CRYPTO-GLASSES
Final Year Project Report
Fall-Spring 2006

Hajj Ali Samer 200500396
Itani Ibrahim 200201832
Assaf Bassam 200503272

Supervisor: Dr Ayman Kayssi
6.0 System Design Constraints .......................................................................................... lxxv
6.1 Economic constraints ............................................................................................... lxxv
6.2 Environmental constraints .................................................................................... lxxv
6.3 Social constraints ................................................................................................... lxxvi
6.4 Political constraints ............................................................................................... lxxvi
6.5 Ethical constraints .................................................................................................. lxxvi
6.6 Health and Safety constraints .............................................................................. lxxvi
6.7 Manufacturability constraints .............................................................................. lxxvii
6.8 Sustainability constraints ..................................................................................... lxxvii
7.0 Conclusion ............................................................................................................. lxxviii
References ................................................................................................................... lxxix
Appendices .................................................................................................................. lxxx
List of figures

Figure 2.1  Block diagram for the second approach ........................................ 12
Figure 2.2  Vigenere cipher matrix ................................................................. 15
Figure 2.3  Logitech QuickCam Pro 4000 ...................................................... 19
Figure 2.4  iREZ K2 camera .......................................................................... 20
Figure 3.1  Primary overall diagram ............................................................... 31
Figure 3.2  uCidimm ColdFire 5272 Architecture .......................................... 36
Figure 3.3  Final design diagram ................................................................. 40
Figure 4.1  Prototype diagram ...................................................................... 57
Figure 5.1  Component – System interaction ............................................... 59
Figure 5.2  Blocks dependencies diagram ..................................................... 63
Acknowledgments

We are very grateful to our supervisor Dr. Ayman Kayssi, who has supported and guided us with encouragement during the whole project.

We would also like to express our sincere thanks to the FEA faculty members, who have provided us with the support needed, and especially Mr. Khaled Joujou.
1.0 Introduction

1.1 Definition of the project problem

Protecting private and confidential data in a very effective way, not subject to any possible violation, is one of the crucial problems in our days. However, with the widespread of ways (in software and hardware) that permit the interception of such important data, the protection of this data is becoming more and more difficult.

That’s why it was important to come up with a new way of protecting important data. This new way, as we thought in our team, must be in great part independent of the conventional technological means. We mean by this that the revealing of such important data, after its secure transmission from an end user to another end user, must not occur on a medium subject to intrusion or violation such that a computer screen.

So, in other terms, if we want to reformulate the project problem, it will go as follows: Assume an end user wants to securely transmit confidential information to another end user somewhere else, where the possibility of hand to hand delivery is very difficult to achieve, what would he do? As everyone would think, the first logical step to protect this data before transmission is to submit it to some form of encryption that would fake its appearance and make it difficult to read. The second step is to transmit the data over a medium that we think is somehow secure and not susceptible to violations (even if the transmitted data is already encrypted). However, the third step is the most difficult one to achieve: it is the process of recovering the original, unencrypted data. How would we do that without possible violations? If we just decrypt the data on our personal computer and read it, we would endanger our data because someone else may be watching us! Can we print the data on paper and start the decryption process by hand? Difficult! That’s why we thought we should come up with a new way that would decrypt the transmitted data without any possible exterior intervention. This decryption process must be only dependant on the end user himself.
1.2 Primary suggested solution

A suggested primary solution for the above stated problem is to build a pair of eyeglasses that would guarantee a safe reading of confidential data. It works as follows:

- Input:
  1. Data in encrypted form.
  2. Key for decrypting the data.

- Output:
  Data in original form (before encryption), but ONLY the user can see this data.

In other terms, imagine someone sent you a confidential, encrypted email via the internet. The role of the eyeglasses is to capture this important email, decrypt it and display it to the end user only. But, how can this be done? This is one of the problems we will deal with in the design part of the project.

1.3 Project benefits and applications

The most important benefit from this project is to permit both a secure transmission and decryption of important data, without the possibility of exterior interference or interception.

This project has two main applications:

- Transmission and decryption of confidential data using a computer. An application could be the transmission of such data via the email from one person to another and then doing the decryption process by a process that links the CRYPTO-GLASSES and the computer.

- Transmission and decryption of confidential data using paper. An application could be the transmission of such data via the post office from
one person to another and then doing the decryption process by a process that links the CRYPTO-GLASSES to the papers.

1.4 The work already done

The problem of securely transmitting and decrypting confidential data from one end user to another end user is a historical problem. Over the decades, the concentration was especially on the encryption/decryption process as a means for securing the transmission of data. So many new algorithms were conceived making it more and more difficult, and even impossible, to break them and steal the confidential data.

However, one project called “Your eyes only screen” is somehow similar to our own project, but it is approached in a very different manner (as showed in the quotations below):

“Stop people peeping at what you are looking at on the screen with the following simple device. The screen displays random noise, apart from the precise area (maybe 1 sq cm) that you are looking at. This can easily be determined using Body Position Garment and Glance Direction Specs (see links below). The view area displays what is really meant to be on the screen at that position. You will see a completely normal display, but a casual observer will see only noise…

(The principle of this idea, i.e. displaying only the viewed area and changing this in response to the viewer’s eye movement, is not original. It is described by Daniel Dennet in Consciousness Explained (I think), from a psychological perspective. He reports that it really works. I believe my application to be original, however others may know better.” (Halfbakery, 2000).

Our project has some advantages over this previous solution:
• It permits a bigger view of the screen or the paper at one time, so the user could see the entire document (or parts of it bigger than 1 squared cm) at one shot.

• It preserves the secrecy of the documents. In fact, with the solution quoted above, no security measures are taken (maybe this is not the purpose of the project), so the documents can easily be stolen.

Additional research showed that the idea of our project was simply suggested as a better solution than the above quoted one, without being implemented or developed at the end of the year 2004:

What if the screen was encrypted and the glasses you wore un-encrypted it? The glasses would have to have a code you could key into one of its arms to trigger the decryption program (which would match the code you keyed in for the monitor). (Halfbakery, 2000)

1.5 Summary of this report

The sections that follow in this report will explore all the necessary points that will lead us to the final problem solution and design:

• A section that reviews the background knowledge necessary for completing the one or more approaches which constitute solutions for the problem.

• The alternatives solution that permit solving the same problem, compare them and select the best one according to suitable criteria.

• Modeling and analysis of the proposed solution.

• Indication of the budget necessary and the manner by which the required hardware, software, and/or services are going to be made available.

• Implementation and testing to prove that the design specifications have been satisfied.

• Verification and validation applied at all stages.

• Critical appraisal of the project, indicating the rationale for the design/implementation decisions, lessons learned during the course of the project, and evaluation of the product and process.
- System design constraints
- References.
- Appendix.
2.0 Various design approaches and alternatives

In this chapter, we will start by showing the approaches we can use in order to accomplish our aim of designing the CRYPTO-GLASSES. Then, in a second part, we will show the alternatives for each block in the overall design so that we can choose the best for each block in the design part.

2.1 Approaches for building the system

As we have seen from the above problem statement that we need to design a system that would secure and protect some of the encrypted emails and documents. This would be achieved via an independent device called CRYPTO-GLASSES which would decrypt the text on an eyeglasses shaped device and display them for the user.

Before using our device, the encrypted file would be ready on a text file on the screen of the PC. Our system’s main task, the CRYPTO-GLASSES, is to take the text in a certain way and decrypt it on that device independently from the computer and then displays the original message for the user. Making the device work independently from the computer in use would make the decrypted texts and documents much more secure and protected since no one would have access to that device except the user directly and indirectly.

There are multiple approaches to build a system with the above requirements. However, there exists only one optimal way to do that. The following are two approaches in which the system requirements can be achieved:

2.1.1 Approach 1:

The device can be made up of one main hardware component along with four external hardware components. In addition, a software would be needed to decrypt the encrypted text file. What are the theses hardware components and software that would make up such a device, the CRYPTO-GLASSES?
The hardware components

1- **A Bluetooth receiver device**: it is one of the CRYPTO-GLASSES components that would receive the encrypted text file which is then decrypted before it is displayed to the user.

2- **A Bluetooth transmitter device**: this device would be connected to the computer case which contains the encrypted text file or email. The task of this component would be to transmit the encrypted text file, so that the Bluetooth receiver would receive it.

3- **A keypad**: this is a component that would help in decrypting the encrypted text. We need a very reliable encryption and decryption code. The optimal way to achieve this is to use a powerful encryption code that uses a key each time it encrypts a text. Thus, with the help of such a keypad, the user would be able to supply the device with the appropriate key in which the decryption would be based. How user would know the key? Simply by synchronization between the user and the text transmitter who sets the key for the encrypted text. This can be done by sending the key by a message on the mobile phone.

4- **The LCD screen**: This component is used to display for the user the decrypted or the original file of the text.

5- **The microcontroller**: This is the main component of the CRYPTO-GLASSES system. It connects all the external components together. That is, it is connected to the Bluetooth receiver device, the keypad and the LCD screen; thus, making the interaction between these components possible. A software should be installed on the
microcontroller in order to make all the components interact with each others so that the whole system achieves the required goal.

The Software

In this approach, there is only one software. The task of this software is to take the text being received by the Bluetooth receiver device and to decrypt it based on the key entered via the keypad. The final step of this software is to display the decrypted text or the original text on the LCD screen.

Remark

External components to the microcontroller would not function unless their drivers are installed on that device. Thus, the drivers of the LCD screen, the keypad, and the Bluetooth receiver device should be installed on the microcontroller along the software that is used to decrypt the text. As a result, the software would be able to interact with the different components connected to the controller.

2.1.2 Approach 2

The device can be designed with a difference but totally different techniques. Instead of using a Bluetooth device, we can use a camera! By focusing the device in which the camera is attached to the screen that contains the encrypted text and capturing it. The next step of the system is to convert the picture to a text file using specific software. Finally the system should decrypt the outcome text to its original text. But what would be the hardware components in such a device? Do they differ a lot from the
previous approach? And what is the software that should be used in such a device? Is the software of the previous approach enough for operating this device? Such a system consists up of four hardware components and two softwares that would make up such a device.

