ModCon Serial Protocol

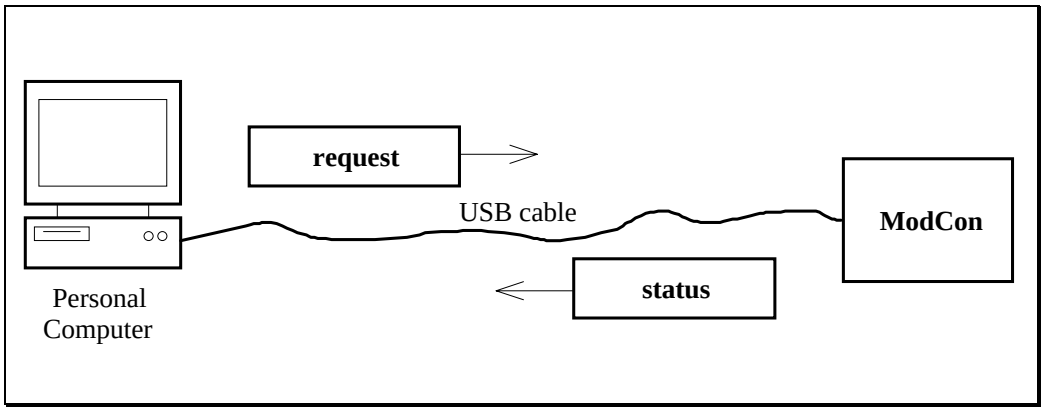
Communication between MC9S12A256B/512 and PC. Conceptual view. Packet structure. Commands. Acknowledgement.

Introduction

This document outlines the communication protocol between the PC and the ModCon microcontroller board.

Conceptual View

The PC or ModCon can initiate a transfer of serial information at any time (asynchronously) using the Universal Serial Bus (USB). Information is transferred in packets, with each packet consisting of 5 bytes. A conceptual view of communication between the PC and the ModCon is shown below:



Conceptual view of information transfer between PC and ModCon

Figure P.1

Sometimes a transmitted packet will request a response from the receiver. In this case, a response to the command is initiated once the command has been carried out (or attempted to be carried out).

P.2

PC and ModCon Communication

A typical picture of the PC and ModCon communicating is shown below:

PC and ModCon communicating

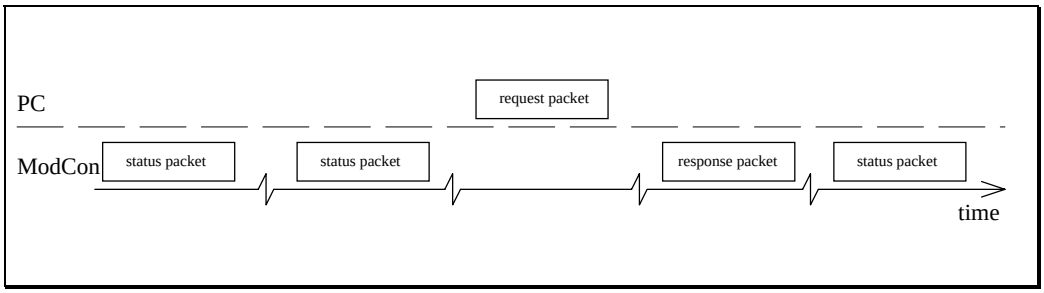


Figure P.2

The ModCon is continually updating the PC with information regarding the state of the various inputs and outputs of the module, such as push buttons, LEDs, A/D results and PWM duty cycles. This state information is only sent on start up and when the state of the ModCon module changes – this prevents the PC from receiving many packets with the same information, which wastes processing time.

The PC will occasionally send a request packet to the ModCon, asking it to carry out some task. It may or may not specify for the ModCon to acknowledge that the request has been carried out successfully. If the PC does specify an acknowledgement, then the ModCon will respond with a response packet.

Packet Structure

Each packet of information contains 5 bytes as follows:

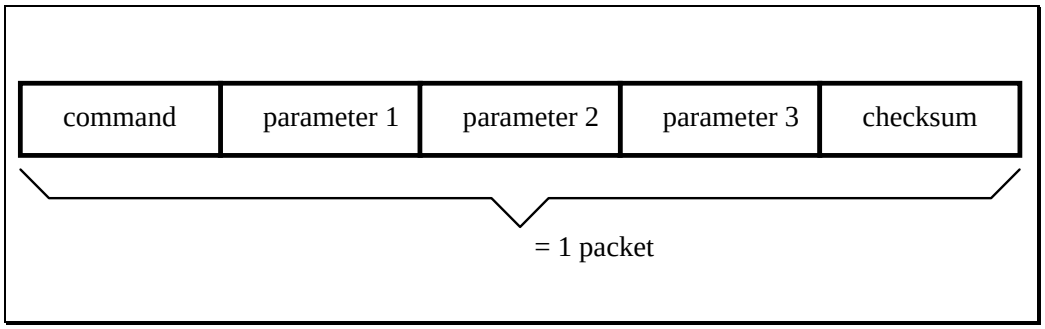


Figure P.3

The command byte contains a command number, as listed in the tables that follow. The use of the three parameter bytes depends on the particular command being sent. The checksum is the exclusive-or (XOR) of the four preceding bytes and is used as a simple means to detect most transmission errors (corruption of a packet) and for packet synchronization.

