

Chapter 1

Introducing the Omnivision OV7670 Camera

In this chapter I cover the Omnivision ov7670 camera. First, a short description of the camera is given followed by some photos of the camera itself. Then key digital camera terminology needed to understand key concepts in this book are covered. I then give a more in depth explanation of the camera including details of each key part and the steps by which an image is captured, processed and transmitted to the Arduino.

What is the OV7670 Camera?

The ov7670 camera is a low cost widely available CMOS camera made by Omnivision Technologies located in Santa Clara, California. It comes in two versions one without frame buffer memory and one with frame buffer memory which is commonly called the FIFO version. In this book we will use the version with the FIFO frame buffer memory. The frame buffer memory holds image data that has been captured from the camera. The image data can then be transferred from the frame buffer memory to the Arduino's memory or to a storage device such as a SD Card. The ov7670 camera can be used with the Arduino through its SCCB interface that is compatible with the Arduino's I2C interface. The camera can be focused manually by turning the camera lens clockwise and counterclockwise which moves the lens outward and inward. The camera lens clockwise and counterclockwise which moves the lens outward and inward. The 1 shows a photo of the back side of an ov7670 camera with frame buffer memory labelled "Averlogic". Figure 1-2 shows a photo of the front of an ov7670 camera. Figure 1-3 shows a picture captured from a ov7670 camera.

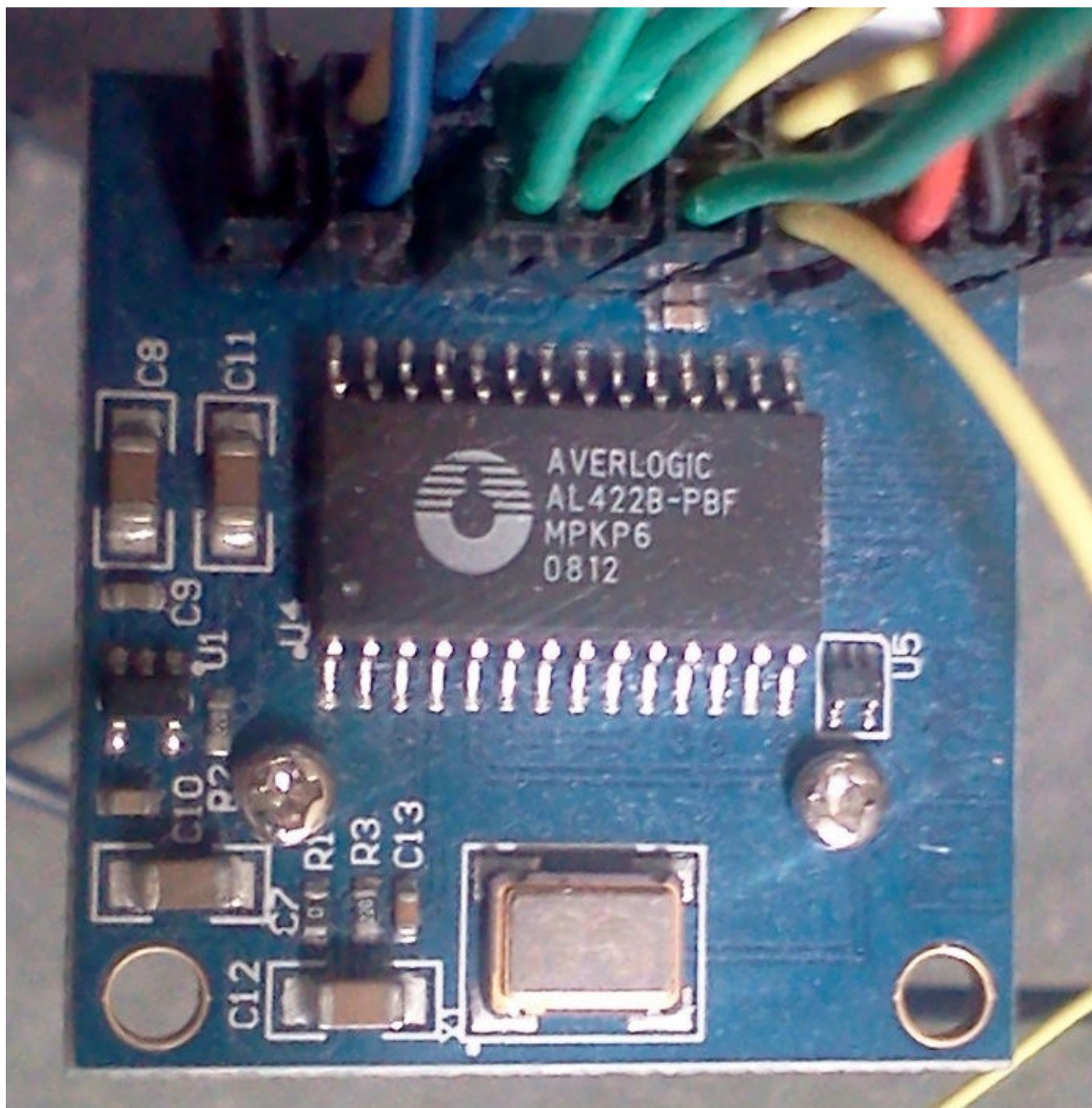


Figure 1-1. ov7670 FIFO camera version back side showing the Averlogic frame buffer memory

