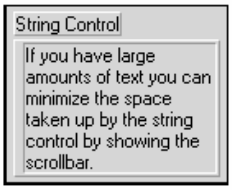
A. Strings

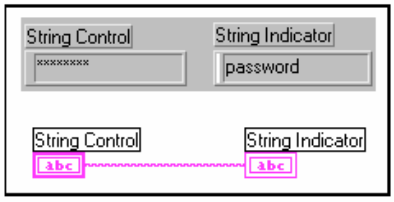
A string is a sequence of displayable or nondisplayable characters. Often, you use strings for more than simple text (for example, ASCII) messages. For example, in instrument control, you pass numeric data as character strings. You then convert these strings to numbers. In many cases, storing numeric data to disk also requires strings, which means that you first must convert numbers to strings before writing the numbers to a file on disk.

Creating String Controls and Indicators

String controls and indicators are in the **String & Table** subpalette of the **Controls** palette. You enter or change text inside a string control using the Operating tool or the Labeling tool. You can enlarge string controls and indicators by dragging a corner with the Positioning tool. To minimize the space that a front panel string control or indicator occupies, use the **Show Scrollbar** option from the string pop-up menu. If this option is dimmed, you must increase the window's vertical size.



You also can configure string controls and indicators for different types of display. For example, you can choose password display by enabling the **Password Display** option from the string's pop-up menu. With this option selected, only asterisks appear in the string's front panel display. On the block diagram, the string data reflects what was typed.

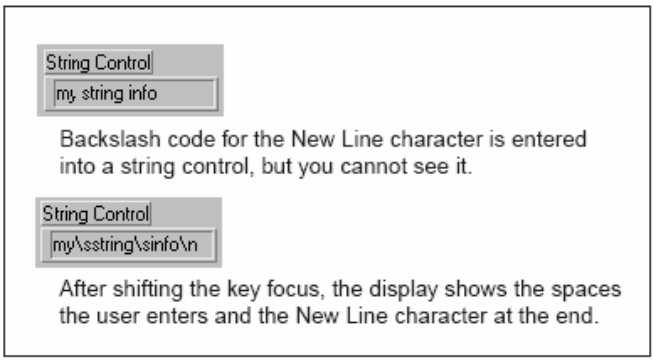


String controls and indicators also can display and accept characters that are usually nondisplayable, such as backspaces, carriage returns, tabs, and so on. To display these characters, choose '\' **Codes Display** from the string's pop-up menu.

In '\' Codes Display mode, nondisplayable characters appear as a backslash followed by the appropriate code. A partial list of codes appears in the table

below. (For the complete table, use the **Online Reference** (**Help** menu) and search on Nondisplayable Characters.) To enter a nondisplayable character into a string control, type the backslash character \, followed by the code for the character. As shown below, after you type text in the string and click the Enter button, any nondisplayable characters appear in backslash code format.

Enter button



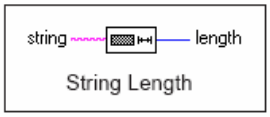
Code	LabVIEW Interpretation
\b	Backspace (ASCII BS, equivalent to \08)
\s	Space (ASCII SP, equivalent to \20)
\r	Return (ASCII CR, equivalent to \0D)
\n	Newline (ASCII LF, equivalent to \0A)
\t	Tab (ASCII HT, equivalent to \09)

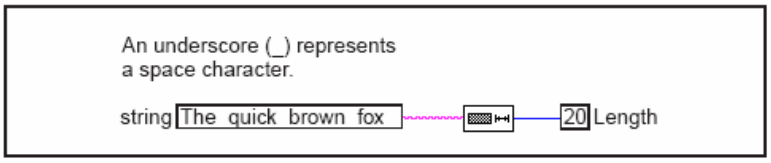
The characters contained in LabVIEW string controls and indicators are represented internally in ASCII format. To view the actual ASCII codes (in hex), choose **Hex Display** from the string's pop-up menu.

B. String Functions

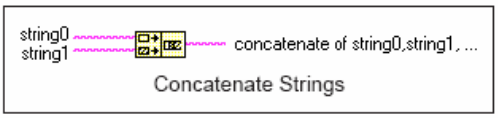
LabVIEW has many functions to manipulate strings. These functions are available from the **String** subpalette of the **Functions** palette. Some common functions are discussed below.

String Length returns the number of characters in a string.



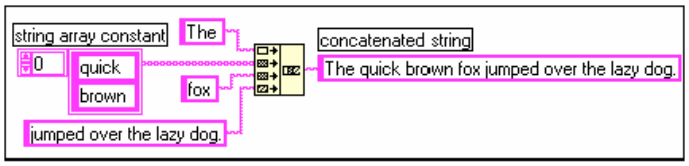


Concatenate Strings concatenates all input strings and arrays of strings into a single output string.

