Institute of Technology, Nirma University

Department of Electronics and Communication Engineering

FPGA based System Design

Basic Image Compression

Abstract :

An picture can be made smaller without sacrificing quality by using an image compression technique. This is accomplished by eliminating the image's spectral and spatial redundancy. Reducing the amount of bits needed to represent the image while maintaining the highest level of visual clarity and resolution in the reconstructed image is the aim.
Lossy and lossless picture compression are the two primary varieties. An image with lossy compression has a smaller file size but a lower quality since part of the image data is lost. Since lossless compression does not eliminate any data, the file size is bigger but the image quality remains unchanged.

Typical methods for compressing images include the following:

- Color quantization: Limits the image's color space to a small number of "representative" hues.
- Dithering: A strategy to prevent being posterized.
- palette for the entire image: Colors used in GIF and PNG file formats are typically 256.
- Block palette: Used in BTC, CCC, S2TC, and S3TC; typically consists of two or four colors each block of four by four pixels.

Keywords: Lossy Compression, Lossless Compression, JPEG (Joint Photographic Experts Group), Run-Length Encoding (RLE).

1. Lossy Compression: Lossy compression achieves its effect at cost of loss in image quality, by removing some image information.
2. Lossless Compression: Lossless compression techniques reduce size whilst preserving all of the original image information, and therefore without degrading the quality of the image.
3. JPEG (Joint Photographic Experts Group): Utilizes lossy compression techniques based on the Discrete Cosine Transform (DCT) to compress images. JPEG is widely used for photographs and web graphics.

State of the art technology available:

1. JPEG Compression
2. JPEG2000 Compression
3. PNG Compression
4. Run-Length Encoding (RLE)
5. Huffman Coding
6. Lempel-Ziv-Welch (LZW) Compression

The limitations or drawbacks of currently available technology on basic image compression include:

- Loss of Quality: The majority of simple image compression methods include some lossy compression, which can lower the quality of the resulting photos. This is especially true for images with a lot of intricacy.
- Limited Compression Ratios: High-quality pictures could have higher file sizes because basic compression methods may not be able to achieve high compression ratios when compared to more sophisticated ones.
- Restricted Colour Depth: A few common compression formats, such as GIF, have a 256 colour maximum, which can cause colour banding or a loss of information in images with a large colour gamut.

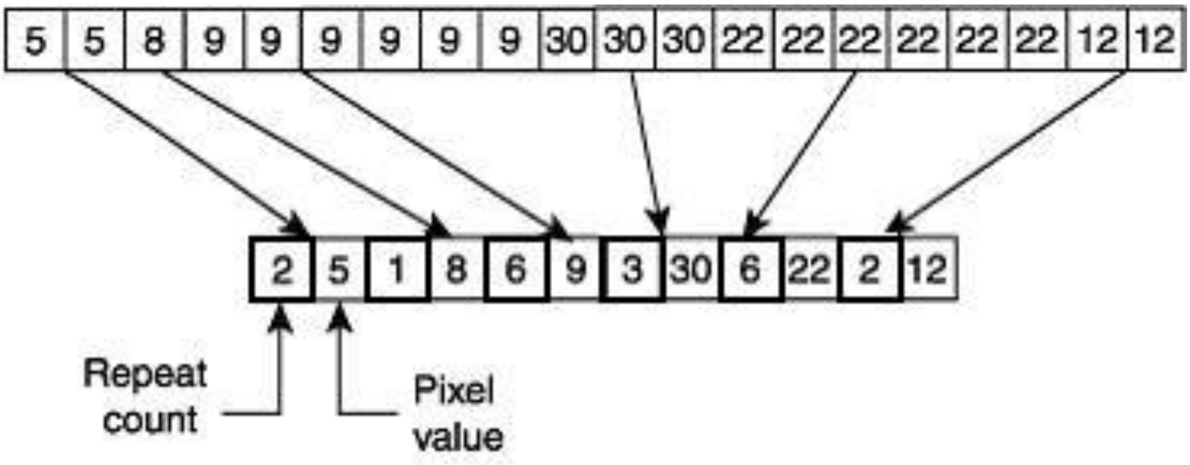
Encoding for run length (RLE)