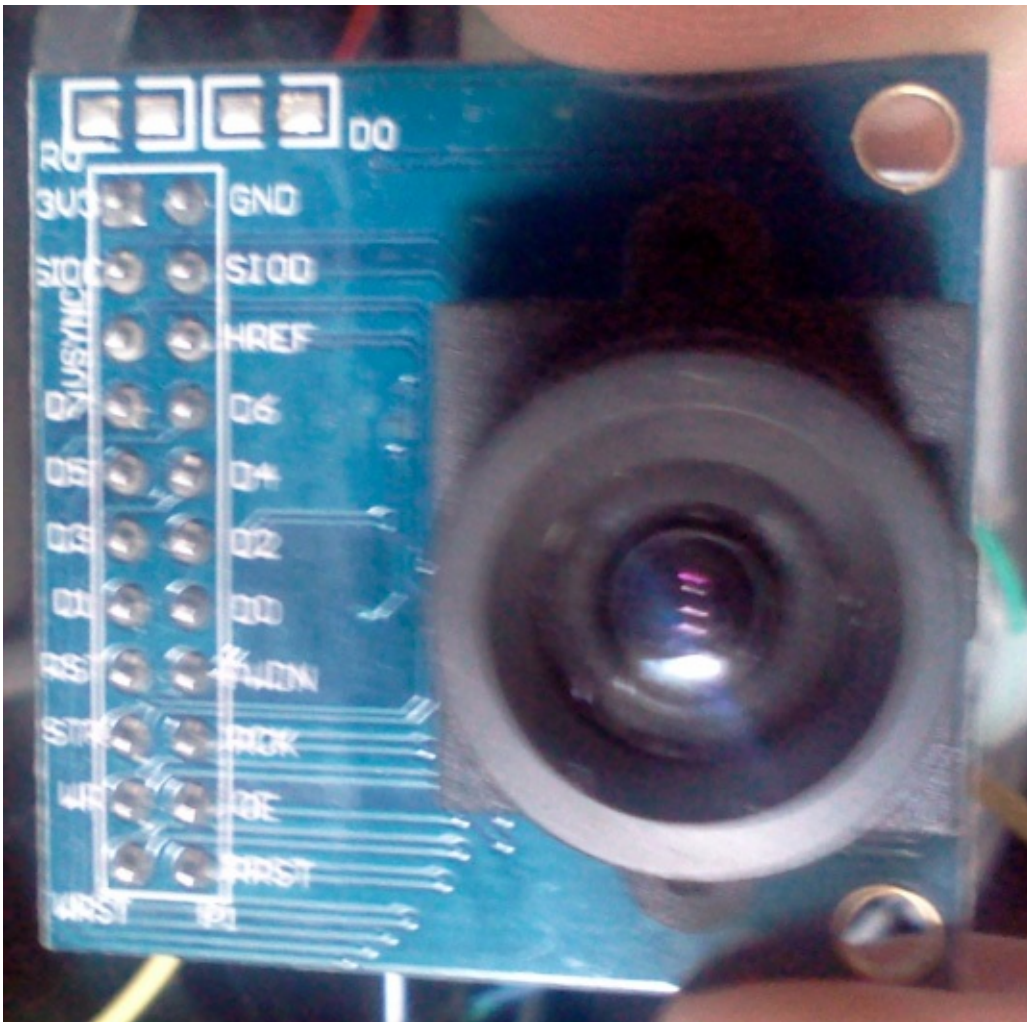


Figure 1-2. ov7670

FIFO camera version front side showing camera lens



Figure 1-3. Picture captured from an ov7670

camera

Key Camera Terminology

This section covers key terms related to digital cameras and traditional cameras.

- Pixel – A pixel is the smallest unit that makes up a digital image. It is generally a small square illuminated element that can take on various colors. For example, in Figure 1-4 you can see that the number “0” is composed of many pixels represented in the figure by black squares.

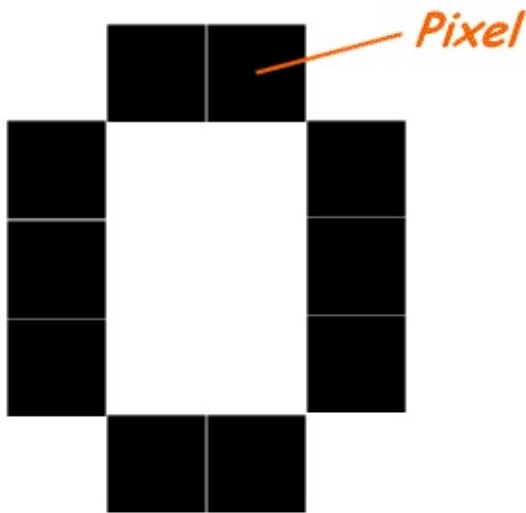


Figure 1-4. Group of pixels representing the image of the letter “O”

- Resolution – Resolution refers to the width and height of an image in pixels.
- VGA – VGA refers to a camera resolution that generates images that are 640 pixels wide and 480 pixels high.
- QVGA – QVGA refers to a camera resolution that generates images that are 320 pixels wide and 240 pixels high.
- QQVGA – QQVGA refers to a camera resolution that generates images that are 160 pixels wide and 120 pixels high. See Figure 1-5 for a comparison of the VGA, QVGA, and QQVGA resolutions.

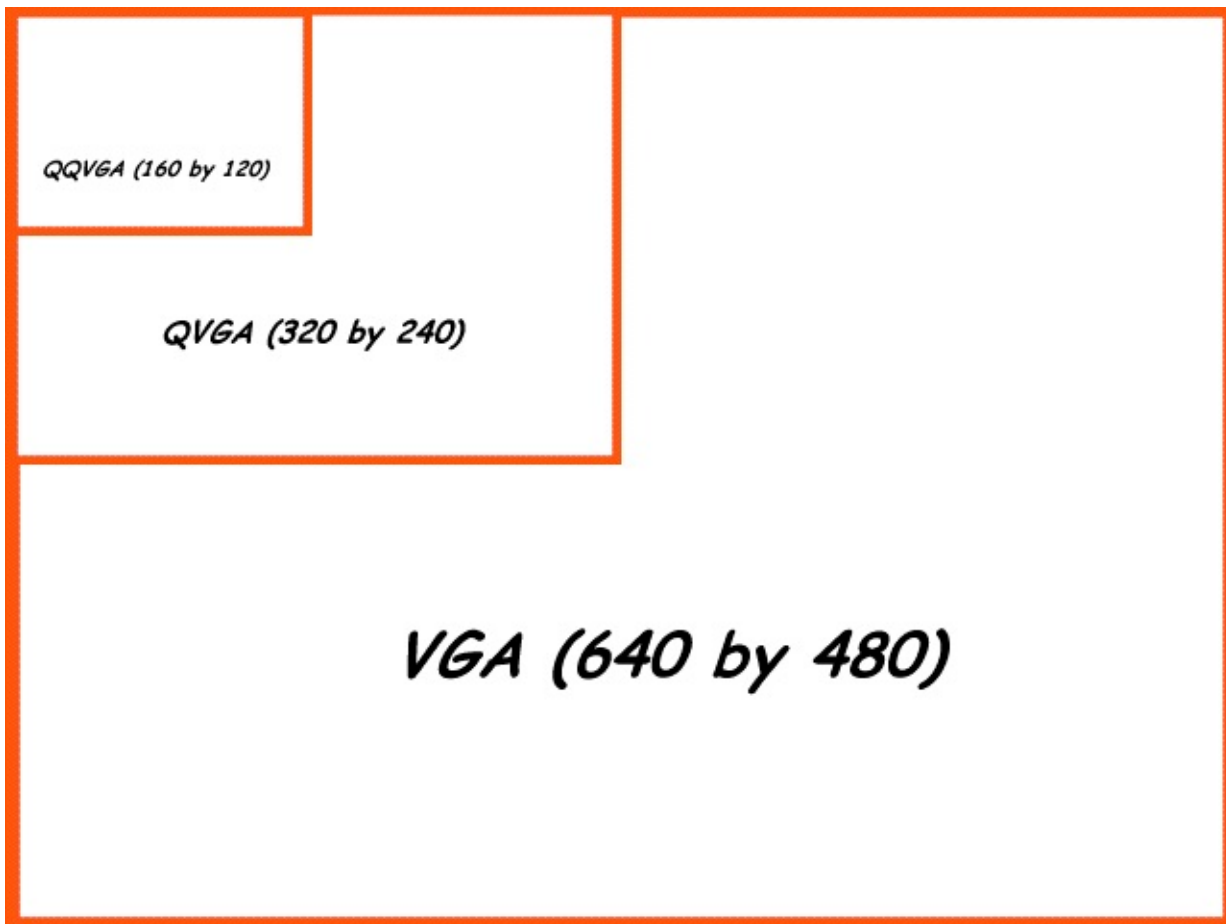


Figure 1-

5. Comparison of VGA, QVGA, and QQVGA resolutions

- CIF – CIF refers to a camera resolution that generates an image that is 352 pixels wide and 288 pixels high. See Figure 1-6 for a comparison with the VGA modes.

