



# *QR CODES DECODER*

---

ENG629

*Name: Marta SANTAMARÍA SANTOYO*

*Supervisor: Barrie BIRMINGHAM*

*3<sup>rd</sup> May 2013*

---

# Author's Declaration

---

## **Statement 1**

This work has not previously been presented in any form to Glyndŵr University or at any other institutional body whether for assessment or for any other purposes. Save for any express acknowledgements, references, and/or bibliographies cited in the work, I confirm that the intellectual content of the work is the result of my own efforts and no other person.

## **Statement 2**

It is acknowledged that the author of this work shall own the copyright. However, by submitting this copyright work for assessment, the author grants to the University a perpetual royalty-free license to do all or any of those things referred to in section 16(1) of the copyright, designs, and patents act 1988 (viz: to copy work; to issue copies to the public; to perform or show or play the work in public, to broadcast the work or make an adaptation of the work).

Signed: \_\_\_\_\_

Date: \_\_\_\_\_

## ACKNOWLEDGMENTS

I must give thanks to Professor Barrie Birmingham for accepting this project under his direction. His support and confidence on my work and his ability to guide my ideas have been an invaluable contribution for the development of this project. I also have to thank the provision he always has offer to carry out all the activities proposed during its development.

I also want to express my sincere thanks to Mr. Logan López Giménez for his important contribution and active participation in the development of this project. I must emphasize on his availability and patience that made the project go on. There is no doubt that his participation has enriched the work done and also has meant a strong friendship.

Special and sincere thank to my family who has always supported me over the years the degree has lasted and also because, with his help, it has been possible to make this project during an ERASMUS program.

Finally, I want to thank the support and help that Iago and Gemma have offered me during this year, and in special to Maximilian, who has calmed me during the difficult times I went through in the realization of this project.

## CONTENTS

1. Introduction .....	2
1.1. Aim .....	2
1.2. Objectives .....	2
2. Research .....	3
2.1. Codes .....	3
2.2. Electronic devices .....	7
2.3. Programming Languages .....	10
2.4. Programs .....	10
2.5. Communication .....	12
2.6. Pixel Formats .....	12
2.7. Digital Image Processing .....	14
3. Design .....	19
3.1. Hardware Circuit .....	19
3.2. Digital Circuit .....	23
3.3. USB Driver .....	25
3.5. Software .....	28
4. Implementation .....	31
4.1. Hardware .....	31
4.2. Software .....	35
5. Testing .....	37
5.1. Simulation .....	37
5.2. Testing .....	38
6. Conclusion .....	41
References .....	43

---

APPENDIX I .....

APPENDIX II .....

APPENDIX III .....

## **1. Introduction**

### **1.1. Aim**

The aim of this project is to develop a device that demonstrates the ability to read barcodes, and is also capable of detecting and analysing QR codes from a picture.

### **1.2. Objectives**

- Investigate microcontrollers available in the market;
- Use of Digital Image Sensors;
- Investigate about reading and decoding barcodes / QR codes;
- Investigate different types of programming languages;
- Program/Hardware device;
- Design, build and test complete PCB;

First of all, a research of information about reading and decoding barcodes will take place. As a result, acquired knowledge will help in making the right decision about selecting the most suitable scanner.

After that, the principal block diagram will be designed giving the way to choose the hardware and develop the software required. It is important to take care about the correct choice of the microcontroller because is the main part of the system. The microcontroller will not only process the data acquired by the scanner but also will send to the computer the information registered about the item.

Finally, the device will be designed to be battery powered with a total cost around £25; also the technology expected to be used is CMOS logic and a flash programmable microcontroller.

## 2. Research

### 2.1. Codes

[By 1970, the Universal Grocery Products Identification Code or UGPIC was written by a company called Logicon Inc. The first company to produce barcode equipment for retail trade use (using UGPIC) was the American company Monarch Marking in 1970, and for industrial use, the British company Plessey Telecommunications was also first in 1970. UGPIC evolved into the U.P.C. symbol set or Universal Product Code, which is still used in the United States]<sup>1</sup>. The need to create a system for disseminating and managing products led to take the initiative to promote the use of standard code identification.

#### 2.1.1. Types of codes

##### 2.1.1.1. One-dimensional codes

Barcodes have been integrated into every aspect of the daily lives; they are located in supermarkets, department stores, pharmacies, etc... It might seem that they are all equal, but not so. Each type of industry has a symbolism that runs as its own standard.

The existence of multiple types of barcodes is due to the fact that the symbols are designed to solve specific problems. That is, there are different symbols for different applications, and each has its own characteristics.

They are just a different way of using numbers and letters encoded as combination of bars and spaces varying on the thickness. Barcodes store data that can be collected on it quickly and with great precision. They are a simple and easy method for encoding text information that can be read by optical devices, which send the information to a computer.

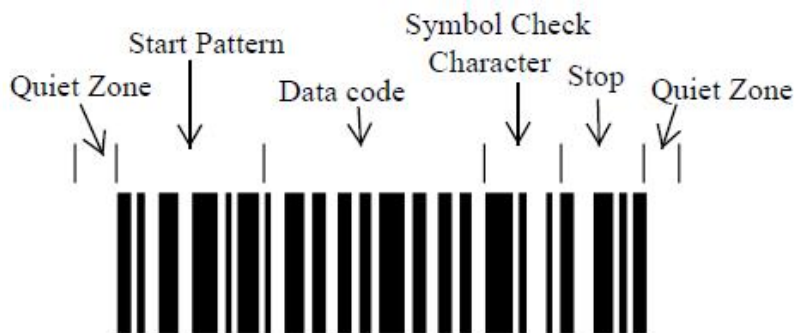
The barcode is the key to access a record in a database where the data actually resides, i.e., the symbols do not contain information about the product, it is a code that identifies the product. It has been created to identify objects and to facilitate data entry, eliminating the possibility of error in the catch.

Some of its advantages over other barcode data collection procedures are:

- It prints at a low cost;
- Very low error rates;
- Fast data capture;
- Teams reading and printing barcode are flexible and easy to connect and install.

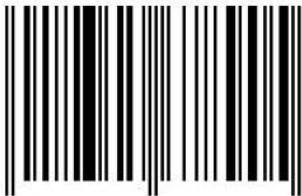
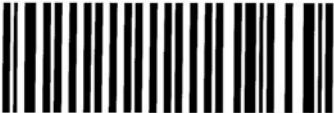
Some characteristics are common to all barcodes, such as density, WNR and the quiet zone.<sup>2</sup>

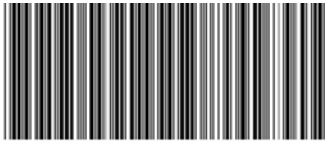
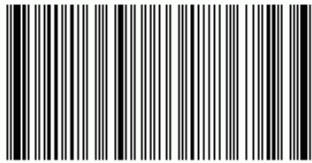
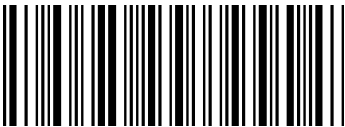
- Density is the width of element (bar or space) inside the narrowest barcode symbol. Is given in mils (thousandths of an inch). A barcode is not measured by its length but by its density physics.
- WNR (Wide to Narrow Ratio) is the reason for the thickness of the narrowest element against wider.
- Quiet Zone is the white area at the beginning and end of a barcode symbol. This area is needed for a convenient reading of the symbol.



**Fig.1:** Barcode characteristics.

**Table1.** Different types of linear barcodes:


Type of linear code	Description	Characteristics
<p><b>EAN code<sup>3</sup></b></p> 	<ul style="list-style-type: none"> <li>• EAN: European Article Number.</li> <li>• International standard created in Europe and worldwide accepted;</li> <li>• It identifies commercial products through the bar code indicating country-product company with a unique international key.</li> </ul>	<ul style="list-style-type: none"> <li>• 13 digits code (the first three digits identify the country, the following six the production company, the next three make reference to the article, and finally a check digit, which gives security to the system).</li> </ul>
<p><b>128 code<sup>4</sup></b></p> 	<ul style="list-style-type: none"> <li>• Used for logistics and parcel.</li> </ul>	<ul style="list-style-type: none"> <li>• High density and widely barcode;</li> <li>• It can only be encoded in alphanumeric or numeric;</li> <li>• With this code it is possible to represent all characters from the ASCII table, including control characters</li> </ul>

<p style="text-align: center;"><b>39 code<sup>5</sup></b></p> 	<ul style="list-style-type: none"> <li>This code is widely used and can be identified by almost any barcode reader.</li> </ul>	<ul style="list-style-type: none"> <li>Capable of representing uppercase letters, numbers and some special characters like the space</li> <li>Low code information density (more space is required to encode data)</li> </ul>
<p style="text-align: center;"><b>93 code<sup>6</sup></b></p> 	<ul style="list-style-type: none"> <li>Same nature as code 39.</li> </ul>	<ul style="list-style-type: none"> <li>Variable length symbology that is capable of encoding all 128 ASCII characters</li> <li>Code 93 offers higher density than Code 39 (it uses additional 4 shift characters to encode other characters)</li> </ul>
<p style="text-align: center;"><b>Codabar<sup>7</sup></b></p> 	<ul style="list-style-type: none"> <li>Codabar has a large installed base in libraries</li> </ul>	<ul style="list-style-type: none"> <li>Linear barcode specially designed to be read without problems even if it is printed by a dot matrix printer.</li> <li>The new symbology allowed containing more information on the same label size.</li> </ul>



### 2.1.1.2. Bi-dimensional codes.

Bi-dimensional codes are two dimensional dot arrays with large storage capacity and fast decoding. Data is encoded in the height and length of the symbol. The main advantage of using 2-dimensional codes is that it can contain a large amount of information that can be read quickly and reliably, without requiring access to a database wherein such data are stored (the case of the codes one-dimensional).<sup>8</sup>

**Table 2.** Different types of two-dimensional codes:

Type of bi-dimensional code	Description	Characteristics
<p style="text-align: center;"><b>PDF417<sup>9</sup></b></p> 	<ul style="list-style-type: none"> <li>The name derives from the symbolism code format. PDF stands for “Portable Data Format” and “417” is derived from the module structure, 4 bars and 4 spaces in a 17 module wide structure.</li> </ul>	<ul style="list-style-type: none"> <li>Variable length code that can encode virtually any letter, number or character.</li> <li>It supports text, numbers, and bytes and can accommodate up to 340 characters per square inch with a maximum capacity of 1,850 characters.</li> </ul>



<p><b>DataMatrix<sup>10</sup></b></p> 	<ul style="list-style-type: none"> <li>• 2D symbology capable of encoding variable length of 128 ASCII characters and a lot of different character sets</li> </ul>	<ul style="list-style-type: none"> <li>• It can hold up to 500 MB per square inch with a data capacity of characters from 1 to 2.355.</li> <li>• It has a high degree of redundancy and resistance to printing defects.</li> </ul>
<p><b>QR codes<sup>11</sup></b></p> 	<ul style="list-style-type: none"> <li>• QR: Quick Response.</li> <li>• It stores alphanumeric information in a two-dimensional array of points.</li> </ul>	<ul style="list-style-type: none"> <li>• Its can include different types of information such as textual information, contact information, locations, events, etc...</li> <li>• It can store data in different types of encoding; numeric, alphanumeric, binary and Kanji.</li> </ul>

### 2.1.2. Barcodes Readers

The reader projects a beam of light moving about the code, through it from end to end. It analyzes the patterns of reflected light, transforming them into data that a computer can interpret.

There are hand-held and also fixed scanners, such as those used in supermarket checkouts. They can be connected in various ways, such as USB, serial port, Wi-Fi, Bluetooth and even directly connected to the keyboard port by through an adapter.

