EE 389 Electronics Design Lab Project Report, EE Dept, IIT Bombay, November 2008

Optical Data Encoding

Group No.: D1

Ujjwal Kumar (05002041) <ujjwal@ee.iitb.ac.in>
Arun Chaitanya (05D07039) <arunchaitanya@ee.iitb.ac.in>
Khem Raj Ghusinga (05D07041) <khem@ee.iitb.ac.in>

Supervisor: Prof. Dipankar

Abstract

This report is about the project done for the course EE 389. The aim was to develop an imaging device which can be used to decode and process data encoded in the image. We have been able to write an algorithm for decoding the data after the image is available and the same has been implemented and tested on MATLAB. A rather basic form of the algorithm is implemented on the microcontroller and has large possibility of improvement. The other major part of taking image, we have not been able to achieve. The basic design procedures followed, circuit diagrams and components etc. are discussed in following sections.

1. Introduction

The project aims to develop a small cell phone like device which has a camera and LCD display. Printed images of encoded data can be picked up by the device, and then suitably run on the hand-held unit. The data could be MP3 files, programs, image-data, slide-shows etc. The idea of this project is to allow a conventional and mature media- that of printing on paper/books, to allow one to access multimedia with a small hand-held device. Potentially these strategies can be incorporated in normal cell phones.

2. Design Approach

2.1 **Hardware**

The hardware part consists of following components:

- Image Capturing Device
- External Memory Interface
- LCD Display
- Microcontroller interface with these components