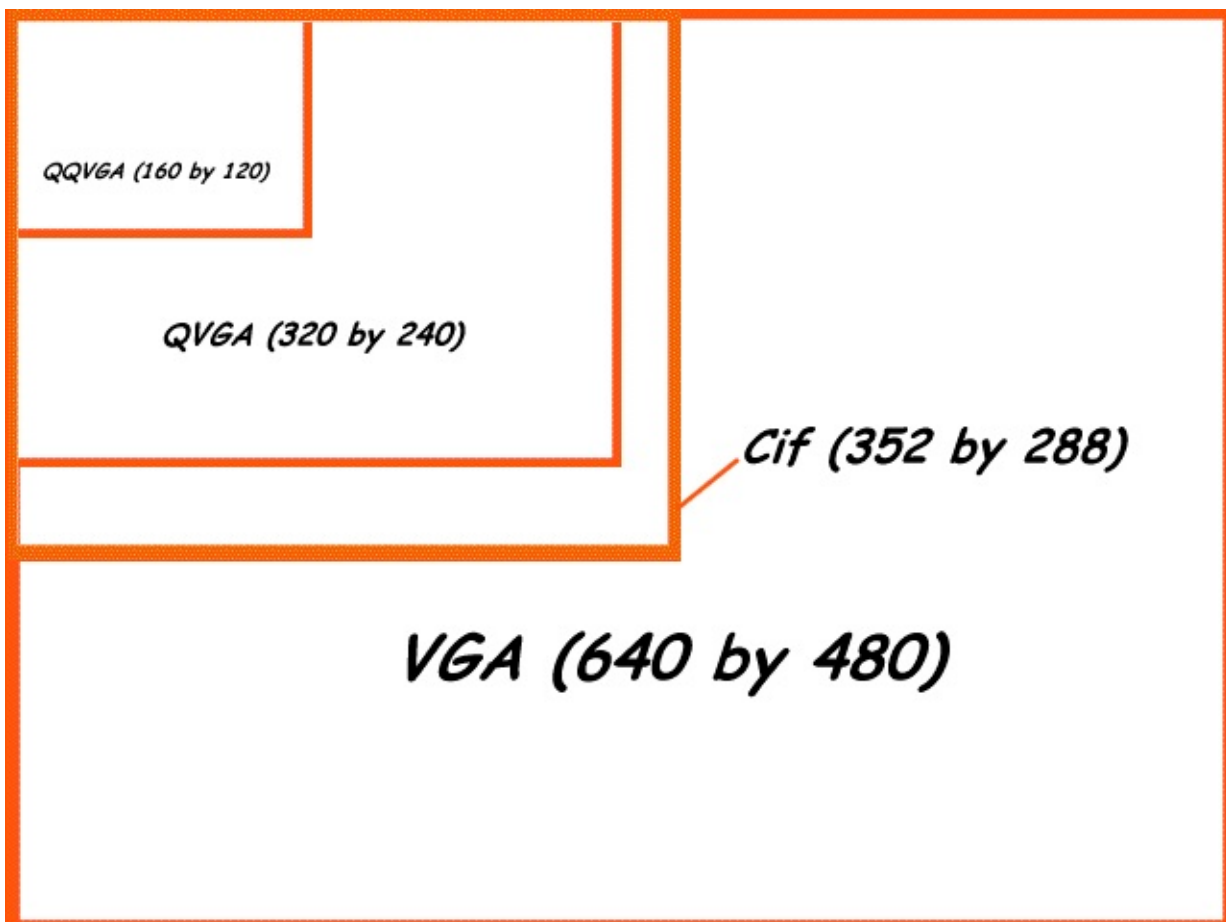


Figure 1-

6. Comparison of CIF with VGA, QVGA, and QQVGA resolutions

- **YUV** – YUV is an image encoding method and is discussed in more detail later in this book. The pixels that make up a digital image must be represented internally in an image format and YUV is one of the available image formats used to represent these pixels. The YUV format incorporates luminance or brightness values and color values.
- **YCbCr** – YCbCr is an image encoding method and is discussed in more detail later in this book. The pixels that make up a digital image must be represented internally in an image format and YCbCr is one of the available image formats used to represent these pixels. In the YCbCr image format a pixel incorporates luminance or brightness values, blue intensity values, and red intensity values.
- **RGB** – RGB is an image encoding method and is discussed in more detail later in this book. The pixels that make up a digital image must be represented internally in an image format and RGB is one of the available image formats used to represent these pixels. In the RGB image format each pixel contains a red, green, and blue component. The red, green, and blue components are added to get a final color. The red, green, and blue components set at the highest setting add up to white light. The red, green, and blue components set to the lowest setting represent the color black. See Figure 1-7.

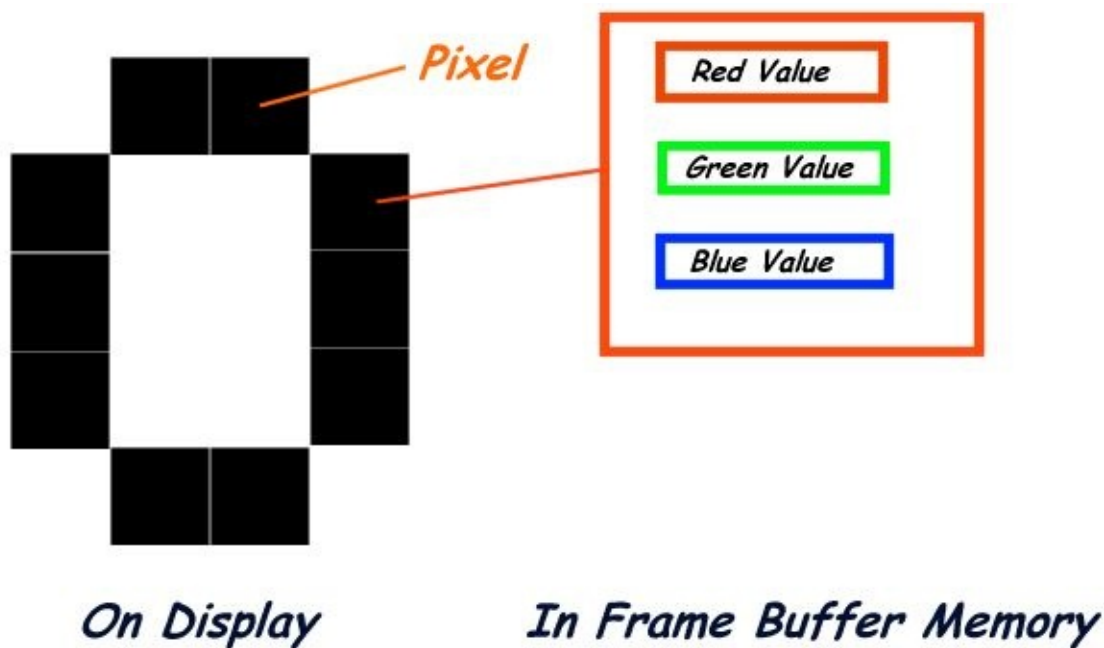


Figure 1-7.

