THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

Serial Port Application

User’s Guide

Mladen Milošević

Wil Gilmore

Table Of Contents

1. Overview

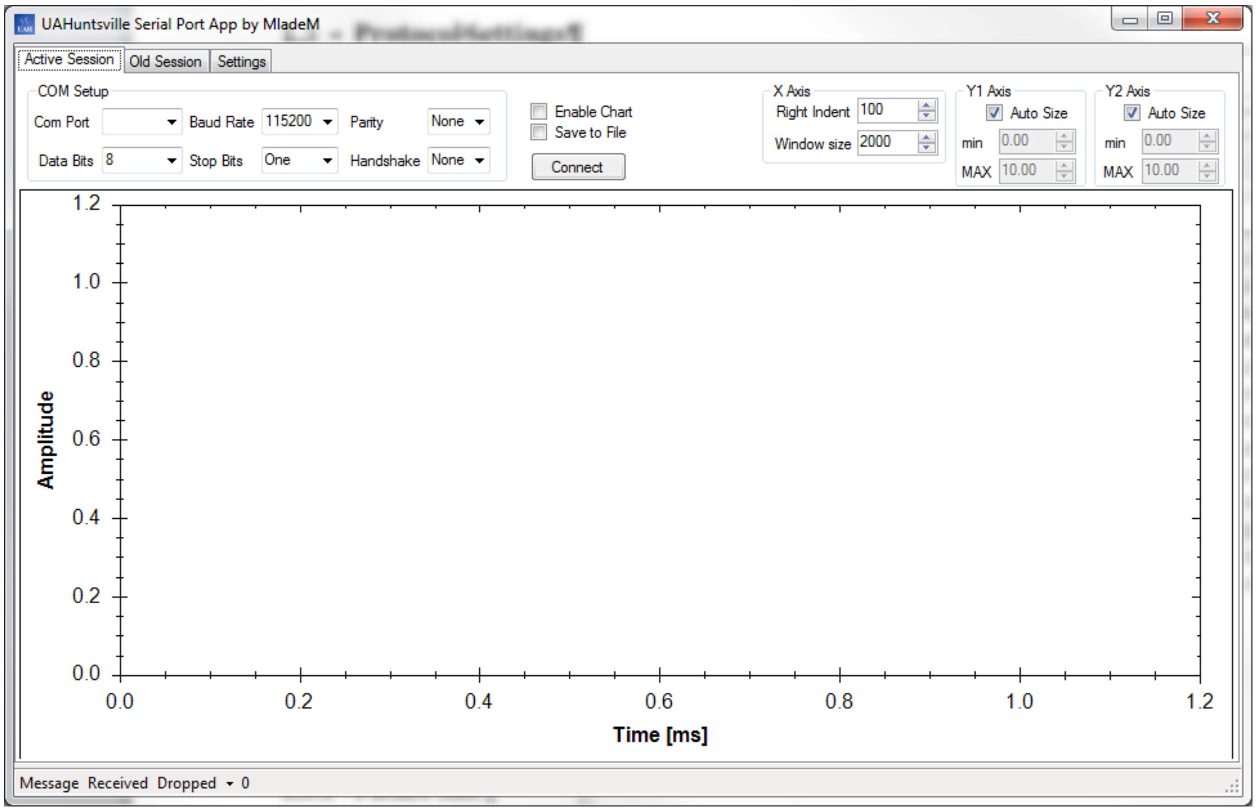
The UAHuntsville Serial Port Application (SPA) is a Windows application for plotting and logging data received via a PC's serial COM port. It is typically used to plot and record data that coming from an embedded platform that connected to the PC using an RS232 communication interface. The Serial Port Application can be configured to receive data packets with a varying number of channels (each channel is plotted as a line on the graph), with a configurable type of data (e.g., uint8, uint16, uint32, etc). In addition to real-time plotting of the data from a COM port, the SPA supports loading of pre-recorded data from a file and plotting them on a graph.

The purpose of this document is to provide the user with a clear, concise and easily understood manual of the application, referencing in detail each feature of the application, as well as a complete walkthrough of the application using a sample scenario to demonstrate how the application may be used. This manual will first introduce the user to the basics of the application, explaining how to configure serial port, graph, and data packet settings, as well as connecting to the serial port itself. Afterwards, additional features such as loading and saving data will be explored. Once all of the features of the application have been introduced, a sample walkthrough of the application will be explored. In addition, a troubleshooting section will be provided for diagnosing and resolving common user issues with the application.

This document assumes the user has basic knowledge on serial communication and associated terms, including but not limited to: baud rates, parity, stop bits and checksums.

2. Getting Started

When starting the application, the user is greeted with the main screen, as shown in **Error! Reference source not found.**. The main window of the application consists of three tabs. The *Active Session* tab is where the currently received data can be plotted and is it is the active tab when the application is started. Additionally, the COM port setup can be configured from this tab, which will be covered later. The *Old Session* tab is primarily for loading and displaying previously recorded data. Finally, the *Settings* tab is for configuring various application settings, including graph and protocol settings. These settings and their functions will also be covered later.


UAHuntsville Serial Port App by MladeM

Active Session | Old Session | Settings

COM Setup: Com Port; Baud Rate 115200; Parity None; Data Bits 8; Stop Bits One; Handshake None

Enable Chart; Save to File; Connect

X Axis: Right Indent 100; Window size 2000

Y1 Axis: ✓ Auto Size; min 0.00; MAX 10.00

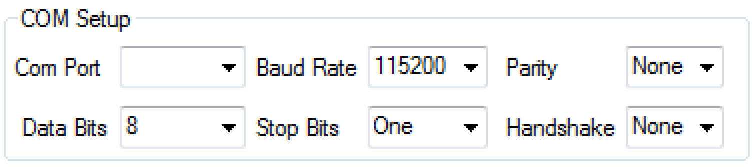
Y2 Axis: ✓ Auto Size; min 0.00; MAX 10.00

Message Received Dropped 0

Figure 1: Serial Port Application Main Window

1.1 COM Setup

A close-up of the *COM Setup* is shown below in Figure 2. From here, the user can configure various communication settings. Please see the "troubleshooting" section (Section 5) if problems occur with COM port setup.