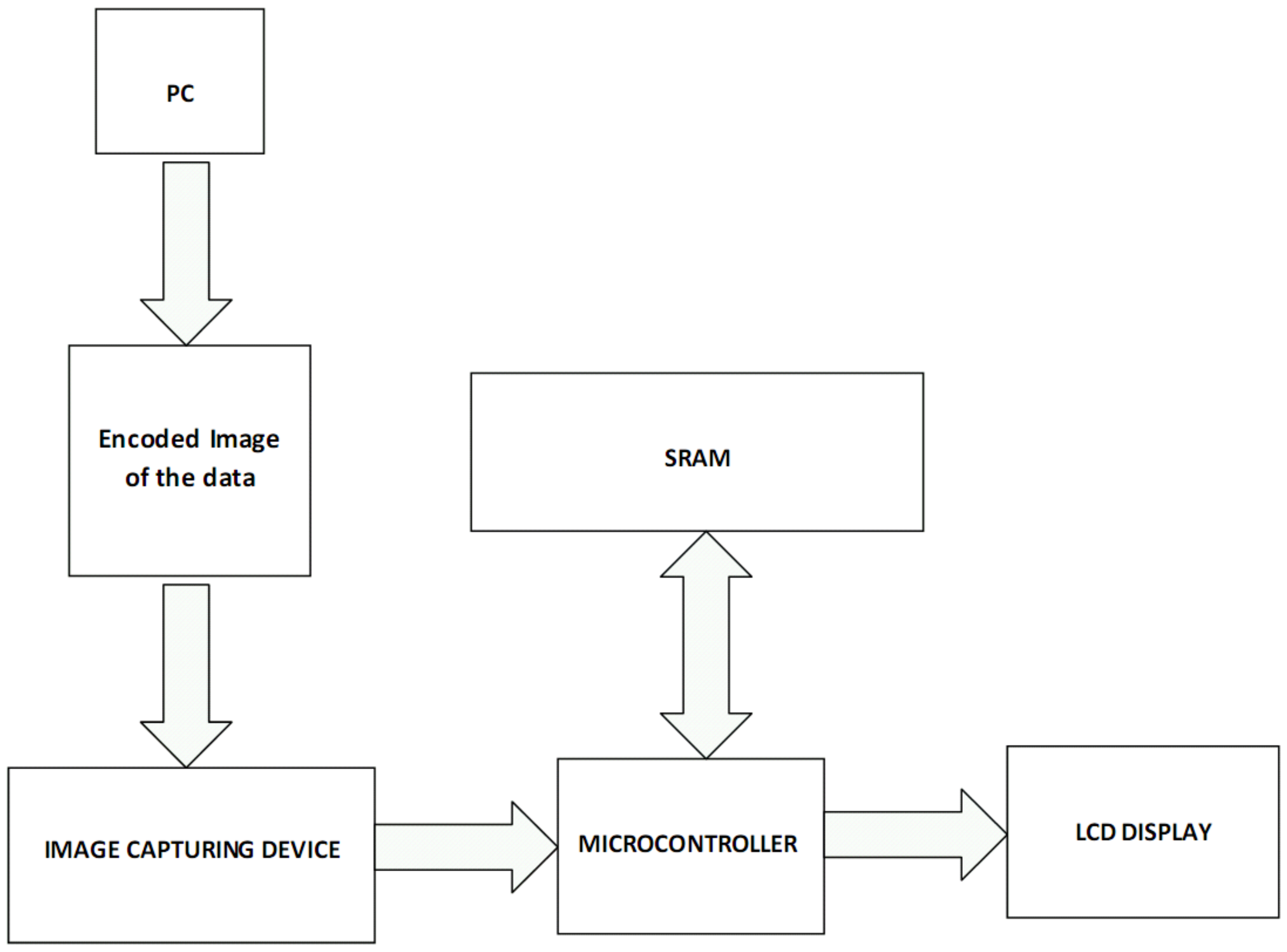


Figure: Block Diagram of the Hardware Module

2.1.1 Image Capturing Device:

This was the toughest and most crucial part of the project. It consumed most of our time. In EDL-I, we had tried to use Image Sensor (KAC9619) to capture the encoded image. But, we couldn't make it to work. In EDL-II, we wanted to use Webcam for capturing the image. As USB Webcam modules are widely available, our main task was to make a USB host controller using the microcontroller, so that to be able to work with USB device (Webcam). We worked on this part till one week after the mid semester examinations but found it quite hard to implement. Then, we came across a parallel port webcam. It was a Windows 98 compatible Webcam and we couldn't use it because drivers its compatible were not available. We tried using this module on another Operating System (Linux) but couldn't really make it to work.

2.1.2 **External Memory Interface:**

As the data was quite large, an external memory interface to the microcontroller was required. We have successfully completed this part using a CY 62256 SRAM.

2.1.3 **LCD Display:**

A LCD display module is used to show the data which was transferred.

2.1.4 Microcontroller-to-Computer Data Transfer:

We have used RS232 with MAX232 to transfer the data between computer and microcontroller. The baud rate we have used is 4800.

2.1.5Choice of Components

We were to use ATmega32 which has required speed to interface image sensors, but finally to implement the algorithm we have used the AT89C55 microcontroller, which was chosen largely due to larger program memory.
CY62256 was used as external memory which has 65 Kbytes of memory, sufficient for images we have used.
A 16 x 2 LCD Display was used to display the data, due its wide availability and easy interfacing.

2.2 Software

The software part consisted of two parts - Encoding and Decoding.

2.2.1 Encoding

Wrote a C program to generate the encoded image from a given text file. This program takes a text file as input and creates a file with binary values, using which a postscript file is generated to print the encoded image.
The encoded image consists of white and black blocks corresponding to a '0'and '1' data bit respectively. The image has 'start' bit indicating start of the data and header information consisting of number of bits and data block number.

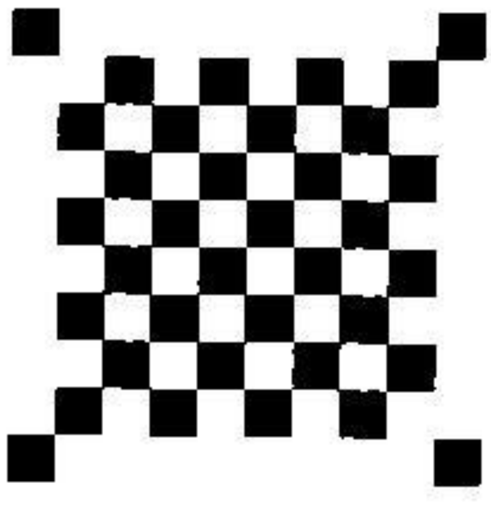


Figure: A typical *encoded* image

2.2.2 Decoding

The Algorithm used to decode the image taken from the image sensor works for the case when the rectangular block consisting of data in the image is tilted up/down by 45 degrees (which is sufficient for all practical cases).
The steps of Algorithm are as follows:

a) Let p=position of the first black pixel and q= position of second black pixel. Scan the image row-wise and note the first appearance of black pixel (‘p’).

