforms to prove your conclusions.