A scanner for reading barcodes comprises a scanner, a decoder and a cable that acts as an interface between the decoder and the terminal or computer. The scanner reads the bar code symbol and provides an electrical output to the computer corresponding to the bars and spaces of the barcode. The function of the decoder is to recognise the barcode symbology, analyse the content of the read barcode and transmit the data to the computer in a traditional data format.

The scanner can have the decoder built into the handle or it may be scanner without a decoder that requires a separate box, called interface or emulator.

Barcodes are read by passing a small spot on the bar code symbol printed. Only see a thin red line emitted from the laser scanner is visible. A device in the scanner receives the reflected light and converts it into an electrical signal.

The laser begins to read the bar code on a white space (fixed area) before the first bar and continue going to the last line, ending in the white space that follows it.<sup>12</sup>

There are four main types of readers:<sup>13</sup>

- Single Point LED;
- Laser;
- CCD (Charge Coupled Device);
- Omni-directional laser.




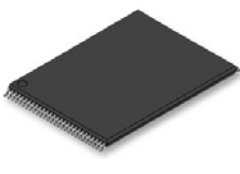
### 2.1.3. Printers and scanners

Barcodes can be printed in several ways. Some methods are described here.

- Master film. This method is used to print barcodes on printers and is mainly used in commercial packaging.<sup>14</sup>
- Laser. A laser printer can be used to print labels of low volume or serialized documents which are printed occasionally.<sup>15</sup>
- Thermal printing. This is the best technology for printing high volumes of labels. Medium and high speed industrial printers are used to print on paper, thermal paper, plastics, and sometimes on metal.<sup>16</sup>

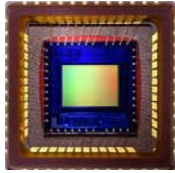
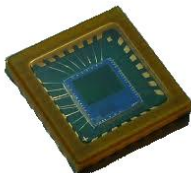
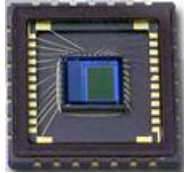
## 2.2. Electronic devices

### 2.2.1. Memories

Specifications	Device			
	CY7C1049DV33 <sup>17</sup>	CY62136EV30LL <sup>18</sup>	AL422B <sup>19</sup>	A43L0616BV <sup>20</sup>
				
<b>Memory type</b>	Static RAM	Static RAM	FIFO	DRAM
<b>Memory size</b>	4Mbit	2Mbit	3Mbit	16Mbit
<b>Memory Configuration</b>	512K x 8-bit	128K x 16-bit	384K x 8-bit 384K x 8-bit	512K x 16 x 2 Banks
<b>Power Supply</b>	3.0~3.6V	2.2~3.60V	5V or 3.3V	3.0~3.6V
<b>Package</b>	TSOP (SMD)	TSOP-2 (SMD)	SOP (SMD)	TSOP-2 (SMD)

**Table 3.** Different types of memories and characteristics.

### 2.2.2. Digital Image sensors

Parameters		Device		
		MT9V131C12STC <sup>21</sup>	MT9V011 <sup>22</sup>	OV7670/OV7171 <sup>23</sup>
				
Optical format		1/4-Inch (4:3)	1/4-Inch (4:3)	1/6-Inch
Frame rate	VGA (640 x 480)	Maximum of 30 fps	Maximum of 30 fps	Maximum of 30 fps
	CIF (352 x 288)	Programmable up to 60 fps	Programmable up to 60 fps	Programmable up to 60 fps
	QVGA (320 x 240)	Programmable up to 90 fps	Programmable up to 90 fps	Programmable up to 90 fps
	QCIF (176 x 144)	-	-	Programmable up to 120 fps
Active pixels		640H x 480V (VGA)	640H x 480V (VGA)	640H x 480V (VGA)
Pixel size		5.6µm x 5.6µm	5.6µm x 5.6µm	3.6µm x 3.6µm
ADC Resolution		10-bit	10-bit	8-bit
Color filter array		RGB Bayer pattern	RGB Bayer pattern	RGB Bayer pattern
Package		44-Ball iCSP 48-Pin CLCC	28-Pin PLCC	24-Pin CSP2
Power Supply		2.8V±0.25	2.8V±0.25	2.45~3V
Serial Bus Description		2-wire serial Interface	2-wire serial Interface	Standard SCCB Interface

**Table 4.** Different CMOS Digital Image Sensors and characteristics.

Some distributors, such as AMAZON, offer the OV7670 digital image implemented on a small PCB, including optical lens.<sup>24</sup>

Another module can be found, the digital sensor added to the FIFO AL442B memory chip. This module is also manufactured on a small PCB, and also includes the lens.<sup>25</sup>



**Fig.2:** OV7670 Digital Image Sensor with Optical Lens.

### 2.2.3. Microcontrollers

Features	Device			
	MC9S08QG8 <sup>26</sup>	PIC18F4550 <sup>27</sup>	PIC18F2550 <sup>28</sup>	COP912C <sup>29</sup>
Voltage Supply	5V	4.2~5.5V	2~5.5V	6V
Package	24-Pin	40-Pin	28-Pin	20-Pin
Core size	8-bit	8-bit	8-bit	8-bit
RAM	512B	2KB	2KB	64B
CPU Speed	20MHz	48MHz	48MHz	5MHz
XOSC	YES	YES	YES	YES
ICS	YES	YES	YES	YES
ACMP	YES	YES	YES	YES
ADC	8-ch	13-ch	10-ch	8-ch
DBG	YES	YES	YES	YES
I2C	YES	YES	YES	YES
IRQ	YES	YES	YES	YES
KBI	8-pin	8-pin	8-pin	8-pin
MTIM	YES	YES	YES	YES
SCI	YES	YES	YES	YES
SPI	YES	YES	YES	YES
USB	NO	YES	YES	NO
TPM	2-ch	4-ch	4-ch	2-ch
I/O pins	12 I/O	35 I/O	24 I/O	16 I/O
Package Types	24 QFN	40 DIP	28 DIP	20 DIP

**Table 5.** Different microcontrollers and main characteristics

### 2.2.3. Controller Voltage Supply

All electronic circuits require one or more continuous stable voltage sources. Here, some options are described.

#### USB<sup>30</sup>

The USB connection allows the possibility to act as a controlled voltage supply. For the version 2.0, the constant voltage supply is 5V with a tolerance of 5%, and a maximum current value of 500mA.

#### Voltage Regulator<sup>31</sup>

This electronic device maintains a constant voltage level. There exists a wide range of voltage regulators depending on its use. For its implementation, the voltage regulator needs a input voltage supply to generate a fixed output voltage.

### 2.3. Programming Languages

There exist a lot of types of languages for programming microcontrollers. Below, some of them are described in more detail.

#### **C**<sup>32</sup>

The C language was always distinguished as highly expressive and potentially very economical language due to its small number of keywords and the power of certain operators (e.g.: pointers.) At present, however, there is often a desire to support increasingly complex programming structures, whereby C language implementations tend to appear dark and unreliable versus other languages.

#### **C++**<sup>33</sup>

The purpose of its creation was to extend the successful C programming language with mechanisms for manipulating objects. In that sense, from the point of view of object-oriented languages, the C + + is a hybrid language. A special feature of C++ is the ability to redefine operators, and to create new types that behave like fundamental types.

#### **Assembler**<sup>34</sup>

Assembly language or assembler is a programming language for low-level computers, microprocessors, microcontrollers, and other programmable integrated circuits. It implements a symbolic representation of the binary machine code and other constants needed to program the CPU architecture and it is the direct representation of the machine code.

#### **BASIC**<sup>35</sup>

BASIC (Beginner's All-purpose Symbolic Instruction Code) Language is intended to facilitate the problems of complexity of older languages, this new language was designed specifically the simple user, who was not interested in speed, but the fact to be able to use the machine. It is very easy to use, even for beginners, so an understanding of computer hardware is not required.

## 2.4. Programs

### 2.4.1. Hardware

#### **Proteus**<sup>36</sup>

PROTEUS is a program to simulate complex electronic integrated circuits including developments made with various types of microcontrollers; it is a high performance tool with impressive graphics capabilities.

The program is able to fully replace the circuit board and help design the automatic trace PCBs. Furthermore, the program can simulate analogue or digital circuitry qualitatively.

This simulator has 2 major modules such as the ISIS module, which is used for simulation and debugging of digital circuits, and ARES module, used for developing PCB or printed board components.

### **Multisim<sup>37</sup>**

NI Multisim is a very useful tool for designing electronic circuits and simulations. With NI Multisim it is easy to design an electronic circuit from the really beginning using all kinds of components, simulating its performances and analyzing each of its sections.

The interactive components offer the possibility of controlling and removing data from instrumentation during the modelling system process, and also the possibility of measuring analogue and digital signals.

Specialists can optimize their projects minimizing errors and reducing the number of iterations in the design using Multisim in combination with Ultiboard software; used to design the PCB layout.

### **MPLAB<sup>38</sup>**

MPLAB is a tool for writing and developing assembly language code for PIC microcontrollers. MPLAB provides all the necessary tools for the completion of any project, as well as a text editor, and a simulator in which you can execute the code step by step to see its evolution and the state in which their registers are at any time.

It is a program that runs under Windows; it presents, among others, the classic bar program, menu, and tool condition. The MPLAB environment has text editor, compiler and simulation (not real time).

### **Circuit Maker 2000<sup>39</sup>**

Circuit Maker 2000 is a tool that allows designing analogue and digital circuits in a very simple way. It is able to simulate the circuit behaviour with many electronic devices, and also the possibility of measuring data such as voltage, and current, allowing the analysis of the circuit.

## **2.4.2. Software**

### **CCS C Compiler<sup>40</sup>**

It is a very useful tool for programming PIC microcontroller family, the CCS PCW Compiler, is a compiler that allows writing programs in C language instead of assembler, which achieves a lower development time, and very easily programming.

Once debugged, the program is able to create a simulation that can be implemented in a circuit simulation program such as PROTEUS due to its good integration and adaptability.

### **CodeWarrior<sup>41</sup>**

CodeWarrior is a tool based on an IDE (Integrated Development Environment). It integrates editing files, optimized compilation, simulation code, debugging and programming of different devices using Freescale processors. This program brings the versatility to program in assembly language, C, C++.

### **Visual Basic<sup>42</sup>**

Visual Basic is a graphical environment for developing applications for the Microsoft Windows operating system. The applications created with Visual Basic are based on objects and are handled by events. Visual Basic was derived from BASIC language, which is a structured programming language. However, Visual Basic programming model employs an event-driven.

## **2.5. Communication**

The interface with a computer can be performed by any of the known external ports: serial, parallel or USB. The parallel port is hardly found in today's computers, so just serial and USB ports will be described.

### **RS232<sup>43</sup>**

RS232 (Recommended Standard 232, also known as Electronic Industries Alliance RS-232C) is an interface that designates a standard for the exchange of a serial binary data between a DTE (Data Terminal Equipment) and a DCE (Data Communication Equipment). It defines typical mechanical, electrical, functional and procedures of a protocol oriented to physical link. This standard is based on asynchronous communication, that is, the data can be transmitted at any time, so the transmission and reception must be synchronized.

### **USB<sup>44</sup>**

The USB (Universal Serial Bus) is a Serial transmission and reception between host and serial devices. This port is designed to connect multiple peripherals to a computer. It is a serial bus architecture for the computer and telecommunications industries, which allows to install peripherals without having to open the machine for putting on the hardware, i.e. it just need to connect the peripheral in the back of the computer. Nowadays, all the computers, both laptops and desktop computers, are provided with this port.

## **2.6. Pixel Formats**

### **Greyscale<sup>45</sup>**

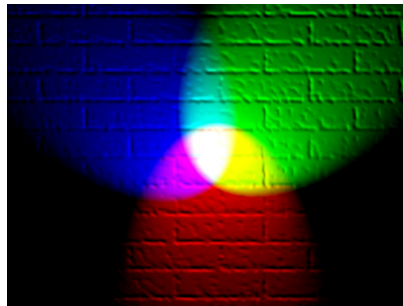
Each pixel is stored as 8 bits, representing gray scale levels from 0 to 255. Where 0 is black, 255 is white, and the intermediate values are grey.