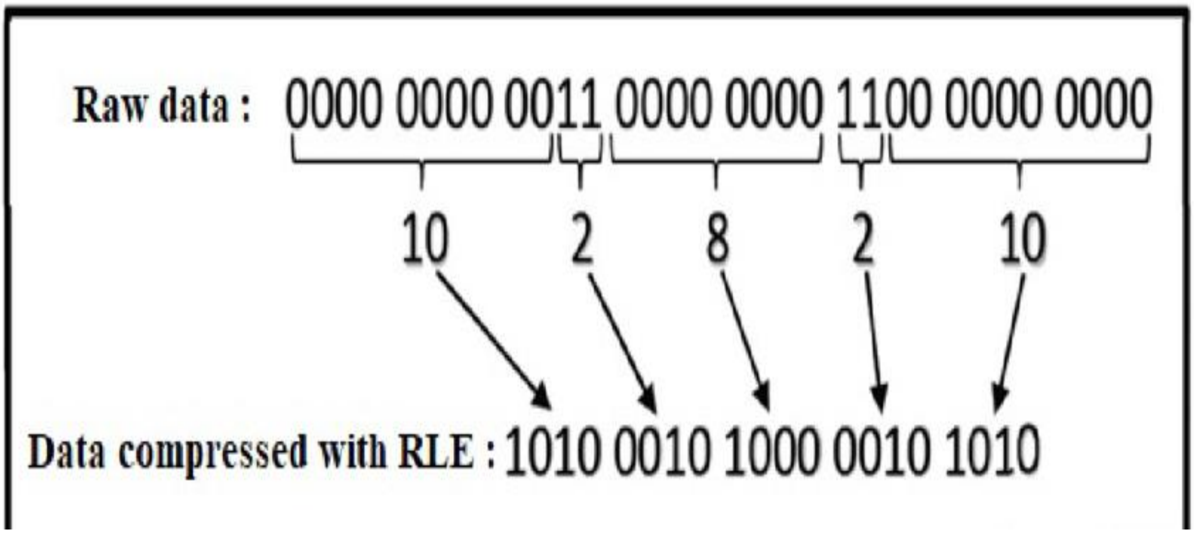
Run length encoding (RLE) is perhaps the simplest compression technique in common use. RLE algorithms are lossless, and work by searching for runs of bits, bytes, or pixels of the same value, and encoding the length and value of the run. As such, RLE achieves best results with images containing large areas of contiguous colour, and especially monochrome images. Complex colour images, such as photographs, do not compress well – in some cases, RLE can actually increase the file size. There is a number of RLE variants in common use, which are encountered in the TIFF, PCX and BMP graphics formats [4]. Run-length encoding represents a string by replacing each subsequence of consecutive identical characters with (char; length). The string 11112222333111 would have representation (1; 4) (2; 4) (3; 3) (1; 3). Then compress each (char; length) as a unit using, say, Human coding. Clearly, this technique works best when the characters repeat often. One such situation is in fax transmission, which contains alternating long sequences of 1's and 0's. The distribution of code words is taken over many documents to compute the optimal Human code.[14]

The proposed enhanced RLE algorithm :