COM Setup

Com Port; Baud Rate 115200; Parity None

Data Bits 8; Stop Bits One; Handshake None

Figure 2: COM Setup

1.1.1 Com Port

The user is able to select which serial port to listen to for incoming data. The application automatically lists the ports available on the computer in the drop-down menu. Be default, no port is selected. The user must determine the port packets are to be received on.

1.1.2 Baud Rate

The user is able to configure the baud rate by selecting one of the baud rates from the *Baud Rate* drop-down menu. It is up to the user to ensure that the baud rate selected matches the baud rate of the device transmitting the data packet.

1.1.3 Parity

The user is able to configure the parity selecting one of the configurations from the *Parity* drop-down menu. The application supports odd, even and no parity. The user must ensure that the selected parity matches that of the data packet.

1.1.4 Data Bits

The user is able to configure the number of data bits by selecting from the *Data Bits* drop-down menu. The application supports seven and eight bit data bits. The user must ensure that the selected number of data bits matches that of the number of data bits within the data packet.

1.1.5 Stop Bits

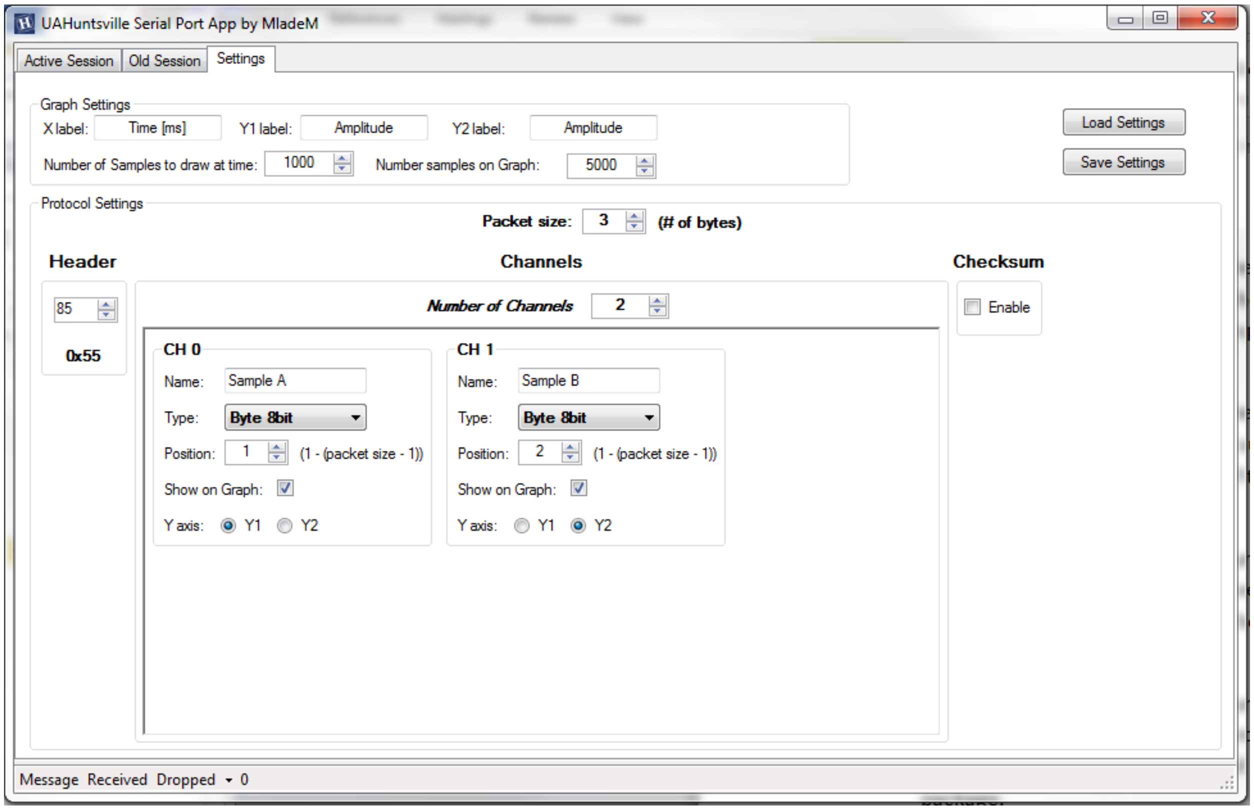
The user is able to configure the number of stop bits by selecting from the *Stop Bits* drop-down menu. The application supports zero, one, two and one and a half stop bits. The user must ensure that the selected number of stop bits matches the number of stop bits in the data packet.

1.1.6 Handshake

The user is able to configure the handshake protocol by selecting from the *Handshake* drop-down menu. The application supports XOn/XOff, RequestToSend, and RequestToReceive handshaking protocols. The user must ensure that the handshake protocol selected matches the protocol used for the data packet.

1.2 Graph Settings

The graph settings for the application can be found in the *Settings* tab, pictured below in Figure 3. From this tab, graph and protocol settings can be configured. This section deals with the graph settings. A close-up of the graph settings can be seen in Figure 4.


UAHuntsville Serial Port App by MladeM

Active Session | Old Session | Settings

Graph Settings
X label: Time [ms] Y1 label: Amplitude Y2 label: Amplitude
Number of Samples to draw at time: 1000 Number samples on Graph: 5000

Load Settings
Save Settings

Protocol Settings
Packet size: 3 (# of bytes)

Header
85
0x55

Channels
Number of Channels: 2

CH 0
Name: Sample A
Type: Byte 8bit
Position: 1 (1 - (packet size - 1))
Show on Graph: ✓
Y axis: [✓ Y1] Y2

CH 1
Name: Sample B
Type: Byte 8bit
Position: 2 (1 - (packet size - 1))
Show on Graph: ✓
Y axis: Y1 [✓ Y2]

Checksum
Enable

Message Received Dropped 0

Figure 3: Settings Tab

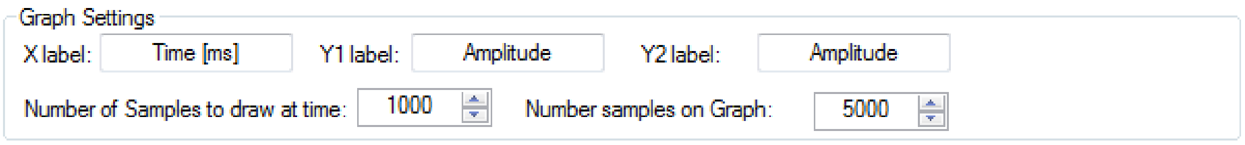

Graph Settings
X label: Time [ms] Y1 label: Amplitude Y2 label: Amplitude
Number of Samples to draw at time: 1000 Number samples on Graph: 5000

Figure 4: Graph Settings

1.2.1 Axis Labels

The user is able to change the X and Y-axis names by typing in names in the *X label, Y1 label* and *Y2 label* text boxes. These are the names that will be displayed on the axes of the graph when data is plotted.

