

GENERAL OVERVIEW

The ECU-121 provides several methods to control data flow between it and the host. In this application note, we will present and discuss examples of how a host communicates with the ECU-121 using hardware handshaking. Examples will be given in Visual Basic using the MSCOMM control. A complete simple application can be downloaded from the Bowser Electronics website.

PORT SETTINGS

The ECU-121 uses a fixed set of communication parameters. The host com port must be set the same. The following shows the initial MSCOMM properties.

- 'ComPort' – Whatever port your PC uses
- 'Settings' – '57600,N,8,1'
- 'Handshaking' – 'comRTS'
- 'InBufferSize' – 2048
- 'OutBufferSize' – 512
- 'InputMode' – 'comInputModeText'
- 'InputLen' – 0
- 'SThreshold' – 0
- 'RThreshold' – 0

All data is sent and received as strings. Therefore, each byte of binary data is represented as a character.

FLOW CONTROL

The ECU-121 implements the CTS signal on pin 6. This signal needs to be buffered and inverted using external circuitry to be understood by the host (Refer to application note AN-001). The minimum time to hold the CTS Signal low can be set by using command '0x16'. The default time is 4.096ms. If your PC misses the 'comEvCTS' Events, increase this value by 5mSec at a time until your PC no longer misses the event.

Host hardware control is implemented in the MSCOMM control using the 'Handshaking' property and setting it to 'comRTS' (3) prior to opening the com port. The CTS signal only has an effect on data transmission. When sending data from host to chip, transmission will only take place when the 'CTSHolding' property is 'True'. As the ECU-121 receives data from the vehicle bus, it is retransmitted byte-by-byte to the RS-232 bus. Once the message is complete, CTS is toggled for the minimum time and the system screens and processes the message as defined by command 0x13.

When external message handling is enabled, the CTS transition indicates the start of a 100ms timer within which the reply message must be sent. The first byte of the reply must be a single byte indicating how many response messages will be sent.

Each message will then be transmitted as follows: The first byte of each message will indicate how many bytes are in the message not including the length byte. The message must be properly formatted for the vehicle bus and CRC/Checksum appended. The ECU-121 performs no checks or processing of the message and simply repeats it on to the vehicle bus. Once the final byte is received, the CTS line is pulsed while the response messages are transmitted on to the vehicle bus. The end of the CTS pulse indicates to the host to begin to transmit the next message. This repeats for each message. When all messages have been sent, the system returns to monitoring the vehicle bus for the next incoming message.

PROGRAMMING EXAMPLES

Implementing communications as a self contained function allows data to be requested and returned from/to any point and any form within your program and provides the most flexibility. The ECU-121 Function Library, in the example application available from the Bowser Electronics website, provides a sample communication function, supporting functions, as well as functions for calculating CRC and Checksum. This communication routine can be used to send commands and receive replies to/from the ECU-121.

Receiving from and replying to vehicle requests is easiest handled through Event programming. Your program should build a table array of requests/ responses that your program updates to suit your requirements. When a vehicle request is made, your event handler will be triggered for the MSCOMM control comEvCTS event. If the CTSHolding property is true, retrieve the completed request from the receive buffer, lookup the response in the table, and transmit messages as needed. This method allows the most recent data to be sent as quickly as possible to the ECU-121. For a specific example of how to implement the MSCOMM event, see the ECU-121 demo program available on the Bowser Electronics website.

Below are examples of code to transmit and receive data to/from the ECU-121. If these are to be implemented in a single function, code should be included to wait for the CTSHolding to transition to 'False' between the transmit and receive blocks.

Sending Data to the ECU-121

```
Dim X, Y, SendStream As String
'Convert Text type Hex String to Binary type data
'SendStream would contain '05686AF10100'
Y = ""
For X = 1 To Len(SendStream) Step 2
    Y = Y & Chr(Hex2Dec(Mid$(SendStream, X, 2)))
Next
'Clear Input and Output Buffers
MSComm1.InBufferCount = 0
MSComm1.OutBufferCount = 0
'Set a start time reference
X = Timer
'Wait for chip ready to accept data
Do
'Exit if more than 5 seconds
If Timer > X + 5 Then Exit Function
Loop Until MSComm1.CTSHolding = True
'Move data to output buffer
MSComm1.Output = Y
```

Reading Data from the ECU-121

```
Dim X, Y, Z
'Set a start time reference
X = Timer
'Wait for chip finished
Do
'Exit if more than 5 seconds
If Timer > X + 5 Then Exit Function
Loop Until MSComm1.CTSHolding = True
'Read Input buffer
X = MSComm1.Input
'Convert to Text type data
Y = ""
For Z = 1 To Len(X)
    Y = Y & Dec2Hex(Asc(Mid$(X, Z, 1)))
Next
```