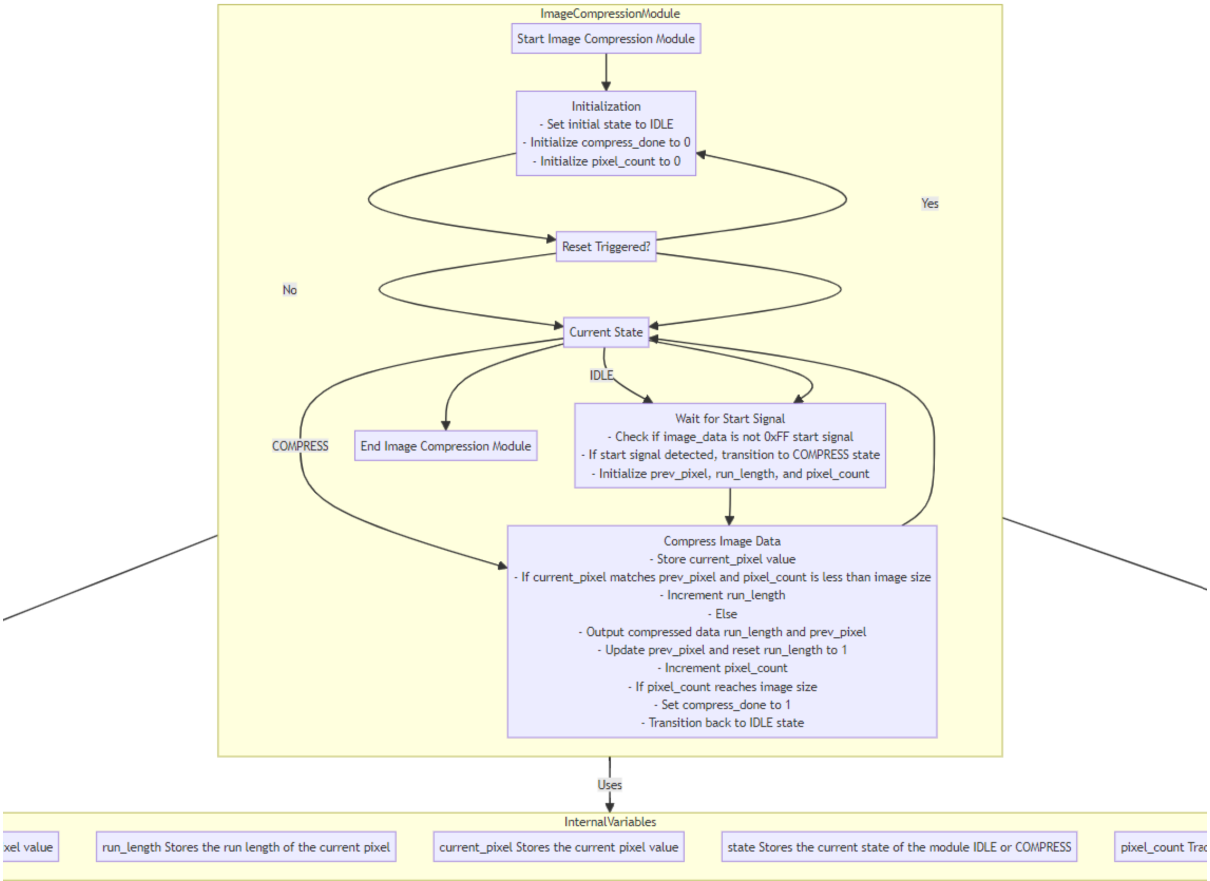
1- Read the BMP image.
2- Get the height N and the width M for the image
3- Create an array , let it M ain(N,M) ,each element of this array consists of three fields for R,G,B colours.
4- Convert the image to the main array ; Main(N,M).

5- Let X=Main(0,0) ; Main(0,0) is the first element in an array. Let TH=10, TH :the threshold .
6- For I= 0 to N-1
7- For J=0 to M-1
8- If X-Main(I,J) <= TH then Let C=C+1 Else Let X=Main(I,J) and C=0
9- End