1.2.2 Number of Samples to Draw at a Time

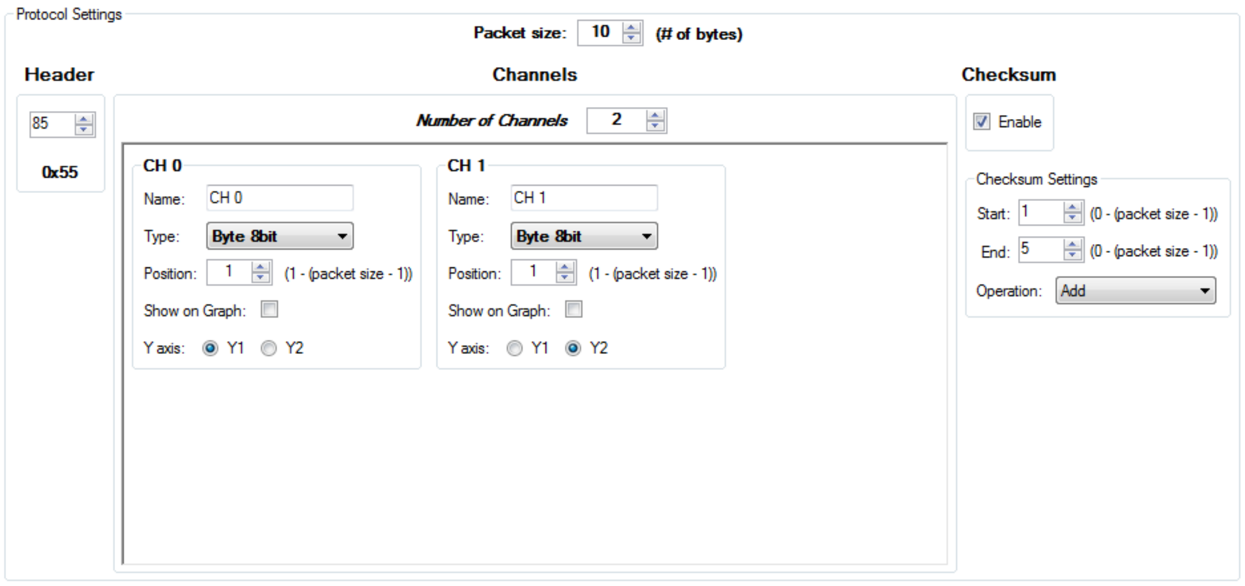
The user can select the number of samples that will be plotted at a time. In other words, the application will wait until the specified number of samples has been collected on the serial port before displaying the samples on the graph.

1.2.3 Number of Samples on Graph

This is the number of samples that may appear on the graph window at one time. For example, if 2000 is selected, the graph will display at most 2000 samples. If more than 2000 samples are collected, the graph will display the 2000 most recently collected samples on the graph.

1.3 Protocol Settings

The user is able to configure application settings pertaining to the size and format of the data within the packet itself, including the expected size, expected first byte, and the expected checksum if enabled. In addition, the user is able to configure individual channels to monitor the data packet, displaying the information on the graph if desired. A close-up of the protocol settings is provided below in Figure 5.



Protocol Settings

Packet size: 10 (# of bytes)

Header

85

0x55

Channels

Number of Channels 2

CH 0
Name: CH 0
Type: Byte 8bit
Position: 1 (1 - (packet size - 1))
Show on Graph: ☐
Y axis: [✓ Y1] Y2

CH 1
Name: CH 1
Type: Byte 8bit
Position: 1 (1 - (packet size - 1))
Show on Graph: ☐
Y axis: Y1 [✓ Y2]

Checksum

☑ Enable

Checksum Settings
Start: 1 (0 - (packet size - 1))
End: 5 (0 - (packet size - 1))
Operation: Add

Figure 5: Protocol Settings

1.3.1 Packet Size

This is the size, in bytes, that the received packet is expected to be, excluding parity and stop bits. In other words, this is the size of the data itself within the packet, including the header byte and checksum byte. The application expects a minimum packet size of two bytes and supports up to one hundred byte size data packets.

1.3.2 Header

This is the expected first byte in the data packet. For example, if 0x55 (85) is selected as the header, the application will be expecting the first byte in the data packet to be equal to 0x55.

1.3.3 Checksum

This option enables or disables the expected checksum byte(s), which is located at the end of the data packet if a checksum is enabled. If unchecked, the application does not expect a checksum byte(s) at the end of the data packet. If checked, an additional *Checksum Settings* menu, like the one in Figure 6 appears. From this menu, the computational settings of the checksum can be configured, including the starting and ending bytes from which the checksum is computed and the checksum operation used. The application supports "Add" and "XOR" checksum operations.

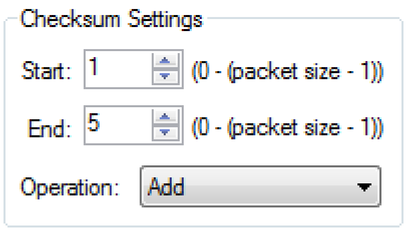

Checksum Settings
Start: 1 (0 - (packet size - 1))
End: 5 (0 - (packet size - 1))
Operation: Add

Figure 6: Checksum Settings

1.3.4 Number of Channels

Here, the user selects the number of channels desired to monitor data within the received packet. Each channel can be configured individually to monitor individual byte(s) within the packet, as well as whether or not to display it on the graph and which Y-axis on the graph the data will use, among other settings. Figure 7 shows how channels are individually configured. A more detailed explanation of the settings follows.