Packets Transmitted from ModCon to PC

The following table lists the packets that can be sent by the ModCon to the PC:

Command	Data Stream
0x04	**ModCon Startup** Parameter 1: 0 Parameter 2: 0 Parameter 3: 0 *Note: The ModCon will issue this command upon startup to allow the PC to update the interface application and the ModCon. Typically, setup data will also be sent from the ModCon to the PC.*
0x08	**EEPROM – Stored data** Parameter 1: address LSB Parameter 2: address MSB Parameter 3: data
0x09	**Special – Master alarm** Parameter 1: 5 Parameter 2: 0 Parameter 3: 2 *Note: If the PC receives this then the ModCon is overloaded.*
0x09	**Special – Number of Starts** Parameter 1: 's' = starts Parameter 2: LSB Parameter 3: MSB
0x09	**Special – ModCon version** Parameter 1: 'v' = version Parameter 2: Major Version Number Parameter 3: Minor Version Number (out of 100) *Note: e.g. V1.3 has a major version number of 1 and a minor version number of 30.*
0x0A	**Protocol – Mode** Parameter 1: 1 Parameter 2: 0 = asynchronous 1 = synchronous Parameter 3: 0
0x0B	**ModCon Number** Parameter 1: 1 Parameter 2: LSB Parameter 3: MSB *Note: The ModCon number is an unsigned 16-bit number.*
0x0C	**Time** Parameter 1: 'i' = intervals Parameter 2: seconds Parameter 3: minutes

0x0D	**ModCon Mode** Parameter 1: 1 Parameter 2: LSB Parameter 3: MSB *Note: The ModCon mode is just an unsigned 16-bit number – it is usually declared as an enumerated type.*
0x30	**ATD – Value** Parameter 1: analog channel (0-15) Parameter 2: LSB Parameter 3: MSB
0x31	**ATD – Mode** Parameter 1: analog channel (0-15) Parameter 2: 0 = "raw / calibrate" mode, 1 = "normal" mode Parameter 3: 0
0x50	**Analog Input – Value** Parameter 1: analog channel (0-7) Parameter 2: LSB Parameter 3: MSB
0x51	**Analog Output – Value** Parameter 1: analog channel (0-1) Parameter 2: LSB Parameter 3: MSB

0x60	**Wave** Parameter 1: 0 = status Parameter 2: output channel (0-1) Parameter 3: 0 = off 1 = on
0x60	**Wave** Parameter 1: 1 = get waveform Parameter 2: 0 = sine 1 = square 2 = triangle 3 = sawtooth 4 = noise 5 = arbitrary Parameter 3: 0
0x60	**Wave** Parameter 1: 2 = get frequency Parameter 2: LSB Parameter 3: MSB Note: The frequency that is received is $f \times 256$ (truncated to an integer).
0x60	**Wave** Parameter 1: 3 = get amplitude Parameter 2: LSB Parameter 3: MSB Note: The amplitude that is received is $A \times 204.8$ (truncated to an integer).
0x60	**Wave** Parameter 1: 4 = get offset Parameter 2: LSB Parameter 3: MSB Note: The offset that is received is $DC \times 204.8$ (truncated to an integer).

For example, if the ModCon were informing the PC that its version number is 1.3, then the packet would be:

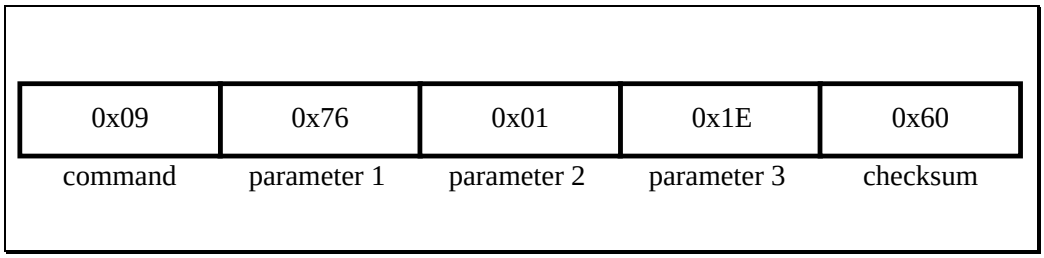


Figure P.4

Packets Transmitted from PC to ModCon

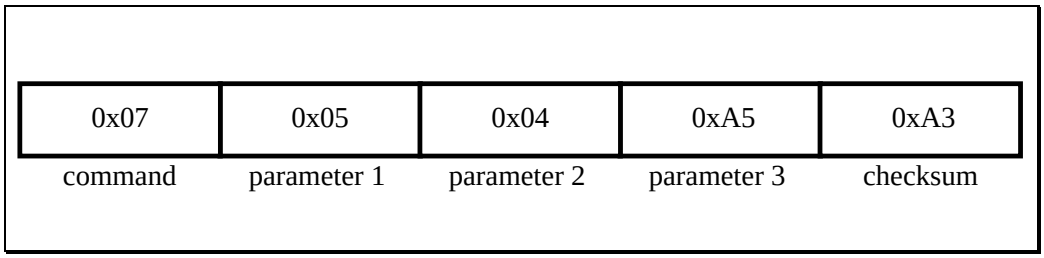
The following table lists the packets that can be sent by the PC:

Command	Data Stream
0x04	**Special - Get startup values** Parameter 1: 0 Parameter 2: 0 Parameter 3: 0 *Note: The PC will issue this command upon startup to retrieve the state of the ModCon to update the interface application.*
0x07	**EEPROM – Program byte** Parameter 1: address LSB Parameter 2: address MSB Parameter 3: data *Note: The address has the range of 0x0400 to 0x1000.* *An address of 0x1000 will ERASE the* entire *EEPROM.*
0x08	**EEPROM – Get byte** Parameter 1: address LSB Parameter 2: address MSB Parameter 3: 0 *Note: The address has the range of 0x0400 to 0x0FFF.*
0x09	**Special – Get values** Parameter 1: 'g' Parameter 2: 'i' Parameter 3: CR *Note: The PC may issue this command upon startup to ensure switch positions are correct at the PC end. The ModCon will send switch, counter and analog values, which the PC may or may not use.* *Note: This "packet" can be sent from HyperTerminal by typing TAB+"gi"+CR+LF. (Mnemonic is [g]et [i]nfo)*
0x09	**Special - Start bootloader** Parameter 1: 'b' Parameter 2: 'l' Parameter 3: CR *Note: This "packet" can be sent from HyperTerminal by typing TAB+"bl"+CR+LF. (Mnemonic is [b]oot[l]oader)*
0x09	**Special – Toggle debug mode** Parameter 1: 'd' Parameter 2: 'j' Parameter 3: CR *Note: This "packet" can be sent from HyperTerminal by typing TAB+"dj"+CR+LF. (Mnemonic is [d]ebug [j]unk)*

0x09	**Special – Number of starts** Parameter 1: ‘s’ Parameter 2: ‘}’ Parameter 3: CR *Note: This “packet” can be sent from HyperTerminal by typing TAB+”s}”+CR+LF. (Mnemonic is [s]tarts [}])*
0x09	**Special – Get version** Parameter 1: ‘v’ Parameter 2: ‘x’ Parameter 3: CR *Note: This “packet” can be sent from HyperTerminal by typing TAB+”vx”+CR+LF. (Mnemonic is [v]ersion [x])*
0x0A	**Protocol – Mode** Parameter 1: 1 = get Protocol mode 2 = set Protocol mode Parameter 2: 0 = asynchronous for a “set”, 0 for a “get” 1 = synchronous for a “set”, 0 for a “get” Parameter 3: 0
0x0B	**ModCon Number** Parameter 1: 1 = get ModCon number 2 = set ModCon number Parameter 2: LSB for a “set”, 0 for a “get” Parameter 3: MSB for a “set”, 0 for a “get” *Note: The ModCon number is an unsigned 16-bit number.*
0x0D	**ModCon Mode** Parameter 1: 1 = get ModCon mode 2 = set ModCon mode Parameter 2: LSB for a “set”, 0 for a “get” Parameter 3: MSB for a “set”, 0 for a “get” *Note: The ModCon mode is just an unsigned 16-bit number – it is usually declared as an enumerated type.*
0x31	**ATD – Set Mode** Parameter 1: analog channel (0-15) Parameter 2: 0 = “raw / calibrate” mode, 1 = “normal” mode Parameter 3: 0
0x32	**ATD – Get State** Parameter 1: analog channel (0-15) Parameter 2: 0 Parameter 3: 0