**Fig.3:** Image in greyscale format

### RGB<sup>46</sup>

The RGB colour model approach is the decomposition of any colour in red, green and blue light at different intensities. The intensity of each light can go from 0 to 255, where 0 is the absence of light, and 255 is the maximum intensity.



**Fig.4:** RGB lights

### YCbCr<sup>47</sup>

YCbCr is a format in which a RGB colour can be encoded. The Y or luminance component is the amount of white light of a colour, and the Cb and Cr are the chrome components, which respectively encode the blue and red levels relative to the luminance component.



**Fig.5:** Decomposition of an image into its Y, Cb and Cr components.



## 2.7. Digital Image Processing

When reading a QR code, some difficulties in identifying the code can be found, such as uneven brightness images with different angles and distances, and geometric distortion problems due to the image device capture.

The capture of the code will not be sufficient to identify the data it contains. The code can be rotated, stretched or with different types of zoom. Therefore, it is necessary to process the image before applying an algorithm to decode QR codes.

The process to recognize QR codes is as follows:

### 2.7.1. Binarization of the image

The binarization of the image is very important in the process of identifying QR codes. It is the process of reducing the information, where only two values persist: true and false. In a digital image, these values, true and false, can be represented by the values 0 and 1, or more frequently, by the colours black (gray value 0) and white (gray value 255).

A simple but effective way to separate the black and white values shown below:

$$g(x, y) = \begin{cases} 1 & \text{if } f(x, y) > \text{threshold} \\ 0 & \text{if } f(x, y) < \text{threshold} \end{cases}$$

The key point is the choice of binarization suitable threshold value. This can be done manually by assigning the desired threshold, although there are methods for calculating the optimal threshold value automatically.

A good method to calculate the threshold value automatically is the Otsu method, which has been established as standard.

- **Otsu Method**<sup>48</sup>

The Otsu method uses statistical techniques to choose the best threshold. Specifically, the variance is used to measure the dispersion of the gray levels.

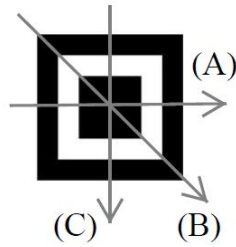
Otsu's method calculates the threshold value so that the dispersion within each segment is as small as possible, but at the same time the dispersion is as high as possible between different segments. For this, the quotient between the two variances is calculated, and a threshold value for which this ratio is maximal is searched.

### 2.7.2. Skew correction

<sup>49</sup>

When the image is captured, often it appears rotated and needs a rotation operation to be corrected. The correction process operates as it follows.

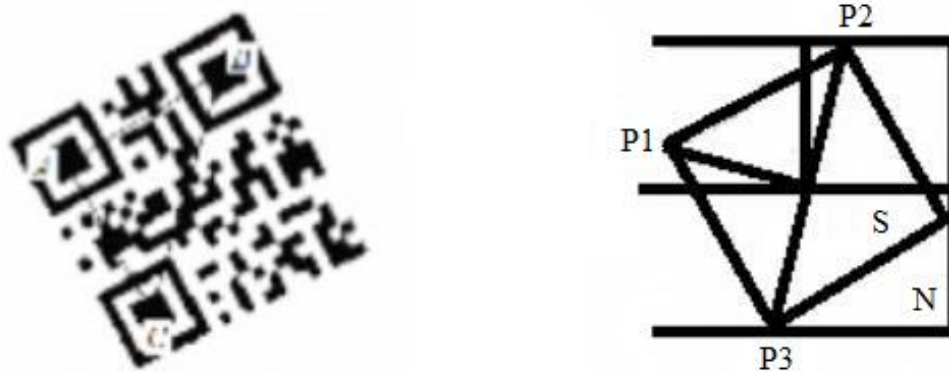
It is known that whatever the rotation patterns A, B and C are, they all have the same size and shape. The detection patterns consist of three overlapping concentric squares, the modulus ratio between black and black in all directions will 1:1:3:1:1.



**Fig.6:** Possible rotation patterns

To detect the three corners of the code, a row per row swept of the picture is done. A black pixel may belong to a corner of an edge, so, the other pixels of the same row are read while the proportion is met. If the proportion is met, the square detected can probably be a corner, but it is needed to verify vertically looking for the other square. The next step is to take horizontally the central point of the row where the proportion is detected, and from there, is checked up and down if the column meets the proportion. If it happens, it is definitely a corner, and the centre of the area of the column where we have the proportion is stored. To detect the following squares it is needed to limit the study to different regions of the image. Then, the rotation angle is determined.

As the figure shows, the area of square S is the original QR Code image, the square N is the area after the rotation, and P1, P2 and P3 are the centres of the detection patterns (also called "vertices").



**Fig.7:** Detection of the position of the corners

Assuming the rotation angle  $\Theta$ , the greater distance between the centre points of the three vertices, P2 and P3, is calculated. To calculate the slope of P2, P3:

$$k = \frac{(y_3 - y_2)}{(x_3 - x_2)}$$

**Eq.1:** Calculation of the slope

If  $k > -1$  and  $k < 1$ , then a rotation in the clockwise direction is required.

$$\Theta = \arctan((k - 1)/(k + 1))$$

Otherwise, a rotation in the opposite clockwise direction is required.

$$\theta = \arctan(- (k - 1)/(k + 1))$$

After determining the coordinates of the image centre, the length of one side is calculated:

$$LewLenght = \sqrt{2} * OriginalLength * \sin \theta$$

**Eq.2:** Calculation of the new length

After the rotation is carried out, the image interpolation is not distorted. The “nearest neighbour”, bilinear and trilinear methods are the most used to rotate the image.

**I. Geometric correction of the image**

The geometric distortion of the image can be due to the angle of shot, the rotation of the image if the surface on which the code it is located is not flat, etc. This distortion can decrease the rate of recognition of codes.

The algorithm for the geometric correction is explained below:

- **Obtaining the four corners of the image**

Due to the rotation process, the external noise has been removed, so the code is scanned line by line in the 8 possible directions (up, down, left, right and diagonals).

The code is scanned in straight line until two or more intersections find a black square (vertex). After the scanning in the 8 directions, at least 16 points are found. The point shown in both directions is the vertex.

When these steps are completed, three or more vertices would have been found. Based on the distance between the vertices and the centre of the detection patterns, the three shorter vertices are obtained, which are the centres of the patterns.

- **Obtaining the new values and vertices**

The vertices P1, P2, P3 and P4 are the four corners of the QR code image without distortion.

P1', P2', P3' and P4' are the vertices of the current image. f(x,y) is the value at coordinate (x,y) in the corrected image, f(x',y') the value at coordinate (x',y') in the uncorrected image.

Assuming the coordinates of Pi

$$P_i(1 \leq i \leq 4) \text{ are } (x'_i, y'_i),$$

and the coordinates of Pi'

$$P'_i(1 \leq i \leq 4) \text{ are } (x_i, y_i).$$

The following relationship exists:

$$\begin{aligned} x_i &= ax'_i + by'_i + cx'_iy'_i + d(1 \leq i \leq 4) \\ y_i &= nx'_i + my'_i + px'_iy'_i + q(1 \leq i \leq 4) \end{aligned}$$

**Eq.3:** Relation between the coordinates

The parameters a, b, c, d, m, n, p, q can be solved by Jacobi's iteration method, which makes the transformation from a parallelepiped to a square.

- **Obtaining the new pixel value**

When the point (x',y') becomes the point (x,y) by correcting the geometrical distortion, a bilinear interpolation to determine the value of f(x, y) must be performed. Assuming the 4 adjacent coordinates (x',y'); (x<sub>0</sub>, y<sub>0</sub>), (x<sub>1</sub>, y<sub>1</sub>), (x<sub>2</sub>, y<sub>2</sub>), (x<sub>3</sub>, y<sub>3</sub>), and the values of the coordinates f(x<sub>0</sub>, y<sub>0</sub>), f(x<sub>1</sub>, y<sub>1</sub>), f(x<sub>2</sub>, y<sub>2</sub>), f(x<sub>3</sub>, y<sub>3</sub>), the interpolation bilinear formula is:

$$\begin{aligned} f(x_i, y_i) &= [f(x_1, y_0) - f(x_0, y_0)](x - x_0) + [f(x'_1, y'_0) - f(x'_0, y'_0)](y - y'_0) \\ &+ [f(x'_1, y_1) + f(x_0, y'_0) - f(x_0, y_1) - f(x'_1, y'_0)](x - x'_0)(y - y'_0) \\ &+ f(x'_0, y'_0) \end{aligned}$$

**Eq.4:** Interpolation bilinear formula

In the new versions of QR codes, an alignment pattern is added and makes this task easier.

- **Image normalization**<sup>50</sup>

Obtained the geometrically corrected image, the image processing is almost finished. The last step is to normalize the image so that the decoder understands the information is introduced.

First, the version number of the decoding algorithm for the QR code must to be obtained. Then the image is split in small QR code matrix (n x n). The value for n must be commensurate to the version number of the algorithm, which obtained in the previous step. The centre of each matrix is searched and set as a sampling point. From there, the value of the symbol of the matrix can be obtained by making an average of the values contained in it.

### 2.7.3. Correction of the error

<sup>51</sup>

Although these operations can change some pixels of the image, the QR can correct the errors. There are four levels of error correction depending on the percentage of recovering.

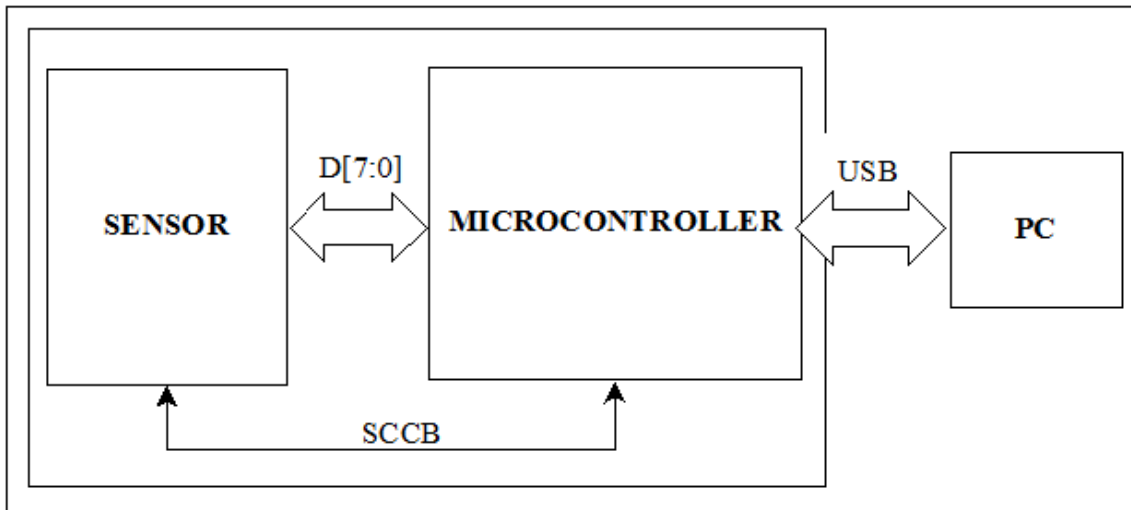
Capability of the QR code to correct errors	
L level	7%
M level	15%
Q level	25%
H level	30%