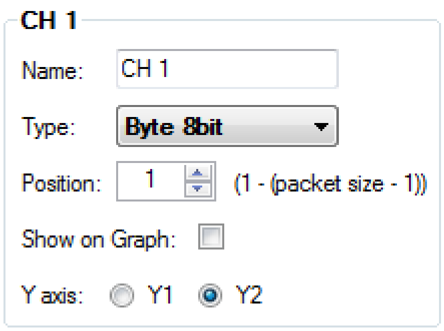

CH 1
Name: CH 1
Type: Byte 8bit
Position: 1 (1 - (packet size - 1))
Show on Graph: ☐
Y axis: ○ Y1 ● Y2

Figure 7: Channel Settings

Name – This is the name of the channel. This name is used by the legend of the graph as the displayed name next to the corresponding colored line used to plot the data monitored by this channel.

Type - This is the data type the application is expecting the monitored byte(s) in the data package to be. For example, if a channel is monitoring the third byte in a seven byte package as an unsigned 32-bit integer, the channel will look at the third through the sixth bytes in the package as a single, 32-bit (4 byte) number, displaying this value if desired. The application supports single byte (8-bit), signed and unsigned 16-bit, 32-bit and 64-bit integers, single (32-bit) and double (64-bit) data types. Note: it is up to the user to ensure that they have the appropriate data type(s) selected for the data packet.

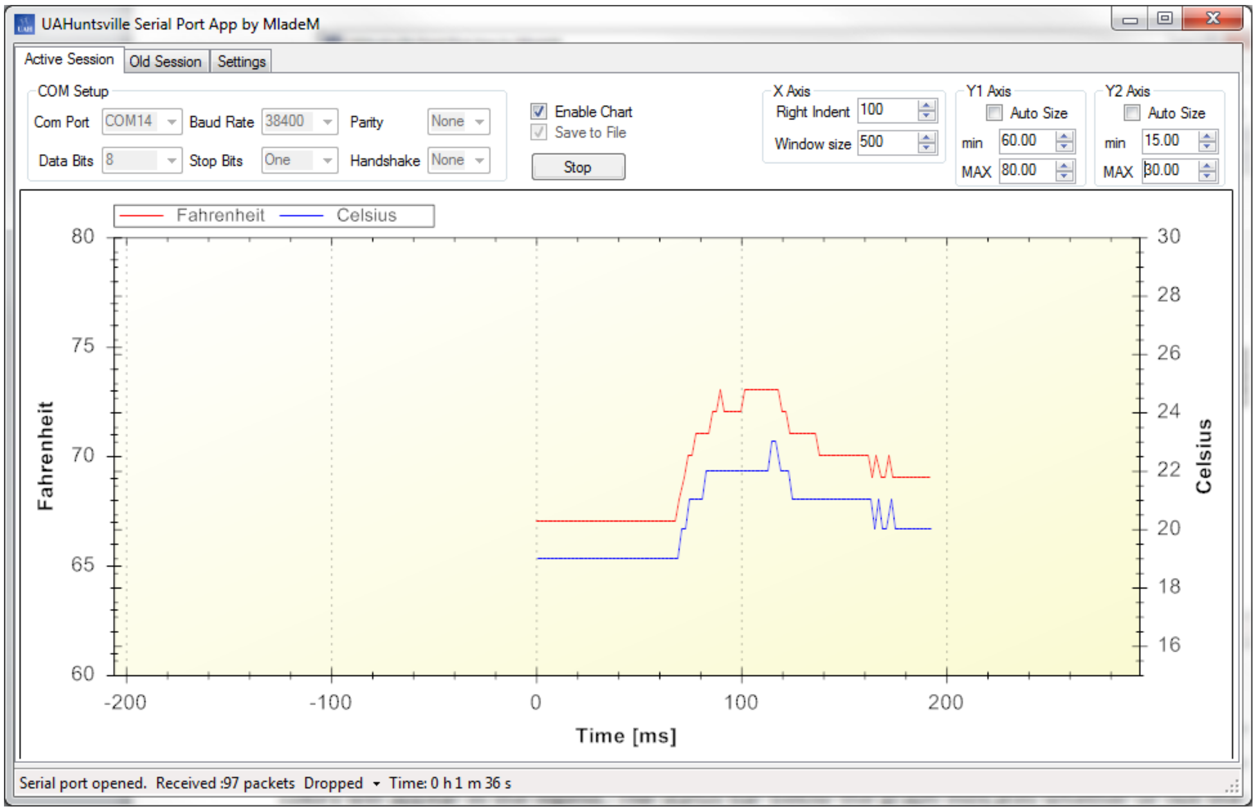
Position - This is the position within the data packet that the channel monitors. As stated previously, the data at this position can be of several different data types.

Show on Graph – This option enables or disables graphing for the channel. If checked, the data from the channel is displayed on the graph; it is not displayed if unchecked. Note that this option does not affect whether or not data is logged to a file.

Y axis: - This option determines which Y-axis on the graph the data is plotted with respect to. The data will use the Y1-axis (left axis on the graph) if Y1 is selected, and the Y2-axis (right axis on the graph) if Y2 is selected.

1.4 Connecting and Displaying Data

In order to receive data, the application must first connect to the serial port that the data packet is to be received from. From the *Active Session* tab, once the appropriate serial port is selected, the user presses the *Connect* button to connect to the serial port. In order to graph the received data, however, graphing must be enabled by checking the *Enable Chart* checkbox located to the right of the *COM Setup* settings. Figure 8 below shows an example data plot.