RGB image format

- **Raw Bayer RGB** – Raw Bayer RGB is an image format where each pixel consists of raw sensor data of either red, green, or blue depending on the color filter at that pixel location. Raw Bayer is the format the photo is initially captured in before further processing. Raw Bayer is discussed more in depth later in this book.
- **Demosaicing** – Demosaicing is the process by which a raw bayer RGB image can be transformed into a full color image.
- **Exposure** – Exposure is the amount of light per unit area and can be increased by capturing a photo over a longer period of time or decreased by capturing a photo over a shorter period of time. The end result is that the longer the exposure the brighter the captured image will be. The shorter the exposure the darker the captured image will be.

- AEC – AEC stands for Automatic Exposure Control and means that the camera will adjust the exposure setting according to certain parameters.
- AGC – AGC stands for Automatic Gain Control and controls the luminance or brightness of the photo that is taken.
- White Balancing – White Balancing is the adjustment of the colors in an image generally the colors red, green, and blue for correcting neutral colors such as gray or white so that they appear grey or white in the photo.
- AWB – AWB stands for Automatic White Balancing which means that the camera will automatically adjust the colors of the image so that neutral colors such as grey or white will appear grey or white in the captured photo.
- BLC – BLC stands for Black Level Calibration which adjusts the level of black in the image with the objective of matching true black (zero brightness) in the environment to true black in the corresponding captured image.
- ABLC – ABLC stands for Automatic Black Level Calibration that automatically adjusts the black level in the captured image according to certain parameters.

OV7670 Camera with AL422B FIFO Memory Overview

This section gives an in depth description of the ov7670 camera. First, the general capabilities of the camera are summarized. Then each functional component of the camera is described in detail. This is followed by a step by step description of how an image is captured by the camera and then transferred to the Arduino.

General Summary Of Capabilities

- Good low light operation by using NightMode
- Low operating voltage (3.3 Volts) suitable for embedded portable apps such as Arduino based projects
- Maximum frame capture rate of 30 frames per second using VGA resolution
- Compatible with Arduino though use of the camera's SCCB interface which is compatible with Arduino's I2C interface
- Supports raw Bayer RGB, RGB, YUV, and YCbCr image formats as output
- Supports VGA, QVGA, QQVGA resolutions
- Automatic image control functions including: Automatic Exposure Control, Automatic Gain Control, Automatic White Balance, Automatic Black Level Calibration.
- Supports other image processing features such as edge enhancement, denoise operations, and color correction.
- 384K (393,216) bytes frame buffer memory which is enough to hold a VGA screen capture in raw Bayer format.

Camera Functional Block Diagram

This section discusses the individual components of the Omnivision ov7670 camera. Each component is labeled with an alphabet enclosed in a circle. Each of these components is then discussed in detail. See Figure 1-8 for the full camera functional block diagram.

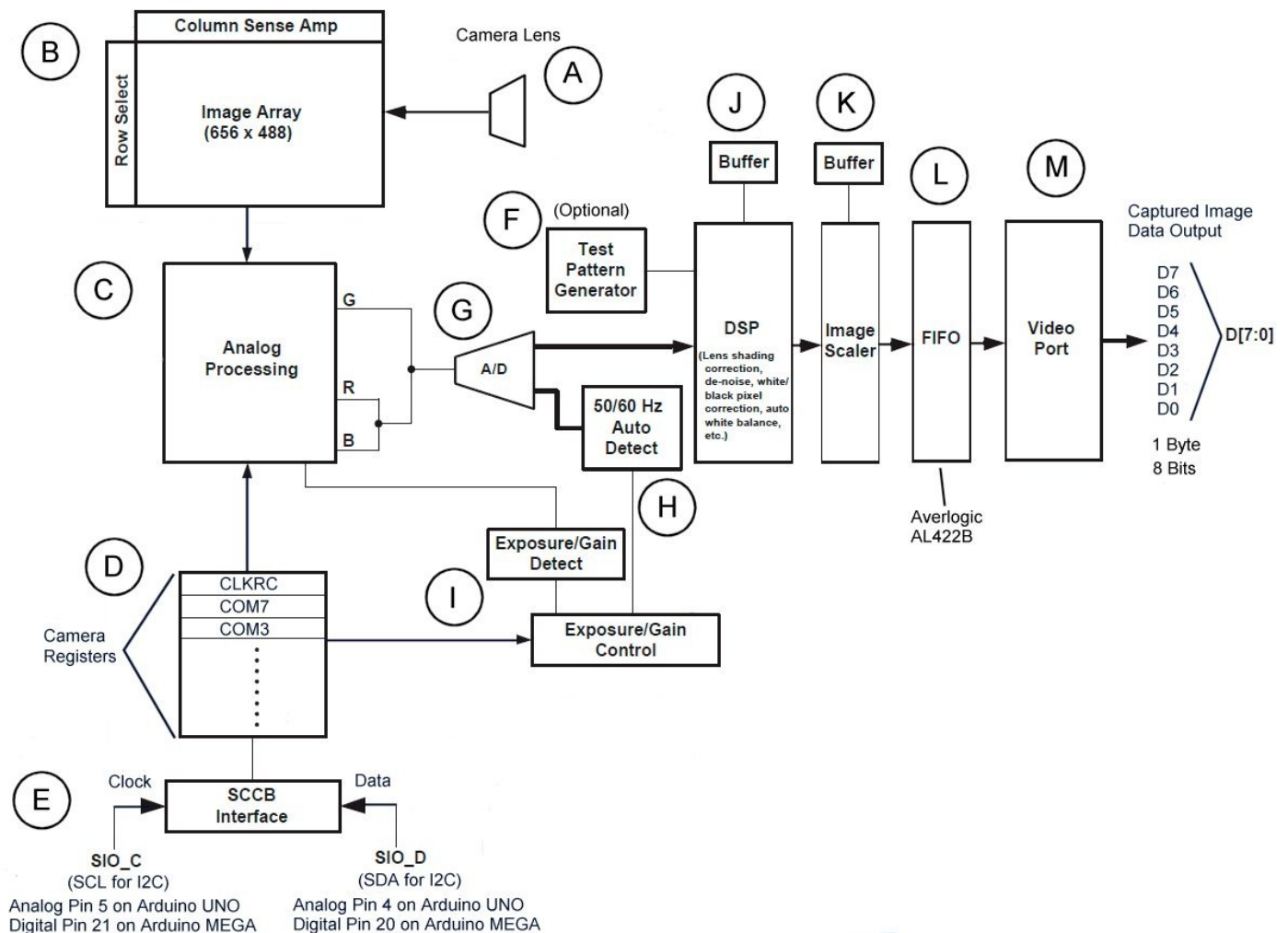


Figure 1-8. OV7670 Camera Functional Block Diagram

A. Camera Lens

The ov7670 has a lens that can be adjusted by screwing it in or out to adjust the focus of the image to be captured. Light first comes through this lens before hitting the camera's image array. See Figure 1-9.

