Super Exciting Hardware Device

RS232 Serial Communication

Contents

1 Introduction

This is the fictional documentation which accompanies a likely expensive piece of hardware you bought. The hardware advertises RS-232 communication, so you're very excited to write some LabVIEW code to interact with it so that you can avoid the buggy, frustrating vendor-supplied software with which it shipped.

2 Command Structure

This section describes the structure of the string which is communicated to/from the device for each command. Each command has the same set of components, which are detailed below:

- <Command> : This is the name of the command, for example, "STATUS". All commands are case sensitive.
- <TC> : This is the 'Termination Character'. It is a line end, or '\n' character.
- <Data> : This is a string representing the data transmitted with the command as an instruction. It may be an empty string.
- <Response> : This is the reply data sent with the message response from the device. It may be an empty string.

Computer to Device Format

<Command>,<Data><TC>

Device to Computer Format

<Command>,<Data>,<Response><TC>

3 List of Commands

3.1 STATUS

<Data> : None.
<Response> : 'Hello World!' string, without quotation marks.

3.2 ADD

<Data> : List of comma separated floating point values.
<Response> : Floating point value equal to the sum of the data values.

3.3 MUL

<Data> : List of comma separated floating point values.
<Response> : Floating point value equal to the product of the data values.

4 Error Response

If an unexpected command is received by the device, the device will transmit "Error" in response.

5 LabVIEW Tips

Below are some tips that you might find useful in implementing a LabVIEW 'driver' for the Arduino device. Feel free to try first without reading, if you want slightly more of a challenge!

- When the device is first communicated with, it sends a starting message. You need to handle this message before you can get it to respond properly. Consider using 'VISA Clear'.
- The error you are most likely to receive is the Timeout error, which in LabVIEW has code number -1073807339.
- Ensure that your VI will allow you to send an empty array of data to the Add and Multiply functions, and that the data returned is as you expect.
- Although it's possible to use the 'Bytes at Port' method to try and read a specific number of bytes (and this is very commonly seen in real drivers, and on forum posts asking for help), if you have the ability to use a termination character, you definitely should. Ensure that the number of bytes you wire to the 'VISA Read' node is larger than the number you expect to read in one message if using a termination character.