UAHuntsville Serial Port App by MladeM

Active Session | Old Session | Settings

COM Setup
Com Port: COM14 Baud Rate: 38400 Parity: None
Data Bits: 8 Stop Bits: One Handshake: None

[✓] Enable Chart
[✓] Save to File
Stop

X Axis
Right Indent: 100
Window size: 500

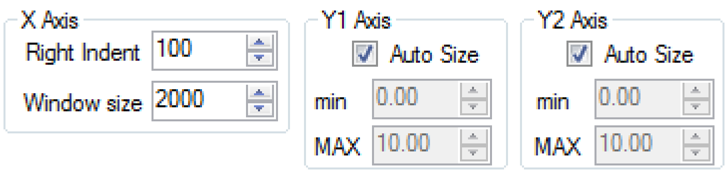
Y1 Axis
[ ] Auto Size
min: 60.00
MAX: 80.00

Y2 Axis
[ ] Auto Size
min: 15.00
MAX: 30.00

Serial port opened. Received :97 packets Dropped Time: 0 h 1 m 36 s

Figure 8: Example Data Plot

In Figure 8, the data is plotted on the graph as a red line, as indicated by the legend above the graph. If multiple channels are plotted, the channel names and their corresponding line colors will appear in the legend. The status bar below the graph indicates whether or not the serial port was opened successfully, the number of packets received, the number of bytes dropped and how long the serial port has been opened. While the graph is displayed, the user is able to alter a few settings to better view the plotted data. These settings include Right Indent and Windows Size for the X-axis and Auto-Sizing and minimum and maximum values for the Y-axes. Figure 9 provides a close-up of these settings. If data is not being displayed on the graph properly or not at all, or if the status bars indicates that it could not open the serial port, refer to section 4 (troubleshooting.)


X Axis
Right Indent: 100
Window size: 2000

Y1 Axis
[✓] Auto Size
min: 0.00
MAX: 10.00

Y2 Axis
[✓] Auto Size
min: 0.00
MAX: 10.00

Figure 9: Graph Display Settings

1.4.1 Right Indent

This is the number of samples from the right the newest samples are plotted. In other words, the graph is ended over this number of units from the right.

1.4.2 Window Size

This is the number of units the X-axis encompasses. In other words, it is the length of the displayed X-axis in samples.

1.4.3 Y-axis Scaling

Each Y-axis can be individually set to automatically scale to the received values using the *Y1 Auto Size* and *Y2 Auto Size* checkboxes. When checked, manual scaling of the corresponding axis is disabled and the application itself determines the appropriate minimum and maximum value of the Y-axis. When disabled, the user must manually set the minimum and maximum value to a value appropriate for the receive data. *Y1 min* and *Y2 min* are used to set the minimum value of the Y1 and Y2 axes, and *Y1 max* and *Y2 max* are used to set the maximum value of the Y1 and Y2 axes.

1.5 Loading and Saving Data

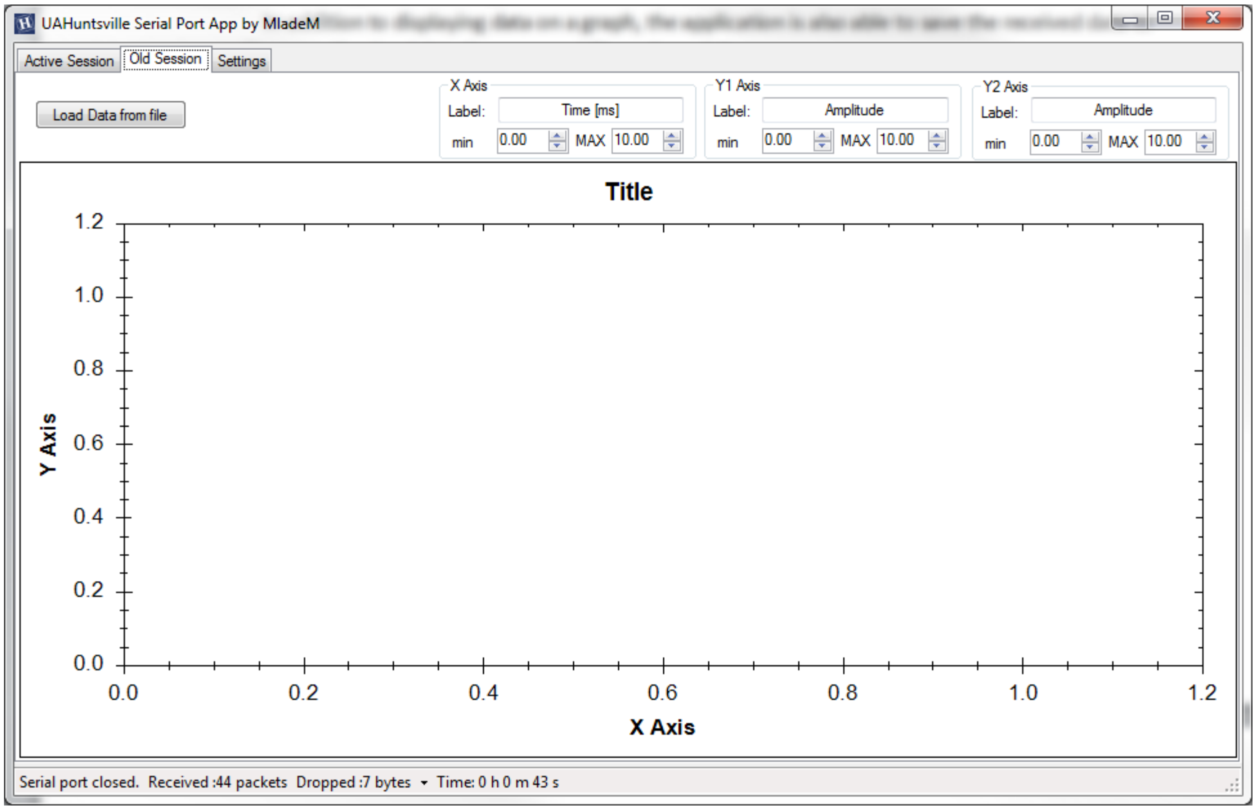
In addition to displaying data on a graph, the application is also able to save the received data to a file, as well as load data from a saved file and graph it.

1.5.1 Saving Data

To save data to a file, the user must check the *Save to File* checkbox, located below the *Enable Chart* checkbox, before connecting to the serial port. Once the application connects to the serial port, this checkbox cannot be checked or unchecked, that is, the user must first disconnect from the port before toggling this option. The saved data file contains the data monitored from each channel, sorted in labeled columns. The text file where the data is written is named after the time at which the application connected to the serial port. For example, if the application connected to the serial port and begins monitoring data on November 5, 3012 at 11:13:37 AM, the data is saved to a file named *3012_11_4_11_13_37.txt* located in the same directory from where the executable of the application is located.

