CHAPTER 1
INTRODUCTION

1.1 Introduction

Toll gates are usually considered as difficulty for travelers not only the cost of the toll gates, but also for the time delay at entrance of toll gates. So this drawback can be resolved while keep up a convenient toll gate system. For this Security systems can also be added, which will further improve the system.

The toll gate system using ANPR technologies facilitate the electronic collection of toll payments. This technology has been considered by researchers and applied in various highways, bridges, and subways requiring such a process. This system is having ability of determining if the car is recorded in the database or not, and then informing the authorities of toll payment destructions, debits etc. The most evident advantage of this technology is the opportunity for remove jamming in tollbooths, especially during joyful seasons when the traffic tends to be heavier. It is also a method by which to control complaints from vehicle drivers or owners regarding the problems involved in manually making payments at the tollbooths. Other than this noticeable advantage is that it could also benefit to the toll operators.

The benefits for the motorists through this system can be fewer queues at toll plazas, faster and more efficient service (no transferring of toll fees by hand), and the ability to make payments by keeping a balance on the registered credit card. The use of postpaid toll statements (no need to request for receipts). Other general advantages for the motorists contains fuel savings and reduced mobile emissions by eliminating decelerations, waiting time, and acceleration. Meanwhile, the benefits through this system are poorer toll collection costs, better audit control by centralized user accounts, and increasing capacity without building more infrastructures.

AUTOMATIC license plate recognition (LPR) plays a main role in several applications such as unattended parking lots security control of confined areas traffic law implementing blockage pricing and automatic toll collection. Due to the different working environments, LPR techniques vary from appliance to appliance. Pointed cameras create dynamic scenes when they move, criticize or zooming. A dynamic scene image may be contains multiple license plates or no license plate at all. So, when they do appear in an image, license plates may have random size directions and positions. And, if complex backgrounds are exists, then sensing of license plates can become quite a challenge. Usually, an LPR process consists of two main stages (1) locating license plates and (2) identifying license numbers. In the first stage, license plate candidates are determined based on the features of license plates. Features commonly employed have been derived from the license plate format and the alphanumeric characters constituting license numbers. The features regarding License plate format include shape, balanced height-to width ratio color texture of grayness spatial frequency and variance of intensity values Character features include line splash the sign transition of gradient magnitudes, the aspect ratio of characters the distribution of intervals between characters and the alignment of characters. In reality, a small set of healthy, reliable, and easy-to-detect object features would be enough.

CHAPTER 2

EXISTING SYSTEM

The existing system includes controller, RFID reader, RFID Tag and stepper motor. The readers retrieve the information about the ID number and recognize the vehicle. Then for the tax to be collected and the bill is printed at the time of exit. Here the stepper motor is used to open and close the gate automatically.

In this system, although there is an RFID reader, the tax collection is manual and it will not automate. There are none of security features such as identifying the stolen vehicle etc. The toll tax is collected for all the vehicles and it is the same and the tax is collected according to the weight of the vehicle.

2.1 Existing System based on ANPR:

2.1.1 Character Segmentation:

To segment the characters in the binary license plate image, the method named peak-to-valley is used. In this the first segments the picture in digit images getting the two bounds of the each digit segment according to the statistical parameter DIGIT_WIDTH = 18 and MIN_AREA = 250. For this purpose, it uses a recursive function which uses the graph of the sums of the columns in the LP binary image as shown n Figure 2.1. This function shows in the graph from left to right, bottom-up, incrementing at each recursive step the height that is examined on the graph as showed by the red lines on Figure 2.1 if it is greater than DIGIT_WIDTH, then the function is recursively called after incrementing the height which is examined on the graph, as illustrated for the segments corresponding to the number 2, 7, 1, and 8 on Figure 2.1. Or else, if the bandwidth is good, the two bounds of the signal with this bandwidth are taken as a digit segment, and the function is recursively called for the part of the image which is at the right side of the digit segment. This is done until the whole width of the picture has been passed over.

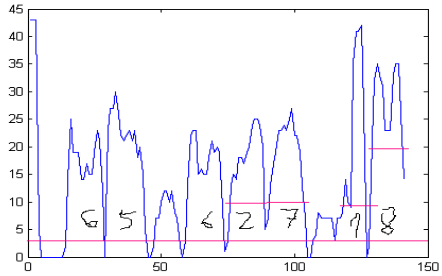


Figure2.1: Characters segmentation - Columns sum graph

Once this segmentation has finished, the method keeps in the result only segment for which the area of the smallest rectangle containing them is more than MIN_AREA; then, it keeps only the seven segments in the result with largest area, and in case less than seven segments were found, it attempts to recall the whole method, after making the separation between the already found segments clearer (by cleaning the bits which are there).

Figure2.2: The digit images obtained by the character segmentation machine

2.1.2 Rationale:

License plates have always clear signature which corresponds to strong white level variations at somehow "regular" intervals. Due to noises, as shown in Figure 2.1, the variations are not always ideal and our algorithm permits to repair those variations. The method proved to be very accurate.

2.1.3 Possible Problems/Weaknesses:

In some rare cases, digit may be cut or two digits may appear in the same segment; this is especially the case when the image is blurred due to motion or when the contrast of the LP is very poor.

2.1.4 Existing method of ANPR

- Color and character features based license plate extraction
- It consists texture type of segmentation

2.1.5 Drawbacks of existing of ANPR

- It having sensitive to noise and it may leads error during detection.
- Computational complexity and time taken will be high.

CHAPTER 3
PROPOSED SYSTEM

3.1 Block Diagram

The proposed system makes sure that the traffic at the toll gates make more efficient and high security is also present. The tax which is collected according to weight of the vehicle. Through this system we can also identify stolen vehicles.