**Table 6:** Percentage of error correction

### 3. Design

#### 3.1. Hardware Circuit

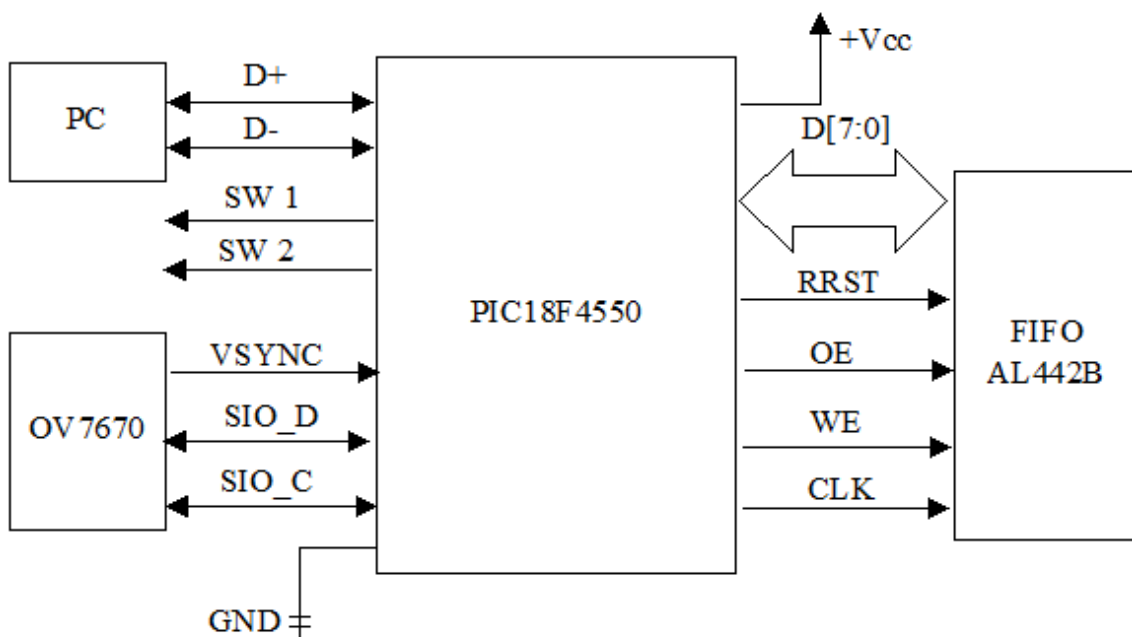
The block diagram below describes the flow of data in the circuit. The hardware is divided into two blocks; the first block is represented by the PIC microcontroller, which is connected to the PC via USB Interface. And the second block, the sensor, which is controlled by the microprocessor via the SCCB Interface and the bi-directional bus of data.



**Fig.8:** Circuit Block Diagram.

The two blocks are described in more detail below.

##### 3.1.1. PIC18F4550 Microcontroller



**Fig.9:** PIC18F4550 Connections Diagram

The diagram above shows the acquisition and processing of the data. The OV7670 can be accessed by microcontroller via the SCCB Interface, which is compatible with the I2C Interface.

The PIC interfaces with the PC via USB through the two differential data signals, labelled as D+ and D-.<sup>52</sup>

### **3.1.2. OV7670 Digital Image Sensor + FIFO AL442B + Optical Lens**

The design of the QR codes reader is developed with this module. Thus, the hardware design is reduced and there is no need to use an external memory to store the data from the camera sensor.

Another advantage of using this module is that the optical lens is already implemented on the chip, which facilitates and reduces the problem of the design and implementation of the camera.

#### **Operation**

The image sensor can be configured and controlled through the SCCB Interface, Serial Camera Control Bus, which is compatible with I2C.

The sensor can output several types of frame formats:

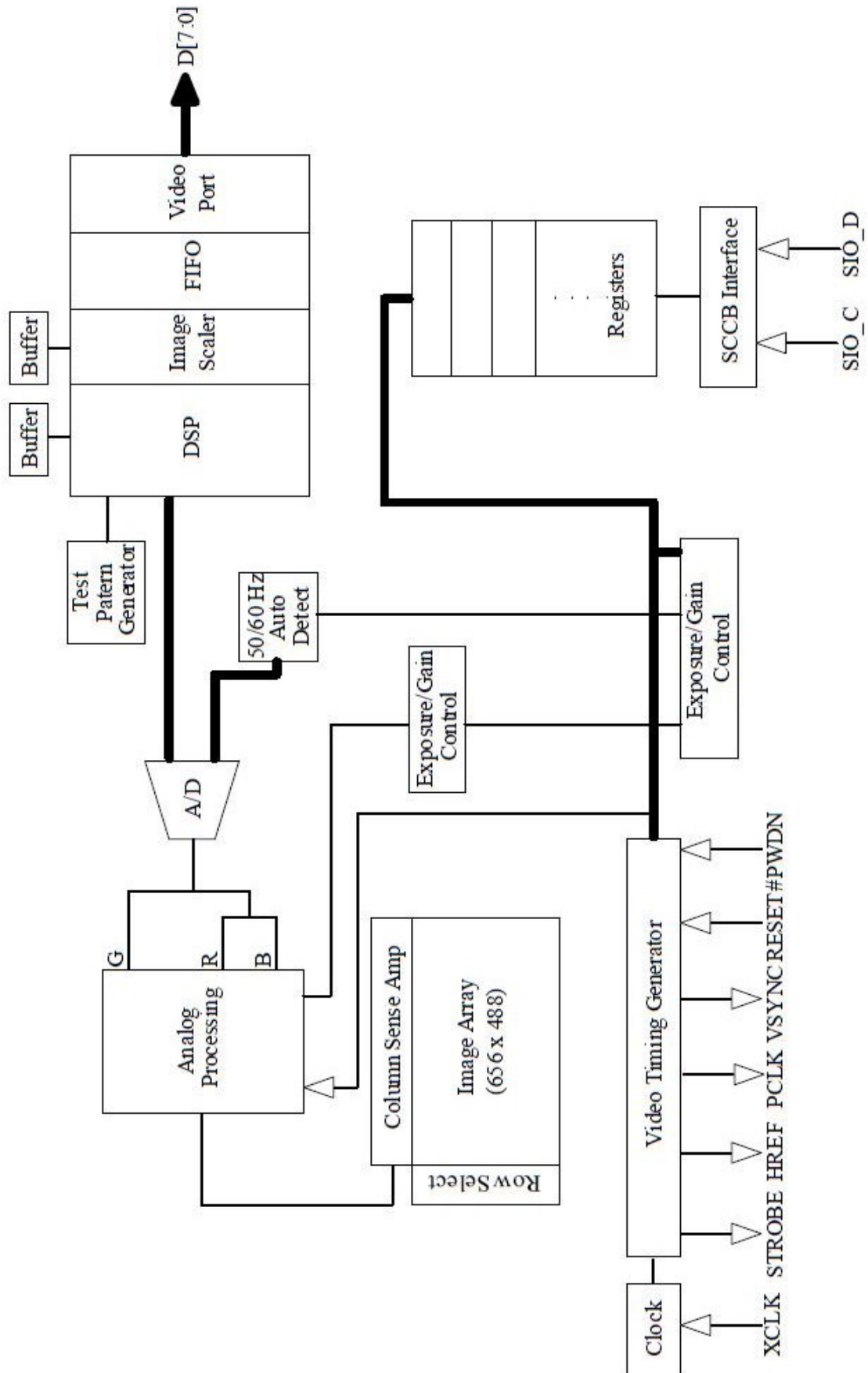
- YUV/YCbCr 4:2:2;
- RGB565/555/444;
- GRB 4:2:2;
- Raw RGB.

The frame is sent through a 8-bit bus, D[7-0], and some commands that help to control the camera are:

- VSYNC – Vertical sync output – Low during frame;
- HREF – Horizontal reference – High during active pixels of row;
- PCLK – Pixel clock output – Free running clock. Data is valid on rising edge.

The OV7670 uses its own clock; a 24MHz oscillator is already implemented and tracked in the module.

The internal memory AL422B uses two different frequencies for reading and writing modes. During the writing process, it also uses the oscillator from the module, for the reading process, an external clock must be connected to the RCK pin.<sup>53</sup>



**Fig.10:** OV7670 CMOS VGA Camerachip Functional Block Diagram



### 3.1.3. Components

The components used in the design of the circuit are described below. For the schematic circuit, look to the Fig.14: Schematic Circuit.

#### Header and lead (SEN1, 15cm LEAD)

The main reason for using a 24-pin header (SIN1) with a 15cm lead to connect the sensor to the PCB is, basically, to make the sensor mobile. The lead offers the possibility of moving the sensor without losing any pin-connection.

#### 40-Pin socket (PIC1)

The use of a 40-pin socket, labelled PIC1, for the PIC microcontroller is due to the fact that if the microcontroller has to be reprogrammed, and it facilitates to unplug it from the socket avoiding, this way, desoldering and resoldering the device, which could cause the deterioration of the PCB.

#### Ceramic Capacitors (C1, C2, C3, C6)

The ceramic capacitors used in the design are radial decoupling capacitors. They are placed close to the power pins (Vcc and GND) of each device, not only to keep the inductance as low as possible, but also to avoid the creation of antennas that cause the major EMI.

The value for capacitors C1, C2 and C3 is 100nF, which is the recommended value for decoupling in the PIC18F4550 Datasheet in base of the current that flows through the pins.

The capacitor C6 is connected between the VUSB pin of the microcontroller and GND, and its value, 470nF, is also recommended in the Datasheet of the PIC.

#### Resistors (R1, R2)

The value for these resistors is calculated in base of the current the PIC can hold up for its input pins.

The maximum value for that current  $I_{MAX}$  is  $25mA^{54}$ , and the voltage 5V:

$$V = IR$$

**Eq.5:** Ohm's Law

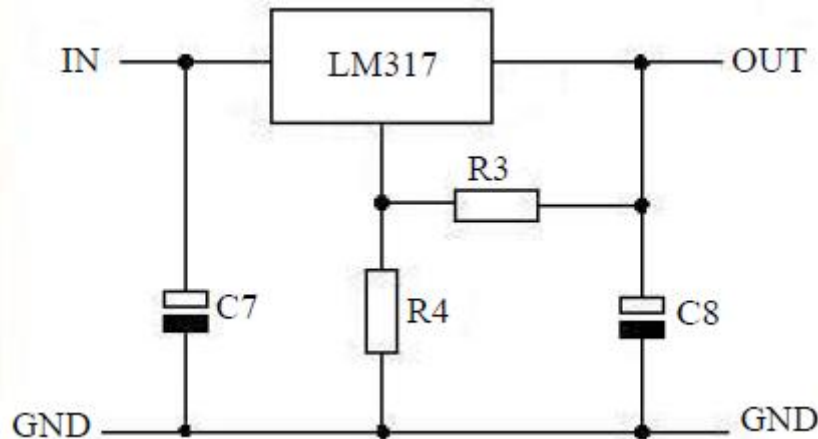
$$R > V/I = 5V/25mA = 200 \Omega$$

**Eq.6:** Calculation of the values for the resistors R1 and R2

Thus,  $R > 200\Omega$ , for facility and availability, the value for the resistors R1 and R2 is  $1K\Omega$  and are high-precision resistors with a tolerance of  $\pm 1\%$ . The case style is axial leaded which is the easiest style for soldering in the PCB.

### Electrolytic Capacitors (C7, C8) and Resistors (R3, R4)

The use of electrolytic capacitors C7 and C8 is required for the correct operation of LM317 Voltage Regulator. The values are recommended in the datasheet; for the capacitor C7 is  $0.1\mu F$  and  $1\mu F$  for the C8.<sup>55</sup>



**Fig.11:** LM317 Voltage Regulator Connections

$$V_{OUT} = 1.25(1 + R3/R4)$$

**Eq.7:** Relation between Voltage and Resistors<sup>56</sup>

From the equation above, the values for the resistors R3 and R4 can be calculated. For a desired  $V_{OUT}$  of 3V, and the typical value for R3 of  $240\Omega$ , R4 must be  $336\Omega$ .