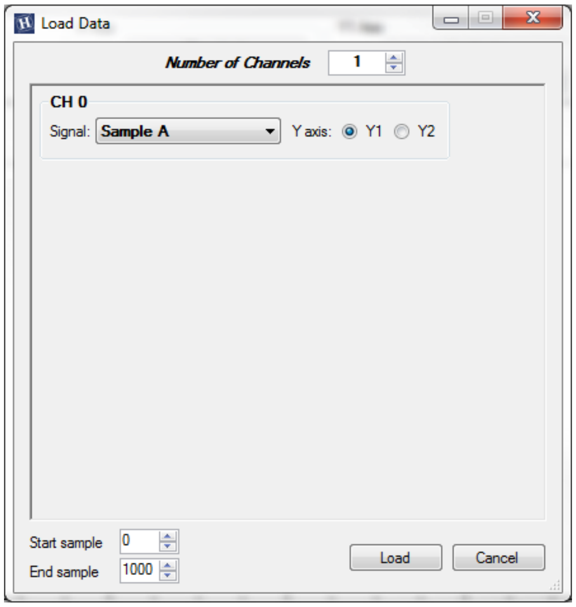
1.5.2 Loading Data

To load saved data, navigate to the *Old Session* tab, pictured below in Figure 10. Then, click the *Load Data from File* button, which will prompt the user to select a file from which to load the saved data. After selecting a file, the user is prompted with load data configuration window as shown in Figure 11.



Serial port closed. Received :44 packets Dropped :7 bytes Time: 0 h 0 m 43 s

Figure 10: Old Session



Number of Channels: 1

CH 0

Signal: Sample A Y axis: [✓ Y1] Y2

Start sample: 0

End sample: 1000

Figure 11: Load Data Configuration

From here, the user can select from which channels in the file to load data by selecting the *Number of Channels*. For each individual channel, the user can configure on which Y-axis to display the data from the channel and which dataset in the file to graph. In addition, the user can select a subset of samples from the loaded file by changing the *Start sample* and *End sample* parameters. In other words, only samples from the specified start sample to end sample will be loaded and displayed on the graph. To load the data, click the *Load* button to display the data on the graph or the *Cancel* button to close the window without loading any data from the file.

Once loaded, the data is displayed on the graph, as shown below in Figure 12.

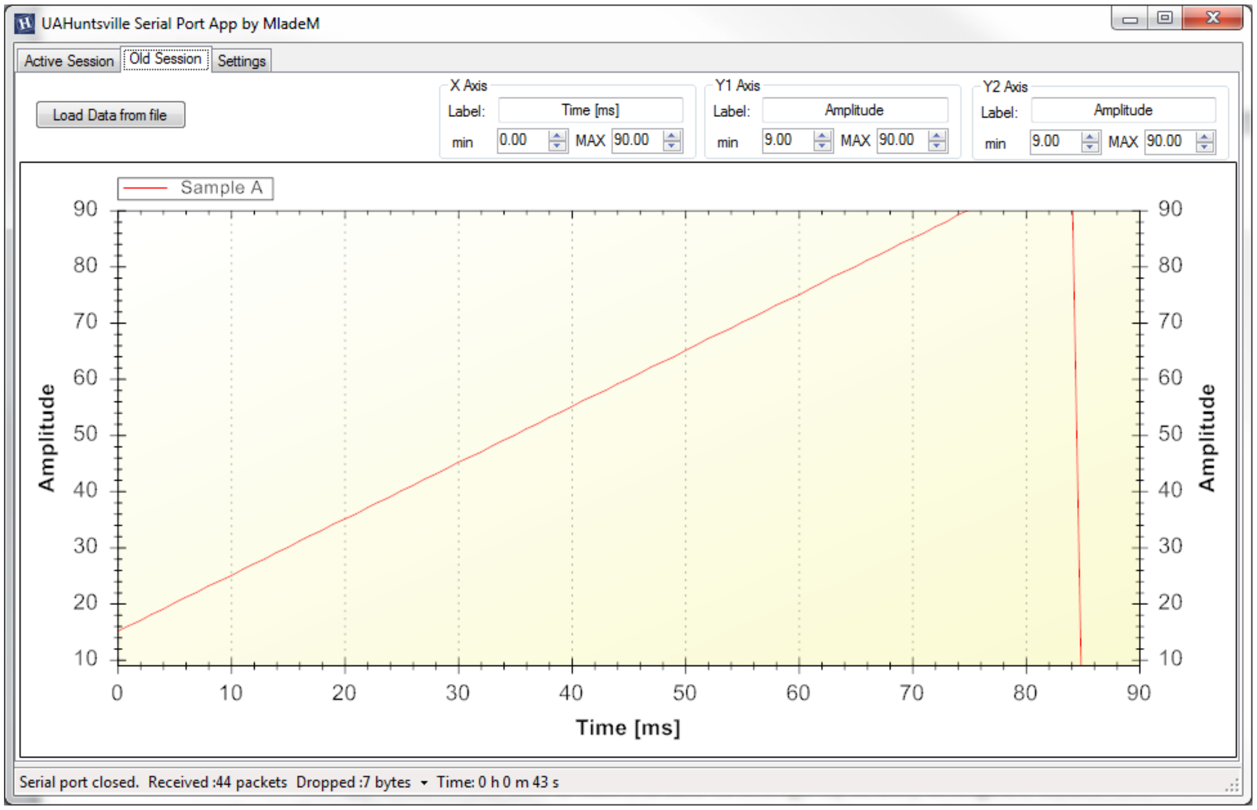


Figure 12: Displayed Data Loaded From File

The minimum and maximum values of the X and Y axes are automatically scaled to accommodate the loaded data. The user may individually configure these values by changing the *min* and *MAX* values. The user may also specify axis labels by typing in a desired name in the *Label* text box.

1.6 Loading and Saving Settings

The application is also able to load and save its settings. By default, settings are loaded from the *config.dat* file located in the directory of the executable when the application is started.