0x60	**Wave** Parameter 1: 0 = get status Parameter 2: 0 Parameter 3: 0
0x60	**Wave** Parameter 1: 1 = set waveform Parameter 2: 0 = sine 1 = square 2 = triangle 3 = sawtooth 4 = noise 5 = arbitrary Parameter 3: 0
0x60	**Wave** Parameter 1: 2 = set frequency Parameter 2: LSB Parameter 3: MSB Note: The frequency that is sent is $f \times 256$ (truncated to an integer).
0x60	**Wave** Parameter 1: 3 = set amplitude Parameter 2: LSB Parameter 3: MSB Note: The amplitude that is sent is $A \times 204.8$ (truncated to an integer).
0x60	**Wave** Parameter 1: 4 = set offset Parameter 2: LSB Parameter 3: MSB Note: The offset that is sent is $DC \times 204.8$ (truncated to an integer).
0x60	**Wave** Parameter 1: 5 = on Parameter 2: 0 Parameter 3: 0
0x60	**Wave** Parameter 1: 6 = off Parameter 2: 0 Parameter 3: 0
0x60	**Wave** Parameter 1: 7 = active channel for settings Parameter 2: 0 = Channel 1 1 = Channel 2 Parameter 3: 0

For example, if the PC were requesting the ModCon to write 0xA5 to EEPROM address 0x405, then the packet would be:



Example packet sent from PC

Figure P.5

Packet Acknowledgement

The command ID has bit 7 (the most significant bit) reserved for packet acknowledgement purposes. The transmitter sets bit 7 to 1 to request an acknowledgement, and leaves it cleared to 0 for no required acknowledgement.

The receiver, upon receiving a packet with an acknowledgement request, will firstly attempt to carry out the requested command. It will then either set bit 7 to indicate that the command was carried out successfully (an ACK), or clear it to 0 to indicate that the command could not be carried out (a NAK), and send the packet back to the requester.

For example, if the PC sends the following packet to write 0xA5 to EEPROM address 0x405:

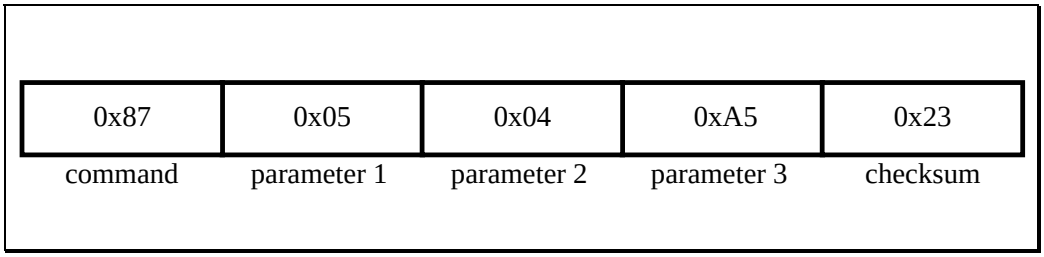


Figure P.6

then the ModCon will respond with the same packet, indicating it was carried out successfully.

However, if the PC asked to write to address 0x1001 (which does not physically exist), then the ModCon would respond with:

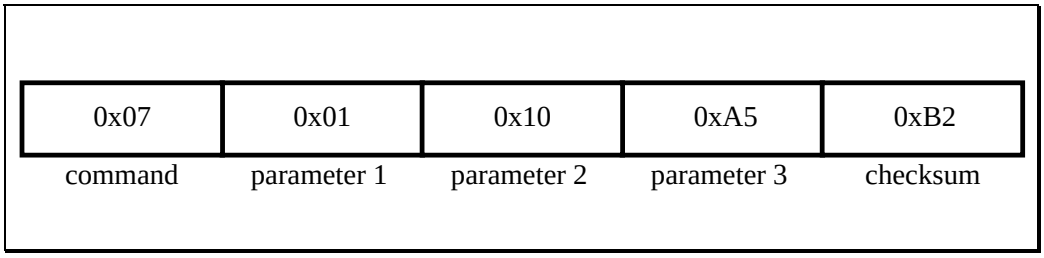


Figure P.7

Note that the most significant bit of the command has been cleared, indicating that the requested command could not be carried out.

PC Communication Parameters

Baud Rate and Virtual Com Ports

The USB drivers on the PC side implement a virtual COM port.

If the ModCon board has a MC9S12A256B, then the PC uses COM5.

If the ModCon board has a MC912A512, then the PC uses COM6.

Baud Rate and Data Format

The baud rate used is selectable and is either 38400 or 115200 baud.

The data format used is 8N1 (8 data bits, no parity, 1 stop bit).