Commands

('M', port, 0, function)	
	A binary "0" corresponds to the target output function which is followed by one byte with the desired output function. There are five predefined functions: 0 = sine, 1 = triangle, 2 = sawtooth, 3 = rectangle, and 4 = DC. Other values can be used for custom memories. Depending on the used FPGA education board a large number of custom wave memories could be utilized, but for most applications it should be sufficient to limit the number of possible functions to 256.

M', port, 1, frequency high, frequency low)	
	A binary "1" after the target out-port indicates that the frequency is going to be set. Due to the range of 512Hz two bytes are needed to specify the output frequency.

('M', port, 2, 0..255)	
	A "2" corresponds to the internal amplifier of the signal which can be adjusted between 0 to 255 within one byte, which is interpreted as percentage value. Thus it is possible to scale the signals out of range to generate different signal forms. For example, a tuned out triangle produces a trapezoid.

('M', port, 3, phase high, phase low, related port)	
	The last type "3" is the phase of the target output, which can be adjusted in degrees between 0 to 360 with 2 bytes followed by one byte for the related outport to which the phase should be adjusted. The update logic then compares the next requested PWM step of both ports and sets the desired port in relation to the reference port.

('M', port, 4, parameter index 0..3)	
	Additionally the state of each port can be requested by sending a binary "4" followed by one byte with the requested type. As a result two bytes will be sent back with the desired internal value.

(`C', memory index, address high, address low, data high, data low)	
	The wave memories can be filled by starting a command with a "C" and then followed by one Byte for the index of the custom wave memory and 2 Bytes for the address and 2 Bytes for the PWM value. A 9~Bit PWM signal refers to a 512x9 Bit memory and a 10 Bit PWM signal refers to a 1024x10 Bit memory. To fully update an internal memory 6 Bytes multiplied by the 1024 or 512 PWM steps are required. In case of a 9 Bit PWM the internal memory can be updated with 65 Hz. For a 10 Bit PWM the update rate decreases to 32 Hz via UART. Based on an Arduino UNO with an SPI clock of 8 MHz the internal custom memory can be updated at 325 Hz for a 9 Bit PWM or 162 Hz for a 10 Bit PWM. The maximum update rate at 50 MHz SPI clock is 2034 Hz for a 9 Bit PWM or 1017 Hz for a 10 Bit PWM. Whenever a custom memory is read by an PWM generator while a new value is written, the previous content will be processed to the output.