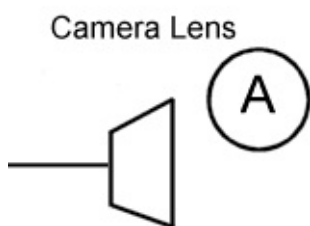


Figure 1-9. Camera Lens

B. Image Array

The camera's image array captures the incoming image and is 656 pixels wide and 488 pixels high. See Figure 1-10.

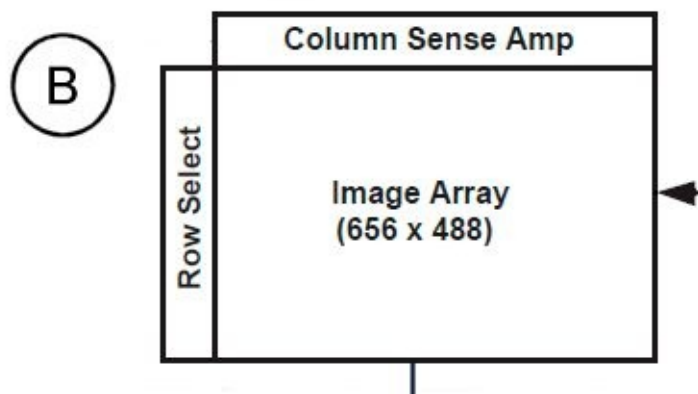


Figure 1-10. Camera's Image Array

The image array is covered with color filters arranged in a blue-green/green-red pattern. That is one row would contain color filters alternating between blue and green covering the sensor pixel one row would contain color filters alternating between blue and green covering the sensor pixel 11.

Bayer Filter Pattern

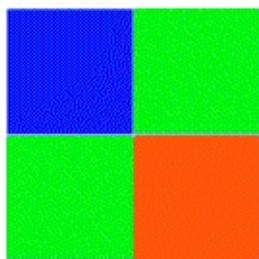


Figure 1-11. BG/GR Bayer Filter Pattern

The way the bayer color filters work is that the red, green, and blue filters only allow in the red green, or blue component of the light reflected from the image and this intensity level is measured by the image sensors located on the pixel cells of the image array. See Figure 1-12. In case A only the red light component is measured by the pixel cell sensor. In case B only the green light component is measured by the image sensor. In case C only the blue light component is measured. Thus, each pixel in a raw bayer format image represents the intensity level of either red, green, or blue light. Each final image pixel must contain red, green, and blue information for the pixel to be correctly displayed. Therefore, the raw bayer image must go through a process called demosaicing to estimate the missing two color components needed to display the pixel correctly.

The YUV and YCbCr camera output formats use the camera's built in demosaicing algorithms to generate the final correct image. The values of each pixel in the final image are determined by the light hitting that pixel directly as well as the light hitting the surrounding pixels. The camera can also generate GRB, RGB555/RGB565 formats which are converted from YUV/YCbCr. The raw bayer images can be demosaiced using a free public domain program called FFMPEG. I discuss FFMPEG later on in this book as well as these image formats.

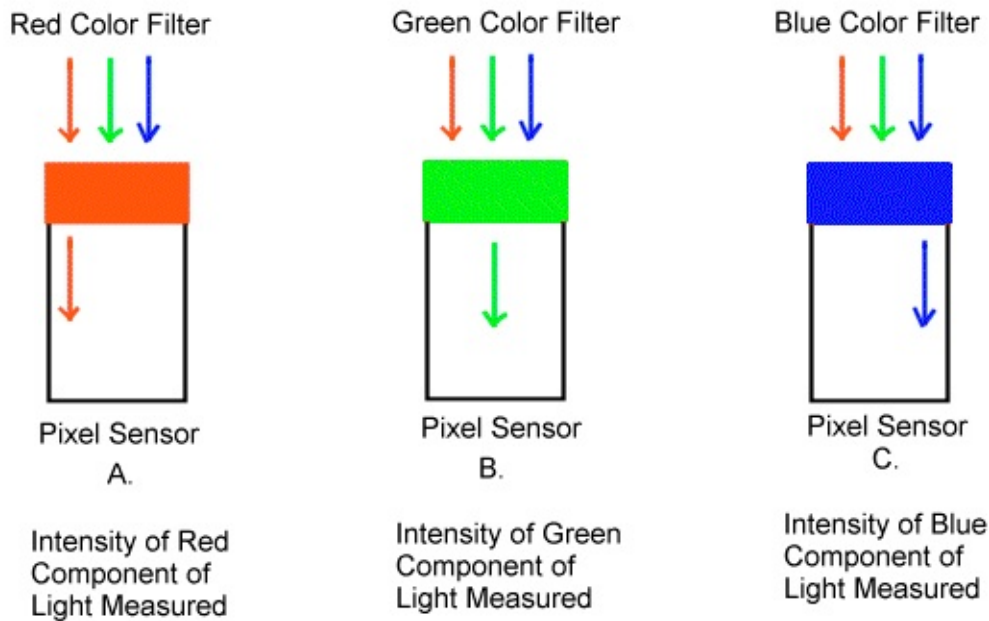


Figure 1-12. How

Bayer Color Filters Work

C. Analog Processing

The ov7670's analog processing includes exposure control, gain control, and black level calibration control. Gain controls can be set to manual or automatic. The term gain refers to the luminance or brightness of the image. Setting the gain to automatic tells the camera to control the image's brightness automatically without any other control inputs supplied by the user. Black level calibration can be set to manual or automatic and adjusts the black color in the captured image as close to the actual image as possible. The exposure control can be set to manual or automatic. The (AEC) automatic exposure control methods used can be average based or histogram based. (AEC) Automatic exposure control and (AGC) automatic gain control share the same algorithms and are used together to adjust the overall luminance or brightness of the image. See Figure 1-13.

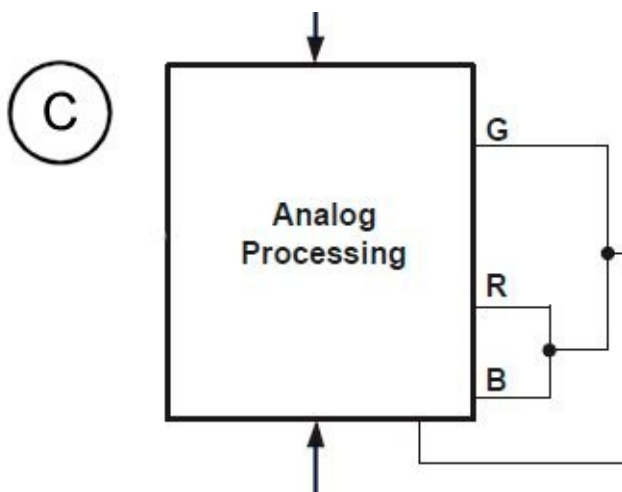


Figure 1-13. Analog Processing

The strategy in average based control of AEC and AGC involves changing the exposure and gain fast if the measured luminance is outside the control zone. Once the luminance is within the control zone the exposure and gain is changed in smaller amounts until the measured luminance is within the stable operating region. Once within the stable operating region there are no further changes to the camera's exposure and gain. See Figure 1-14.