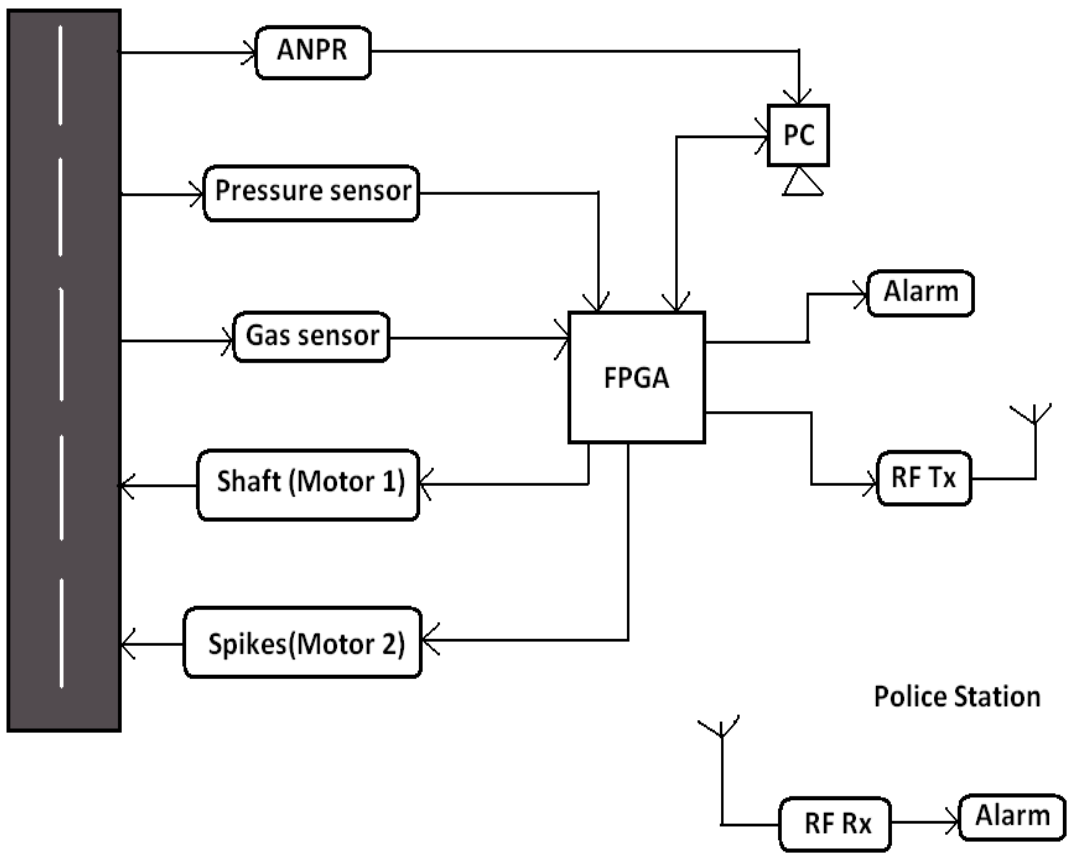


Figure 3.1 Block Diagram of proposed system

3.1.1 Description of Proposed System:

Figure 3.1 shows the block diagram of proposed system. Then the ANPR system compares the input number plate image with the database. PC already consists of the entire database which is updated continuously. Now, after reading the information in database, PC compares the recognized number plate data in the database and allows the access accordingly by opening/closing the gate. This data is used to print a daily or monthly bill for toll collection from the vehicles. This way even stolen vehicle can be also identified.

The pressure of the vehicle is obtained using the pressure sensor and accordingly the weight of the vehicle is displayed on the screen. A counter is used to count the number of vehicles which are passing through the toll gates. The amount on the basis of weight & the count of vehicles is also displayed on the screen. The amount to be paid is automatically deduced from the respective bank account.

If a vehicle carries any kind of gas, the gas sensor detects the gas in the vehicle. Else if there is any kind of gas that is detected, the RF transmitter is used to alert the near police station and an alarm is get enabled to alert the all nearest areas. At the same time motor 1 is used to close the gate and simultaneously motor 2 is used to pull up the spikes in order to damage the vehicle. The flowchart for the entire process is shown in figure 3.2