The hardware components

1- **A small digital camera:** the mean to send the encrypted text from the PC screen to the CRYPTO-GLASSES device is done via a camera and not via a Bluetooth channel. By focusing the camera to the encrypted text, we can clearly capture a picture with high quality especially if the digital camera being used a high number of pixels or very high performance.

2- **A keypad:** the keypad used in this approach is the same as the one which would be used in the above approach. As we said that it is a component that would help in decrypting the encrypted text. We need a very reliable encryption and decryption code. The optimal way to achieve this is to use a powerful encryption code that uses a key each time it encrypts a text. Thus, with the help of such a keypad, the user would be able to supply the device with the appropriate key in which the decryption would be based. How user would know the key? Simply by synchronization between the user and the text transmitter who sets the key for the encrypted text. This can be done by sending the key by a message on the mobile phone.

3- **The LCD screen:** This component is important whatever the approach is. It is used to display for the user the decrypted or the original file of the text after the system finishes from its decryption operations.
4- **The microcontroller:** It connects all the external components together. It is connected to the camera, the keypad and the LCD screen; thus, making the interaction between these components possible. A software should be installed on the microcontroller in order to make all the components interact with each other so that the whole system achieves the required goal. However, more than one software is needed in building this approach. As a result, the microcontroller is the main component of the CRYPTO-GLASSES system.

**The Software**

A CRYPTO-GLASSES using a camera to capture a text needs two software to decrypt it to its original text. The first software that should be used is one that can be able to convert the captured text from a picture to a text file after a screen shot of the encrypted text is captured from the Screen using the camera. The second software that should be used is the same as the one used in the previous approach. After the first software has changed the picture to a text, the second software would be ready to decrypt it based on the key entered via the keypad. The next step of this software is to display the decrypted text or the original text on the LCD screen.

**Remark:**

As the case of the system described in the above approach, external components to the microcontroller would not function unless their drivers are installed on that device. Thus, the drivers of the LCD screen, the keypad, and the camera should be installed on the microcontroller along the softwares that are used convert a picture to a text and perform the decryption. As a result, the
software would be able to interact with the different components connected to the controller.

**Pros and Cons of the first Approach**

Sending the encrypted text file via a Bluetooth would reduce the effort of having a software at the microcontroller in changing a picture to a text file as in the second approach. Thus, we might see that the first approach is more convenient for such a device. However, this is not the case. A Bluetooth device broadcast its message. That is, when the Bluetooth transmitter device sends a message, it does not send it to its specific destination. As a result, any Bluetooth receiver on any device would receive the message being sent. This would be very bad for in terms of security and privacy. Although the text being sent would be encrypted, we would not like to send it to devices other than that of the destination since this would make people curious and would make problems in the future. They might try to break the encryption! However, if a very strong encryption code is used, this approach would be a second alternative. More, as in the case of sending, the Bluetooth receiver of the device would be able to receive any file being transmitted by any Bluetooth transmitter device. In such cases, if someone is trying to use the CRYPTO-GLASSES device at the time an outsider, near the user, is trying to send a file to someone via a Bluetooth device, the CRYPTO-GLASSES operations would be corrupted and this would lead to a bad performance. In addition, it is convenient to use the CRYPTO-GLASSES device in front of the PC containing the encrypted text and not from any place near the PC as the case with the first approach.

**Pros and Cons of the second Approach**

When using a camera device in the CRYPTO-GLASSES device instead of a Bluetooth devise, the user have to sit in front of the PC facing the screen. This would be more convenient as it has been described above. When we capture an image via the camera, this image would only go to the CRYPTO-GLASSES device only not any other device as in the case when using a Bluetooth device as
illustrated in the first approach. More, the device would not be corrupted by other external devices. However, using a camera instead of a Bluetooth device requires some effort in making a software that would convert a picture into a text file. This would be a hard task to do since we need a very precise software that can be very accurate in performing such an operation in order to get good results. If that software is not accurate the encrypted text would be decrypted in a wrong way and not to its original form which is the case that we want to avoid. As a result, we need a powerful software that would perform such an operation. This would be the part of the alternative and design phase.

**Which approach to use?**

As we can see from the cons and the pros of both approaches, it is more practical to go with the second approach which requires the use of a camera. The following shows a block diagram for the second approach with the camera as the component which will receive the encrypted text. In the design, we would emphasis more on the components of that block diagram.

**Figure 2.1**

Block diagram for the second approach
In the following parts of this chapter, we will study the design of those blocks of the project which have alternatives. This will be done by reviewing the related literature for each block and presenting the possible alternatives.

2.2 Alternatives for the decryption block

The choice of the encryption/decryption algorithm strongly depends on the probability to break it. We are interested in having a “hard-to-break” algorithm. However, there is also another constraint that also matters; it is the complexity of the algorithm, because after all, it will affect the performance of our embedded system inside the CRYPTO-GLASSES.

We are going to present in this section two ciphers, that are completely opposite in terms of performance and complexity.

1. Vigenere cipher: easy to break, and very simple.
2. Rijndael cipher (also known as AES): impossible to break using today’s computers, and very complex.

2.2.1 Vigenere cipher

This cipher is based on the Caeser cipher which is one of the simplest and most widely known encryption techniques, based on shifts and substitutions. It was called Caeser cipher because it is thought that it was used by Julius Caeser to communicate with his army during wars. It simply consists of shifting each alphabet letter by a given number of letters. For example, the sentence: “Hello World” encrypted with a Caeser cipher with a number of 3 would become: “KHOOR ZRUOG”.

The Vigenere cipher is very close to the Caeser cipher but it is more accurately a series of Caeser cipher cascaded one after the other. In fact, the Vigenere cipher uses the letters of a keyword, called key to perform the necessary shifts. In other terms, it uses what we call a shift matrix which corresponds to each letter its encrypted peer according to the corresponding letter of the keyword.

Figure 5.1 shows the Vigenere matrix:
a. Encryption method

During the encryption process, the rows in Figure 2.2 below represent the letters of the key while the columns represent the letters of the original text (to encrypt). For example, if the original text (to encrypt) is “Hello World” and the key is “FYP”, the encryption process goes as follows:

1. Look at the row starting with F, the first letter of the key.
2. Look at the column starting with H, the first letter of the original text.
3. Check their intersection: it is the letter M.
4. M is the encrypted correspondence of H using the key “FYP”
5. Repeat the same procedure by moving to the next letter in the original text and the next letter in the key (repeat the key when it is over)…

So the encrypted form of the original text “Hello World” will be “MCAQM LTPAI”

During the decryption process, the rows also represent the letters of key, so we search in the row for the encrypted letter and its place would indicate the column of the decrypted text.

If we continue with the previous example, the decryption of “MCAQM LTPAI” will be “Hello World”.
b. Advantage

The Vigenere cipher has one single advantage. It uses very simple operations, and thus takes a very little time of the processor so that it can perform the encryption/decryption.

c. Drawback

One big disadvantage affects the performance of the Vigenere cipher: it is easy to break. We do not intend here to explain in details the process of breaking the cipher but we will only show how it works:
“To break Vigenere encryption, one guesses a period \( p \) and then, by comparing the histogram of observed frequencies of \( s^{th} \) letters to the histogram of English letter probabilities, one is led to the correct value of \( k_s \). A wrong guess for the period \( p \) leads to relatively flat histograms for all or most of the values of \( s \). The code breaker in this case repeats the analysis with a new trial period.”

One remark

We notice that usually, the Vigenere algorithm does not encrypt numerical data or other symbols widely used (such that?!&….) as shown in the matrix. In our project, we need the encryption block to encrypt such symbols and numeric. This can easily be done by extending the matrix and include the additional data types that we desire.

2.2.2 Rijndael cipher (also called AES)

This algorithm, also called AES (Advanced Encryption Standard) is one of the strongest encryption algorithms know world wide. It is adopted by the US government to protect its confidential data rated as “TOP SECRET”.

a. Encryption method

In this report, we don’t want to explain extensively how the AES algorithm works because it needs a lot of theoretical and especially mathematical background, but we will rather state the steps involved in its working:

Each round of AES consists of four stages:

1. **SubBytes** — a non-linear substitution step where each byte is replaced with another according to a lookup table.
2. **ShiftRows** — a transposition step where each row of the state is shifted cyclically a certain number of steps.

3. **MixColumns** — a mixing operation which operates on the columns of the state, combining the four bytes in each column using a linear transformation.

4. **AddRoundKey** — each byte of the state is combined with the round key; each round key is derived from the cipher key using a key schedule. (Wikipedia, 2006)

**b. Advantage**

AES is one of the strongest encryption algorithms today. In fact, breaking the AES algorithm with a 128-bit key requires $2^{120}$ operations, which cannot be done now right now.

**c. Drawback**

The main drawback of the AES algorithm is that it is a very complex algorithm, which needs a lot of operations to encrypt or decrypt the data, thus consuming a lot of time from the processor.

**2.3 Webcam Overview**

Webcams, like most things, range from simple to complex. But if we understand the essence of a simple Webcam, increasing the complexity is only a matter of adding functionality through software, custom code and/or equipment connections.

A simple Webcam setup consists of a digital camera attached to the computer, typically through the USB port. The camera part of the Webcam setup is just a digital camera and the “Webcam” nature of the camera comes with the software. Webcam
software grabs a frame from the digital camera at a preset interval and transfers it to another location for viewing. In case of using the webcam for video application it is better to choose a Webcam system with a high frame rate. Frame rate indicates the number of pictures the software can grab and transfer in one second. For streaming video, you need a minimum rate of at least 15 frames per second (fps), and 30 frames per seconds is ideal.