1.6.1 Saving Settings

To save settings, click the *Save Settings* button on the *Settings* tab. The user is prompted to enter a name for the settings. The saved file will contain the current graph and protocol settings. Note: as stated above, default settings are loaded from the *config.dat* file. Overwriting this file overwrites the default settings of the application.

1.6.2 Loading Settings

To load settings, click the *Load Settings* button on the *Settings* tab. The user is prompted to select a save file from which to load settings. Once a file is selected, the saved graph and protocol settings are applied by the application.

3. A Sample Walkthrough

This section contains a sample walkthrough of the application using a Texas Instruments® MSP430 Experimenter's Board. In this walkthrough, the MSP430 periodically counts from 0 to 100 and sends the current value in a 2 byte packet every second. The first byte in the packet is the header file with a value of 60 (0x3C.) The second byte may contain any of the values 0-100. The MSP430 is configured for a baud rate of 115200, no parity, no handshaking, 8 data bits and one stop bit. Also, the MSP430 uses no checksum byte.

To retrieve the data sent from the Experimenter board, the application must first be configured to match the UART settings of the MSP430. To do this, use the following settings as shown in Figure 13:

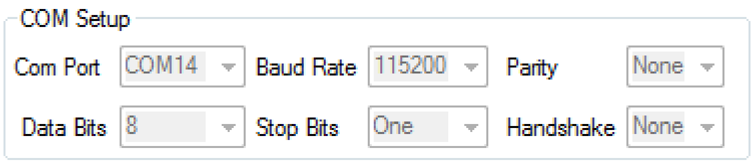

COM Setup
Com Port: COM14
Baud Rate: 115200
Parity: None
Data Bits: 8
Stop Bits: One
Handshake: None

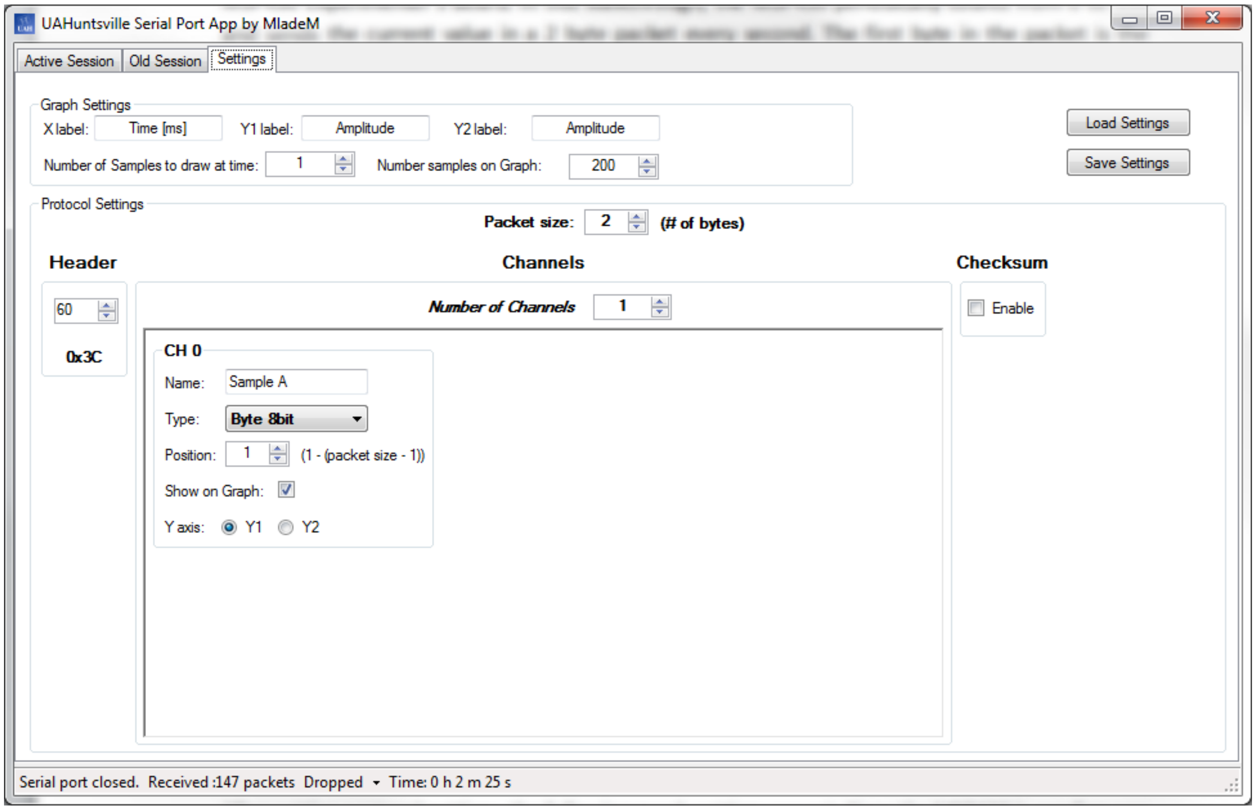
Figure 13: COM Setup Configuration for Walkthrough

For this walkthrough, COM14 was the serial port used. The MSP430 is using a baud rate of 115,200 for communication, so the baud rate of the application must be set to this value. Also, the MSP430 is using 8 data bits and one stop bit. Since the MSP430 is not using a parity bit or a handshaking protocol, they are configured as "None."

Next, protocol settings must be configured to match the MSP430 by navigating to the *Settings* tab. The MSP430 is not using a checksum byte, so this needs to be disabled in the application by unchecking the *Enable* checkbox in the *Checksum* groupbox. The packet size from the MSP430 is two bytes, so *Packet Size* must be set to 2 bytes. For this walkthrough, one channel will be used to monitor the second byte of the packet as an 8-bit byte, displaying the data with respect to the Y1 axis on the graph.

After setting protocol settings, set the following graph settings. Since the MSP430 is sending one packet every second, it is appropriate to set the *Number of Samples to draw at a time* to 1. For this walkthrough, set the *Number of Samples on Graph* to 200.

Figure 14 shows the protocol and graph settings as configured for this walkthrough.