3.2 Overall Flowchart:

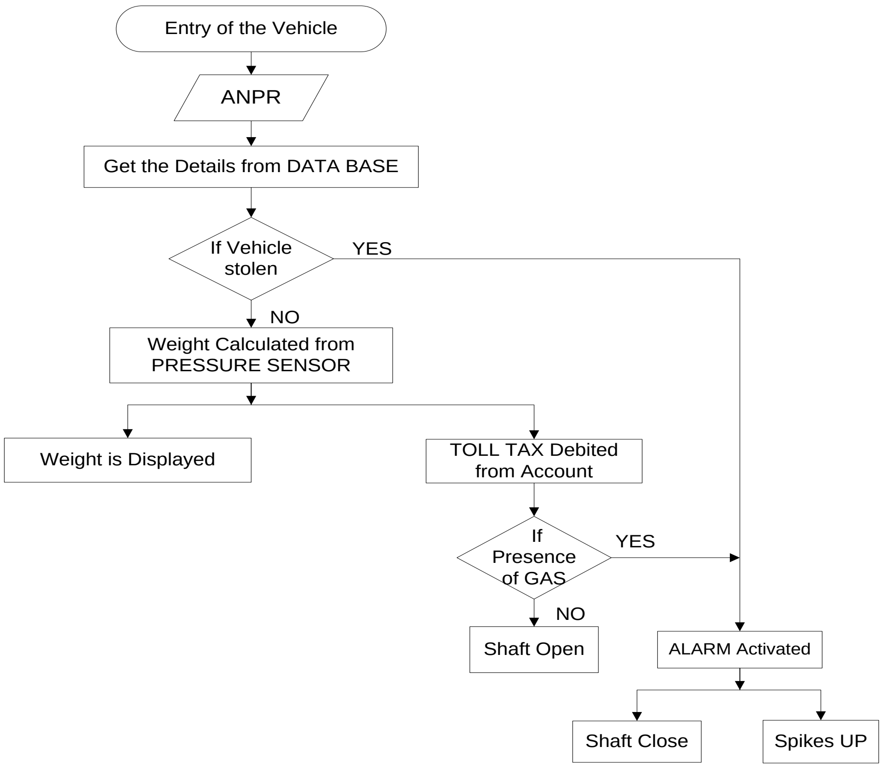


Figure 3.2: Overall flow chart

CHAPTER 4

ANPR SYSTEM

4.1 BLOCK DIAGRAM

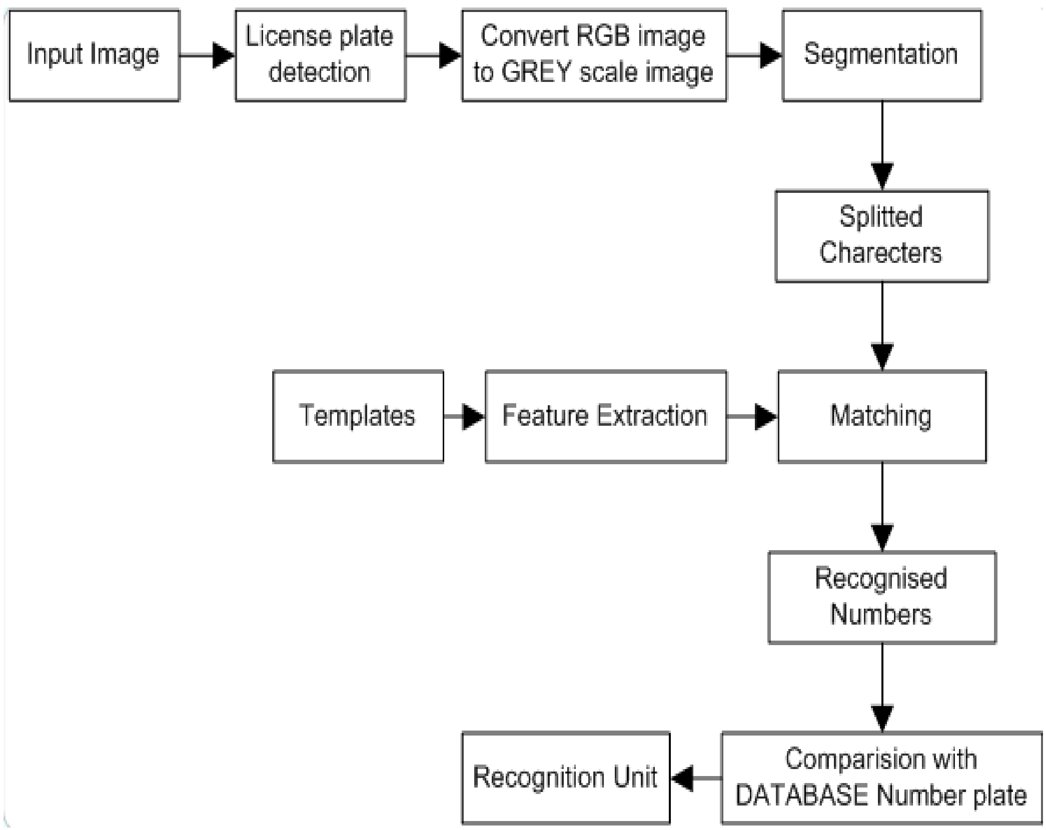


Figure 4.1 Block Diagram of ANPR System

4.2 Segmentation

License Plate Detection

The objective of image segmentation is to gather the pixels into most significant image areas, i.e., regions similar to individual areas, objects, or natural elements of objects. In computerized segmentation refers to the method of changing a digital image to multiple segments. The goal of segmentation is to form easy and/or modification the illustration of a picture into something that's extremely meaningful and simple to analyze the Image segmentation is often used to find objects and boundaries (lines, curves, etc.) in pictures. Image segmentation is that the method of allocating a label to each pixel in a picture such pixels with the same label shares certain visual characteristics.

The results of image segmentation may be a set of segments that collectively cover the complete image, or a collection of counters extracted from the image. Every of the pixels in a region is same with relevance some characteristics, like color, intensity, or texture, adjacent regions are considerably completely different with respect to identical characteristics.

Number of segmentation methods was proposed in the past decades, and few methods are necessary to present properly here. The methods presented in this chapter are therefore rather a categorization regarding the emphasis of an approach than a strict division.

The following methods are used:

- **Threshold Segmentation.** To Segment an Image, Histogram and Slicing Techniques are used. Histogram and Slicing techniques can be applied directly to an image, and also it can be used with pre-processing and post-processing techniques.
- **Edge segmentation.** With this technique, the represented object boundaries are assumed by detected edges in an image and used to identify these objects.
- **Region segmentation.** Edge segmentation finds the object boundaries and then, after it locates the object itself by filling them. Whereas region segmentation is exactly opposite to that of edge segmentation, i.e. the region based technique starts in the middle of an object and then it will "grow" outward until it meets the object boundaries.

Each pixel is assigned to the correct object in Perfect image segmentation; the process of segment usually cannot be achieved. Because of the Digital image is obtained, this is impossible, since a pixel may straddle the "real" boundary of objects such that it partially belongs to two or more objects. Numbers of methods are presented and attempts are made to single segment to assign a pixel. This approach is most used in number of applications. The method is assigned to segment with probability distribution to each pixel called Probabilistic. These methods are more accurate by theoretically, and for measurements of specific objects the probabilistic approach is used. Therefore by using the above approach the measurement of an object is accurate. However, both in the sense of concept and implementation the probabilistic techniques are complexity to segmentation process and the technique is still used.

