

Task 2: Pulse Width Modulation (PWM)

Modify the program in Task 1 to obtain Pulse Width Modulation (PWM). The frequency should be fixed, but the duty cycle should be possible to change. Use two push buttons to change the duty cycle up and down. Use interrupt for each pushbutton. The duty cycle should be possible to change from 0 % up to 100 % in steps of 5 %. Connect the output to an oscilloscope, to visualize the change in duty cycle.

Task 3: Serial communication

Write a program in Assembly that uses the serial communication port0 (RS232). Connect a computer to the serial port and use a terminal emulation program. (Ex. Hyper Terminal) The program should receive characters that are sent from the computer, and show the code on the LEDs. For example, if you send character A, it has the hex code \$65, the bit pattern is 0110 0101 and should be displayed with LEDs On for each 'one'. Use polled UART, which means that the UART should be checked regularly by the program.

Serial communication, Wikipedia: https://en.wikipedia.org/wiki/Serial_communication

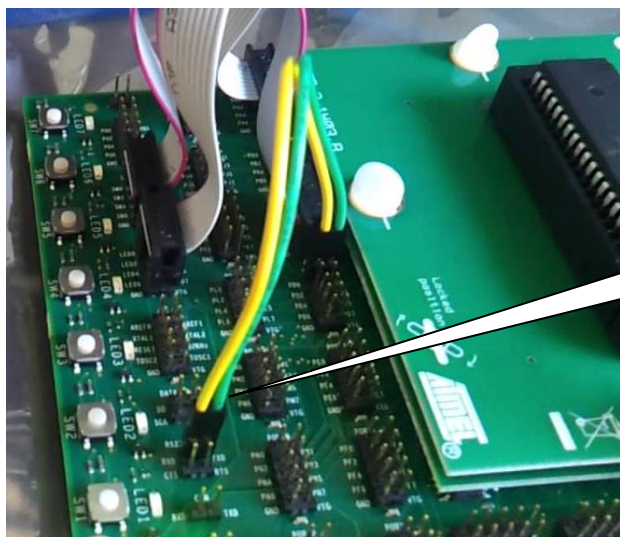
Task 4: Serial communication with echo

Modify the program in task 3 to obtain an echo, which means that the received character should also be sent back to the terminal. This could be used as a confirmation in the terminal, to ensure that the character has been transferred correctly.

Task 5: Serial communication using Interrupt

Do task 3 and 4, but use Interrupt instead of polled UART. (USART, Rx Complete, USART Data Register Empty and USART, Tx Complete)

RS232 serial cable with Male/Female DB9 connectors. Connect to RS232 SPARE on STK600 and to serial port on PC (COM1).



Connect jumper cable between PE0/PE1 and RXD/TXD – RS232 SPARE.