Two Webcams were being tested and evaluated during the first phase of our project:

- **Logitech QuickCam Pro 4000**

  **Technical Data:**
  
  - **Image sensor:** VGA CCD
  - **Still image-capture resolution:** Up to 1,280 x 960 pixels, 1.3 mega pixels (software-interpolated)
  - **Still image-capture format:** JPEG
  - **Video-capture resolution:** Up to 640 x 480 pixels (software-interpolated)
  - **Video-capture format:** AVI
  - **Compression:** Fixed data rate hardware intra frame
  - **Frame rate:** Up to 30 frames per second (with recommended system)
  - **Audio recording:** Yes (includes microphone)
  - **Lighting:** Ambient indoor incandescent, fluorescent (50 or 60 Hz flicker-free), daylight
  - **Controls:** Digital zoom
  - **Built-in flash:** No
  - **Onboard memory:** None
  - **Included memory cards:** None
  - **Interface:** USB
  - **Compatibility:** PC, Macintosh
  - **Power:** Powered through USB
- **Weight**: 1.3 pounds
- **Warranty, parts**: 2 years (hardware)
- **Warranty, labor**: 2 years (hardware)
- **Price**: $95.99

**Figure 2.3**

Logitech
QuickCam Pro
4000

➢ **iREZ K2 Camera**

**Technical Data:**

- **Image Sensor Size**: 1/3"; 0.35 Mega pixels
- **Resolution**: Up to 640x480
- **Focus Distance**: 5cm to Infinity
- **Field of View**: 54°
- **Colors**: Up to 32 bits RGB
- **Frame Rate**: Up to 30 fps at 640x480 (VGA)
- **USB Cable Length**: 2 meters
- **Manual Focus**: Yes
- **Auto White Balance**: Yes
- **Auto Exposure Control**: Yes
• **LED Indicator**: Yes
• **Price**: $99.95

2.4 Filtering block

Filtering the image captured by the camera is a very important step that we need to do as good as we can in order to ensure that the output of the filtering process is a clear picture, with minimal or no noise. In fact, the more the picture is noisy and hard to read, the more the OCR will find it difficult to recognize the characters in the picture.

The filtering process that we will use consists of improving the image contrast and enhancing it using the Linux libraries. This process includes the following functions: Contrast, Enhance. Those two functions will ensure that the image will be enhanced, thus the noise reduced or eliminated, and that it will be subject to a contrast increase in order to increase the contrast of the block words that we try to read.
2.5 Design of the OCR block

2.3.1 Overview

The field of **Optical Character Recognition** is a much studied and well researched field. It is a logical and technical extension of **Optical Mark Recognition** technology. It is generally distinguished from OCR by the fact that a recognition engine is not required. That is, the marks are constructed in such a way that there is little chance of not reading the marks correctly. In OMR, there is no image kept after acquisition. That is, the image is immediately interpreted, and the original image is thrown away. The high accuracy of OMR (typically 99.9%) reduces the need for keeping any record other than the scanned value.

**Optical Character Recognition** refers to the branch of computer science that involves reading text from paper and translating the images into a form that the computer can manipulate (for example, into ASCII codes). An OCR system enables you to take a book or a magazine article, feed it directly into an electronic computer file, and then edit the file using a word processor. Our CRYPTO-GLASSES system includes a digital camera for taking pictures and generating an image file that will be fed next to the OCR. Most OCR systems use a combination of hardware (specialized circuit boards) and software to recognize characters, although some inexpensive systems do it entirely through software (like our case). Advanced OCR systems can read text in large variety of fonts, but they still have difficulty with handwritten text. The potential of OCR systems is enormous because they enable users to harness the power of computers to access printed documents.

Older OCR systems match these images against stored bitmaps based on specific fonts. The hit-or-miss results of such pattern-recognition systems helped establish OCR's reputation for inaccuracy. Today's OCR engines add the multiple algorithms of neural
network technology to analyze the stroke edge, the line of discontinuity between the text characters, and the background. Allowing for irregularities of printed ink on paper, each algorithm averages the light and dark along the side of a stroke, matches it to known characters and makes a best guess as to which character it is. The OCR software then averages or polls the results from all the algorithms to obtain a single reading.

The problem of **Optical Character Recognition** is relatively easy to deal with. The specific application to Latin-script text is a solved problem. The technique is simple to describe. Once the input file is generated by the digital camera, the font that the text is written with is identified using predetermined lookup tables that are easy to construct. The main problem lies in initially identifying the particular font that one is dealing with. Other extensions to OCR include reading the Barcode on items, the Postal Code on letters (a problem often made easier by restricting the location of the code). This can also be extended to recognizing fixed character sets that need not necessarily belong to an alphabet.

### 2.3.2 Various design alternatives for the OCR

The OCR is one of the basic components of our CRYPTO-GLASSES project since it is essential to recognize and convert actual image (which is a set of pixels) into a text file (which is a set of characters). Once we were able to convert our pixels into characters or bits, it became easy to feed these bits to a processor where we can process these data and generate the exact output.

Two OCRs were being tested and evaluated during the first phase of our project under Linux based platform:

- **GOCR:**
  a. **General Overview**: An optical character recognition program, released under the GNU General Public License. It reads images in many formats and outputs a text
file. Possible image formats are pnm, pbm, pgm, and ppm, some pcx and tga image files. Other formats like pnm.gz, pnm.bz2, png, jpg, tiff, gif, bmp will be automatically converted using the netpbm-progs, gzip and bzip2 via UNIX pipe. The user does not have to train the program or store large font bases. He can simply call GOCR from the command line (Linux Shell) and get the wished results.

b. **How to use?**: One main advantage of using GOCR is that it is accessed and controlled by the command line interface; by this way it will be easy for us to use this program in small systems such as embedded systems.

In order to realize an OCR operation, we need first to supply our program with an image and then receive the output as a text file.

The command that will be used during our project is the following

```
./gocr -v 1 file.jpg > out.txt # generate text file
```

gocr: The actual process that must be supplied with the correct parameters in order to execute in a convenient manner.

file.jpg: The input image file that will be present in the same directory of the gocr executable file.

out.txt: This file will be created and generated automatically when the OCR process terminates correctly without any run time error. This file contains the actual recognized characters with some errors.

c. **Program Limitations**:

- **Memory Limitations**: GOCR requires 8.75 MB for storing an A4 picture in memory, for this reason we must make sure either to have enough RAM installed on our system or we can alternatively cut the picture into small pieces.

- **Others Limitations**: 

- Fonts should be in the range between 20 to 60 pixels
- It takes around 1 to 2 sec for the OCR to execute and generate the text file.
- Try to use larger fonts in order to fight against the interference of one character and the other.
- Can not recognize the following fonts: Chinese, Arabic, Egyptian, Cyrillic or Klingon.
- Handwritten text recognition feature is not supported.
- Only alpha numeric font’s recognition is supported.

d. **Program Strengths:**

- GOCR is a free source program where you can download it for free from the following link: http://jocr.sourceforge.net
- Simple installation procedure and easy to use scenario in order to execute and generate outputs.
- Very powerful alpha-numeric recognizer (even with small fonts).
- GOCR is interfaced by a command line that can be done from any regular UNIX shell → this is an advantage while implementing this project on an embedded system.
- Program ability to receive several kinds of image formats such as: pnm, pbm, pgm, ppm, some pcx, tga, pnm.gz, pnm.bz2, png, jpg, tiff, gif and bmp.

➢ **OCRAD:**

a. **General Overview:** GNU Ocrad is an OCR (Optical Character Recognition) program implemented as a filter and based on a feature extraction method. It reads a bitmap image in pbm format and outputs text in ISO-8859-1 (Latin-1)
charset. It can be used as a stand-alone console application, or as a backend to other programs.

b. **Usage**: As stated above, the main advantage of using OCRAD is that it is accessed and controlled by the command line interface like in GOCR; by this way it will be easy for us to use this program in small systems such as embedded systems.

In order to realize an OCR operation using OCRAD program, we need first to supply our program with an image and then receive the output as a text file. The command that will be used during our project is the following:

```
./ocrad -x test1.orf examples/test1.pbm
```

- **ocrad**: The actual process that must be supplied with the correct parameters in order to execute in a convenient manner.
- **test1.pbm**: The input image file that will be present in the same directory of the ocrad executable file. Ocrad only receives pbm image format!
- **test1.orf**: This file will be created and generated automatically when the OCR process terminates correctly without any run time error. Orf stands for OCR results file where we can recollect the recognized characters with some errors.

c. **Program Limitations**:

- **Memory Limitations**: OCRAD requires 20 MB for storing an A4 picture in memory, since it does not apply any compression techniques during execution time. For this reason we must make sure either to have enough RAM (much more RAM than GOCR) installed on our system or we can alternatively cut the picture into smaller pieces.

- **Others Limitations**:
- For better results the characters should be at least 20 pixels high, the more we increase the fonts the better the results will get on average.
- Merged characters are always a problem. Try to avoid them.
- Very bold or very light (broken) characters are also a problem.
- Always see with your own eyes the pbm file before feeding it to Ocrad in order to verify if it is clear and net.
- By comparing it to GOCR, OCRAD is relatively weak in recognizing regular alpha numeric font.
- Handwritten text recognition feature is not supported.
- OCRAD is only capable of receiving .pbm image font and usually these pbm files requires a lot of memory because they are not compressed again Memory problem.

d. Program Strengths:

- OCRAD is a free source program where you can download it for free from the following link:
- Simple installation procedure and easy to use scenario in order to execute and generate outputs.
- Very powerful symbol recognizer (even with small fonts), an example of symbols that can be very well recognized are the following: %$#%^*() +- and so on…
- OCRAD is interfaced by a command line that can be done from any regular UNIX shell this is an advantage while implementing this project on an embedded system.
2.6 The Microcontroller

The main component of the whole system is the microcontroller. A microcontroller is a highly integrated chip that contains all the parts needed to make up a controller. Typically, this includes a CPU, RAM, ROM, I/O ports, and timers. The role of a microcontroller is designed to perform a specific task to control a specific system unlike a general-purpose computer. Thus, the parts can be reduced making the size of the microcontroller very small so the whole system can be held by hand and is portable.

- The ROM is the component in which we use to program the system. We shall place in it the suitable programs that should control the CRYPTO-GLASSES system to perform the required task. The decryption program, which is the Seizer, as well as the OCR, which is the GOCR, are going to serve as the main software of the system in the ROM.

- Our microcontroller had to interface with three components, the camera, an LCD screen and a keypad that is used to enter the key along which the seizer is based to decrypt the message captured by the camera. Thus, we need to have three I/O ports that can be used to interface the microcontroller with the three external parts mentioned above.

- The Ram serves an important component in the controller as it does in a regular PC. The main purpose of our system is to analyze and perform operations on the screen message in the form of a picture and bits. Thus, after a screen shot is taken by the camera, it has to be saved on a certain place so that the microcontroller can perform the required tasks on it. These tasks are: converting the image to a text and decrypting it. This place is said to be the RAM of the microcontroller.