The value for R4 of  $336\Omega$  does not exist; thus, it is necessary to approach the value of the resistor so that the output voltage is never less than 3V. The value chosen is  $348\Omega$ .

## 3.2. Digital Circuit

### 3.2.1. Communication

#### RS232

- + Easy Interface between UART and serial port connector;
- + Can support up to 50 meters of distance between the hardware and the computer;
- Computers rarely have serial ports nowadays;
- 9-pin connector.

## **USB**

- + Fast;
- + 4 pin connector;
- + Every computer has USB input;
- Can support up to 5 metres.

The use of RS232 would make the PCB design much more complex because of its 9-pin connector, and the distance between the computer and the hardware is not going to be greater than 1 metre. Thus, the USB presents more advantages than the RS232 and is implemented on the circuit.

### **3.2.2. Microcontroller choice**

#### **MC9S08QG8**

- + Good immunity to external interferences;
- + SPI Interface;
- + Individual KBI Interruption Configuration;
- 12 I/O;
- No USB Interface.

#### **PIC18F4550**

- + USB Interface
- + SPI Interface
- + 35 I/O;
- Low immunity to external interferences;

#### **PIC18F2550**

- + USB Interface;
- + SPI Interface;
- + 24 I/O;
- Low immunity to external interferences;

#### **COP912C**

- + SPI Interface;
- + Normal immunity to external interferences;
- No USB Interface;
- 16 I/O.

When performing the selection of the microcontroller, it is necessary to look if the microcontroller offers the USB interface due to the fact that the results are going to be displayed on the computer. Also, several connections between the hardware and the microprocessor have to be made, thus, the PIC18F4550 is the most suitable.

### **3.2.3. Controller Voltage supply**

#### **Voltage Regulator**

- + Good line isolation;

- + Good output voltage regulation;
- External power supply needed;
- Need of filtering components (more costs);

### **USB**

- + Very good and stable output voltage regulation (5% tolerance);
- + Low cost;
- + No external power supply needed (the power comes from the computer);
- Only 5V power supply;

The decision of using the USB as controller voltage supply is supported by the fact that it is used for other functionalities in the circuit, and also because the output voltage is more stable and the circuit does not need more than 5V of supply for any of its components.

#### **3.2.4. Further circuit features**

##### **External clock (Y1)**

The external clock is more accurate than the internal oscillator of the microcontroller, and it is necessary because the microcontroller uses USB communication. The value for the external clock (Y1) is 20MHz. From the datasheet of the PIC18F4550, for the correct operation of the external clock, two ceramic capacitors of 22pF (C4 and C5) must be connected in parallel with the crystal.

##### **Switches (I1, SW1, SW2)**

The circuit uses three switches; two press-button switches and one interrupter. The power that supplies the PIC is controlled by the interrupter (I1). Thus, when the USB is connected to the PCB, the PIC does not start running till the user switches it ON.

The two press-button switches, SW1 and SW2, activate the RESET of the FIFO contained in the camera sensor and the initialization of the camera, respectively.

### **3.3. USB Driver**

#### **3.3.1. Windows XP Driver**

A configuration via HID driver has been implemented, that it is preloaded with Windows XP so at the time of connecting the device to the computer, Windows XP will recognize it and associate a virtual com port to it

After that, it is able to communicate like a serial port in a bidirectional way

#### **3.3.2. VIP/PID**

VID (Vendor ID) numbers are managed through the usb.org. The user has two ways to get a VID number from them:

- Become a member of usb.org;

- Buy a VID Number.

However, because a device from Microchip is being used, and it has associated a VID number, the user can use this identification in order to configure his device.

- VID number for Microchip devices is: 0x04D8

PID means Product ID Number and is assigned through the company that has a VID licensed.

Currently, Microchip can assign to the user a PID for free if they agree with its sublicensing terms.<sup>57</sup>

- PID number by default: 0x0000

Take care that we can only use this PID for:

- Internal development purposes;
- During initial production without exceeding 10000 units of our product.

At the time of exceeding that quantity, the sublicense will be revoked and the user will have to get a Vendor ID from USB-IF, Inc (aka usb.org).<sup>58</sup>

### 3.3.3. User Software discussion

At the very beginning of the project, Visual Basic is used to communicate with the PIC because it is the easiest way to code in the computer.

After checking the correct behaviour of the device, all the software of the PIC microcontroller is coded in C language, that is be very useful in order to migrate to other platforms such as Linux and also it is be free of licenses.

## 3.4. Software

### 3.4.1. Languages

#### C<sup>59</sup>

- + Uses simple language core, with significant added functionality, such as mathematical functions and file handling, provided by libraries;
- + Flexible language that allows programming with multiple styles;
- + Uses small set of keywords;
- + Very useful in order to migrate to other platforms;
- The refreshment is can present some difficulties.

#### C++<sup>60</sup>

- + Object Oriented;
- + Variable source code;
- + Fast programming language;
- Very complex syntax.

### **Assembler<sup>61</sup>**

- + Flexibility;
- + Size efficiency in memory;
- Low-level language (greater programming time);
- Lack of portability (each machine has its own language).

The decision of programming the software in C language is due to the large use and acceptance of this language in the area of programming microcontrollers. C is very easy to implement and very fast

### **3.4.2. Hardware simulation**

#### **Proteus**

- + PIC's simulation in almost real time;
- + Able to read files in assembly code for PIC, AVR, 8051, HC11, and ARM/LPC200 microprocessor's families, and simulate perfectly their behaviour;
- + It incorporates a library with more than 6,000 models of digital and analogue devices;
- + Composed by the ISIS module, used for simulation and debugging of digital circuits, and the ARES module, used for developing PCBs;
- + Very good integration with CCS C Compiler;
- Not as simple and intuitive as other simulators.

#### **Multisim**

- + Very easy to manage;
- + Good simulation of analogue and digital electronic circuits;
- + Very good integration with Ultiboard (PCB design software);
- + It incorporates a library with a large number of components;
- Small range library of microcontrollers.

#### **MPLAB**

- + Uses assembly language code for Microchip PIC;
- + Composed by text editor, compiler and simulator;
- No real time simulation.

#### **Circuit Marker 2000**

- + Very easy to use;
- + Voltages and currents can be measured;
- Microcontrollers cannot be simulated;
- No related to any PCB design software.

Due to the reason that a microcontroller from the PIC family is used in this project, the easiest way to simulate the circuit is using "PROTEUS". This program can be easily managed by its libraries and the main advantage is the simulation of PICs, which can be run in real time.

### 3.4.3. Software simulation

#### CCS C Compiler<sup>62</sup>

- + Uses easy and manageable high level language;
- + Includes a powerful C editor;
- + Automatic Documentation Generator;
- Generates very large HEX files (no optimization of code).

#### CodeWarrior

- + Very powerful compiler;
- + Supports many architectures, offers a wide range of options;
- + Easy to program and detect errors;
- Presents some difficulties when used.

The CCS C Compiler is chosen not only because of the election of PROTEUS as simulator and the excellent integration that exists between them, but also because of the facilities it offers for programming PIC microcontrollers.

To implement the digital image processing software, the Visual Basic program is used. It is a graphical environment to develop applications for the Microsoft Windows operating system. It is very easy to use, and the resulting project is a executable file that can be implemented in all the computers that use Microsoft Windows operating system without installing any further program.

### 3.5. Software

The program actions are controlled and defined by different interrupts on the PIC microcontroller. In Fig.13 and Fig.14 the flowchart of the software is shown.

First of all, the variables, USB Interface and the sensor are initialized in the main program when switching on the PIC microcontroller. The program will not start running until the user presses the START button, which corresponds to the routine TAKE\_PICTURE(). Now, the PIC enables and initializes the FIFO memory through the routine INITIALIZE\_FIFO(), and the data coming from the sensor starts to be written synchronously in the registers of the memory. When the complete frame is stored, the sensor sends a pulse through the pin VSYNC, which is detected through and interruption by the PIC.

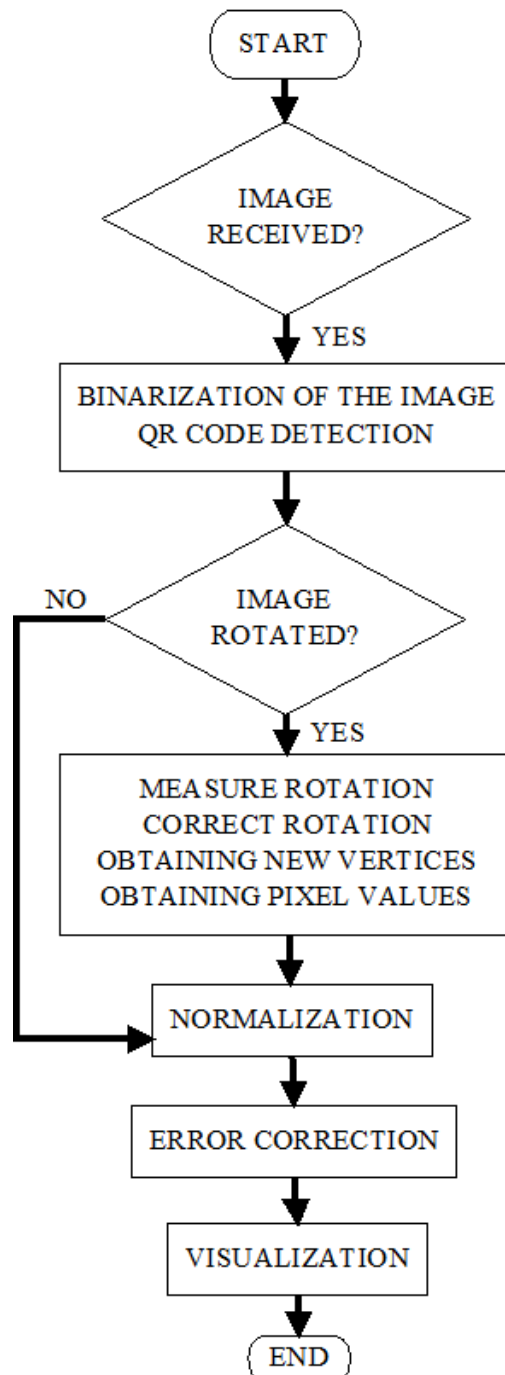
With this pulse, the sensor stops writing on the FIFO and the pointer of the memory is reset (RRST). Here, the routine stops and the timer routine starts to run.

In this routine, the PIC microcontroller starts reading from the FIFO, transforming the frame; which is on YCbCr format, to greyscale and sending the pixels through the USB interface to the computer, where they are stored byte by byte.

When the complete frame is received on the computer, the first function of the QR decoder program saves the picture as a \*.bmp file. After this action, the second function

reads the picture saved previously, and decodes the image with the help of a library which contains the algorithm for decoding QR codes. Finished this step, the results are shown on the computer screen.