Due to the occurrence of “over and under segmentation” the Perfect image segmentation is not much used. In case1 the pixels belonging to the same object are classified as belonging to different segments. A single object may be represented by two or more segments. In case2, the pixels belonging to different objects are classified as belonging to the same object i.e. opposite to that of case1. Figure 4.2 shows a simple example of over and under-segmentation. Original image is shown at the top left. The image on the top right shows the correct segmentation. Each segment has been indicated by a unique grey value here. The bottom left and right images show examples of over segmentation and under segmentation respectively.

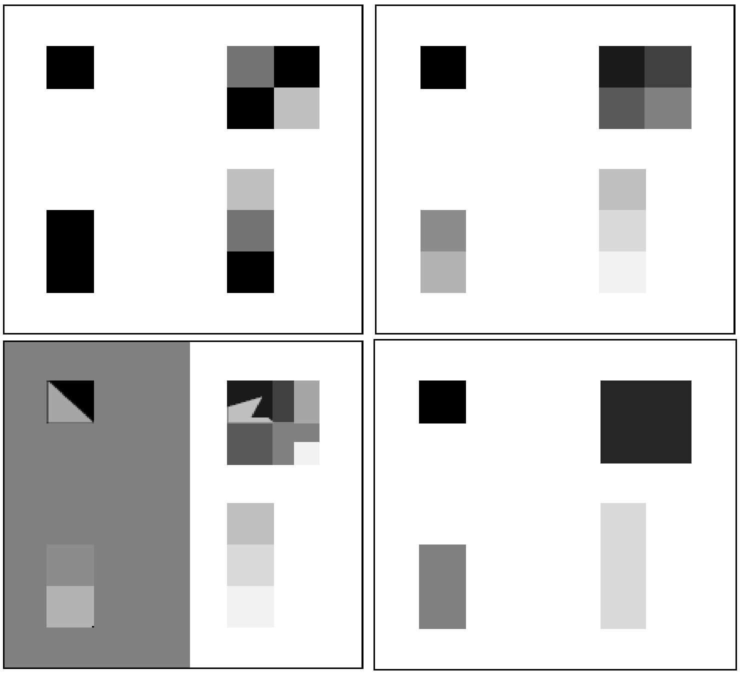


Figure 4.2: Over and Under Segmentation

4.3 Threshold segmentation

Thresholding is the most frequently used technique to segment an image. The thresholding operation is a grey value remapping operation g defined by:

$$g(v) = \begin{cases} 0 & if \quad v < t \\ 1 & if \quad v \geq t \end{cases}$$

Where v represents the grey value and t represents the threshold value. By using Thresholding technique converts a grey valued image to binary image. After the operation of thresholding, the image is segmented into two segments, which are identified by the pixel values 0 and 1 respectively.

If the image contains bright objects on a dark background, then the thresholding is used to segment an image. For an example see fig4.3. the grey values of objects are different from the background value in many types of images, thresholding technique is the best suited method to segment an image into objects and background.

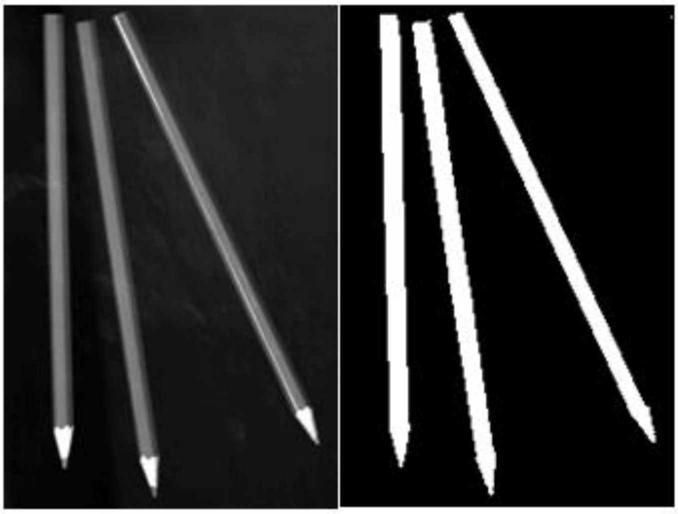


Figure 4.3 Segmentation by thresholding.

The original image with bright objects on a dark background is on the left side of above image. By using an appropriate threshold segments the image into objects (segment with value 1) and background (segment with value 0).

Number of methods are exists to select the suitable threshold value for segmentation process. The most common method is to set the threshold value. The user manipulates the value and review the threshold result till the satisfied segmentation is obtained. In establishing a suitable threshold value the histogram is a valuable tool. Figure 4.4 show the image from the previous figure 4.3 together with its histogram, and the four different threshold values are obtained from the histogram by using thresholding results.

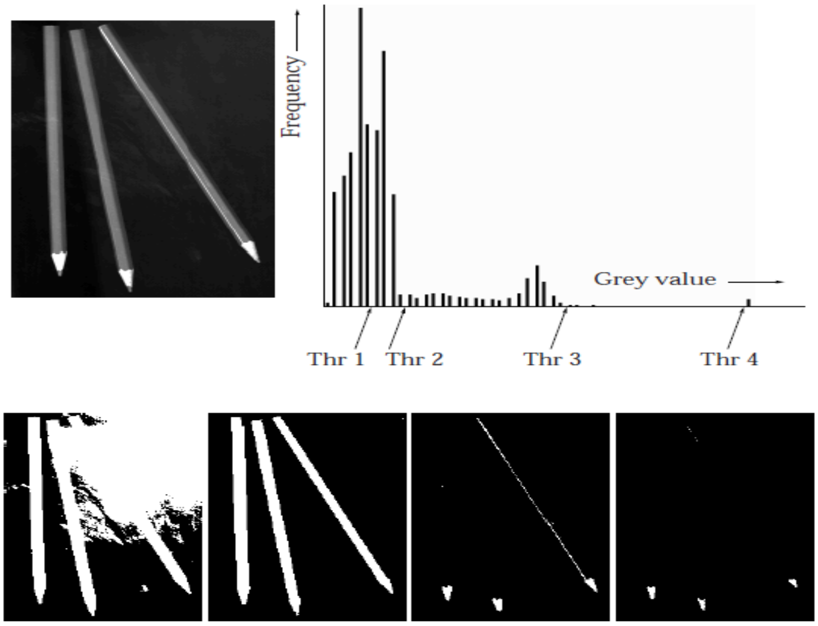


Figure 4.4 Threshold selection from the histogram.