- The third important component of the microcontroller is the CPU. All the actions that are performed in the microcontroller are done by its CPU. The OCR need to analyze the image in order to convert it to a text file. Thus, a high speed CPU is
needed in order to have good decisions without overhead. More, the decryption task is also done by the **CPU**. Thus, the controller that we should use should be selected based on the high performance of its **CPU**.

- The timer is needed in any device taking actions of multiple actions to perform a certain task. We have to synchronize the different actions taking place. We should be accurate in specifying the delays in order to make our system as fast as possible without interfering with the different actions taking place.

**The required tasks from the microcontroller**

The drivers of the LCD, keypad and that of the camera would be installed on the ROM of the microcontroller in addition to the different softwares (OCR and the Seizer). Thus, we need our ROM to be somehow big between 1.5 and 3 MB in size but not too big in order not to have delays in the system.

Referring to the components mentioned above, we need to have specific microcontroller with the appropriate RAM, ROM, I/O and CPU. We need our CPU to be of high speed in order to accomplish the different tasks in fraction of seconds. The ROM needs to have a sufficient space in order to fit the drivers of the camera, keypad, and the LCD. It should also have enough space for the other softwares which include the OCR and the Seizer. In our microcontroller, the RAM is mainly used as a workplace where we can manipulate the image being captured. The image would be in the form of JPEG files. As a result, we need our RAM to be big enough so that it can hold such files and allow us to use some other space for manipulation. The microcontroller should also be selected so that its I/O ports would be compatible with the external components (Camera, LCD, and keypad.)
3.0 Design

In this chapter, we will start by stating the alternatives that were presented in chapter 2 and we will compare those possible alternative designs for the blocks and select the most suitable ones.

3.1 Selection of the suitable design alternative for each block

In this part, we will select the best alternative between those presented above, based on practical experimentation and testing.

3.1.1 Comparison between the two decryption algorithms

A comparison between the two ciphers (Vigenere and Rijndael) according to the constraints of our project is necessary so that we can decide which one suits our project the best. Although it seems a good idea to use both algorithms so that we can give the user a choice between using a strong encryption scheme or a not very strong one, many criteria should be considered and those criteria should decide on the choice. But in order to decide on the best algorithm to use, we will assign to each criteria a given weight and finally calculate the points that each algorithm takes and decide on the one to use:

a. Effectiveness of the encryption (Weight = 6)

One important factor in our design is the use of a cipher that is not very easy to break. In fact, we don’t want that while the CRYPTO-GLASSES are decoding the data, someone else, watching us, is capable of tracking the sequences of encoding and breaking the encoder algorithm.
That’s why, according to this criterion, the Rijndael encryption algorithm takes over the Vigenere algorithm.

\[ Rijndael: 10 \times 6 = 60 \]
Vigenere: $2 \times 6 = 12$

b. **Compatibility with the OCR (Weight = 9)**

This is a crucial criterion. In fact, due to the constraints in the OCR technology, the OCR cannot read all given characters, and also it cannot always read correctly even simple characters with optimal text size and font.

**Some facts and some solutions (if any):**

- **Fact 1:** The AES encoder encrypts any given text into a set of characters that are very hard to almost all OCR to recognise.
- **Solution 1:** In order to find a solution to this problem, we decided to use an AES algorithm that provides at its output a base 64 encoding instead of the various characters it usually outputs, and takes as input also a base 64 input to decrypt it. The primary overall design of our project is shown in Figure 5.2, where the output characters are supposed to be exactly like the input characters.
- **Fact 2:** The OCR technology is not that accurate as we said before. An error in reading one of the base 64 characters would cause a propagation of errors in a part of the output characters which is equivalent to a decryption error.
- **Solution 2:** We couldn’t really find a solution for the above problem because it is inherent in the AES algorithm.

**Let’s take an example:**

i. Input text (to encrypt): Hello World
ii. Key: fypfyp
iii. Output encrypted and transformed into base64:
    cRSmT1rFmlq6SnJeg/iSsKg4M0ZRkSm31Qr2
iv. Assume the OCR reads one character wrong:
    cRSmT1rFmlq6SnJeg/iSsKg4MÖRkSm31Qr2
v. Text after decryption:
    â•|U% Ü Sd
We see that the whole output is wrong. This was caused by only reading one character wrong!

While if one character in the vigenere encryption, only this character will be affected because there is no correlation between the characters.

That’s why, the Vigenere algorithm strongly takes over the Rijndael algorithm:

→ Rijndael: 0*9 = 0

Vigenere: 10 * 9 = 90

That’s why, given the two criteria in question, the Vigenre encryption algorithm is selected because it is more convenient to our design.

**Input to our crypto-glasses**

![Figure 3. Primary overall diagram](image-url)
3.1.2 Webcam selection

As we can notice from the specifications above in the alternatives part, Logitech QuickCam Pro 4000 is more suitable than iREZ K2 Camera for our project since the ratio resolution over price is better (Logitech’ resolution is 1.3 Mega pixels and IREZ’s resolution is 0.35 Mega pixels).

3.1.3 Selection of the suitable OCR

Our CRYPTO-GLASSES project is mainly a visual application that must be capable of understanding and processing characters and numbers that are familiar to human beings. The aim of this project as stated before is to encrypt our papers, documents, email in order to protect our data from hackers and crackers. The only way a person can decrypt these documents, papers and emails is through the CRYPTO-GLASSES.

After a long period of testing and evaluation we decided to discard OCRAD and use instead GOCR for the following reasons:

- GOCR was designed to recognize English letters and numbers. Several tests were made to show that this program is far more powerful in recognizing alpha-numeric fonts than OCRAD which is basically designed to recognize symbols rather than letters and numbers.

- GOCR has the capability to recognize small font sizes (down to font 8, Times New Roman).

- OCRAD is weak in recognizing small fonts. If we supply a small font picture to OCRAD, most of the time it will not be able to recognize the
letters independently from the other neighboring letters which will lead to the miss recognizing of the whole sentence and the meaning will be totally miss understood!

- In term of Memory usage, using GOCR is much more efficient than using OCRAD. In embedded system, memory is very expansive and limited; therefore it is more convenient to choose GOCR as a suitable program for embedded system.

As you can notice, GOCR is better than OCRAD in three main fields:

1- **Performance**: GOCR is designed for Alpha-Numeric Fonts
2- **Memory Efficiency**: GOCR accepts compressed type images (like jpeg)
3- **Quality**: GOCR produces good results even when using small fonts.

### 3.1.4 The selected Microcontroller

Based on the requirements mentioned above, we found that the uCidimm ColdFire 5272 microcontroller is optimal for our system. The uCidimm ColdFire 5272 microcontroller has a low-cost. Like most microcontrollers, the uCidimm ColdFire 5272 allows the use of only Linux commands. This would be better in terms of efficiency since Linux commands would not take much space in the ROM thus making it possible to use a ROM with small size compatible with a microcontroller. The following are the features found in the uCidimm ColdFire microcontroller:

μCidimm ColdFire Features and Compatibility:

- Motorola RISC core
- 2 MB FLASH ROM
- 8 MB SDRAM
- 10BaseT Ethernet
- 100BaseT Ethernet
- On board MAC for DSP
- 2 RS232 serials
- 1 QSPI (queued) serial
- USB functionality
- 2x 16 & 1x 4bit parallel I/O
- Chip selects available
- 2 PWM output
- Software watchdog timer
- 4 hardware interrupts
- Power management
- 1.7" x 2.7" 144 pin soDIMM form factor
- uClinux OS preinstalled
- TCP/IP SLIP and PPP
- Cost-effective

As we see from the above features of the uCidimm ColdFire microcontroller that our requirements are achieved. This microcontroller has a 2MB FLASH ROM which is not too big and at the same time provide enough space for storing the OCR and the Seizer as well as the drivers for the camera, keypad and the LCD screen. We have two input streams, one getting from the camera in the form of a picture and the other getting from the keypad in the form of a key to decrypt the message. Thus, we need two interrupts and our microcontroller provide us with four which is enough and can be reserved for later use in case we need to have additional accessories to our system. The uCidimm ColdFire microcontroller provides us with 8 MB SDRAM which would be enough for storing the captured JPEG pictures and manipulating it. It also contains 2 PWM outputs in which one would be useful for displaying the decrypted message on the LCD screen of the CRYPTO-GLASSES device. This microcontroller is a RISC based architecture that deals with registers rather than memory during processing time. This increases the speed of the CPU; thus, achieving higher performance. In addition, the uCidimm ColdFire
microcontroller is a cost-effective controller and has the uClinux OS preinstalled prior to its usage. This facilitates our task in programming the controller before installing the different software that serves as the basics for our system (OCR, Seizer, and the drivers of the I/O devices).

**What is inside the Kit?**

“To expedite development work, the uCdimm ColdFire5272 is offered in kit form with a highly functional development platform called the Arcturus uCevolution. The uCevolution includes all appropriate connectors required for development, debugging and accessing features of the uCdimm. The uCevolution also includes over 7 square inches of prototyping space, transmit and receive indicators for serial and Ethernet, reset button, 4 interrupt buttons, 8 bi-directional controls for I/O testing and extension headers for additional functionality or daughter boards. Though designed for prototyping with the uCdimm, the uCevolution is also perfect for one-off designs. The uCdimm ColdFire5272 Development Kit also includes, uClinux ColdFire5272 firmware distribution, power supply, cross-over and straight through Ethernet cables, serial cable, reference guide and programming manual.” (Arcturus Networks, 2005)

**The uCdimm ColdFire 5272 Architecture**

As we can see from the figure shown below that the clock cycle is 48 MHz. This is very good in terms of performance for a microcontroller. More, it supports good features for Ethernet connection. This might be useful for later use.
3.1.5 Small LCD screen

One of the important parts in our intended design is the small LCD screen. In fact, after decrypting the text provided by the OCR, the result of this decryption should be shown to the user, for this is the original data he wanted to see before the encryption process.
To let the user see the data without letting anyone else see them, a small LCD screen will have to be embedded inside the CRYPTO-GLASSES, where the data will be displayed.