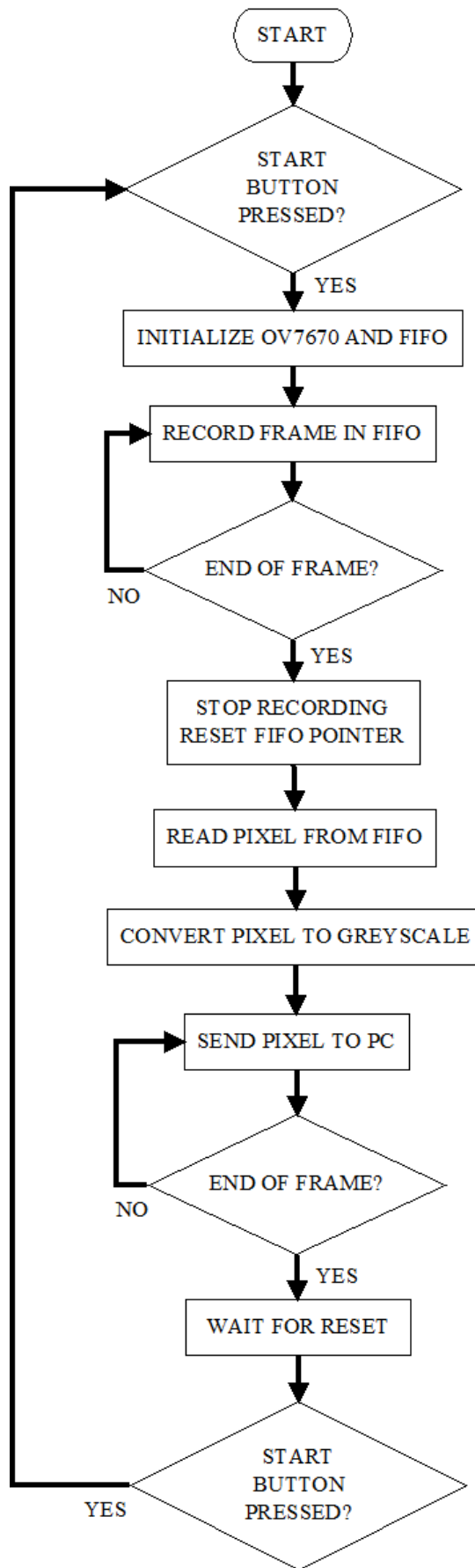
At this point, the RESET button can be pressed to reset the FIFO memory, and another picture can be taken and analyzed by pressing the START button.



**Fig.12:** Digital Image Processing Flowchart

The description of the digital image processing is explained in detail in the Research chapter, section 2.7 Digital Image Processing.





**Fig.13:** Main Software Flowchart

## 4. Implementation

### 4.1. Hardware

The program used for the design of the PCB is “Protel Design Explorer 2002”. It offers a wide range of PCB and Schematic libraries which contains several components that make easier the design of both Schematics and PCB designs.

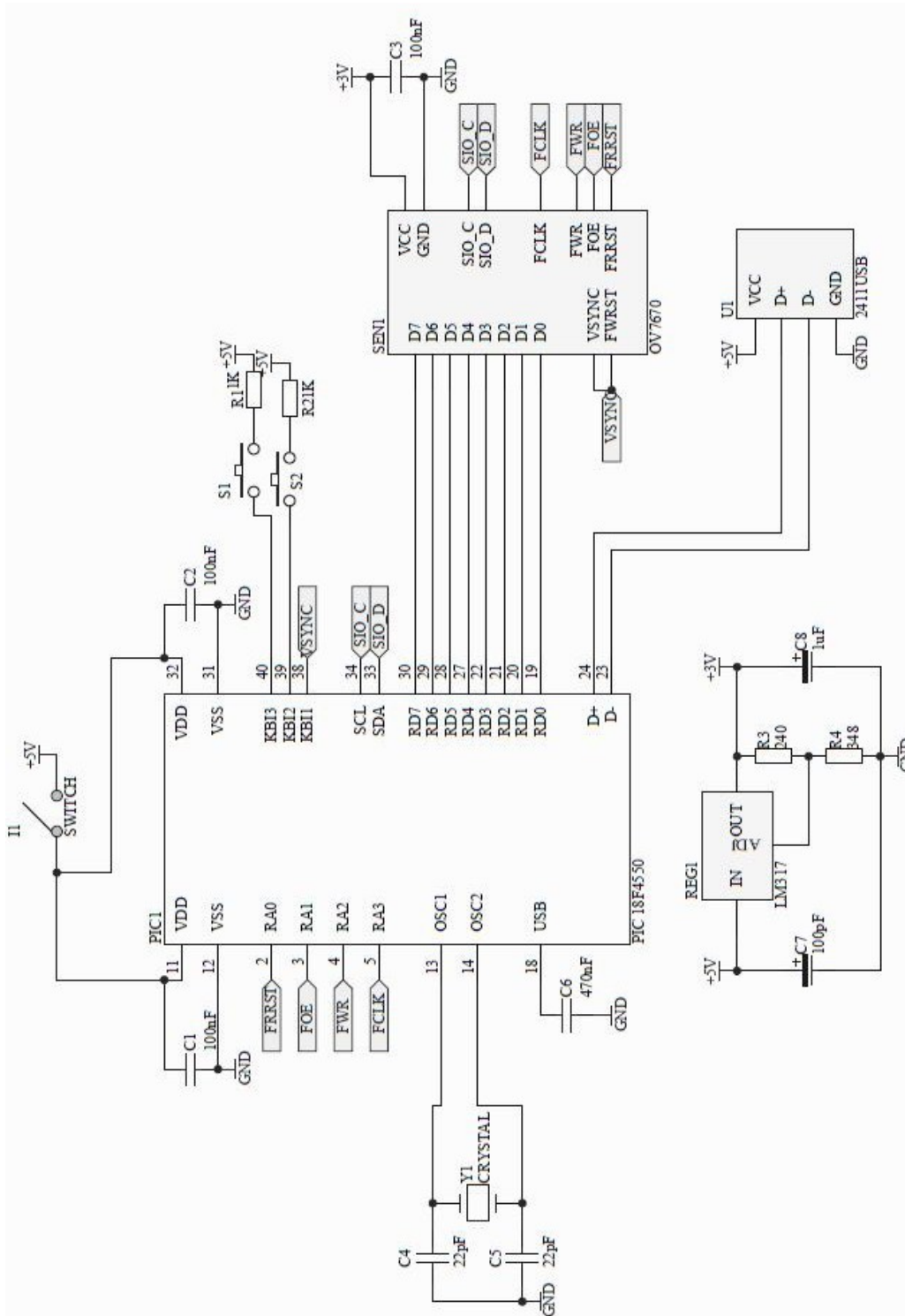


Fig.14: Schematic Circuit

**4.1.1. Parts list**

<b>Design.</b>	<b>Description</b>	<b>Manufacturer</b>	<b>Order Code</b>	<b>Distrib.</b>	<b>Price/u (£)</b>
<b>C1</b>	100nF Radial Ceramic Capacitor	MULTICOMP	1600820	FARNELL	0.017
<b>C2</b>	100nF Radial Ceramic Capacitor	MULTICOMP	1600820	FARNELL	0.017
<b>C3</b>	100nF Radial Ceramic Capacitor	MULTICOMP	1600820	FARNELL	0.017
<b>C4</b>	22pF Radial Ceramic Capacitor	MULTICOMP	1600966	FARNELL	0.015
<b>C5</b>	22pF Radial Ceramic Capacitor	MULTICOMP	1600966	FARNELL	0.015
<b>C6</b>	470nF Radial Ceramic Capacitor	KEMET	1582428	FARNELL	0.019
<b>C7</b>	0.1uF Radial Electrolytic Capacitor	PANASONIC	1973500	FARNELL	0.23
<b>C8</b>	1uF Radial Electrolytic Capacitor	PANASONIC	2282218	FARNELL	0.052
<b>I1</b>	Slide Switch	MULTICOMP	1550246	FARNELL	0.27
<b>PIC1</b>	PIC18F4550 Microcontroller 40-pin DIP	MICROCHIP	9321357	FARNELL	5.19
<b>PIC1</b>	40-Pin DIP Socket	TE CONNECTIVITY	1101352	FARNELL	0.33
<b>R1</b>	1K $\Omega$ Axial Resistor 1%	MULTICOMP	1127016	FARNELL	0.008
<b>R2</b>	1K $\Omega$ Axial Resistor 1%	MULTICOMP	1127016	FARNELL	0.008
<b>R3</b>	240 $\Omega$ Axial Resistor 1%	MULTICOMP	1126959	FARNELL	0.008
<b>R4</b>	348 $\Omega$ Axial Resistor 1%	VISHAY	5812306	FARNELL	0.02
<b>REG1</b>	LM317 3-Terminal Positive Adjustable Regulator	FAIRCHILD	2102581	FARNELL	0.69
<b>S1</b>	Push-button Switch	TE CONNECTIVITY	1555985	FARNELL	0.107
<b>S2</b>	Push-button Switch	TE CONNECTIVITY	1555985	FARNELL	0.107
<b>SEN1</b>	OV7670 VGA with HIGH Quality Lens + AL442B FIFO	CAMERACHIP	OV7670	AMAZON	13.99
<b>SEN1</b>	24-Pin, 2.54mm Latch Header	AMPHENOL	2215267	FARNELL	1.33
<b>U1</b>	USB-2411 02 Connector	LUMBERG	1177885	FARNELL	0.85
<b>Y1</b>	20MHz Crystal Pin Layout HC49	TXC	1842257	FARNELL	0.32
<b>LEAD</b>	Lead, 2.54mm, 150mm, 24-way	AMPHENOL	2217613	FARNELL	1.97
<b>TOTAL (£)</b>					<b>24.25</b>

**Table 7:** Parts list.

#### **4.1.2. PCB Manufacturing, drilling and soldering**

##### **Manufacturing Process**

The manufacturing process can be divided into 4 steps; circuit printing, transference to copper, removal of exceeding copper and tinning bath. The process is explained below.

After having printed the design of the tracks on onionskin paper, it is needed to cut the shape of the PCB, being sure that the PCB is photo-sensible and double-side coppered; there are tracks in both sides.

The first step to take is to place correctly the PCB between the top layer and bottom layer which are printed on the onionskin papers, making the pads on the top layer correspond to those on the bottom layer.

Here, the PCB is placed on the Isolation Plate, where the designed tracks are transferred to the copper. To reveal the tracks, the PCB must be introduced into the developing tank till it is seen that the tracks are correctly defined. This step corresponds to the transference to copper.

Before continuing with the removal of exceeding copper, the PCB must be washed and dried carefully. Now, it can be introduced on the acid tank. This step is the longest one; the acid slowly attacks the copper and removes it, leaving copper only on the tracks. This operation is not hazardous, but it should take place in a ventilated area to avoid the breathing vapours of the acid, and, of course, to avoid splashing even if this acid does not irritate the skin.

Similarly to the previous step, the board must be washed and dried properly. Once dried, the steel wood is used to remove all trace of remaining stoner.

Finally, the board is subjected to a tinning bath, where the tracks and pads are covered with an anti-corrosion coating. As in the previous steps, the board is washed and dried.

##### **Drilling Process**

To drill the board, it is needed to make sure that the holes are centred on the pads of the PCB, and their diameter is adequate to the terminals of the components that are used. Larger or smaller holes will prevent the desired final result.

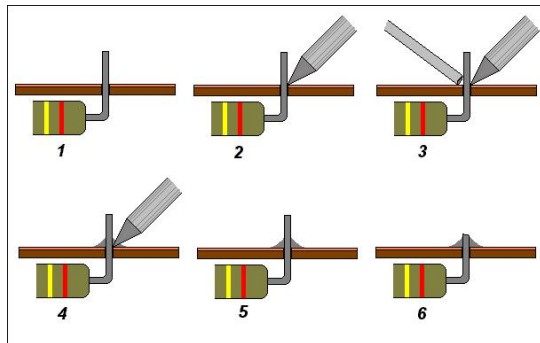
##### **Soldering Process**

During this process it is needed a metal with a low melting point, thus tin is used. With the help of a welder, the pad is heated, and the tin melts joining the elements that are in contact with it. Here, the steps that must be followed for a correct soldering.

- The elements to be soldered (pad and component) must be clean;
- Also the soldering tip must be clean and slightly tinned. To clean a soldering tip all that is needed is a damp sponge.

- Once the welder is heated, it is applied for a few seconds to the welding area, that is, the pad of the PCB, and simultaneously to the terminal component to be welded, and thereupon tin is applied.
- When the tin has been applied, it is needed to keep the welder a couple of seconds to distribute correctly the tin.
- The welder is removed and the union is let cool naturally.
- Finally, with the help of pliers, the exceeding terminal of the piece is cut.

