

4. USART	2
Exercise 1.....	2
Exercise 2.....	3
Exercise 3.....	3
Exercise 4.....	4
Exercise 5.....	5
Exercise 6.....	5
Exercise 7.....	5

4. USART

Exercise 1

Write a code to send the classical message “Hello world” through the USART built-in the ATMEGA328P. Configure the USART to 9600 bauds, 8-bit data, 1 stop bit and non-parity bit.

Tips:

- *Use the macro “BAUD” for setting the bauds.*
- *Add the macro “UBRR_VALUE” for calculating the value to be written in the UBRR register. (Use the equation in the L07_USART slides)*
- *Create two functions:*
 - o *USART0_putchar (unsigned char data), for sending one character.*
 - o *USART0_putString (char *strPointer). This function should call the previous one to send all the characters in the string.*

Use Putty as Terminal Software. (<https://www.putty.org/>)

Exercise 2

Write a code to show on the screen the same character received from a keyboard (Echo test).

Tips before starting:

- Create a header file to include all the macros related to the USART, “USART.h”.
 - o Based on the Exercise 01, add to this file the inline function “USART0_Init (uint_16t valueUBRR)”.
- Create a C file to include all the functions related to USART.
 - o Add the two functions created in Exercise 01:

USART0_putchar (unsigned char data)

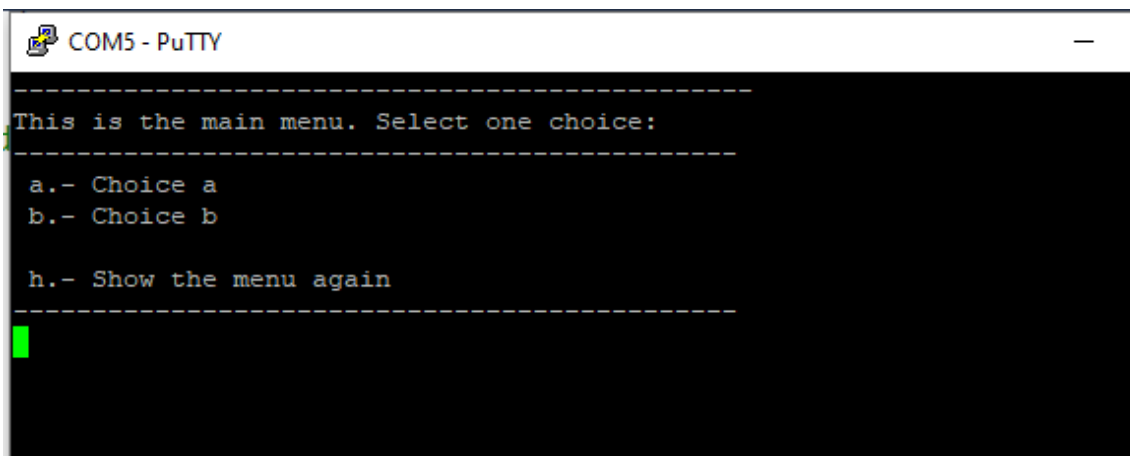
USART0_putString (char *strPointer)

- o Create and add a new function to read a character by means of polling technique, “USART0_getchar (void)”.

Exercise 3

a) Write a code to show on the screen a menu as shown below. Each choice allows printing a different message on the screen.

- When key is ‘a’ prints “You pressed a”
- When key is ‘b’ prints “You pressed b”
- When key is other than ‘a’ or ‘b’, print “I don’t know what to do with that key”.
- When key is ‘h’ clean the screen and show the menu again.

**Tips before starting:**

- The menu must be a function and be called in the setup and every time the ‘h’ key is pressed. Name to this function “mainMenu()” and declare it as “static inline”.
- By sending the following ASCII characters (or the decimal equivalent) you get:

- “/n” and “/r” → Line feed and return page (new line)
- “/f” → Form feed (goes to the start of the following page)

b) Modify the code in order to avoid the blocking “while” included in the function “USART0_getchar ()”.

Tips

- Use a macro, in “USART.h”, for reading the RXC0 bit and to continue only when it is ‘1’.

Exercise 4

Connect a pushbutton to the external interrupt (INT0) in port D, and 4 LED to PORTD (pins 4,5,6 and 7). Use the 9600N1 configuration for the USART.

The system must perform the following actions:

- a. Every time the pushbutton is pressed, a new ASCII character (from 0x20 to 0x7E) will be shown in a serial terminal. Use a transmission based on polling.
- b. In addition, a ASCII character may arrive in anytime by the USART (by means of the keyboard) and must be shown onto the LED (only the 4 LSB bits of the ASCII). Use the RX interrupts.

Draw the proposed electronic schematic.

Tips:

- Firstly, try the a) requirement. Once it works, add the second one, b).
- For testing the b) section, hit the number keys as the corresponding ASCII code are ranged from 0x30-0x39. Therefore, you can watch the binary value in the four LED.

Exercise 5

Connect one pushbutton in an external interrupt, INT0. Use the 9600N1 configuration for the USART and interrupts for TX. Create a code to send the message “Hello world” every time the button is pressed.

- a) Firstly use the `USART0_putString` function. (Blocking)
- b) Then, try to modify the code to avoid the blocking while.

Tips for before starting:

- You should have added two new *static inline* functions in “EXT_INT.h”:
 - o `USART0_enableInterrupt_TX()`
 - o `USART0_disableInterrupt_TX()`
- This exercise reviews a powerful feature of the AVR interrupts: a ISR can enable/disable another type of interrupt. In this case, enable the TX interrupt when the pushbutton is pressed, that is, the INT0 ISR should do it.
- Once the ISR of the TX interrupt send the message, disable the interrupt.

Exercise 6

Write an AVR C program to receive a character from the serial port. If it is a small letter, from ‘a’ to ‘z’, change it to capital letter and transmit it back. Use the 9600N1 configuration for the USART and **interrupt** for RX and TX.

Exercise 7

Connect two pushbuttons each in an external interrupt, INT0 and INT1, and 4 LED to PORTB (3,2,1 and 0). Use the 9600N1 configuration for the USART and interrupts for TX.

- Modify the program created in exercise 4a) in a way that one pushbutton increments the ASCII character and the another decrements it. Show the ASCII character in the serial terminal.
- Besides, the LED in port B must show how many times has been pressed one of the pushbutton (decide what pushbutton you prefer).

Draw the proposed electronic schematic.