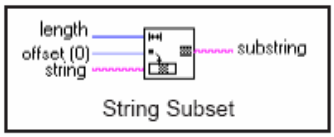


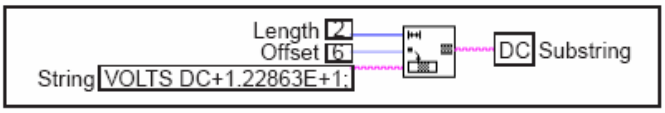
Concatenate Strings function when placed in Diagram window

The function appears as shown at left when you place it in the Diagram window. You can resize the function with the Positioning tool to increase the number of inputs.

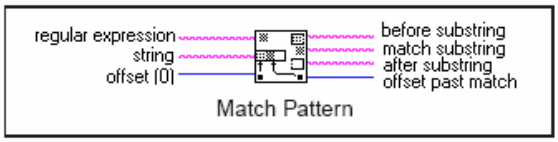


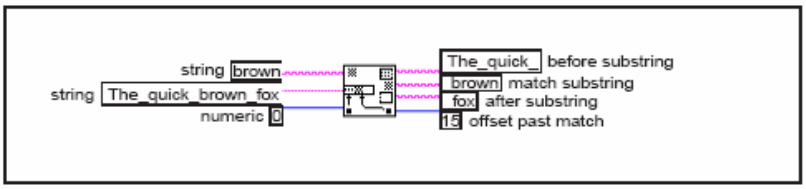
String Subset returns the substring beginning at **offset** and containing **length** number of characters. The first character offset is zero.





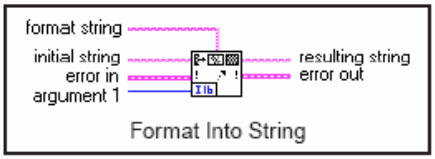
Match Pattern returns the matched **substring**. The function searches for the **regular expression** in **string** beginning at the **offset**, and if it finds a match, splits the string into three substrings. If no match is found, the match substring is empty and the **offset past match** is -1.





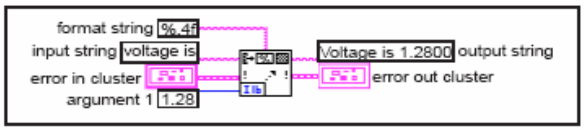
In many instances, you must convert strings to numbers or numbers to strings. The **Format Into String** function converts a number to a string and the **Scan From String** function converts a string to a number. Both of these functions can perform error handling.

Format Into String converts any format **argument** (for example, numeric) to the specified formatted **resulting string**. You can expand the function to have multiple values converted to a single string simultaneously.

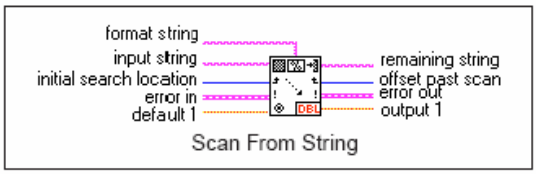


The function can format the output string with the **initial string** and **argument**(s) based on the **format string**.

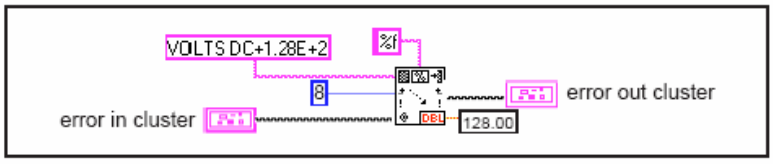
In the example below, the function converts the floating-point number 1.28 to the 6-byte string "1.2800."



Scan From String converts a string containing valid numeric characters (0 to 9, +, -, e, E, and period) to a number. The function starts scanning the **input string** at **initial search location**. The function can scan the input string into various data types (for example, numerics or Booleans) based on the **format string**. This function is expandable to have multiple outputs.

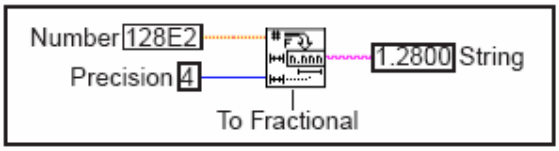


In the example below, the function converts the string "VOLTS DC+1.28E+2" to the number 128.00. The function starts scanning at the ninth character of the string (that is, the +). (The first character offset is zero.)



Both **Format Into String** and **Scan From String** have an **Edit Scan String** interface to create the format string. The format string specifies the format, precision, data type, and width of the converted value. You can access the **Edit Scan String** dialog box by popping up on the node and choosing **Edit Format String** or simply double-clicking on the function. After you configure the format string and select **Create String**, the dialog box creates the string constant and wires it to the format string input for you. See the following example of using the **Edit Scan String** to create the format string for a floating-point number, precision of 2 digits, width of 8 digits, and padded with spaces.

There are additional string formatting functions in the **String » Additional String to Number Functions** subpalette. You can use these functions for specific data types. For example, the **To Fractional** function converts a **number** to a floating-point formatted **string**.



The **From Exponential/Fract/Eng** function converts a string containing valid numeric characters to a floating-point **number**. The function starts scanning the **string** at **offset**.