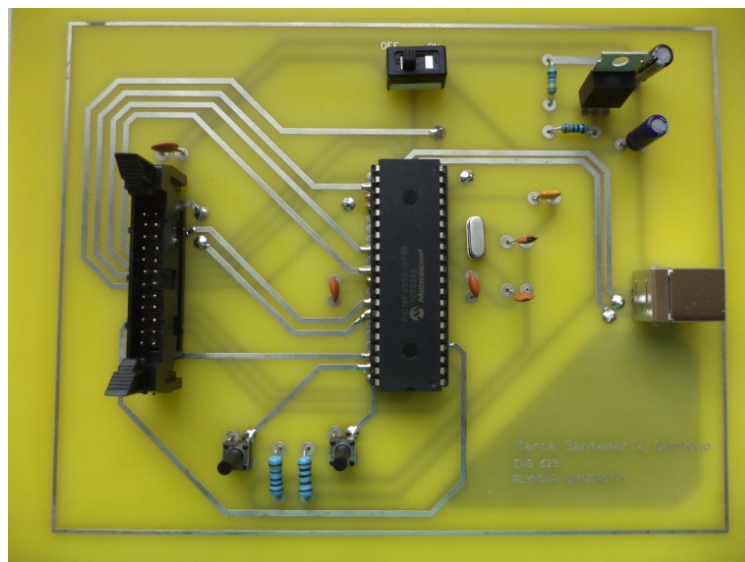
Below, a graph of the soldering process:



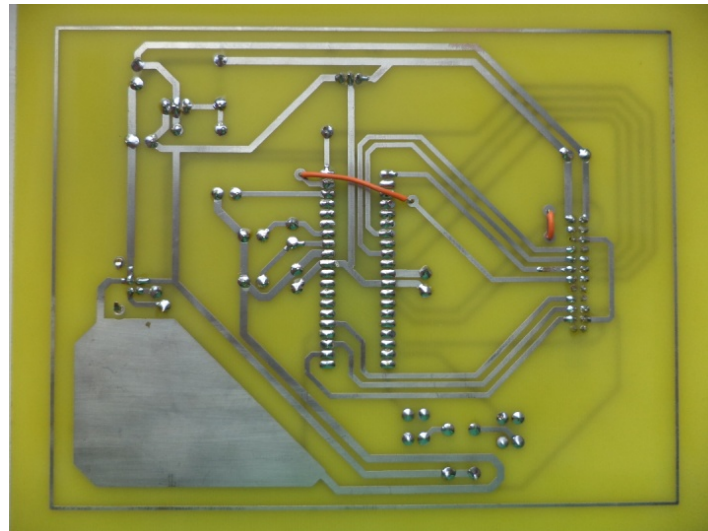
**Fig.15:** Soldering process

### Results

Some pictures of the final PCB after the manufacturing, drilling and soldering processes.



**Fig.16:** PCB Top Layer



**Fig.17:** PCB Bottom Layer

Due to the impossibility of routing some tracks during the design process with Proteus, they were wired with cables during the soldering process.

## 4.2. Software

### 4.2.1. PIC Programmer Board VM134



**Fig.18:** PIC Programmer Board VM134

This card allows programming a wide range of Microchip PIC microcontrollers. The communication between the programmer board and the PC is through RS232 and it is needed to install the program from the CD before the use for the recognition of the device. Following the instructions of the quick guide offered by Velleman, it is very easy to implement the program created in the CCS C Compiler on the PIC microcontroller.

Below, a picture during the programming of the PIC is shown:

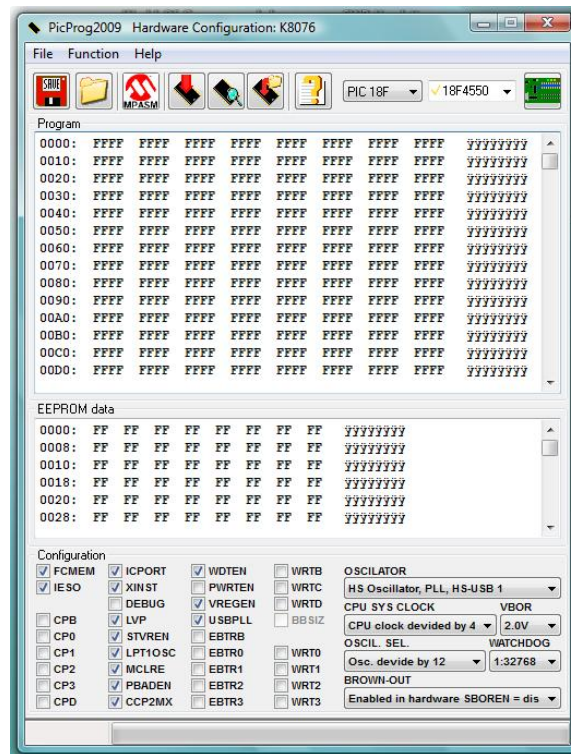


Fig.19: Hardware Configuration of VM134

#### 4.2.2. ISIS. Proteus Simulator

The PCB is also designed on the ISIS program for simulating the operation of the PIC, switches and USB communication. The libraries of the Proteus do not offer the possibility of simulating an image sensor, thus, some switches and LED diodes are used on its place.

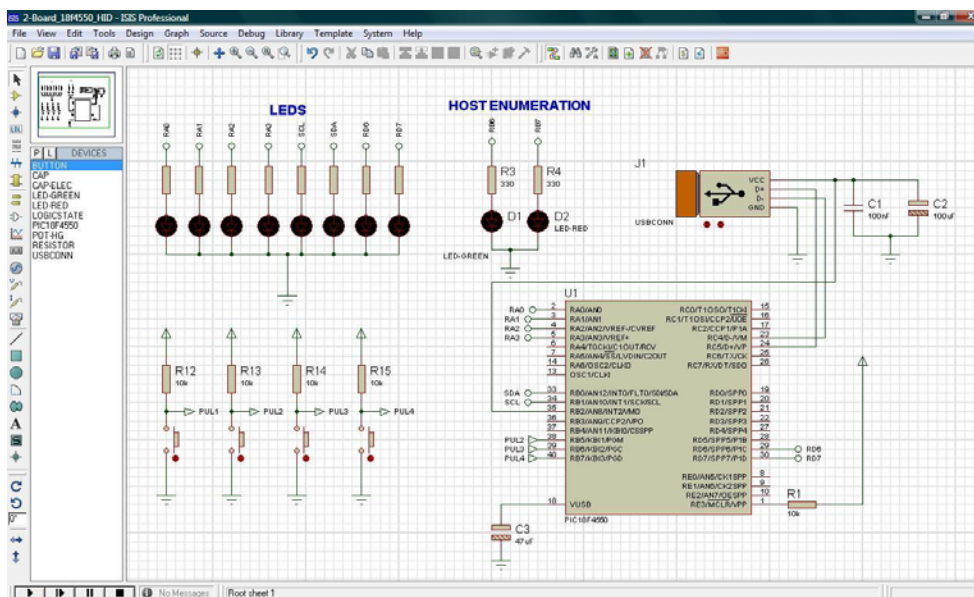


Fig.20: Design on the circuit on ISIS (Proteus simulator program)



## 5. Testing

The testing chapter is divided into two parts; the simulation of the hardware with the PROTEUS simulation program and the CCS C Compiler, and the testing done in the laboratory.

### 5.1. Simulation

The way the CCS C Compiler is integrated with the Proteus Simulation Program helps to understand how the circuit operates with the compiled code.

Before testing the PCB, the circuit is simulated on ISIS (Proteus Simulating Program) to check the behaviour of the circuit and make the appropriate changes in the code, avoiding unplugging the PIC from its socket and reprogramming it every time the code needs an arrangement.

The simulation process is divided into several parts. Each code function is simulated individually to check if the programming code is correct. The simulations are described below.

#### 5.1.1. USB Interface

The USB Interface configuration is programmed and compiled with the CCS Compiler, and then simulated with the ISIS. The PC can detect the virtual port and recognises it as a HID (Human Interface Device).

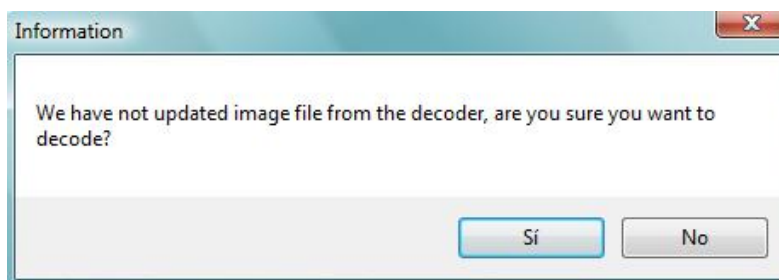
#### 5.1.2. Interrupts

The project uses two kinds of interruptions; interruptions-on-change, which are activated when a change in the input of a pin has been detected, and timer interruptions, which are executed automatically every defined “x” time.

Both interrupts are configured so that, when they occur and are executed correctly, LED diodes light. The interrupt-on-change is connected to a switch, when it is activated, the corresponding LED lights. The timer interrupt is defined such that every 2 seconds a LED diode lights.

#### 5.1.3. Image decoder

To test if the image decoder is working, the software is modified. A frame containing a QR code is stored in the (C:) memory. From the executable, the button “Decode” must be clicked, a warning message appears as in Fig.



**Fig.21:** Warning message of the executable



If the “YES” button is clicked, the decoder takes the image stored in (C:), and analyses and decodes it. The resulting box is shown in Fig22. If the “NO” button is clicked, the program will wait until it receives a frame through the USB.



Fig.22: Box showing the decoded value of the QR code

## 5.2. Testing

Below there are some tests that were done to verify the correct operation of the different parts of the project.

### 5.2.1. PCB

The tests that have been done to the PCB are:

- i. Check the wiring.

**Result:** PASS

With the multimeter all the tracks and pins are looked for continuity.

- ii. Applying the voltage supply to the PCB through the USB cable, measure the output voltage from the USB (5V) and from the voltage regulator (3V).

**Result:** PASS

USB Output Voltage: 5.12V

Voltage Output Regulator: 3.079V

- iii. Measure Crystal Frequency.

**Result:** PASS

Frequency: ~20MHz

### 5.2.2. USB Interface

To test the correct operation of the USB interface, a code with just the configuration of the USB interface is created and programmed in the PIC microcontroller. The test which has been done is the following:

- i. Connect the PCB to the computer and check if the PC recognises the device.

**Results:** PASS

### 5.2.3. SCCB Interface

The tests that have been done to check the correct behaviour of the SCCB Interface have been:

- i. Prove the SCCB communication

**Result:** PASS

The OV7670 answers with an ACK after it has been address properly.

- ii. Modification of registers

**Result:** PASS

First, the register is read, the desired bits are modified, and then it is written back to the OV7670.

- iii. The 7 bit SCCB/I2C address is 0x21; this translates to 0x42 for write address and 0x43 for read address.

**Result:** PASS

- iv. Debug

**Result:** PASS

For debugging purposes, some registers are read and checked, they contain their default values, i.e. the reading of the register 0x01 returns the value 0x80.

### 5.2.4. Digital Image Sensor

The Image Digital sensor has been subjected to several tests.

- i. Check the wiring, pin configuration and clock configuration

**Result:** PASS

- ii. Grab a frame

**Result:** PASS

Only one frame is grabbed during this test. This operation starts from VSYNC falling edge and finishes with VSYNC rising edge. This procedure is repeated multiple times and it is confirmed that the number of bytes per snapshot is constant.

- iii. Read data from the output of the FIFO

The camera lens is covered, and it is verified that the snapshot have the following information (in bytes): 128 0 128 0 128 0 128 0 ... i.e. every even byte is 128 and every odd byte 0. This corresponds to a pitch black image.