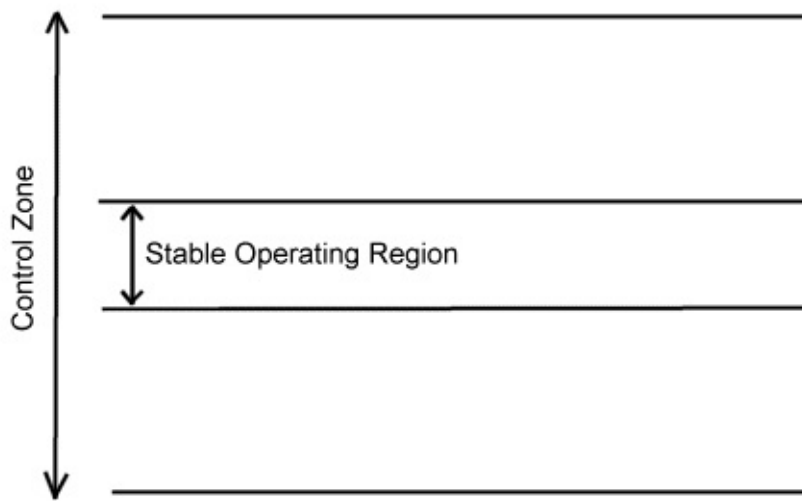


Figure 1-14. Average AEC/AGC

For the histogram method the exposure and the gain are changed until the luminance histogram reaches the desired distribution. We discuss the average based and histogram based AEC and AGC control methods more in depth later in this book.

D. Camera Registers

The registers in the ov7670 camera are memory cells that are 1 byte or 8 bits in length and hold values that are used to control the camera's functions such as resolution, image output format, exposure, gain, frame rate, etc. If you are new to digital design I discuss bytes and bits later in this book so don't worry if you are unfamiliar with these terms. You can set and read the values of these registers through the camera's SCCB interface using the Arduino. See Figure 1-15.

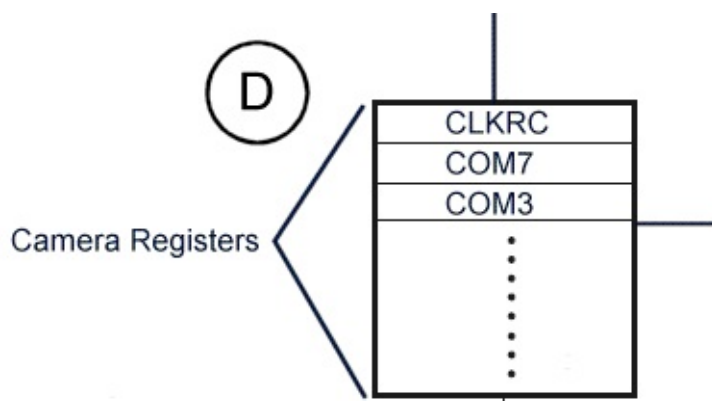


Figure 1-15. Camera Registers

E. SCCB Interface

This interface is used to read data from the camera's registers and to write data to the camera's registers. The SCCB interface on the camera is compatible with the Arduino's I2C interface and code used to activate and use the I2C interface will work with the camera's SCCB interface without any modifications. There are two pins which are the clock which is labeled the SIO_C and the data which is labeled SIO_D. The SIO_C is the same as the SCL on the I2C interface and the SIO_D is the same as the SDA on the I2C interface.

The SIO_C is connected to the Arduino UNO through analog pin 5 and is connected to the

Arduino MEGA through digital pin 21. The SIO_D is connected to the Arduino UNO through analog pin 4 and connected to the Arduino MEGA through digital pin 20. See Figure 1-16.

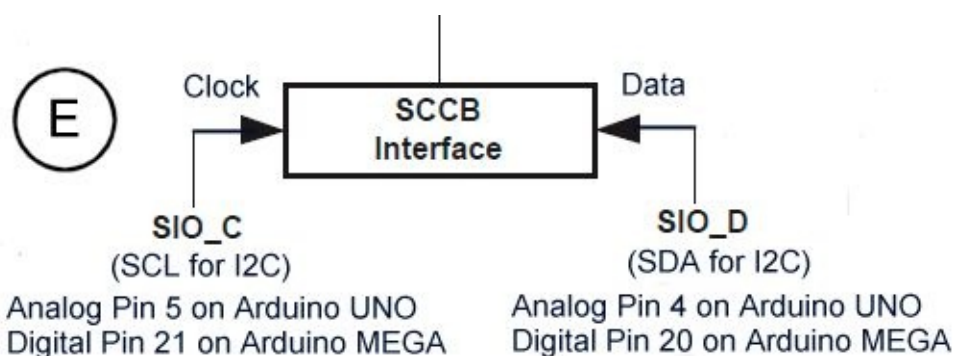


Figure 1-16. The SCCB Interface

F. Test Pattern Generator

The test pattern generator is used to display a standard set of vertical colored bars that are used to determine if the camera is working properly. Not only should a vertical group of colored bars be displayed clearly but the colors must also be in the right order. We get into more detail regarding the test pattern generator later in this book. See Figure 1-17.

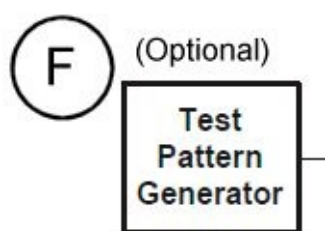


Figure 1-17. The Test Pattern Generator

G. Analog to Digital Converter

The analog to digital converter converts the raw bayer image from the image array to a digital format using a 10 bit converter. See Figure 1-18.

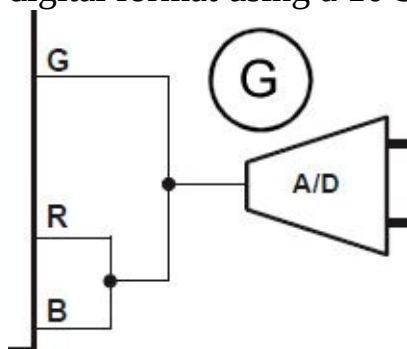


Figure 1-18. A/D Converter

H. 50/60 Hz Auto Detect

The 50/60 Hz auto detect can automatically detect the frequency of artificial light such as florescent light used in an office or home. This feature can be used with the camera's band filter features to remove any light banding that may occur in an image. See Figure 1-19.