Top row contains the original image and histogram. Four grey values are chosen which are indicated by 1, 2, 3 and 4. At each of the values the bottom row shows the respective threshold results. At a first glance, the original image appears to have only three grey values. This is indeed the case. The result of threshold 1 shows that the peak between thresholds 1 and 2 also corresponds to background values. The result of threshold 2 shows the desired segmentation; every grey value to the right of threshold 2 corresponds to the pencils. Threshold 4 shows that the right-most little peak corresponds to the bright grey value in the tips of the pencils.

When several desired segments in an image can be distinguished by their grey values, threshold segmentation can be extended to use multiple thresholds to segment an image into more than two segments, the value smaller than the first threshold are assigned to segment 0, the values between the first and second threshold are assigned to segment 1, the values between the second and third threshold are assigned to segment 2 in all the pixels respectively and so on. If n thresholds are used then (t_1, t_2 . . . tn). After thresholding, the image is segmented into n + 1 segments and identified by the grey values 0 to n respectively.

4.3.1 Threshold selection

Number of methods is there to find a suitable threshold for segmentation. The simplest method is the interactive selection of a threshold by the user, possibly with the aid of the image histogram a method that is usually accompanied by a graphical tool, which lets the user immediately assess the result of a certain choice of threshold.

4.3.1.1 Using histogram extrema

As shown in figure 4.5 the typical objects which is on a background image will have a bimodal histogram. Generally the peaks of the histogram will not be as sharp as we like them; because of the image degrading factors such as partial volume effects, noise and illumination artifacts. In practice, certain objects may overlap in corresponding with the histogram. When this happens, the segmentation based on global thresholding will occur an error. If there is no overlap in the histogram, can use maxima i.e. peak of the histogram to establish threshold segmentation. This threshold‘t’ will be midway between two peaks with grey values p1 and p2.

$$t = \frac{p1+p2}{2}$$

Or better still, it may be the grey value at the minimum between the two peaks:

$$t = \arg \min_{v \in [p_1, p_2]} H(v).$$

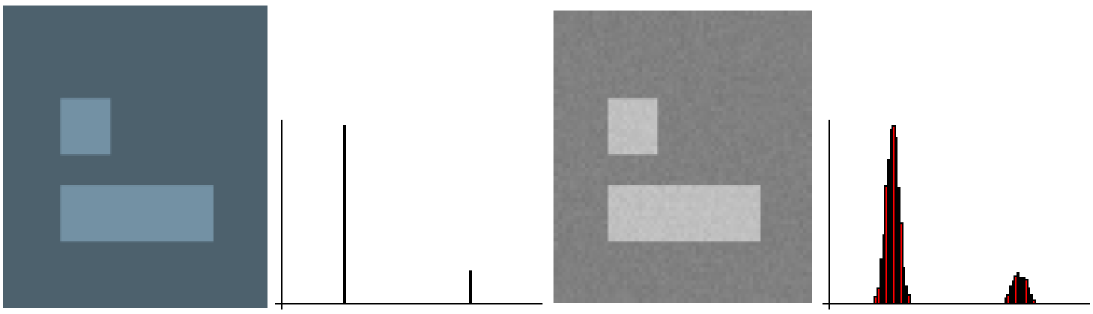


Figure 4.5: An Object on a Background Image- Top left of the image contains only two grey values is shown in above figure. The histogram (next to it) shows only two peaks. In practice, image artifacts will cause the image and its histogram to degrade from this ideal situation. In particular, the histogram blurs as is shown in the image and histogram on the right. In practice, it is quite possible that the two curves in the histogram start to overlap.

Where *H* (*v*) is the grey value v of the histogram value and by let the p1 is lesser than the p2.

As shown in the below figure 4.6, the histogram is not as smooth as someone likes. So, it is most useful to smooth the histogram beforehand for many histogram analysis tasks for example finding a global extrema. This analysis can be done by the convolution method with a Gaussian or a discrete averaging kernel.

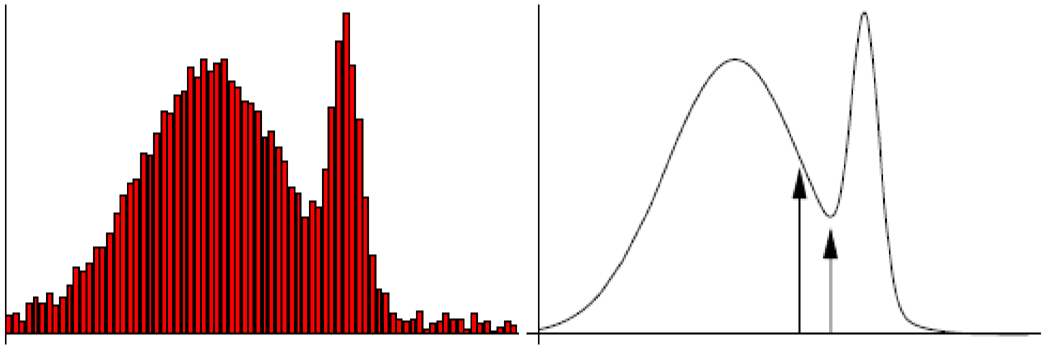


Figure 4.6 Comparing the threshold midway between peaks and the minimum between peaks: the top left image is an histogram. The right image is an approximation of the histogram i.e. thick curve using two Gaussians i.e. thin curves. The left arrow shows the threshold value when used by the minimum between peaks criterion the optimal threshold shows in the right arrow.