b) Scan in columns > 0.7 x Total number of columns. Note the first appearance of black pixel (q_1).

c) If $p \neq q_1$, the image is clockwise rotated. Using values of p and q, we can calculate the angle of rotation Ø. Now we have to move in Ø direction for scanning the rows and get the data.

d) If $p=q_1$, the image is anticlockwise rotated. In this case, scan in columns < 0.3 x Total number of columns and find appearance of second black pixel. Let the position be q_2. Then we interchange q_2 and p, and find Ø from them and then proceed in the same way as above.

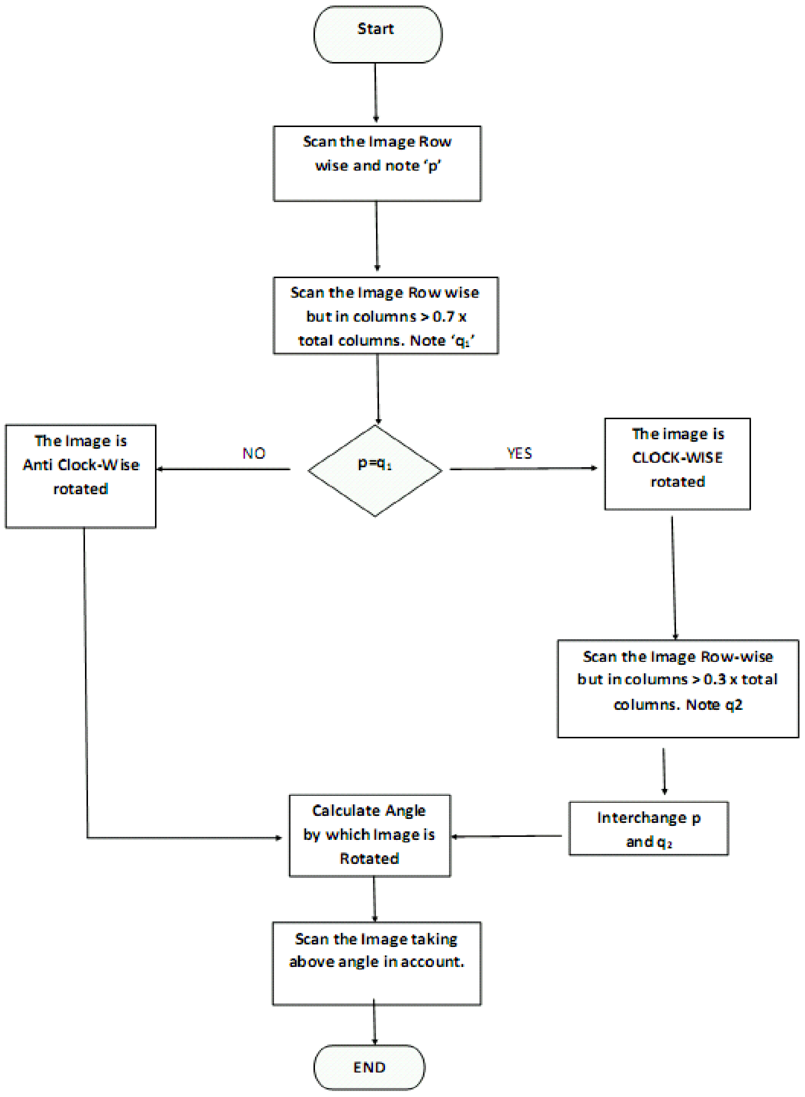

Start
Scan the Image Row wise and note ‘p’
Scan the Image Row wise but in columns > 0.7 x total columns. Note ‘q₁’
p=q₁
NO
YES
The Image is Anti Clock-Wise rotated
The image is CLOCK-WISE rotated
Scan the Image Row-wise but in columns > 0.3 x total columns. Note q2
Interchange p and q₂
Calculate Angle by which Image is Rotated
Scan the Image taking above angle in account.
END

Figure: Flow Chart of the Decoding Algorithm

3. Design of Circuit

The presently working module of our project consists of Microcontroller (AT89C55) with external memory interface (SRAM CY62256) and a LCD display. It takes input image file from the computer, implements the decoding algorithm on it and displays the decoded data on LCD.