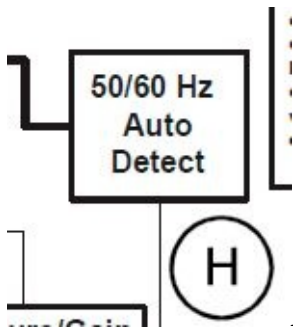


Figure 1-19. 50/60 Hz Auto Detect

I. Exposure/Gain Detection and Control

This component of the camera is responsible for detecting and controlling the exposure and gain of the image that is processed in the analog processing block. It receives control information from the camera's registers and then sets the exposure and gain accordingly. For automatic exposure and automatic gain control the camera automatically controls the exposure and gain based on the exposure and gain detected in the incoming image. See Figure 1-20.

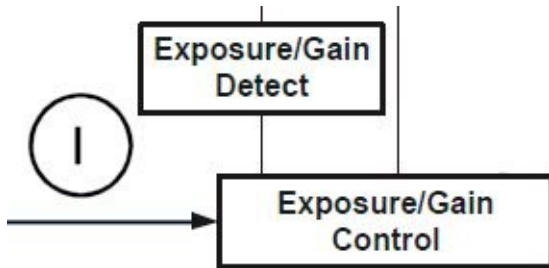


Figure 1-20. Exposure/Gain detection and control

J. Digital Signal Processor (DSP)

The digital processor or (DSP) receives digital image data from the analog to digital converter and is responsible for:

- White Balance Control
- Gamma Control
- Color Matrix
- Sharpness Control
- De-Noise
- Automatic Color Saturation Adjustment
- Defect Pixel Correction

See Figure 1-21.

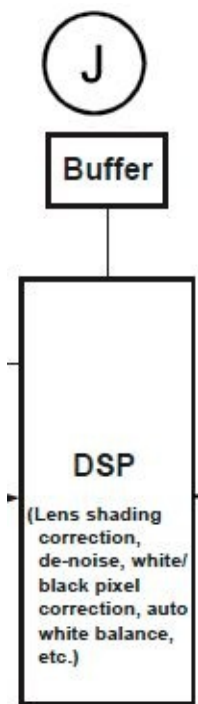


Figure 1-21. Digital Signal Processing (DSP)

White Balance Control

The white balance control for the camera allows for both manual and automatic control. The objective of white balance control is to make the white colors in the image white regardless of light color and can be set to normal (simple) mode or advanced mode.

The normal mode for automatic white balance makes the average values of the red, green, and blue colors for all the pixels in the image equal by changing the red, green, and blue gains. It assumes that the average of all the colors in the world is gray. The normal or simple mode does not depend on the characteristics of the camera lens being used to take the photo.

The advanced mode for automatic white balance uses the color temperature to adjust the red, green, and blue gains. The advanced mode depends on the characteristics of the specific lens that is being used to take the picture.

A separate pre-gain value for the red, green, and blue channels is also supported.

Gamma Control

Gamma control provides gamma correction to the image which controls its luminance or brightness. The user can set individual values that define a gamma curve that is used to lighten or darken the image.

Color Matrix

The color matrix can perform color correction and color conversion on the camera's image. The color matrix is used in conversion from raw bayer RGB to YUV/YCbCr. The matrix itself is 3 by 3 and is active in image formats YUV/YCbCr and image formats derived from YUV/YCbCr such as RGB565, RGB555, and RGB444.

Raw RGB values are converted to Cr and Cb values by multiplying the RGB value of a pixel by the ColorMatrix. The Y value is taken directly from the camera's sensor and is not affected by the ColorMatrix. See Figure 1-22.

$$\begin{bmatrix} Cr \\ Cb \end{bmatrix} = ColorMatrix \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Figure 1-22. YCbCr color matrix use

The ColorMatrix is built from the values in camera registers MTX1, MTX2, MTX3, MTX4, MTX5, and MTX6. All that is needed is to set those registers and the camera will use the new values in converting the original picture into the final image. See Figure 1-23.

$$ColorMatrix = \begin{bmatrix} MTX1 & MTX2 & MTX3 \\ MTX4 & MTX5 & MTX6 \end{bmatrix}$$

Figure 1-23. The ColorMatrix value

The same matrix is used for conversion of RGB values to YUV. See Figure 1-24.

$$\begin{bmatrix} V \\ U \end{bmatrix} = ColorMatrix \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Figure 1-24. YUV color matrix use

Sharpness or Edge Enhancement Control

The sharpness control can be either set to manual or automatic. The sharpness feature only works on processed bayer, YUV/YCbCr images or those that are derived from them. The raw bayer image does not contain any digital processing including sharpness or edge enhancement adjustments. If the sharpness control is set to automatic then the sharpness will vary according to a limits supplied by the user in the camera registers REG75 and REG76. In automatic mode the sharpness changes inversely with the gain. For example, the higher the gain the lower the sharpness.

De-Noise

The camera has a built in de-noise function that can be set to manual or automatic mode. In automatic mode the de-noise level is proportional to the gain. That is the greater the gain the stronger the de-noise applied to the image. The de-noise function will work on processed bayer RGB, YUV/YCbCr or any derived format such as RGB555, RGB565, and RGB444. De-noise will not work on raw bayer RGB since that format does not go through the digital signal processor.

Automatic Color Saturation Adjustment The camera can automatically adjust color saturation based on gain. The higher the gain the weaker the color.

Defect Pixel Correction The camera has built in pixel error correction to compensate for bad pixels on the image array.

K. Image Scaler

The image scaler reduces the size of the VGA image (if desired) that is output by the

camera's digital signal processor. All other resolutions are produced by scaling down the VGA image. See Figure 1-25.

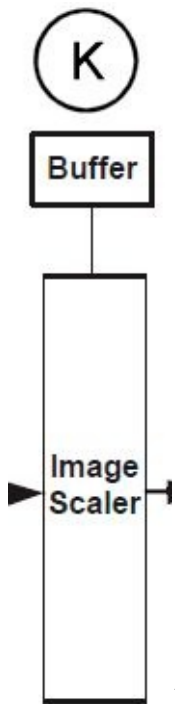


Figure 1-25. Image Scaler

L. FIFO Frame Buffer Memory

The FIFO frame buffer memory is made by AverLogic and the model is AL422B. It holds the image so that it can be read in by the Arduino. It is 384K which is enough to hold a VGA raw bayer RGB image of 1 byte per pixel with a resolution of 640 pixels wide by 480 pixels high. See Figure 1-26. An important item to be aware of is that the FIFO memory can only hold 1 byte per pixel at VGA resolution. If you attempt to write more than 1 byte per pixel to the frame buffer at VGA resolution such as trying to use the YUV image format mode with VGA then the image you get will be incorrect.



Figure 1-26. AverLogic AL422B FIFO memory

M. Video Port

The image from the FIFO memory can be output through the camera's video port. The video port is 1 byte or 8 bits in length. Thus, we will need to read the image data from the camera's video is 1 byte or 8 bits in length. Thus, we will need to read the image data from the camera's video 27.