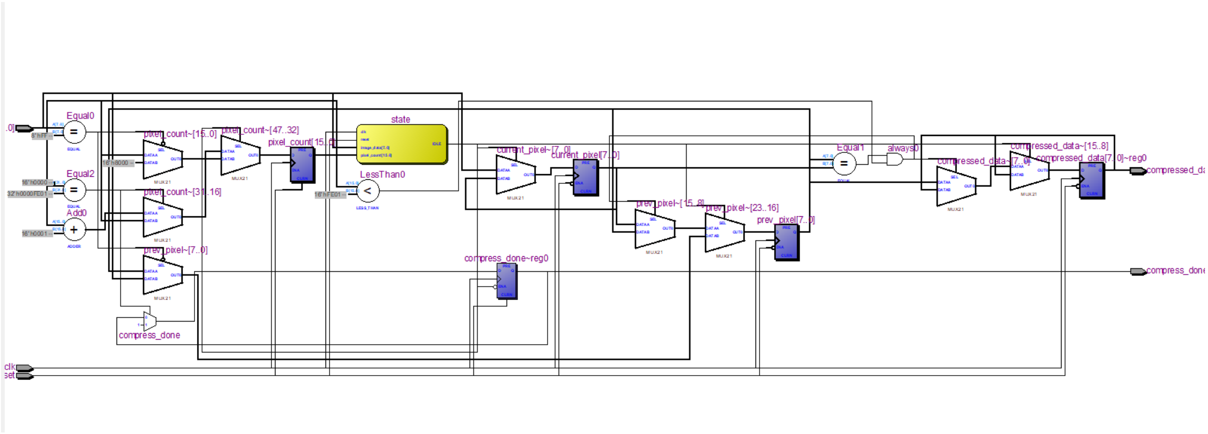




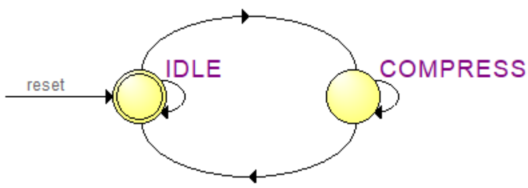
FLOW CHART FOR CODE:



RTL:



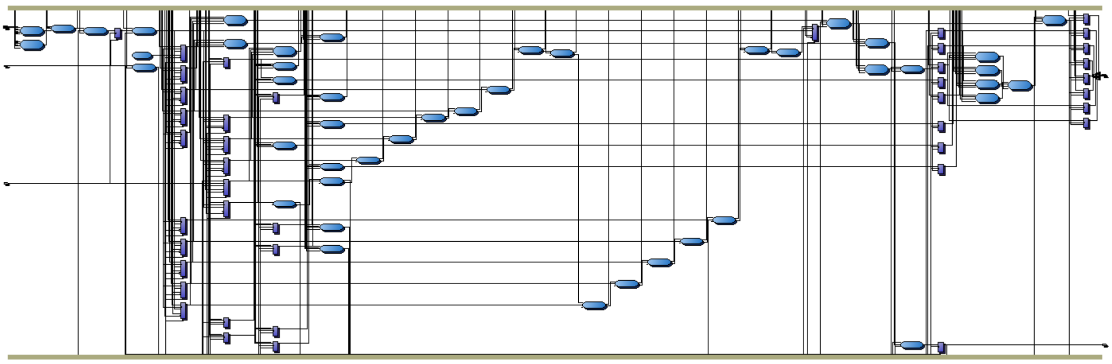
STATE DIAGRAM:



STATE TABLE:

	Source State	Destination State	
1	COMPRESS	COMPRESS	
2	IDLE	COMPRESS	(!image_data[0]) + (image_data[0]).(!image_data[1]) + (image_data[0]).(image_data[1]).(!image_data[2]) + (image_data[0]).(image_data[1]).(image_data[2]).(!image_data[3]) + (image_data[0]).(image_data[1]).(
3	IDLE	IDLE	(image_data[0]).(image_data[1]).(image_data[2]).(image_data[3]).(image_data[4]).(image_data[5]).(image_data[6]).(image_data[7])

TTL:



EXPEREMENTLE RESULTS:

Image Name (BMP)	Original Size (KB)	New Size (KB) Using RLE	Compression Ratio Original Size/ Size Using RLE	New Size (KB) Using proposed enhanced RLE	Compression Ratio Original Size/ Size Using proposed enhanced
Birds	475	586	0.8:1	133	3.5:1
Heart	919	1137	0.86:1	471	1.95:1
Light	226	298	0.003:1	211	1.07:1
Madina	80	89	0.9:1	38	2.1:1
Mansour	88	116	0.98:1	23	3.8:1
Microsoft	523	526	0.99:1	145	3.6:1
Ramadan	533	669	0.79:1	240	2.2:1
Rawda	998	775	1.28:1	343	2.9:1
Rose	127	160	0.79:1	62	2.08:1
Winnie	170	64	2.65:1	31	5.48:1

<u>CONCLUSION:</u>

In conclusion, measuring the number of neighbouring pixels that have the same gray level value is how Run-Length Encoding (RLE), an efficient lossless picture compression technique, operates. When compressing grayscale photos, RLE can also be used to encode each run of pixels with the same intensity as a pair (run length, pixel value). The pixel value can take up to multiple bits, depending on the amount of Gray levels, whereas the run length typically takes up one byte, allowing for runs of up to 255 pixels. RLE can greatly reduce the size of image files, particularly when dealing with grayscale or big areas of similar colour. RLE, however, might not work well for pictures with intricate textures or patterns, and it occasionally produces larger files.

