

Serial Host Interface

6.0 Introduction

This serial drive provides a simple interface to set drive parameters and to control a DC motor. These drive parameters can be saved into the drives EEPROM and restored automatically on the drive when it powers on. The DC104-SER drive communicates with similar commands as the DC104 PC104 model. The commands and parameters are put into a packet of data. This packet has the same structure for each parameter.

6.1 Communication between a PC and the DC104 drive

The PC communicates to the DC104 drive via a standard RS-232 cable with a null modem connection (the TX and RX lines are crossed). The use of a USB to RS-232 converter allows computers that do not have a RS-232 port to interface to the serial port of the drive. When a computer and drive are connected via the serial port, the computer is the master and initiates all communication to the drive. The computer can either read or write parameters to the drive. When reading a parameter, the computer generates a packet that requests a read of a parameter then the drive will respond to this request with the parameters data in similar packet. When writing a parameter, the computer generates a packet that requests the drive to save the parameters new value the drive will respond with an ACK if it accepts and completes the writing operation. A packet is a predefined byte structure that typically starts with a byte called a header. This header identifies the beginning of a packet, the middle of a packet has the information and the end of the packet is a checksum of all the information. The following are the two packet structures the DC104 drive uses:

Reading Command Packet Structure

```
02h --> Header called STX
AAh --> Reading command (direction)
xxh --> Parameter number 0 to 255
xxh --> Checksum, the sum of 0xAAh + parameter number + high byte + low byte
```

Response to a reading command will be the data requested in the following packet format:

```
02h --> Header called STX
xxh --> Parameter number 0 to 255
xxh --> High byte data

xxh --> Low byte data

xxh --> Checksum, the sum of parameter number + high byte + low byte
```

Writing Command Structure

```
02h --> Header called STX
55h --> Writing command
xxh --> Parameter number 0 to 255
xxh --> High byte data

xxh --> Low byte data

xxh --> Checksum, the sum of 0x55h + parameter number + high byte + low byte
```

Response to a writing command will be either an ACK (hex value of 06h) or a NAK (hex value of 15h).

The two packet structures shown above have two important bytes. The start of the packet is always started with a STX, which has a ASCII value of 2. The second byte of the packet identifies the direction of the information as either reading from the drive or writing to the drive. The packets always end with a checksum, which is the sum of all the bytes between the STX and Checksum. The checksum is not two's complemented.

The serial port needs to be opened at 19200 baud, no parity, 8 data bits and 1 stop bit. The serial port also cannot ignore null or ASCII zero characters.

```
' -----
' Description: Open the RS-232 Serial comm port
' Inputs: CommPort = Comm number of port to open
'
' Outputs: none
'
' Returns: none
'
' -----
Private Sub OpenCommPort(ByRef CommPort As Short)

    MSComm1.CommPort = CommPort
    MSComm1.Handshaking = MSCommLib.HandshakeConstants.comNone 'no handshaking
    MSComm1.Settings = "19200,N,8,1"
    MSComm1.PortOpen = True
    MSComm1.RThreshold = 1
    MSComm1.SThreshold = 0
    MSComm1.InputLen = 0

End Sub
```

Figure 1, Above is a software example in Visual Basic for opening a serial port.

```

'-----
' Description: This function will build the packet following the protocols definitions
'
' Inputs: DC104Parameter = the integer value of the parameter, must be 1 to 255
'         DC104Data      = the value of the data, if reading then should be 0, must be
'         between 0 to 65535
'         DC104Direction = Direction needs to be 0xAA for read or 0x55 for write
'
' Returns: BuildDC104Packet = The packet string
'-----
Private Function BuildDC104Packet(ByVal DC104Parameter As Short, ByRef DC104Data As long, ByRef
DC104Direction As Byte) As String

If DC104Data > 65535 Then
    DC104Error = "Data too large to fit in high and low byte"
    BuildDC104Packet = ""
    Exit Function
End If

Select Case DC104Direction
    Case ReadDC104
        BuildDC104Packet = Chr(2)
        BuildDC104Packet = BuildDC104Packet & Chr(DC104Direction)
        BuildDC104Packet = BuildDC104Packet & Chr(DC104Parameter)
        BuildDC104Packet = BuildDC104Packet & Chr((DC104Parameter + DC104Direction) Mod 256)
        DC104Error = ""

    Case WriteDC104
        BuildDC104Packet = Chr(2)
        BuildDC104Packet = BuildDC104Packet & Chr(DC104Direction)
        BuildDC104Packet = BuildDC104Packet & Chr(DC104Parameter)
        BuildDC104Packet = BuildDC104Packet & Chr(Int(DC104Data / 256)) 'high byte
        BuildDC104Packet = BuildDC104Packet & Chr(DC104Data Mod 256) 'low byte
        BuildDC104Packet = BuildDC104Packet & Chr((DC104Parameter + Int(DC104Data / 256) +
(DC104Data Mod 256) + DC104Direction) Mod 256)

    Case Else
        DC104Error = "Direction is not valid, needs to be 0xAA for read or 0x55 for write"
        BuildDC104Packet = ""
        Exit Function
End Select
End Function

```

Figure 2, Above is a software example in Visual Basic for building a DC104 serial packet in a string. This string is then sent out the serial port to the DC drive.

```

' -----
' Description: This function will check the packet to see if it is valid. It
'             will check to see if the 02h STX is at the header, the parameter
'             is in the valid range and that the checksum is correct.
'
' Inputs: Packet = the string of characters that make up the packet
'           02h --> Header called STX
'           xxh --> Parameter
'           xxh --> High byte data
'           xxh --> Low byte data
'           xxh --> Checksum, sum of parameter+high byte+low byte+direction
'
' Returns: CheckDC104Packet = True if passes test and False if packet is invalid
' -----
Private Function CheckDC104Packet(ByRef Packet As String) As Boolean
Dim CheckSum As Byte

    If Len(Packet) <> 5 Then
        CheckDC104Packet = False
        Exit Function
    End If
    If Mid(Packet, 1, 1) <> Chr(STX) Then
        CheckDC104Packet = False
        Exit Function
    End If
    If Mid(Packet, 2, 1) < Chr(1) Or Mid(Packet, 2, 1) > Chr(MaximumNumberParameters) Then
        CheckDC104Packet = False
        Exit Function
    End If
    CheckSum = (Asc(Mid(Packet, 2, 1)) + Asc(Mid(Packet, 3, 1)) + Asc(Mid(Packet, 4, 1))) Mod 256
    If CheckSum <> Asc(Mid(Packet, 5, 1)) Then
        CheckDC104Packet = False
        Exit Function
    End If
    CheckDC104Packet = True
End Function

```

Figure 3, Above is a software example in Visual Basic for checking a DC104 serial packet coming from DC drive after a parameter read request.