Bitmap File Writer VI

Goal

Use Binary File I/O to write to a file using a specified format.

Scenerio

Write a file storage routine to store image files for an existing LabVIEW-based Drawing Pad. The drawing pad VI returns a drawing as a 2D array of red, green, and blue (RGB) values. Save the data as a 24-bit bitmap file.

Design

You can use binary file I/O to write or read data in any file format, assuming that you have a specification that tells you the file layout for that format. The following section describes the format for a 24-bit Windows bitmap file.

24-bit Bitmap File Layout

Bitmap (`.bmp`) files are a format for storing image data. Bitmaps come in multiple varieties, with differences such as the number of bits used to represent a pixel and the level of image compression used. The easiest type of bitmap file to understand and create is a 24-bit uncompressed bitmap file. Table 1 describes the format of a 24-bit uncompressed bitmap file.

Table 1. 24-Bit Bitmap File Layout

Section Name	Size (bytes)	Notes
BITMAPFILEHEADER	14	Data is provided. Write data to the file.
BITMAPINFOHEADER	40	Data is provided. Write data to the file.
Image Data	3 × Number of Pixels	Stored in BGR order. Image is inverted.

BITMAPFILEHEADER—Contains information about the file such as the file type and file size. A subVI calculates the data for this segment. The subVI returns an array of U8 numerics that you must write to the file.

BITMAPINFOHEADER—Contains information about the image such as the height, width, compression level, and number of bits per pixel. A subVI calculates the data for this segment. The subVI returns an array of U8 numerics that you must write to the file.

Image Data—For a 24-bit image, three bytes represent each pixel in the image. The three bytes represent the RGB values for the pixel and are stored in reverse order, BGR. The pixel array you are given is a 2D array of clusters. Each cluster has an RGB value in it. The rows in the Image Data are stored from bottom to top, and the first pixel stored is the lower left corner of the image.

Inputs and Outputs

The main VI for this program contains no inputs or outputs. Dialog boxes control all user interaction. The Drawing Pad VI displays a dialog box that allows the user to draw a picture. When the user clicks **Save**, the application prompts the user to enter a save location by using a File Dialog VI.

Program Flow

1. Call the Drawing Pad VI to create a picture.

Note The Drawing Pad VI returns the picture as a 2D array of clusters, each containing RGB values. The 2D array has a width that is a multiple of four and is inverted in preparation for writing to file.

2. Display a file dialog box to the user to select a location and filename and open the selected file for writing.
3. Call the BITMAPFILEHEADER VI and pass the dimensions of the pixel array to it.
4. Write the 1D array of unsigned integers returned by the BITMAPFILEHEADER VI to the open file.

Note Disable the **prepend array or string size** option when you call the Write to Binary File function, otherwise LabVIEW inserts the array size at the beginning of the data, which invalidates the file layout.

5. Call the BITMAPINFOHEADER VI and pass the dimensions of the pixel array to it.
6. Write the 1D array of unsigned integers returned by the BITMAPINFOHEADER VI to the open file.
7. Process each pixel in the array by using a pair of For Loops to remove the values from the cluster in BGR order and build them into an array.

Note Use a 3D array to store the processed pixel data, because it allows you to use For Loop auto-indexing and simplify the program. The number of dimensions in the array is not important, because the File I/O functions automatically reformat the array to write to the file.

8. Write the processed pixel array to the open file.
9. Close the file and handle any errors.

Implementation

The files that you need to complete this exercise are here:
`<NI eLearning>\LV Core 2\Using Binary Files\Exercise`.

1. Display the drawing pad.
   - ❑ Open a blank VI.
   - ❑ Save the VI as `Bitmap File Writer.vi` in the `<Exercise>` directory.
   - ❑ Open the block diagram.
   - ❑ Add the Drawing Pad VI, located in the `<Exercise>` directory, to the block diagram.
   - ❑ Right-click the **Drawing Pad VI** and select **SubVI Node Setup** from the shortcut menu.
   - ❑ Check the **Show Front Panel when called** and **Close afterwards if originally closed** boxes.
   - ❑ Click **OK** to exit the SubVI Node Setup dialog box.

Note The subVI node setup lets you specify how to call a subVI. Checking the **Show Front Panel when called** box instructs the VI to show its front panel, even if the VI properties would otherwise prevent it from doing so.

   - ❑ Run the VI and observe the drawing pad. Click the **Save** button to exit. Currently, the program ends when you click **Save**, because you have not yet implemented the file I/O.