The schematic of circuit diagram is shown below:

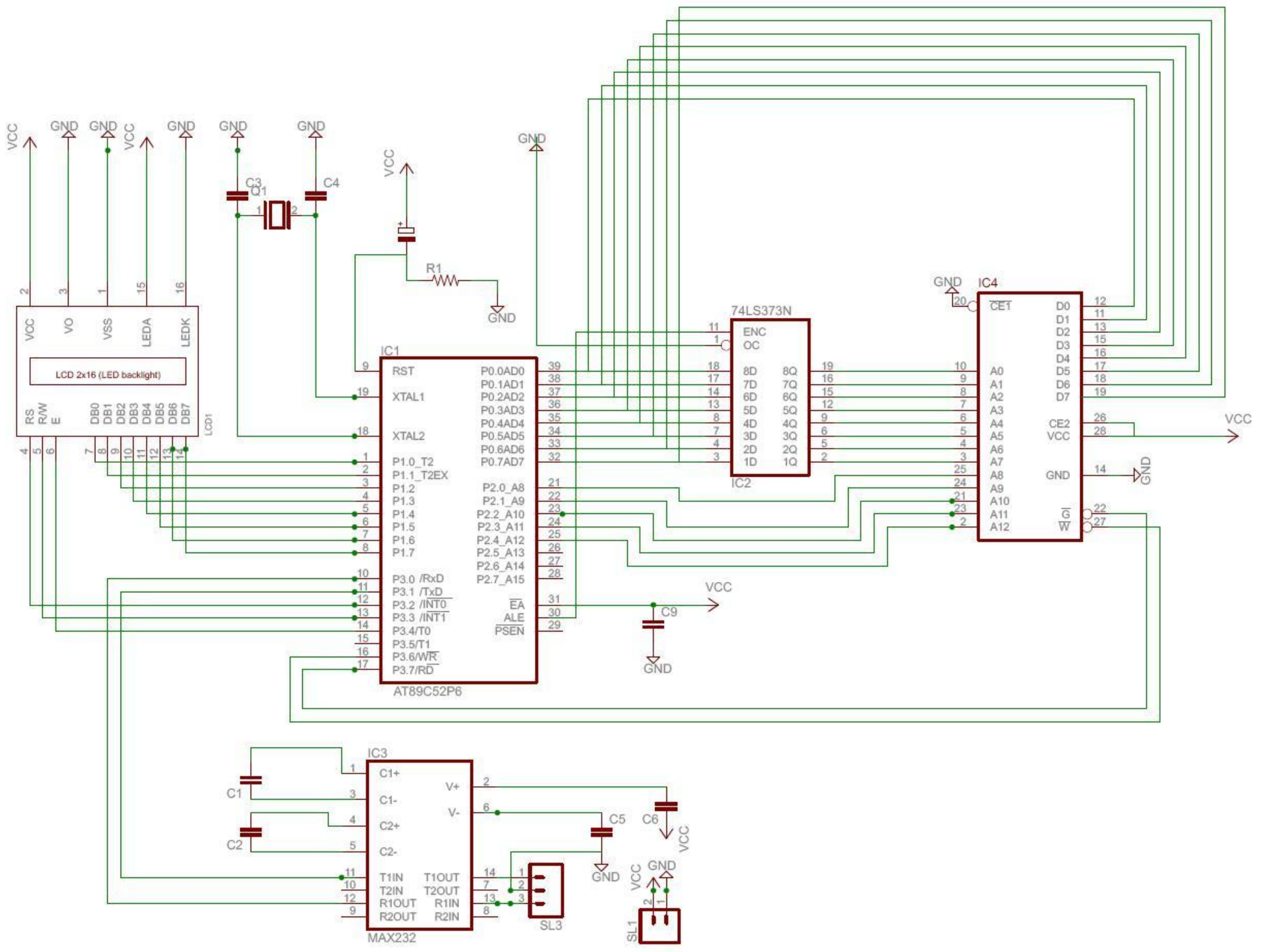
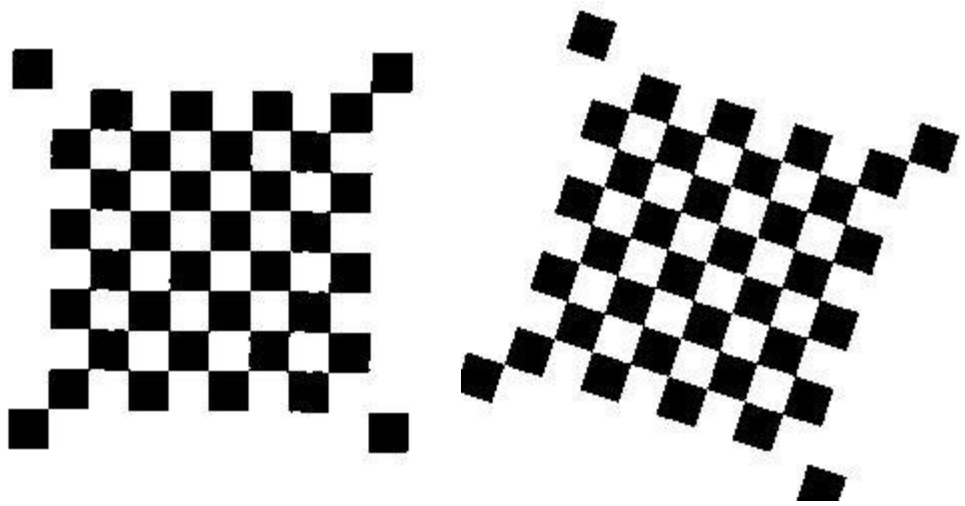