In other terms, in the inside part of the glasses (in front of the eyes of the user), an LCD screen will be there and will display the data that is the result of the whole processing.

The LCD screen doesn’t really have alternatives as an entity but we can only discuss the type of the LCD screen to be used.

The only information we know about this LCD screen is that it should be placed at a distance of around 10 centimeters from the eyes of the user (in order to let him clearly see the information).

As we will explain in the implementation phase of this project, we couldn’t find any LCD screen of about 3 inches of diameter that can be connected to the VGA of the embedded system. This component is not on sale in the Lebanese market.

We have tried to order it from Europe but it costed around $500 so we decided to use a computer prototype instead.

3.2 Designing the product

Now that we have chosen how each block should operate, we can link them together in order to obtain the overall final product.

Before doing so, we will explain as a summary how the overall design should work:
1. The user has an encrypted version of the data he wants to read: this data is either on paper or in a text editor (Windows: MsWord, Notepad; Linux: Emacs, gedit…)
2. The CRYPTO-GLASSES will take a picture of the encrypted text using a camera.
3. The image will be stored inside the memory of the microcontroller and will be ready to start the optical character recognition process.
4. The OCR processing starts to retrieve the useful data from the picture.

Notice: here, in order to let know our useful encrypted data after the OCR processing, the original data will be surrounded by a special tag in order to indicate this data during the decryption.
After many trials and through careful study of the capabilities of the OCR, we found that the tag: _-_ is the best to be read by the OCR.

For example:

- **Input text**: Final year project
- **Key**: fyp
- **Encrypted message with tag**: _-_k62f9 32pw 4w!yj08_-_-_  
- That’s why, the OCR will easily detect the tag, and through software, we will isolate the data between the two tags and decrypt it alone.

Now, how should we link the different blocks together?

1. The first step is to link the microcontroller to its interfaces.
   - The first block we should link to the microcontroller is the camera. In fact, just after the camera takes the picture, the picture should be stored inside the memory of the microcontroller as we have already said.
   
   **We had a problem**: We should write in software the driver of the camera so that we can transform the stream of bits (pixels) that it provides to a jpg picture or some other format.
   
   The problem was solved and we found a camera driver that enabled us to attain our purpose.

2. The second block is the keypad. This device is used in order to supply the decryption program with the necessary key. So the user will be able, alone, without any outside intervention to supply the key.

   **But, how will the user know the key if the text was given to him by someone else?**

   In order to solve this problem, we have though about many alternatives but the best one that we thought about and that guarantees the privacy of the key is to let the first user send the key to the end user through SMS (using the GSM
mobile system). Using this method, virtually no one will be able to know the key other than the end users concerned with the text data that is being sent.

- The third component that we should link to the microcontroller is the LCD display screen. This screen will be used in order to display the final decrypted data which should be the same as the original data to be recovered. This LCD screen will face the eyes of the end user and display the data only to him, privately.

**A question: how will the user be able to see the message knowing that the LCDS screen is very close to his eyes?**

The solution will be solved in the spring term. But we have thought about possible solutions: we could for example simply let the LCD screen be somehow far from the eyes to let the user see the text, or we could use some optical mechanism (like a use of some lenses to diverge the light and let him clearly see the text).

2. The second step is to download on the microcontroller the necessary software that would enable the project to be executed.

This microcontroller is based on a light version Linux operating system. On the flash, we need to download the following programs:

- GOCR
- Vigenere encryption

We have on the flash a 2MB of free memory. The previous programs that we need to download on the flash need around 500KB, so they can easily fit.

So, the final picture of the design will be as follows:
In this project, before we began writing our code, we made a lot of comparisons between several operating systems in order to decide which one is best suitable for our system. The two main operating systems that were tested and compared to each others are **RedHat Fedora Core & Windows Xp**. At the end we decided to work on **RedHat Fedora Core** edition for the following reasons:

**The Big Question is: Why using Linux instead of Windows?**

- **Linux source code is freely distributed**: Tens of thousands of programmers have reviewed the source code to improve performance, eliminate bugs, and strengthen security. No other operating system has ever undergone this level of review.
• **Linux has the best technical support available**: Linux is supported by commercial distributors, consultants, and by a very active community of users and developers. In 1997, the Linux community was awarded InfoWorld's Product of the Year Award for Best Technical Support over all commercial software vendors.

• **Linux has no vendor lock-in**: The availability of source code means that every user and support provider is empowered to get to the root of technical problems quickly and effectively. This contrasts sharply with proprietary operating systems, where even top-tier support providers must rely on the OS vendor for technical information and bug fixes.

• **Linux runs on a wide range of hardware**: Most Linux systems are based on standard PC hardware, and Linux supports a very wide range of PC devices and hardware. And more specifically to our project, the embedded system μCidimm that we will use it in our project is based on a light edition of Linux.

• **Linux is exceptionally stable**: Properly configured, Linux systems will generally run until the hardware fails or the system is shut down. Continuous up-times of hundreds of days (up to a year or more) are not uncommon.

• **Linux has a low total cost of ownership**: Although the Linux learning curve is significant, the stability, design, and breadth of tools available for Linux result in very low ongoing operating costs. So we can get any software we want for free, such that the GOCR…

• **Linux is a free source system**: The cost of the RedHat Fedora Core is actually $0.00.
- **Linux can be heavily oriented by a command line:** That’s why, we could easily use it for embedded systems.

Now, we designed a shell script in order to let the whole program work: after encryption, the program feeds the image captured by the camera to the OCR, which output a text file to the decryptor and the original input is then restored. Using this shell script, the whole process will be automated, enabling us to run all the processes inside the product in one shot.

**Shell Script**

Shell Script is series of command written in plain text file. Shell script is just like batch file in MS-DOS but have more power than the MS-DOS batch file.

Shell in an interactive program. In other words, it means that it accepts commands from the user and executes them. But if the user uses a sequence of n numbers of commands, it is better for him to store this sequence of commands to text file and tell the shell to execute this text file instead of entering the commands. In this manner, all the commands will be executed in a straightforward and automated way. This is known as Shell Script.

**Why to write a Shell Script instead of generating Executable file?**

In order to group and integrate our program modules (OCR and the Decryptor) into one compact unit, we decided to use a simple Shell Script instead of generating an Executable program file for the following reasons:

- Shell script can take input from user, file and output them on screen.
- Useful to create our own commands.
- Save lots of time.
- To automate our commands and execute them in sequence.
- System Administration part can be also automated (System calls)
- Shell script is much more efficient in memory usage than executable file since it is just a text file with a set of commands that are normally short and straightforward.
- Because our embedded system is limited in memory, it is better to use a shell script (light programming) instead of Executable file that is generated by compiling a C code (heavy programming that rely on large libraries).

Note that it is true that Shell Scripting is less powerful than a program that is written in C language, but according to our application, we are using Shell Scripting in order to execute a sequence of commands in the Shell and not to perform a real and complex programming task. So it is much more straightforward and efficient to use Shell Scripting in the case of execution a set of Shell commands.

**Writing the Shell script**

These are the steps that are needed in order to write a Shell Script:

1. Use any editor like vi or emacs to write shell script (we used emacs).
2. After writing shell script set execute permission for your script as follows:
   ```bash
   /> chmod permission your-script-name
   ```
3. Execute your script in the following manner:
   ```bash
   /> .your-script-name
   ```

Note that the . / means current directory and . (dot) means execute given command file in current shell without starting the new copy of shell, the syntax for . (dot) command is as follows:

```bash
/> . command-name
```

**In order to check more information about the shell script, please refer to the appendix where the shell script we wrote is included.**
3.3 Budget

The budget needed for accomplishing the overall design is presented in the table below. This is an estimated budget that may differ of around $100.

<table>
<thead>
<tr>
<th>Component name</th>
<th>Quantity</th>
<th>Price ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microprocessor</td>
<td>1</td>
<td>300</td>
</tr>
<tr>
<td>LCD screen</td>
<td>1</td>
<td>500</td>
</tr>
<tr>
<td>Webcam</td>
<td>1</td>
<td>95.99</td>
</tr>
<tr>
<td>OCR</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Encryption block</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Eyeglasses</td>
<td>1</td>
<td>50</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td><strong>1045.99</strong></td>
</tr>
</tbody>
</table>

In terms of budget, this amount of money is somehow big but we should take into account that this is an amount needed for building one product, not on a manufacturing level, where the price will easily be reduced by 20%.

Concerning the manner by which the software and hardware needed are going to be (or were) procured, the following table includes the details:

<table>
<thead>
<tr>
<th>Hardware (H), Software (S)</th>
<th>Manner by which it will be procured</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microcontroller (H)</td>
<td>Available at the LABs in AUB</td>
<td>As explained later on, this microcontroller was with no software, and had problems to communicate with the computer</td>
</tr>
<tr>
<td>Webcam (H)</td>
<td>Available at the LABs in</td>
<td></td>
</tr>
<tr>
<td>Component</td>
<td>Details</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>LCD screen (H)</td>
<td>Available outside Lebanon</td>
<td></td>
</tr>
<tr>
<td></td>
<td>As explained later on, because of the cost of this LCD screen (around $500), we decided to use a prototype instead.</td>
<td></td>
</tr>
<tr>
<td>Eyeglasses (H)</td>
<td>To be manufactured in Lebanon</td>
<td></td>
</tr>
<tr>
<td></td>
<td>These eyeglasses were not manufactured because of the use of a prototype</td>
<td></td>
</tr>
<tr>
<td>OCR (S)</td>
<td>Downloadable from the Internet</td>
<td></td>
</tr>
<tr>
<td>Encryption algorithm (S)</td>
<td>Idea from the Internet</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Was almost all written</td>
<td></td>
</tr>
<tr>
<td>Webcam driver (S)</td>
<td>Downloadable from the Internet</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Was extensively modified, and needed a kernel update</td>
<td></td>
</tr>
<tr>
<td>Linux Fedora core 4 OS (S)</td>
<td>Downloadable from the Internet</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Needed a kernel update</td>
<td></td>
</tr>
<tr>
<td>Bluetooth driver (S)</td>
<td>Downloadable from the Internet</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The need for this driver is to be explained later on in this report</td>
<td></td>
</tr>
</tbody>
</table>
4.0 Implementation

After analyzing and designing the solution to the CRYPTO-GLASSES problem, the solution is finally implemented. In this section, we will describe how the implementation was done, the tools that were used, the difficulties that were encountered and the way they were overcome.