Graph Settings

X label: Time [ms]   Y1 label: Amplitude   Y2 label: Amplitude

Number of Samples to draw at time: 1   Number samples on Graph: 200

Protocol Settings

Packet size: 2 (# of bytes)

Header: 60, 0x3C

Channels — Number of Channels: 1

CH 0
Name: Sample A
Type: Byte 8bit
Position: 1 (1 - (packet size - 1))
Show on Graph: ✓
Y axis: (● Y1) Y2

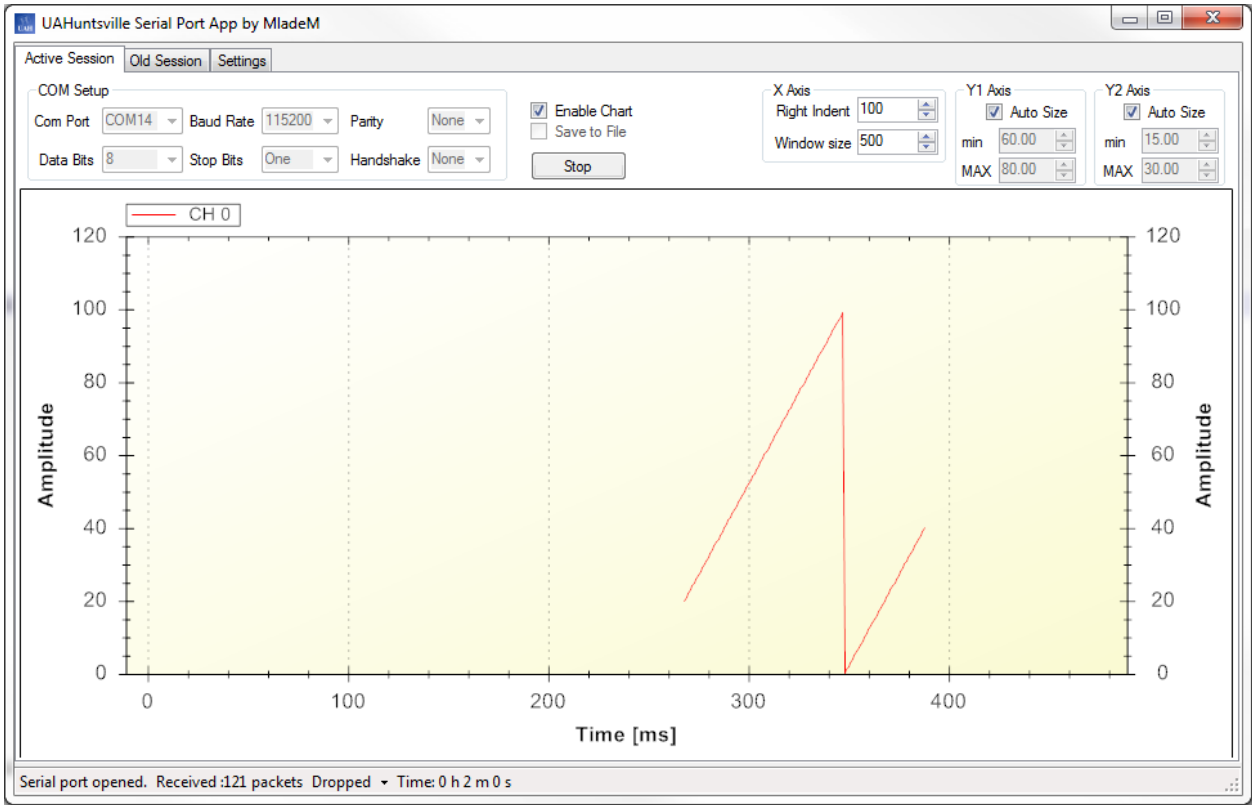
Checksum: Enable ☐

Serial port closed. Received :147 packets Dropped Time: 0 h 2 m 25 s

Figure 14: Graph and Protocol Walkthrough Settings

Before displaying the data from the MSP430, remember to check the *Enable Chart* checkbox on the *Active Session* tab. Also, configure the X-axis *Right Indent* option to 100 and the *Window Size* to 500.

Verify that settings match that of the MSP430, then press the *Connect* button to display the data from the MSP430. If configured properly, the displayed data should appear similar to the data in Figure 15 after collecting several samples. If the application indicates that packets are being dropped or data is simply not being displayed, refer to Section 4 (Troubleshooting.)



UAHuntsville Serial Port App by MladeM

COM Setup
Com Port COM14
Baud Rate 115200
Parity None
Data Bits 8
Stop Bits One
Handshake None

☑ Enable Chart
☐ Save to File

X Axis
Right Indent 100
Window size 500

Y1 Axis
☑ Auto Size
min 60.00
MAX 80.00

Y2 Axis
☑ Auto Size
min 15.00
MAX 30.00

Serial port opened. Received :121 packets Dropped Time: 0 h 2 m 0 s

Figure 15: Displayed Data from MSP430

4. Troubleshooting

This section is to be a quick reference guide, describing common application issues and offer potential solutions.

I cannot connect to the serial port

Verify that the correct serial port has been selected. If the data packet is being sent on one of the computer's built-in serial ports, the port will most likely be COM1 or COM2. If using a peripheral serial port (USB to serial adapter, etc.) the port will most likely be higher in number, such as COM14.

Serial port is connected, but the application indicates that some or all packets are being dropped

This is most commonly due to improper settings. An incorrect baud rate, parity, stop bit, handshake, data bit, checksum and header setting can all contribute to the application dropping packets. Please make sure all these settings match the values on the device transmitting the data packet. Also, make sure the expected number of bytes matches the number of bytes of the data packet.

I cannot see any data on the graph

Verify that the *Enable Chart* checkbox on the *Active Session* tab is checked. Also, for each channel desired, verify that the *Show on Graph* checkbox is checked.

The data being displayed on the graph is not what I expected

Verify that the channel is monitoring the correct byte position and that the data type the channel is monitoring matches the data type of the data at that byte position of the data packet. For data types more than one byte in length, keep in mind the order the data is being sent in (Most Significant Bit, Least Significant Bit, etc.) The application assumes the least significant bit is sent first. For example, consider the following three byte data packet:

0x50	0xFF00
Header	16-bit Unsigned Integer

A channel set to monitor the 2nd byte (position 1 of the packet) for a 16-bit unsigned integer will see it as the least significant bits of the unsigned integer, reading the received number as 0x00FF.