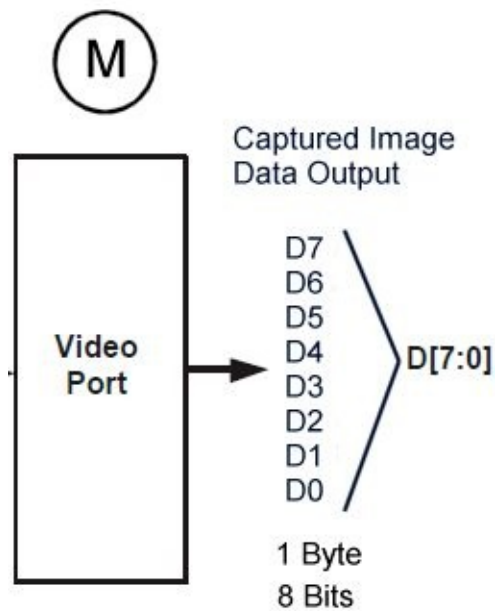


Figure 1-27. Video Port

Summary of Steps Needed for Taking a Photo

This section gives a general overview of what steps occur when an image is captured, processed and transferred to the Arduino using the Omnivision ov7670 digital camera. See Figure 1-28.

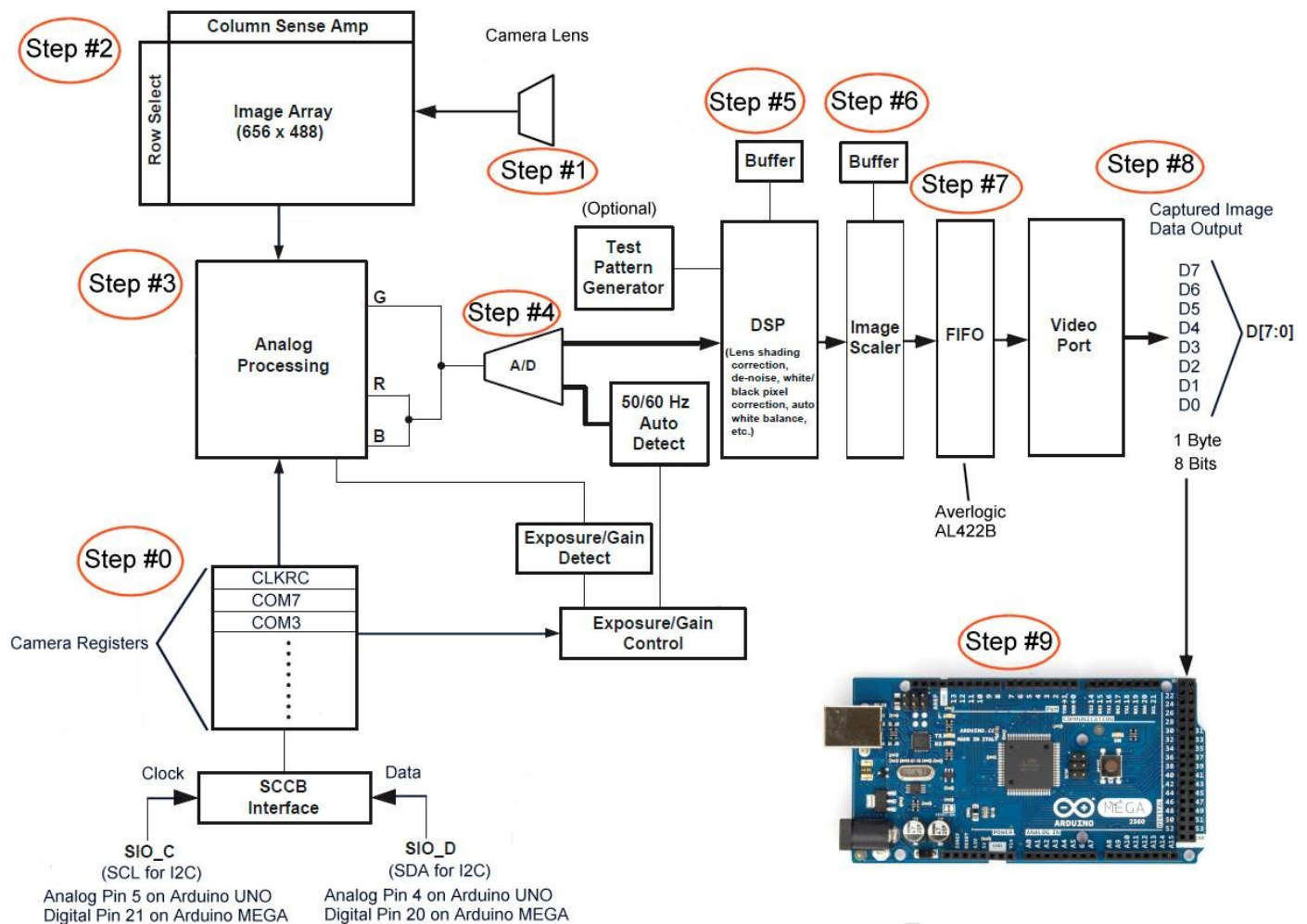


Figure 1-28. Steps in taking a photo

Step #0 – Setting the Camera’s Registers

A preliminary step before capturing the image is to set the camera resolution, set the image output format, and set other image processing parameters that the user desires. This is done by the Arduino using the ov7670 camera’s SCCB interface to write the required values to the camera’s registers.

Step #1 – The Camera Lens

The image that is to be captured by the camera must first go through the camera’s lens. It is here that the focus can be adjusted by the user by screwing the lens clockwise or counterclockwise to get a clear image.

Step #2 – The Image Array

The image array receives the incoming image after it goes through the camera’s lens. Here the camera’s pixel cell sensors detect the red, green, and blue components of the incoming light. These pixel cell sensors are arranged in a raw bayer image format of alternating rows of bluegreen/green-red pattern.

Step #3 – Analog Processing

Next, the image goes through analog processing where the analog items like the image exposure and gain are adjusted according to the camera's register values.

Step #4 – A/D Converter

Next, the analog image is sent through the analog to digital converter that converts the image into its digital form of bytes consisting of 0's and 1's.

Step #5 – Digital Signal Processor

Then, the image is processed by the digital signal processor that handles things like white balance, edge enhancement, and de-noising.

Step #6 – Image Scaler

Next, the digitally processed image is sent to the image scaler where it is reduced in size according to the values in the camera registers that control the size of the final image that is output. Remember that all images are first captured in VGA resolution but can be scaled down using the image scaler.

Step #7 – FIFO

Then, the final scaled image is sent to the FIFO frame buffer memory which holds the image so that it can be read and output to the Arduino.

Step #8 – Video Port

Next, the video port which consists of 8 output pins representing 8 bits or 1 byte is the actual physical point where wires are attached in order to send the image data out to the device that will receive the image data.

Step #9 – Arduino

Finally, the wires from the video port on the camera are connected to pins on the Arduino designated as input pins. From there the image data is read in one byte at a time until the entire image is processed. For example, the image can be saved to a SD card or transmitted via bluetooth to an Android device to be displayed.

Summary

In this chapter I covered the Omnivision ov7670 camera. I started with a discussion of key terms and concepts relating to digital cameras that were essential in understanding the rest of the book. Then I went into a detailed discussion of the camera involving key functions and then I discussed the steps an image went through when being captured, processed and then sent from the camera to the Arduino.