4.1 How the implementation was done?

After analyzing and designing the solution to this project, we moved forward towards the implementation part of the project. During all the implementation of the theory part, we have encountered many problems and difficulties especially in the software part.

a. The first task that was taken into consideration is the image filtering block. This image filtering block is essential for our project since all pictures taken from the webcam are affected by noise and light interferences. To get rid of these noises and light interferences, we decided to develop a program that is able to filter out the noise from the picture by adjusting some digital picture parameters like the brightness, contrast, enhancement and sharpening the letters. For best filtering adjustment, we created three filtering modes schemes as explained below.

This image filtering program has dramatically enhanced the performance of our project. Without the usage of such filter, it will be virtually impossible for the OCR to recognize the characters in the taken image and therefore, the decryption process will fail and incorrect output will be delivered. The main reasons of using an image filtering unit before feeding the original picture to the OCR are:

a) There is no perfect camera that is able to take extremely high quality picture without any kind of noise and light interferences.
b) The OCR is very sensitive to any kind of noise or interference that affects the characters in the picture. In other words, any noise introduction between the characters will lead to wrong character recognition.

b. After testing the image filtering unit, we moved forward to configure and to set up the \texttt{\textmu\texttt{cdimm} microcontroller} in order to program it later. During this phase, we have encountered a lot of problems and difficulties in setting up the microcontroller, some of these problems are:

- Serious problems were encountered during the installation of the \texttt{\textmu\texttt{cdimm}} microcontroller and more specifically the connection problems between the desktop computer and the microcontroller.

- The supported driver and the development toolkit that came with the \texttt{\textmu\texttt{cdimm}} microcontroller were neither compatible nor complete.

- Lack of support and tutorials related to the \texttt{\textmu\texttt{cdimm}} microcontroller.

- The \texttt{\textmu\texttt{cdimm}} microcontroller is built on an old Linux kernel platform, which caused it to be incompatible with new systems such as Linux kernel 2.6.

During this phase, we tried out all the solutions provided in the web to solve the incompatibility problem of the \texttt{\textmu\texttt{cdimm}} microcontroller. At the end, we have decided to use instead of this microcontroller a small portable computer that is supported by a new Linux kernel (version 2.6). This portable computer will act as if it is the microcontroller of the CRYPTO-GLASSES project.
c. After making our choice to use a laptop instead of the μedimm microcontroller, we decided to move forward and start with the webcam. The webcam that we are using in this project “Logitech Pro 4000” is designed for Microsoft Windows platform only. In order to connect it to a Linux platform, we needed to search for a suitable driver that can be compatible with the kernel of Linux. After a long duration of searching and investigation, we found a driver that is under construction and non-functional. We took this driver and we started to hack it in order to complete it. During this hacking phase of the driver, we encountered many problems and difficulties, some of these problems are:

a) Some missing libraries were needed in order to compile the driver code.

b) Some functions and typos were found in the written code of the driver.

c) Problems were found during the MakeFile in Linux due to other kind of missing system libraries.

We overcome these problems by modifying the code of the driver and by adapting it to the Linux kernel after updating this latter to support such drivers.

Once the webcam is connected and installed on the Linux system, we are able now to take pictures and feed it to the image filtering program. Once the image filtering process is completed, the filtered image will be fed to the OCR where the character recognition process will take place. All these programs and processes will be executed by the Laptop.

d. After setting up the camera, we moved forward to set up the LCD screen. The LCD design is quite simple, we require a small LCD (less 6 inches) that can be connected through VGA connection with the Laptop. But why VGA?
This will be explained below in part 4.3.

4.2 The tools that were used

The tools that were used during the implementation of the solution:

**Hardware Tools**

a) **Desktop Computer**: In order to complete our project, we needed a desktop computer for research and development purposes. In other words, we used it in order to search for optimal algorithms on the web on one hand and to develop our own algorithms and application programs on the other hand. Once the programs are done, we test them before moving forward to the integrating phase where the programs are integrated all together in one complete unit.

b) **Portable Computer**: The microcontroller that was proposed in the design section was replaced by a portable computer for two main reasons:

- Serious problems were encountered during the installation of the μcdimm microcontroller and more specifically the connection problems between the desktop computer and the microcontroller.

- The supported driver and the development toolkit that came with the μcdimm microcontroller were neither compatible nor complete.
c) **Webcam**: Since our project is a vision application and image processing related field, the camera is considered one of the most important devices in the final project. Through the camera, we can take a picture of the encrypted message and then feed it to the image filtering program in order to filter out the introduced noise from the original picture. After the filtering phase was done, we feed the filtered picture to the optical character recognizer in order to extract the encrypted message and transform it into alphanumeric characters.

d) **LCD screen**: The aim of the LCD is to interact directly with the user by displaying for him the final decrypted message. The LCD screen must give the user the taste of reality while reading the decrypted output on the screen. The screen must be small, colored and with high quality resolution 800 * 600 or 480 * 640 in order to deliver a high quality picture for the end user.

e) **Crypto-Glasses** system cover: In order to integrate all the hardware components that were stated before, we needed a system cover that will help us integrate all these devices and components in one unit.

**Software Tools**

a) System tools:

- **Linux Operating System “Fedora Core 4”**: Linux is considered one of the most reliable and powerful operating systems. Unlike other operating systems like Microsoft Windows, Linux is a free source system that is rich in major features and flexible while configuring it.
• **Linux Compilers**: In any development project, the choice of the programming language is an important task. In our project, we chose the C and C++ language because they are known to be the most powerful programming languages ever known. In order to compile C or C++ code we have to use the two well known Linux compilers **gcc** (for C language) and **g++** (for C++ language).

• **Webcam “Logitech Pro 4000” driver under Linux**: In order to use the webcam on the Linux system, we must integrate the driver with the kernel and then recompile the kernel.

b) Application tools:

• **Optical Character Recognizer program**: **GOCR** is the optical character recognizer program that is responsible of recognizing the characters in a picture. In other words, the **GOCR** program is responsible of receiving an image file at the first step, and then applies optical character recognition processing techniques in order to transform the content of this image file into real characters. Once the recognition process is done, an output text file that contains all the recognized characters will be generated as a result.

• **Encryption / Decryption program**: **Vigener** program is responsible of encrypting and decrypting a certain message according to a certain key that must be shared between both the sender of the message and the end user that is trying to read the decrypted message while using the **Crypto-Glasses** system.
• **Digital Image Filtering program:** Since perfect camera doesn’t exist in our real world, it is essential to develop a program that is able to filter out the noise and light problems from the picture that is taken by the webcam. With this program we are able to adjust the following characteristic of the digital picture: brightness, contrast, alpha, enhance, sharpen the words and ext... With the help of the digital image filtering program, the optical character recognizer can now easily detect the edges of the words and as a result it will succeed in recognizing the characters most of the time.

**4.3 The difficulties**

During the implementation phase of the project, we have encountered many problems and difficulties as stated above. The main difficulties that were reported are related to the following fields:

• Image filtering block optimization and implementation
• Webcam device driver on Linux
• Microprocessor compatibility and software correctness
• Decryption header character choice
• Small LCD availability

**4.4 The solutions to the difficulties**

• **The image filtering block optimization and implementation** was a real challenge for our project. Since all picture taken from the webcam are interfered with noise that play a negative role when recognizing the
characters, we needed a certain program that is able to filter out these kinds of noises and to produce as a result a good and net picture.

In order to implement this kind of digital image processing, we began this task by using Matlab software in order to master the ability in adjusting the picture characteristics. After two weeks of research and programming on Matlab, we were able to filter out the noises from the picture. After mastering the capability of Matlab to solve such crucial problems, we faced the challenge to transform all the code from Matlab into C code where we can compile it and generate as a result an executable file that can be executed from any laptop or pc.

During the conversion period from .m files to .c files, we encountered many difficulties in reducing the huge number of libraries that were needed to complete such kind of digital signal processing program. At the end and after a lot of work and research, we have succeeded in implementing an optimal image filtering program using the C language (with minimum number of libraries).

After compiling and generating the Filter executable file, we moved forward to the testing phase of the program. During the testing period, we found out that the efficiency of our program is limited by the light in the room. To fix this kind of problem, we have implemented three modes of filtering: Night Mode Filter, Day Mode Filter and Normal Mode Filter:

a. **Nigh Mode Filter**: this filter is suitable when using the CRYPTO-GLASSES in places where there is no enough light. Since there is no enough light, this filter will increase the contrast of the taken picture in such a way that the background of the image will become totally white and the color of the characters will become black sharp.
b. **Day Mode Filter**: this filter is suitable when using the CRYPTO-GLASSES in places where there is excess of light. This excess of light is a drawback for our project since it results in some unwanted reflected lights that can interfere with the characters that must be recognized later by the OCR. Since there is excess of light, this filter will decrease the contrast and the alpha parameter of the taken picture in such a way that the effect of the reflected lights will be suppressed.

c. **Normal Mode Filter**: this filter is suitable when using the CRYPTO-GLASSES in places where there is enough light. Since there is enough light, this filter will neither increase nor decrease the contrast of the taken picture but it will apply an enhancement strategy to sharpen the characters in black in the pictures before feeding it to the OCR.

- **The Webcam driver** that was available with the Webcam device is designed for Microsoft Windows system only. But our system is based on Linux platform and therefore we have faced the challenge to re-write the driver of the webcam in order to be used under Linux while taking a picture. After a long period of research on the web, we found out a driver for Philips camera that can be useful for our Logitech camera. Unfortunately, this driver was neither complete nor correct. To overcome this problem, we tried to hack this peace of code in order to correct the typos and the mistakes of this program. During our hacking period, we found out that there are a lot of missing libraries and incomplete functions. After 2 weeks of programming and testing, we have succeeded in completing the driver and as a result the camera worked correctly on the Linux system.
• **The Microprocessor** that was designed before was replaced by a laptop for the following reason:

- Serious problems were encountered during the installation of the **μcdimm** microcontroller and more specifically the connection problems between the desktop computer and the microcontroller.