4.3.1.2 Minimum variance

If the segment had relatively homogeneous grey values, then it makes sense to select a threshold that minimizes the variance of the original grey values within segments. Alternatively, select a threshold which maximizes the variance between objects and background, or that attempts to optimize both “within” and “between” variances.

4.3.2 Optimal thresholding

As seen in the figure 4.5 an object should cause a single peak in the histogram and theoretically be the uniformly grey value. Practically the distribution of grey values is shown. By using a statistical distribution function such as Gaussian technique the distributed grey value is approximated. The Gaussian technique that approximates the histogram by using weighted sum of distribution functions is optimal thresholding, and then the threshold value is set so that if the number is incorrectly segmented pixels as absorbed from the approximation will be minimum. For an example, let us assume the histogram is bimodal i.e. two segments; objects and background, and thus the histogram can be approximated well using a sum of two Gaussians. In this case, the normalized histogram h as

$$h(v) \approx P_0 g_0$$

$$h(v) \approx P_0 g_0(v) + P_1 g_1(v)$$
$$= (1 - P_1) g_0(v) + P_1 g_1(v),$$

Algorithm for Iterative thresholding

1. Let us assume that corner points of the image is background pixels i.e. part of segment is 0, and then set $\mu 0$ value to the four corner pixels of the grey value. Let all other pixels are object pixels and now set $\mu 1$ to average grey value.
2. Set threshold value t to the below equation and then segment an image.

$$t = 1/2\ (\mu 0 + \mu 1)$$

3. Now recompute μ0 and μ1 which is mean of the original grey values pf two segments.
4. Now if the threshold value changes then go to step 2 and iterate until the threshold value not changes significantly.

Example

If we have an image with the following histogram:

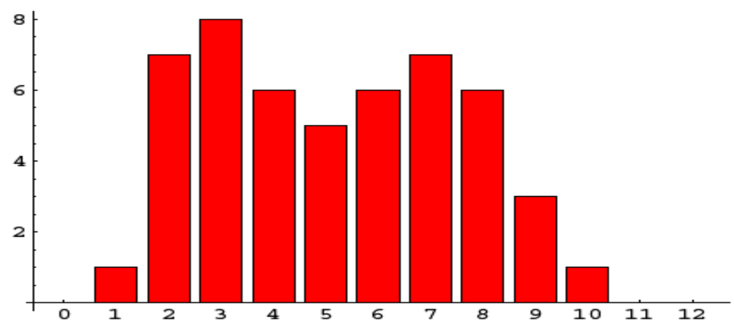


Figure 4.7: Iterative Thresholding graph.

Let the four corner pixels have grey values of 2, 2, 2, and 1 respectively, then the intial value for $\mu 0$ is $\mu 01 = \frac{2+2+2+1}{4} = 1.75$ Then compute the average value of remaining pixels from the histogram or from the image, $\mu 02 \approx 5.46$. There for set the initial threshold as $t(0) = 1/2\ (\mu 01 + \mu 02) \approx 3.61$.

By thresholding, the results will appear in segment 0 which contain all pixels along with the original grey values in the range {0, . . . , 3}, and in segment 1 will appear pixels with grey value in the range {4, . . . , 12}.

By computing the mean values of these two segments will $\mu 03 \approx 2.44$ and $\mu 04 \approx 6.44$. Then the new threshold will appear as $t(1) = 1/2\ (\mu 03 + \mu 04) \approx 4.44$.

After above applied conditions when thresholding the results in segment 0 which contains the pixels with values {0, . . . , 4}, and similarly the segment 1 with the range {5, . . . , 12}. Then by recomputing the mean segment values which gives as $\mu 05 \approx 2.86$ and $\mu 06 \approx 6.96$, so $t(2) \approx 4.91$. The last threshold value which is obtained will be the result in the same segmentation as the previous step, there for end the algorithm here.

4.3.3 Enhancing threshold segmentation

By thresholding the segmentation of the image is totally insensitive to the spatial context of a pixel. Even then the pixel satisfies a threshold criterion a human observer may decide on the basis of spatial context that a pixel is not part of a segment. For example, by thresholding the noise in a image may cause number of very small segments, while in many applications it is known that such small segments cannot physically exist as shown in figure 4.8. Or the segment boundaries may appear bumpy while we know them to be smooth.

Number of techniques are available that address such problems. These techniques can be divided into three categories they are processing of the original image prior to segmentation, processing of the segmented result, and adaptation of the segmentation process. Often, combinations of above three techniques are used in single application. In this section we will give some examples of common approaches.

Below example shows the use of non-linear filters such as a median filter and morphological operators for removing small scale artifacts.

Figure 4.8 Example of noise causing small erroneous segments and object holes.

Example: removing small segments and object holes

In the above figure 4.8 after thresholding the noise can cause small erroneous segments and object holes. There are many possible approaches are there to remove the small artifacts. For example, closing is used to remove the holes, and opening is used to remove the small segments. By using the isolated pixels or segments below certain area, the grey value to the value of their neighborhood. Another approach is the application of a median filter to the segmented result, which also effectively removes all small artifacts.

4.4 Edge segmentation

As the binary object is fully represented by its edges, the segmentation of an image into separate objects can be achieved by finding the edges of those objects. By using edge segmentation approach is (1) compute an edge image, containing all possible edges of an original image, (2) process the edge image so that only closed object boundaries remain, and (3) transform the result to an ordinary segmented image by filling in the object boundaries. In the step 3 the filling of boundaries is not a difficult step. Transforming an edge or edgeness image to closed boundaries also requires the removal of edges that are caused by noise or other artifacts. No edge was detected at the bridging of gaps at locations but it should have logically one. Edge segmentation can be achieved by the following algorithm.