2. Open a new binary file.

- ❑ Add a **File Dialog Express VI** to the block diagram.
- ❑ Click **OK** to exit the Configure File Dialog dialog box. The default values allow the user to select a single new or existing file.
- ❑ Expand the **File Dialog Express VI** to show the **selected path**, **error out**, **error in**, **prompt**, **pattern label**, and **pattern (all files)** terminals by expanding the node and then clicking each terminal to select an input or output.
- ❑ Right-click the **prompt** input of the File Dialog Express VI and select **Create»Constant** from the shortcut menu.
- ❑ Enter `Select File to Save` in the string constant.
- ❑ Right-click the **pattern label** input of the File Dialog Express VI and select **Create»Constant** from the shortcut menu.
- ❑ Enter `Bitmap Files` in the string constant.
- ❑ Right-click the **pattern (all files)** input of the File Dialog Express VI and select **Create»Constant** from the shortcut menu.
- ❑ Enter `*.bmp` in the string constant.

- ❑ Add an **Open/Create/Replace File** function to the block diagram.
- ❑ Right-click the **operation** input of the Open/Create/Replace File function and select **Create»Constant** from the shortcut menu.
- ❑ Select **replace or create** as the value of the constant.
- ❑ Wire the block diagram as shown in Figure 1.

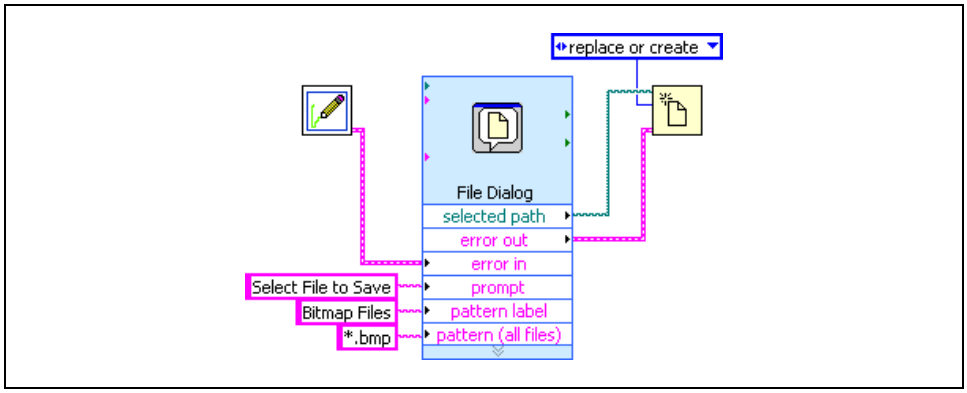

Figure 1. Open Binary File

3. Create bitmap headers.

   ❑ Add the **BITMAPFILEHEADER VI** located in the `<Exercise>` directory to the block diagram.

   ❑ Add the **BITMAPINFOHEADER VI** located in the `<Exercise>` directory to the block diagram.

   ❑ Add an **Array Size** function to the block diagram.

   ❑ Add two **Write to Binary File** functions to the block diagram.

   ❑ Right-click the **prepend array or string size** terminal of each Write to Binary File function and select **Create»Constant** from the shortcut menu.

   ❑ Set the constant values to `False`.

   ❑ Wire the block diagram as shown in Figure 2.

Note You do not need to specify **little-endian** or **big-endian** for the **byte order** input of the Write to Binary File function because the array data type is only one byte.

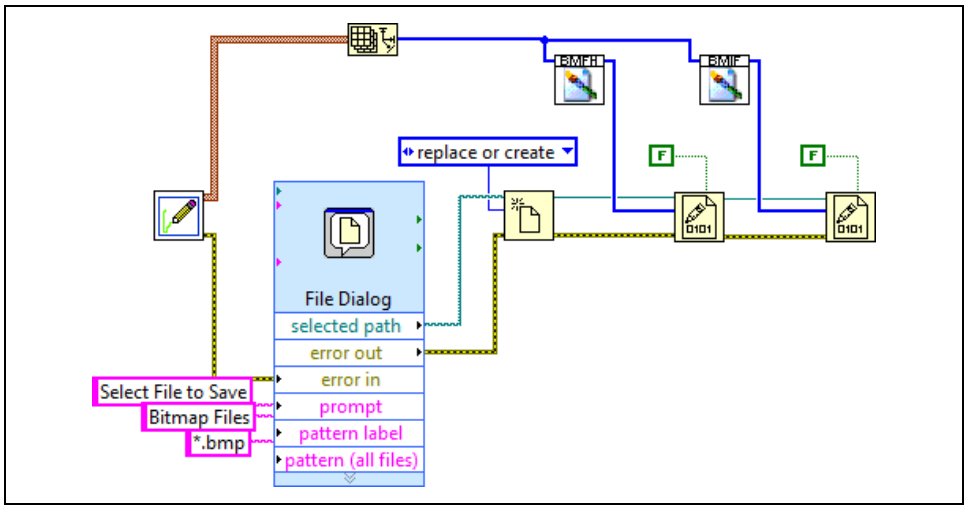

Figure 2. Write to Bitmap Headers

4. Write image data.

- ❑ Add a **For Loop** to the block diagram.
- ❑ Add a second **For Loop** inside the first For Loop.