- The supported driver and the development toolkit that came with the **μcdimm** microcontroller were neither compatible nor complete. So we have contacted the support of the company by email and they sent us back the newer version of the driver because the exact version was too old and not available anymore. Unfortunately this new version driver was not compatible with the hardware and as a result, we couldn’t connect to the microcontroller.

Once we found out that the microcontroller is not working correctly due to the driver that is supported with, we took the decision of replacing it with a small notebook that can help us execute our programs in a correct and faster way.

• **Decryption header character choice** is essential in order to locate the sensitive data that needs to be decrypted. The challenge is to choose a suitable header that can be correctly recognized most of the time even with the presence of high distorting noise.

In other words this header must be the least header that can be affected by the noises that are introduced while taking the picture. The dangerous scenario can take place when the **OCR** misrecognizes the characters of the header. Once the header is not well recognized, the decryptor won’t be able to find the correct header and as a result, no correct decrypted message will be delivered to the user. After a lot of testing scenarios and tests we found out the following header “**000**” is the best for the following reasons:
- The character ‘0’ is a character that is unique. It means that even with the introduction of noise and interference, the ‘0’ will not be affected a lot by these changes. So it is the best character to be used as a header.

- If the OCR doesn’t correctly recognize the character ‘0’ it might recognize it as if it is ‘O’ or at the extreme case ‘D’. It is simple to overcome this problem, we can just add the possibilities of recognizing ‘O’ and ‘D’ in the code of the decryptor when searching for the header.

- In order to deliver the final output to the user, we need a small LCD screen facing his eyes. This LCD must be small (6 inches diagonal) and must be connected to an RGB connection (Laptop). This LCD screen is not available in Lebanon, although we found some LCDs that can be connected through audio/video terminals (analog). To solve this problem, we tried to use a VGA to audio/video converter in order to use this kind of LCD in our project. Unfortunately another problem appeared, it is a resolution problem; the laptop’s resolution is much higher than the LCD screen and as a result, a low quality image is displayed on the LCD screen.

In other words, we have tried to find an LCD screen which can be connected through RCA cables to the S-Video output of the laptop, but the problem was that Linux couldn’t communicate with the S-video port, so we obliged to use the VGA port.

But we couldn’t find an LCD screen less than 10 inches that could be connected directly to the VGA port. So we tried using an RCA LCD screen and use an RCA-VGA converter but the quality of the converter on the Lebanese market was so bad that the resolution was not properly adapted. In fact, the laptop has a higher resolution quality than the available LCDs. There
is no matching in resolution between the Laptop and the LCD screen and as a result, the displayed image on the small LCD is extremely bad.

We found LCD screen implemented inside eyeglasses that could be directly connected to the VGA of a computer, but those are outside Lebanon and cost in average $600. That’s why we decided to use a prototype that will be explained later below.

4.5 The prototype we have used

Because of all the above mentioned problems and solutions we have found to them, and because of the unavailability of an LCD screen on the Lebanese market, and the problems in the available microcontroller, we have decided the below prototype:
As we can see in the above prototype, we have a block that represents the CRYPTO-GLASSES. This prototype includes a webcam and a laptop where all the processing occurs (instead of the microcontroller).

On the other hand, the webcam takes pictures from the LCD screen of a computer, where the encrypted data is displayed. The computer can also be represented by a paper where the encrypted data is printed.

Once the picture is taken and the processing is done on the laptop, (OCR and decryption), the decrypted message is sent to the cellular phone via a Bluetooth connection. This cellular phone represents the LCD screen of the CRYPTO-GLASSES, where the decrypted message is displayed.

So in this prototype we have the following table which describes which component in the prototype represents which component in the actual CRYPTO-GLASSES:

<table>
<thead>
<tr>
<th>Prototype Component</th>
<th>CRYPTO-GLASSES Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>Webcam</td>
<td>Webcam</td>
</tr>
<tr>
<td>Laptop</td>
<td>Embedded microcontroller</td>
</tr>
<tr>
<td>Cellular Phone</td>
<td>LCD screen</td>
</tr>
</tbody>
</table>
5.0 Evaluation of the project

In this part of the report, we will critically appraise the results of our project. We will first start by indicating the underlying principle for the design and implementation. We will then move on to stating and explaining the lessons we have learned during the course of this project. We will end this part by an assessment which will be based on a comparison with the objectives stated at the beginning of the report.

5.1 The underlying principle for the design and implementation

The design we have chosen to complete our project is a result of a whole process that underlies behind it. This process enabled us to arrive at the final design scheme by taking care of all possible alternatives for the design. In other words, the underlying principle behind the design is the “critical comparison and assessment technique” followed in a lot of engineering practices.

This technique starts by stating all the possible design solutions for the parts of the system and then for the system as a whole. The possible alternatives are objectively analyzed, assessed and compared, in order to arrive at a result that would give the most optimal design solution.

---

**Figure 5.1**

Component - System interaction

Component or part design

Overall system design
As we can see in the above figure, the system design is broken into two different parts: “Component or part design”, and “Overall system design”. In their turn, each of the previous two parts is broken into different parts as follows:

### 5.1.1 Component or part design

In this phase of the design, each of the components that constitute the overall system is independently designed. This design considers all the possible choices or alternatives that can be used in order to arrive at the optimal design of this part. To emphasize more on this point, we will start by stating the different components that constitute our product:

- Decryption block
- Camera block
- Filter block
- Optical character recognition block
- Displaying block

For each of the previous blocks, the alternative solutions that contribute to the realization of the function of the block are compared, in order to select the most suitable one, **given the constraints on our project.**

#### a. Camera block

Here, we have the following design goal, and the associated constraints:

**Goal:** Take a picture of the data.

**Constraints:** The picture must be as clear as possible.
As we have said in Chapter 3.1.2, there are two possible cameras that can be used: Logitech QuickCam Pro 4000, and iREZ K2 Camera. Given the constraints, and as said earlier, the Logitech QuickCam Pro 4000 camera is selected.

b. Filter block

Here, we have the following design goal, and the associated constraints:

*Goal:* Filter the image captured by the camera.

*Constraints:* The filtering process should output a clear picture, with minimal or no noise.

As we have said in Chapter 2.6, the filtering process used consists of improving the image contrast and enhancing it using the Linux libraries.

c. Optical character recognition block

Here, we have the following design goal, and the associated constraints:

*Goal:* Extract the text from the picture.

*Constraints:* The OCR process must be accurate and with minimal errors.

As we have said in Chapter 3.1.3, there are two possible OCR algorithms that can be used: GOCR and OCRAD. Given the constraints, and as said earlier, the GOCR algorithm is selected.

d. Decryption block

Here, we have the following design goal, and the associated constraints:

*Goal:* Decrypt the data that is useful to the user.
Constraint: Be able to determine the useful data, and ensure a correct decryption.

As we have said in Chapter 3.1.1, there are two possible decryption algorithms that can be used: Vigenere algorithm and AES algorithm. Given the constraints, and as said earlier, the Vigenere algorithm is selected.

e. Displaying block

Here, we have the following design goal, and the associated constraints:

Goal: Display the useful information to the end user.

Constraints: There are no real constraints here.

As we have said in Chapter 3.1.5, there are two possible OCR algorithms that can be used: Colored LCD screen, and small characters LCD screen. As said earlier, the colored LCD screen is selected.

5.1.2 Overall system design

Now that we have designed each of the blocks that constitute the final product, the next and more important task is to design the product as a whole.

This task is more difficult because it must take into consideration the dependencies between the different blocks. In fact, those dependencies can heavily affect the overall system design and integration.

Let’s draw a table that shows the dependencies between the different blocks:
As we can see in the above figure, each block depends on those that are before him. For instance, the filter block depends on the camera block, while the OCR block depends on the filter block (and thus also on the camera block).

All those dependencies should be taken into consideration when designing the overall product design.

To emphasize more on this point, we will show the dependencies between the different blocks:

- The filter block depends on the camera block because the filtering process depends on the quality of the captured image and the level of noise involved in it.

That’s why we thought about dealing with this dependency by providing the user with three different modes of operation to be set in order to allow a better filtering process:

1. **Day mode**: This mode is used in case of intense light in the environment where the CRYPTO-GLASSES are used.
2. **Night mode**: This mode is used in case of minimal light in the environment where the CRYPTO-GLASSES are used.
3. **Normal mode**: This mode is used in case of normal light in the environment where the CRYPTO-GLASSES are used.
The OCR block depends on the filtering process. In fact, the clearer the image at the output of the filtering block, the more accurate is the OCR process. In particular, we tend to give a high level of importance to detecting the important data to the user through the OCR process. This obliges us to be able to clearly detect the tags that distinguish this important data from other not useful data. That’s why the filtering process should as much as possible enhance the picture and improve its contrast to make the OCR process more accurate.

The decryption block depends also on the OCR block. This is a trivial dependency because if the OCR process renders wrong characters, the decryption process will simply be wrong.

Ensuring a correct decryption process is thus ensuring a good filtering and OCR processes.

Finally, the displaying process depends trivially on all the blocks behind it. If any of the blocks does not function as it should, the displaying process will be distorted.

5.1.3 Overall system implementation

While the system design consisted in designing the structure of the product that can realize the specifications and the goals stated at the beginning of this project, the system implementation consist in translating this structure into a tangible product that can be used in order to operate all the functionalities stated in the design process.

Here, the underlying principle is to translate the design diagram that was conceived into a hardware product. The ultimate goal is to design the CRYPTO-GLASSES and be able to use them in real life applications.

Concerning this part of the project, we have actually starting by implementing each of the different blocks separately, based on the “overall system design” and the
dependencies between the blocks and we have then combined them together in order to finalize the product.

But, before actually building the product, we have tested it using a prototype. Our prototype consisted of using a laptop where all the software (OCR, decryption, image capturing software) were installed. The Logitech camera was connected to that laptop using a USB port. The prototype systems functioned as follows: The camera took the picture of the useful data. The captured picture is then processed on the laptop using the filter, then the OCR and finally the decryption. The final step is to display the decrypted data on the laptop screen of a mobile phone via a Bluetooth connection (which represents in this case the LCD screen in front of the eyes of the user).