Algorithm: Edge segmentation

Given image $\boldsymbol{f}$,

1. Calculate the edgeness image ∇f from $\boldsymbol{f}$.
2. Threshold ∇f to an image $(\nabla f)_t$, so the binary image will obtain by edge pixels.
3. Calculate the Laplacian image Δf from $\boldsymbol{f}$. Any preferred discrete or continuous Laplacian operator may be used.
4. Compute the image $g = (\nabla f)_t \cdot \text{sgn}(\Delta f)$.

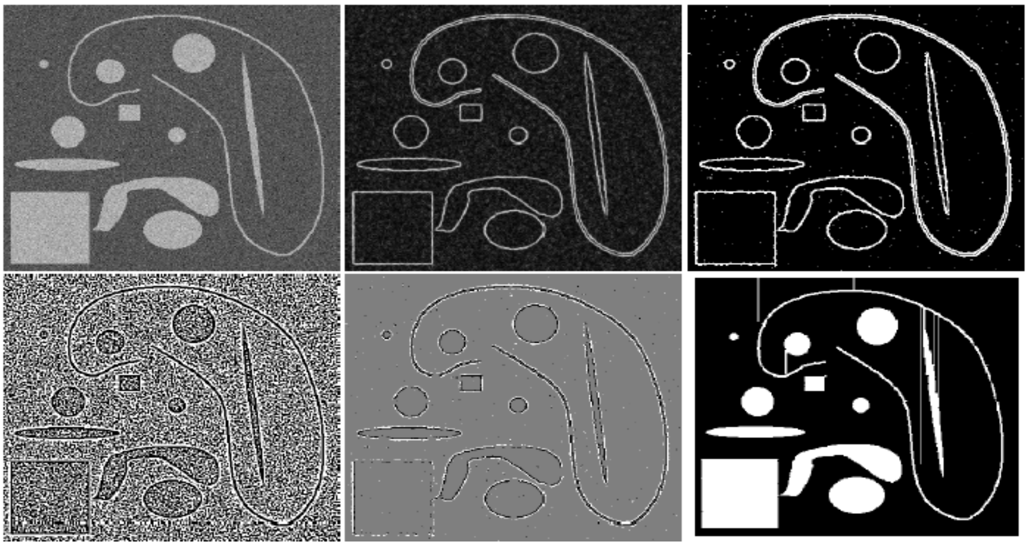


Figure 4.9 Edge segmentation.

Top left: original 400 × 350 artificial image with added noise. Top middle: edgeness image; computed using a scale space operator ($f\ w$) with $\sigma = 1$ pixel. Top right: same image after thresholding. Bottom left: sign of Laplacian image. Laplacian image computed using a scale space operator (fii) with $\sigma = 1$ pixel. Bottom middle: product of Laplacian sign and thresholded edge image. Bottom right: result after filling in of the boundaries as in the algorithm above. The noise was added to the original image to show some of the artifacts that they cause: notches in the edges of segmented objects, and lines where the filling algorithm encounters erroneous sign transitions. If no or less noise is added to the original, the segmentation result is perfect.

Image g contains the boundaries of the objects to be segmented. To facilitate the final steps of algorithm Laplacian is used, turning the boundary image into a segmented image h containing solid objects. If we traverse the image g from left to right, two adjacent pixels with values -1 and 1 means we move into an object, and the values 1 and -1 means we move out of one. The image h can therefore be created by setting all pixel values to zero, except for those pixels that are between the transitions $1 \rightarrow -1$ and $-1 \rightarrow 1$ in each line of g, which are set to 1. If unique values are desired for each separate segment, a labelling algorithm can be run on h.

4.4.1 Edge linking

Edge detection very rarely gives the perfect and closed boundaries need for a direct segmentation. There will be frequently false edges detected which don't happen, and gaps occur where there should be edges. To overcome this problem of partial edges need edge linking to link available partial edges into an object boundary.

4.4.1.1 Hough transform

The Hough transform can be used to find the ideal edges that best fit the partial edges. This technique will work when the objects in the image have simple parametric shapes i.e. it works for few parameters, and for complex shapes when Hough transform is used then time consuming will be more. The disadvantage of the Hough transform is it doesn't have a flexible in sense of unexpected variations in shapes of the object usually cannot be coped. When lines are used, some post-processing of the result is necessary to connect the correct line segments, as shown in figure 4.10.

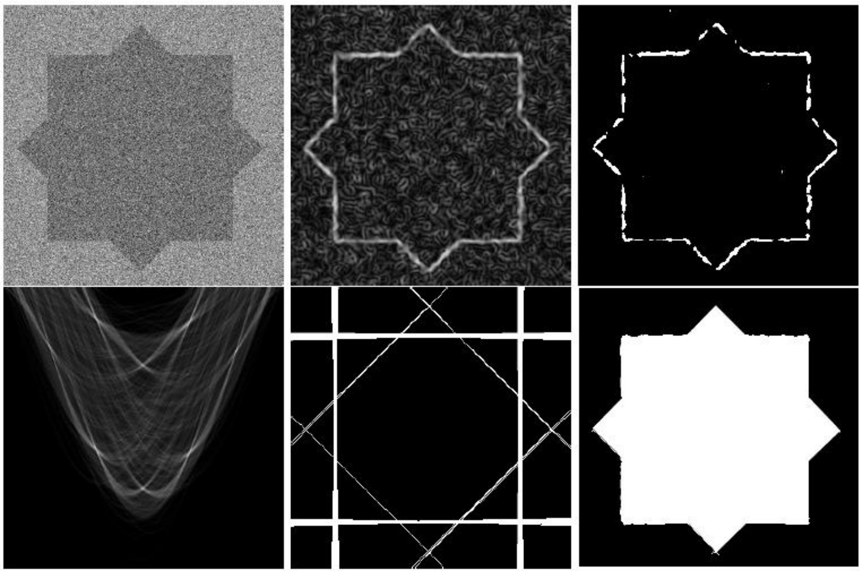


Figure 4.10 Edge linking using the Hough transform.