Figure: Circuit Diagram

4. Testing Procedure

Testing of decoding algorithm program in MATLAB was done using several test images as shown below. The code was able to detect the pixel values correctly even if the image is rotated by some angle ($< 45^o$ is sufficient for all practical purposes)



.

Figure: Test Images for Decoding Algorithm

For implementing the algorithm using the microcontroller, we have created images using MATLAB (160 x 160 pixels), transferred them to the Microcontroller using RS 232, MAX 232 Serial data transfer.

5. Results, Limitations and Possible Enhancements

We have successfully implemented the decoding algorithm on the microcontroller, which works well for basic case of image which does not have any noise and is not rotated. The image for testing algorithm was generated by MATLAB. We have tested using 160 pixel x 160 pixel images, but it can be easily scaled to any size depending on the SRAM, image is ideal.
Limitations are for the case where image is rotated or there is some scattered noise bits present in the image, we could check the algorithm for that case.
Possible enhancements would be inclusion of various non-ideal images in the algorithm and interfacing an imaging device to capture data feed to the microcontroller rather generate it artificially.

6. Conclusions

We set out using an image sensor and USB webcam as the imaging device which we could not interface. This part of project remains not done. We have implemented the decoding algorithm on Microcontroller, which is in basic stage and needs a lot of improvement to tackle real world cases.

References:

[1] Peter Weingartner, *A First Guide to PostScript*
[2] Adobe Systems Incorporated, *PostScript Language Reference, third edition*
[3] Jan Exelsion, *USB Complete, third edition*
[4] AVR USB Example Projects Website (http://www.obdev.at/products/avrusb/index.html)
[5] http://www.beyondlogic.org
[6]Mazidi & Mazidi, *The 8051 Microcontroller and Embedded Systems*
[7]Datasheets of all the components used

Appendix

A. User’s manual

The image encoding can be done using either the C++ code. The data to be encoded is put the file ‘*text.txt*’ and after running the C++ codes, the output data is ‘*out.ps’*. It can also be done using MATLAB file ‘*create_image.m’*.
To transfer the image data serially from MATLAB use the file ‘*serial_transfer.m’*.
The decoding algorithm can be run from the Microcontroller.