5.2 The lessons we have learned during the course of this project

Throughout the course of this project, we had the opportunity to learn many lessons. If we combine all the lessons together, we would get the result of learning how to use an “engineering approach” to solve our problem.

We are going to list those lessons and emphasize on each of them independently:

1. **Learning the goal of such projects**

   The most important lesson we have learned during the course of this project is the following: “process” is as important as “product”. In fact, knowing how to develop the desired product and understanding the steps involved in this development is as important (or we could even say more important) than developing the product itself.

2. **Applying our knowledge in practical applications**

   One of the important skills that we have acquired was using the knowledge we have accumulated during our studying period at the university, in a real life project that
could be used in practice. Thus, we have learned how to apply our theoretical knowledge in practical issue.

3. **Searching for useful sources of information**

   Another important lesson is get to know the ways of finding relevant sources of information that could help us in the accomplishment of our project. Such ways enabled us to distinguish high quality information from low quality information through the place of publication of this information, the number of readers, the reputation of the author(s), the references…

4. **Setting and studying the requirements of our project**

   We have also learned how to set the objectives we have to attain at the end of this project and work in a way to accomplish these objectives on time and on schedule. This was done after a thorough study of the requirements and their relations with the constraints.

   This is a crucial step that should be followed in all projects. In fact, studying the requirement enabled us to know if the project can or cannot be accomplished using the current technology, the available time limits and budget, along with the resources (in terms of number of people working on the project, which is in our caser three).

5. **Considering all the possible options to solve a problem**

   While executing this project, we have tried to come up with the best solution to the problem, by making sound arguments on why we have chosen a particular design and not the other one. In fact, we have stated all the possible solutions or alternatives to solve our problem at the component level and at the overall system level. The alternatives were objectively compared and assessed in order to arrive at the chosen final system design that would be the best solution to solve our problem, give the constraints.
6. Acquiring good planning skills

Throughout this project, we have always been planning our work. We have learned that planning is one of the most important practices during a project. In fact, if planning is not accurate, the project may face a failure. Although planning depends somehow on the intuition of the project manager who does the planning, it is also dependent on material facts, such that the constraints, the risks, the resources…

7. Being ready to face project risks

During the course of our project, we have faces many obstacles. From hardware problems, to hardware not available in the markets, to software malfunction…, the project was in a lot of times at the risk of failure.

But through the planning skills we have acquired, we always knew how to deal with the risks in order to either get over them or minimize their effects.

For example, we could state that finding an LCD screen with dimensions in the nearby of 5’’ was a very difficult task. In fact, we find some LCD screens that could be connect to the S-video output to our laptop, but we couldn’t use them because the S-video cable for our laptop was missing in the market and because no driver was available on Linux for switching the view to the S-video operation mode.

5.3 Evaluation of the product and process

In this part, we will evaluate the product and the process. There are two different ways to evaluate the project: either by comparison of what was done with the project objectives stated at the beginning and in the course of this report, or with what someone else did to meet these objectives.
In our case, we will use the first method to evaluate our project because this is a new project: we will compare what we have achieved at the end of the project with the objectives stated at the beginning and in the course of this report. There are thus two comparisons we should make: comparison of the realized project with the project objectives, and comparison of what we have done with the process objective. Project objectives involve mainly what we have done, while process objectives involve how we have done the project.

a. If we want to reformulate the project objectives, they will be as follows:

1. **Design and build a CRYPTO-GLASSES device.**

   This primary objective was not completely realized. We have actually built a prototype of the CRYPTO-GLASSES device that can be used to ensure the privacy of the information by displaying it to the end user only. In fact, we were obliged to do this because the LCD screen was not available in the Lebanese market and its cost was big to import it from outside Lebanon. So we have used a Bluetooth connection to a mobile phone where the decrypted information will be displayed: this solution replaces the LCD screen. The other alteration is that we have actually used a laptop (portable computer) instead of the microcontroller because this latter was not functioning. So for further development, all software that is being used through the laptop can be installed on a microcontroller device representing an embedded system.

2. **Ensuring that the displaying process is only seen by the end user to ensure the privacy of reading important data.**

   This is a crucial objective. The main objective of our project is to let the end user read from a computer screen or from a paper some encrypted and classified data. This objective was clearly met because the CRYPTO-GLASSES device that we have built was designed in such a way to let only the end user see the LCD screen
in front of his eyes. This objective is also met using our working prototype because the information is displayed on the mobile phone of the user.

3. **Realize the project within the budget, time, and resources constraints.**

   This objective was also met. We have finished conceiving the final product and developing its prototype before the deadline on May 23rd, using an affordable budget, and without need of outside staff resources.

b. If we want to reformulate the **process** objectives, they will be as follows:

1. **Ensure a good captured image.**

   This objective was met by: using a good quality webcam, and by using a good image filtering strategy.
   In fact, the Logitech QuickCam Pro4000 is one of the best webcams available in the market. The quality of its picture is very good and its resolution is also acceptable. So the pictures that we have been capturing using this webcam were of good quality.
   On the other hand, the filtering stage that we have added was very useful because it significantly enhanced and improves the quality of the captured image, thus making it ready for the text recognition process.

2. **Ensure a correct recognition of the encrypted information.**

   This objective was met by using a good and “well-reputed” optical character recognition program over Linux, which is the GOCR program.
   Together with the image filtering process which removed the noise inside the captured image, the GOCR program was very accurate in extracting the text from the filtered image.
3. **Ensure a correct decryption of the encrypted information.**

Because the GOCR program was performing a good recognition, the tags that separated the useful information were *almost* always well recognized. Thus the useful information was well determined and was decrypted using the Vigenere decryption algorithm, thus making this objective met.
6.0 System Design Constraints

As it is stated in the ECE undergraduate program outcomes, one of them is "an ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability."

That’s why, in this part of the report, we will assess our system in terms of all the above stated constraints.

6.1 Economic constraints

As we have stated in the design part of this report (specifically in part 3.3), the budget needed for the accomplishment of this project is around $1045.99. In terms of economic figure, this amount is somehow big but we should take into account that it includes an LCD screen that costs around $500, and a microcontroller that costs around $300. Those costs are those for items bought as individual parts, so if the product is to be designed at factory level, the cost will be easily reduced by 20%, thus a cost of around $850.

This cost is reasonable in terms of the processing done by the system. In fact, if we take a look at eyeglasses on the market which only enable the user to view the computer screen in private (with virtually increasing the size of the screen to around 40 inch sometimes), those cost in average $600.

So, compared to the tasks that can be done by our system, a cost of $850, those around $250 more expensive then the above stated system, the cost of our system is reasonable.

6.2 Environmental constraints

Our system is not really influenced by environmental constraints. It is a system that directly interacts with the system user and provides him with private reading of confidential data.
On the other hand, only one environmental factor related to the brightness and the level of light in the surroundings, will affect our system. However, we dealt with this as said before by choosing three types of usage: day mode, night mode, normal mode.

6.3 Social constraints

Our system provides privacy to an end user who wants to read encrypted confidential data from a computer screen or from a paper, in a public place. Thus, through the functionality provided by this system, no social constraints affect it.

6.4 Political constraints

No political constraints affect our system because there is no political involvement in it. Although it can be used by political departments to save their important data, there are really no constraints on using it, except the conservation of the encryption key, which allows the access to using the system.

6.5 Ethical constraints

Our system abides fully by ethical values. There is no functionality in the system that is limited by ethical considerations because our system is completely ethical.

6.6 Health and Safety constraints

Our system does not endanger or affect the health or the safety of its user or the people around him. The only thing we should take into consideration is that the fact that the LCD screen is somehow close to the eye of the user may be a little bit difficult to adapt to during the first usage, but this will not affect the eye once it is used to the usage of the system.
6.7 Manufacturability constraints

All the parts used in our system are available on the market (although not all of them are on the Lebanese market, such that the LCD screen). So it is a little bit difficult to manufacture the system in Lebanon because of the missing parts, but the manufacture process is completely doable in Europe, Eastern Asia, and America…, where all the parts are available.

6.8 Sustainability constraints

Our system is completely sustainable. In fact, although it does only need to be maintained once in a while, maintaining it is easy, and even replacing the bad parts (if they exist) is easy.
7.0 Conclusion

As a conclusion, we will first review what we did during this project and in this report and then state the final outcomes of this project.

In this report, we started by defining the problem, presenting its benefits and applications and showing how useful it is. We then presented the design approaches we could use in order to produce the final product, and examined all the alternatives for all the components of the product. In the third part, we studied the design of the product, in terms of deciding which of the alternatives we should choose for each block, and explaining how the different parts of the product will coordinate so that they can deliver the desired output. We then moved to the fourth part where we explained the implementation and testing of the product, explaining all the problems that we encountered and the solutions that we found when applicable. The fifth part included the overall evaluation of the project, including the process and methodology that were followed. The final and sixth part studied this project in terms of the constraints that affect its operation and that were taken into account when designing it, such that the economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability constraints.

During this semester, we faced a lot of problems while trying to design the product, because of many reasons, including:

- The LCD screen was not on sale in the Lebanese market and importing it from outside costed a lot of money.
- The microcontroller that was available in the labs at the university had its software missing, and it couldn’t communicate with the computer in order to be able to configure it.

That’s why, we decided to conceive a working prototype of this product, which would perform all its functionalities and prove the correctness of the applied concepts. This prototype included a mobile phone where the decrypted message is displayed (instead of the LCD screen) and a laptop (instead of the microcontroller) where all the data processing occurs.
References


Appendix

In this appendix we will show two of the testing results that we have obtained:

1. **Testing 1:**
   
   The picture initially taken by the camera was:

   ![Initial Picture]

   The picture after filtering became:

   ![Filtered Picture]
The output after OCR processing is: 000 ismm2 xp5me 000
The output after decryption is: hello world
The correct output is: Hello World

2. **Testing 2:**

Other testing of course included some part of the output including wrong characters, either because the recognition was not good or because the image quality (even after filtering) was not very much enhanced. This however occurs in extreme cases. In the following example, a bad CRT monitor was used and the picture quality was bad, so we obtained the following:

The picture initially taken by the camera was:

![Initial Picture](image-url)

The picture after filtering became:
The output after OCR processing is: 000 ;smm2 xp$5me 000
The output after decryption is: ello world
The correct output is: Hello World