- ❑ Add an **Unbundle by Name** function to the inner For Loop.

- ❑ Add a **Build Array** function to the For Loops.
- ❑ Wire the **Image Data** array through the For Loop borders to the **Unbundle By Name** function.
- ❑ Expand the **Unbundle by Name** function so that three elements areshown.
- ❑ Choose **Blue**, **Green**, and **Red**, in order, for the elements.

Note The bitmap file definition specifies that pixels must be stored in blue, green, red order. Storing the pixels in another order causes the colors in your image to be incorrect.

- ❑ Add a **Write to Binary File** function to the block diagram.
- ❑ Right-click the **prepend array or string size** input of the Write to Binary File function and select **Create»Constant** from the shortcut menu.
- ❑ Set the constant value to `False`.

5. Close the file and handle the errors.

- ❑ Add a **Close File** function to the block diagram.

- ❑ Add a **Simple Error Handler VI** to the block diagram.
- ❑ Wire the block diagram as shown in Figure 3.

6. Save the VI.

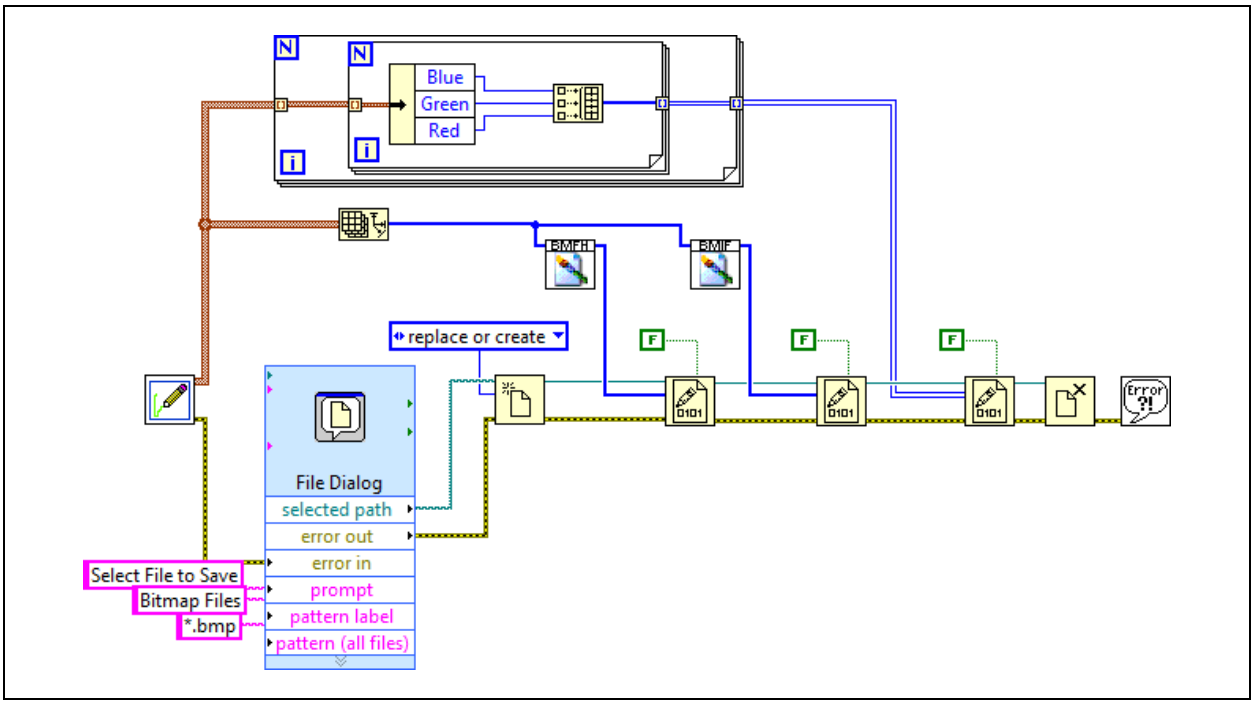

Figure 3. Completed Block Diagram

Test

1. Run the VI.
   - ❑ Switch to the VI front panel window.
   - ❑ Run the VI.
   - ❑ Draw a picture in the drawing pad.
   - ❑ Click **Save**.
   - ❑ Select `My Image.bmp` from the `<Exercise>` directory as the file to save.
2. Open the image in an image viewer.
   - ❑ Open the `<Exercise>` directory in Windows Explorer.
   - ❑ Double-click the image to open it in your default image viewer. Verify that the image displayed is the picture you created.
3. Close the VI.

End of Exercise

Notes