1- Run length encoding algorithm is a method of compressing images depend on the number of adjacent pixels value in the image.
2- RLE algorithm is failing mostly when using it for compressing color images, because we need new field to store the number of repeated adjacent pixels.
3- Run length encoding algorithm is an efficient compression method with images have less various between values of adjacent pixels, but fail with images have high difference between adjacent pixels value.
4- When increasing the values of threshold in the proposed enhanced RLE algorithm this will yield increasing the compression ratio, and vice versa.
5- Controlling the compression ratio depends on the value of the threshold, which depends on the type of domain that image used it .

Code:

```
module gh (
   input  clk,
   input  reset,
   input [7:0] image_data,
   output reg [7:0] compressed_data,
   output reg compress_done
);

// Internal variables
parameter IMAGE_HEIGHT=255;
parameter IMAGE_WIDTH=255;
reg [7:0] prev_pixel;
reg [7:0] run_length;
reg [7:0] current_pixel;
reg [3:0] state;
reg [15:0] pixel_count;

// States
localparam IDLE = 2'b00;
localparam COMPRESS = 2'b01;

always @(posedge clk or posedge reset) begin
   if (reset) begin
      state <= IDLE;
      compress_done <= 0;
      pixel_count <= 0;
   end
   else begin
      case(state)
```

```
IDLE: begin
    // Wait for start signal
    if (image_data != 8'hFF) begin
        state <= COMPRESS;
        prev_pixel <= image_data;
        run_length <= 1;
        pixel_count <= 1;
    end
end

COMPRESS: begin
    // Compress image data using run-length encoding
    current_pixel <= image_data;
    if (current_pixel == prev_pixel && pixel_count < (IMAGE_WIDTH *
IMAGE_HEIGHT)) begin
        run_length <= run_length + 1;
    end
    else begin
        // Output compressed data
        compressed_data <= {run_length, prev_pixel};
        prev_pixel <= current_pixel;
        run_length <= 1;
    end

    // Check if all pixels are compressed
    if (pixel_count == (IMAGE_WIDTH * IMAGE_HEIGHT)) begin
        compress_done <= 1;
        state <= IDLE;
    end
    else begin
        pixel_count <= pixel_count + 1;
```

```
            end
          end
        endcase
    end
end

endmodule
```

REFERENCES:

1. Umbaugh,S.E. (1998), Computer Vision and Image Processing: A Practical Approach Using CVIPtools, First Edition, Prentice Hall, New York,253.

2. Salomon, D. (2007), Data Compression", Fourth Edition, Springer-Verlag,London,28-36.

3. http://searchsoa.techtarget.com/definition/imagecompression

4. Adrian, B. (9 July 2003), Image compression, Digital Preservation Guidance note 5, DPGN-05, 1-9,www.nationalarchives.gov.uk/documents/image_compression.pdf

5. Srinivasan,V., Image Compression Using Embedded Zero-tree Wavelet Coding (EZW),2, www.umiacs.umd.edu

6. Blelloch,E. (25 September 2010) , Introduction to Data Compression, Computer Science Department ,Carnegie Mellon University,www.cs.cmu.edu/afs/cs/project/pscicoguyb/realworld/www/compression.pdf

7. Carey,L. (August 2008), Understanding Digital Image, Conserve O Gram, Sponsored by the Park Museum Management Program, 22/2, 1-2, www.nps.gov/history/museum/publications/conserveogram/22-02.pdf

8. http://en.wikipedia.org/wiki/Image_file_formats

9. www.contentdm.com/USC/tutorials/image-file-types.pdf

10. www.wfu.edu/~matthews/misc/graphics/formats/formats.html

11. www.scantips.com/basics09.html

12. Miano, J. (1999), Compressed image file formats: JPEG, PNG, GIF, XBM, BMP, ACM Press/Addison-Wesley, New York.

13. www.fastgraph.com/help/bmp_os2_header_format.html

14. Jessica, F. (12 February 2002), ”Algorithms for M assive Data Sets Context-based Compression”, CS 493,1:1-3,www.docstoc.com/docs/54164044/Run-length-encoding

15.https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.researchgate.net%2Ffigure%2FRLE-Algorithm-compression-example_fig3_370355521&psig=AOvVaw0o_Ut7Q3wVN0UYOtXocAci&ust=1713003502943000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCNDslry5vIUDFQAAAAAdAAAAABAE