**Result:** PASS

- iv. The electrical/software part is working

**Result:** PASS

- v. The grabbed snapshot is visualized in a PC

The luminance (Y) channel is used to visualize the frame on the computer, i.e. only the even bytes of the snapshot are used.

**Result:** FAIL

The image cannot be visualized on the PC, so we cannot check if the focus is well regulated.

- vi. Configure camera focus

The regulation of the distance from the camera lens to the image sensor is regulated by trial and error by varying the camera focus until a clear image results.

**Result:** FAIL

The image cannot be visualized on the PC, so we cannot check if the focus is well regulated.

## 6. Conclusion

The aim of this project was to develop a device capable of detecting and analysing QR codes. This aim has been reached; a device has been manufactured capable of taking pictures, store them and analyze and decode QR codes.

At the beginning of the design process, the MT9V131C12STC CMOS sensor was going to be included in the project instead of the OV7670. Some difficulties appeared with its implementation; a socket to adapt 48-Pin CLCC package to DIP package could not be found, and the soldering of this chip to the PCB without the socket was difficult to do. Thus, another sensor with similar characteristics was found; the MT9V011 CMOS sensor. The difference between this sensor and the previous is the package pin; 28-Pin CLCC. There exists sockets to adapt this package to DIP package, but a problem with the optical lens aroused; no optical lens were found. Finally, the OV7670 with FIFO included solved the problems above, so the decision of implementing the OV7670 VGA sensor was taken.

Due to the tests and simulations that the device has been subjected to and the results obtained, the following conclusions have been reached:

From the PCB test, all the tracks and connections of the board are well wired and the voltage regulator operates correctly.

From the tests done to the SCCB Interface, there is no problem of communication between the digital image sensor and the microcontroller. The SCCB interface is well defined and configured on the code.

From the test of USB interface, the USB communication fails. It is not possible to establish the communication between the device and the PC. However, in the simulation of the USB communication, there are no problems of communication between the device and the computer; the PC can recognise the virtual device as a HID device. The USB interface is well defined and configured on the code.

All the connections of the circuit are revised, and the design of the hardware results to be correct. Thus, it is concluded that there failure is on the hardware manufacturing process.

The distributor of the components is FARNELL, the time deliberation of the package is around 2~3 days. The more expensive components are the OV7670 VGA sensor; which costs £13.99, and the PIC18F4550 microcontroller; which costs £5.17. The total cost of the project is £24.25.

The aim of manufacturing a device capable of decoding QR codes which a cost less than £25 is reached.

The recommendations to improve this project are the following. If the user wants the digital image processing to be executed by the microcontroller, it must have high level specifications and features. Many data is processed at a very high speed, and the

normal microcontrollers cannot process and store all the data that is required for the decoding.

It is also recommended to use the OV7670 VGA Digital Image Sensor which does not include de FIFO memory. It is very difficult to work with a FIFO memory if the microcontroller has to read and write in specific registers or address in it. Thus, an external memory is needed to support the microcontroller's memory. An example of suitable memory for this project could be the CY7C1049DV33 Static RAM.

## References

---

- <sup>1</sup> <http://www.photography.imageidentity.info/2012/02/history-of-barcode/>
- <sup>2</sup> [http://www.gs1au.org/assets/documents/info/technical/tfs22\\_barcode\\_symbol\\_characteristics.pdf](http://www.gs1au.org/assets/documents/info/technical/tfs22_barcode_symbol_characteristics.pdf)
- <sup>3</sup> <http://mdn.morovia.com/manuals/bax3/shared.bartech.php#Symbology.Codabar>
- <sup>4</sup> <http://grandzebu.net/informatique/codbar-en/code128.htm>
- <sup>5</sup> AIDC Memoirs, David C. ALLAIS, PathGuide Technologies, Inc
- <sup>6</sup> AIDC Memoirs, David C. ALLAIS, PathGuide Technologies, Inc
- <sup>7</sup> <http://www.barcodeman.com/info/codabar.php>
- <sup>8</sup> <http://archive.is/20120915/http://www.qrcode.com/en/aboutqr.html>
- <sup>9</sup> <http://www.morovia.com/education/symbology/pdf417.asp>
- <sup>10</sup> [http://www.gs1.org/docs/barcodes/GS1\\_DataMatrix\\_Introduction\\_and\\_technical\\_overview.pdf](http://www.gs1.org/docs/barcodes/GS1_DataMatrix_Introduction_and_technical_overview.pdf)
- <sup>11</sup> <http://www.qrcodegeneratordownload.net/three-thing.html>
- <sup>12</sup> [http://www.atsi.cc/barcode.htm?&lang=en\\_us&output=json&session-id=19c293e8593ab3a67851da563f5dc89f](http://www.atsi.cc/barcode.htm?&lang=en_us&output=json&session-id=19c293e8593ab3a67851da563f5dc89f)
- <sup>13</sup> [http://www.gunpos.com.au/articles.asp?id=161&lang=en\\_us&output=json&session-id=19c293e8593ab3a67851da563f5dc89f](http://www.gunpos.com.au/articles.asp?id=161&lang=en_us&output=json&session-id=19c293e8593ab3a67851da563f5dc89f)
- <sup>14</sup> [www.barcoding.com/barcode/barcode-printers.shtml](http://www.barcoding.com/barcode/barcode-printers.shtml)
- <sup>15</sup> [www.barcode.ro/tutorials/barcodes/laser-scanners.html](http://www.barcode.ro/tutorials/barcodes/laser-scanners.html)
- <sup>16</sup> <http://www.printronix.com/products/overview.aspx?id=1116>
- <sup>17</sup> Static RAM CY7C1049DV33 Datasheet. CYPRESS PERFORM.
- <sup>18</sup> Static RAM CY62136EV30LL Datasheet. CYPRESS PERFORM.
- <sup>19</sup> AL422B Datasheet. AVERLOGIC.
- <sup>20</sup> A43L0616BV 512K X 16 Bit X 2 Banks Synchronous DRAM Datasheet. AMIC.
- <sup>21</sup> MT9V131: 1/4-Inch SOC VGA CMOS Digital Image Sensor Datasheet. MICRO.
- <sup>22</sup> MT9V011: 1/4-Inch VGA Digital Image Sensor. APTINA.
- <sup>23</sup> OV7670 CMOS VGA CAMERACHIP with OmniPixel Technology Datasheet. OMNIVISION.
- <sup>24</sup> <http://www.amazon.com/OV7670-Camera-Module-640X480-Compatible/dp/B009SIZG86>

- <sup>25</sup> [http://www.amazon.com/sunkee-OV7670-640X480-Compatible-Interface/dp/B00AZWVZKW/ref=sr\\_1\\_fkmr0\\_2?s=electronics&ie=UTF8&qid=1367324606&sr=1-2-fkmr0&keywords=OV7670+%2B+al442](http://www.amazon.com/sunkee-OV7670-640X480-Compatible-Interface/dp/B00AZWVZKW/ref=sr_1_fkmr0_2?s=electronics&ie=UTF8&qid=1367324606&sr=1-2-fkmr0&keywords=OV7670+%2B+al442)
- <sup>26</sup> MC9S08QG8 Datasheet. FREESCALE SEMICONDUCTOR.
- <sup>27</sup> PIC18F4550 Datasheet. MICROCHIP.
- <sup>28</sup> PIC18F2550 Datasheet. MICROCHIP.
- <sup>29</sup> COP912C Datasheet. TEXAS INSTRUMENT.
- <sup>30</sup> 7.3.2 Bus Timing/Electrical Characteristics. Universal Serial Bus Specificaiton. USB.org
- <sup>31</sup> [http://www.analog.com/en/content/ta\\_fundamentals\\_of\\_voltage\\_regulators/fca.html](http://www.analog.com/en/content/ta_fundamentals_of_voltage_regulators/fca.html)
- <sup>32</sup> The Development of the C Language, Dennis M. Ritchie, 2003.
- <sup>33</sup> *The C++ Programming Language* (Third edition), Stroustrup, Bjarne (1997) ISBN 0201889544.
- <sup>34</sup> Assemblers and Loaders, David Salomon (1993).
- <sup>35</sup> Structured BASIC programming, J.G.; T.E. KURTZ, KEMENY (1986). New York, USA: John Wiley & Sons. ISBN 0-471-81087-8.
- <sup>36</sup> <http://www.labcenter.com/index.cfm>
- <sup>37</sup> <http://www.ni.com/multisim/>
- <sup>38</sup> <http://www.microchip.com/pagehandler/en-us/family/mplabx/>
- <sup>39</sup> [http://users.wfu.edu/matthews/courses/p230/CM6/cm\\_usermanual.pdf](http://users.wfu.edu/matthews/courses/p230/CM6/cm_usermanual.pdf)
- <sup>40</sup> [http://www.ccsinfo.com/content.php?page=ide\\_advantages](http://www.ccsinfo.com/content.php?page=ide_advantages)
- <sup>41</sup> <http://www.docstoc.com/docs/5932247/Getting-Started-with-CodeWarrior-IDE-from-Freescale>
- <sup>42</sup> <http://visualbasic.about.com/>
- <sup>43</sup> <http://www.arcelect.com/rs232.htm>
- <sup>44</sup> [http://www.usb.org/developers/devclass\\_docs/usbcdc11.pdf](http://www.usb.org/developers/devclass_docs/usbcdc11.pdf)
- <sup>45</sup> <http://www.techterms.com/definition/grayscale>
- <sup>46</sup> Pages 234-236, Digital Video and HDTV: Algorithms and Interfaces, Charles A. Poynton, Morgan Kaufmann, 2003. ISBN 1-55860-792-7.
- <sup>47</sup> Pages 161-163, Principles of Television Reception, W. Wharton & D. Howorth, Pitman Publishing, 1971.
- <sup>48</sup> <http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html>
- <sup>49</sup> [http://qrbcn.com/imatgesbloc/Three\\_QR\\_Code.pdf](http://qrbcn.com/imatgesbloc/Three_QR_Code.pdf)

<sup>50</sup> [http://qrbcn.com/imatgesbloc/Three\\_QR\\_Code.pdf](http://qrbcn.com/imatgesbloc/Three_QR_Code.pdf)

<sup>51</sup> [http://qrbcn.com/imatgesbloc/Three\\_QR\\_Code.pdf](http://qrbcn.com/imatgesbloc/Three_QR_Code.pdf)

<sup>52</sup> <http://www.usbmadesimple.co.uk/index.html>

<sup>53</sup> OV7670 CMOS VGA CAMERACHIP with OmniPixel Technology Datasheet.  
OMNIVISION.

<sup>54</sup> Page 367, PIC18F4550 Datasheet, DS39632E, MICROCHIP.

<sup>55</sup> Page 5, LM317: 3-Terminal Positive Adjustable Regulator Datasheet, March 2013,  
FAIRCHILD SEMICONDUCTORS

<sup>56</sup> Page 5, LM317: 3-Terminal Positive Adjustable Regulator Datasheet, March 2013,  
FAIRCHILD SEMICONDUCTORS

<sup>57</sup> <http://ww1.microchip.com/downloads/en/DeviceDoc/APPLICATION%20FOR%20SUBLICENSE%20TO%20USB%20VID%20revised%2012110.pdf>

<sup>58</sup> <http://ww1.microchip.com/downloads/en/DeviceDoc/APPLICATION%20FOR%20SUBLICENSE%20TO%20USB%20VID%20revised%2012110.pdf>

<sup>59</sup> The Development of the C Language, Dennis M. Ritchie, 2003.

<sup>60</sup> *The C++ Programming Language* (Third edition), Stroustrup, Bjarne (1997)  
ISBN 0201889544.

<sup>61</sup> Assemblers and Loaders, David Salomon (1993).

<sup>62</sup> [http://www.ccsinfo.com/content.php?page=ide\\_advantages](http://www.ccsinfo.com/content.php?page=ide_advantages)