Top left: original nois 256 × 256 image. Top middle: edge image, computed using the *fw* operator at a scale of $\sigma = 2$ pixels. Top right: threshold of this image, showing gaps in the object boundary. Bottom left: the Hough transform (lines) of the thresholded edge image, showing eight maxima. Bottom middle: the corresponding line image of the maxima in the Hough transform. Note that intelligent processing of this image is necessary to obtain only those line segments that make up the original object. (In this case we dilated the binary edge image, took the logical 'and' with the Hough lines image, and performed an opening by reconstruction of erosion.) Bottom right: result after processing and filling; we now have a closed boundary.

4.4.1.2 Neighborhood search

The easy approach for edge linking is searching around neighborhood in edge pixel for possibility to link the edge pixel. A typical procedure is to select a starting edge pixel then find the best suitable link around neighborhood and first link this pixel then start a new search around newly linked pixel, and so on. The process is stopped when one of the following occurs: (1) If the suitable link is not found, (2) If the image boundary is reached, (3) if the pixel about to link is already part of the current linked boundary. Many criteria can be used to select a pixel B which is suitable to link the current pixel A. For example:

- The pixel value B should have high edgeness value or the value of the edgeness should be close to the value of edgeness in A.
- The direction of the gradient in B should not differ too much from the gradient direction in A, or the direction from A to B should not differ too much from the direction of previous links.

The latter criterion ensures that edges cannot bend too much, which ensures that the search is kept on the right path when we reach a point where edges cross. It has also proven to be a necessary heuristic for noisy images. A disadvantage, however, is that edges at places where objects really have sharp angles are not linked. In these cases, a post-processing of all found linked boundary segments is necessary, where the boundary segments themselves are linked (using proximity and other criteria) to form closed boundaries. In addition, some form of post-processing is usually necessary to remove those linked edge segments that are unlikely to belong to realistic boundaries. The removal criteria are often application dependent, but commonly edge segments below a certain length are removed.

The starting process for the search process may be indicated by the user or it may determine as per the application dependent. It also possible to use (let say) hundred points, which is highest edgeness value for starting points. This condition works well when the contrast of objects and background is almost same for all objects. If this is not the case, then no starting points are picked at low contrast objects. An approach to this problem is to pick hundred starting points with highest edgeness value to do edge linking, then pick a new set of starting points with highest edgeness, disregarding all pixels tracked in the first run and possibly their neighborhoods too. To local search process number of enhancements is proposed, and two out of them are given below to address common problems. Because of noise and occurrence of thick edges the problem occurs that the tracked edge segments are more convoluted then they should realistically. To ensure, the edge tracking remains central to a thick edge, an additional criterion to use in the edge tracking is the demand that the ridgeness of edgeness is high. This is shown in figure 4.11.

Another problem is that the noise will disrupt locally an edge, follow a spurious edge branch that usually ends quickly instead of following the main edge by causing edge tracking to locally. Linking the edge pixel to the best linking candidate and then moving on to the candidate, this can be obtained by the adaption of search process. If linking of the best candidate results in a short dead end, we backtrack and see if the next best candidate results in a longer branch.

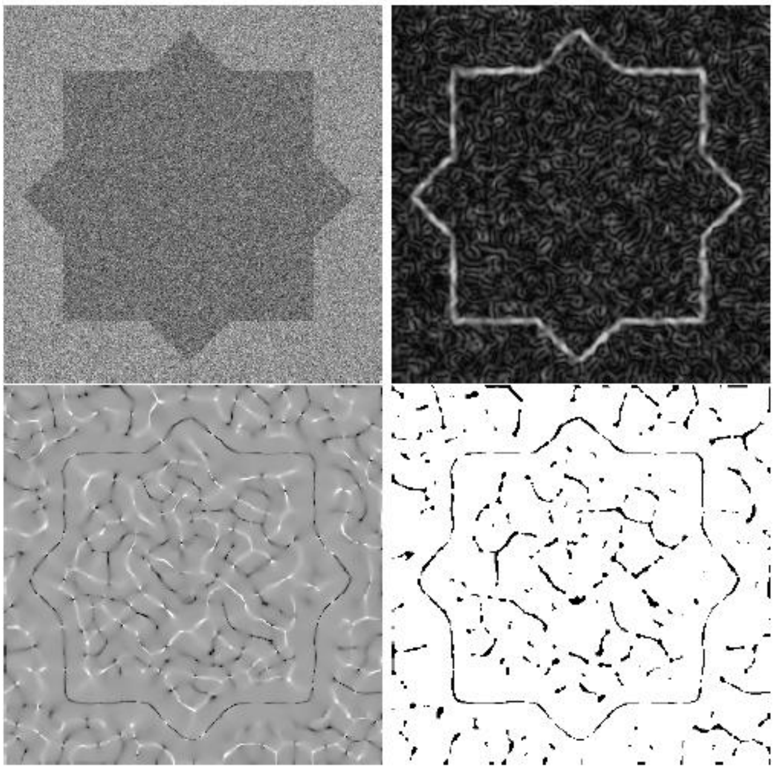


Figure 4.11 Noise and thick edges may cause edge tracking to meander.

The use of a ridgeness in addition to (or instead of) the edgeness image in the tracking process may be beneficial, since the ridge is by definition a narrow structure. Top left: Original 256 × 256 noisy image. Top right: edgeness image (*fw*, σ = 2 pixels). Note the thickness of the edges. Bottom left: ridgeness image of the edgeness image (fvv/fw, σ = 4 pixels). Bottom right: thresholded ridgeness image. Note that the ridges are much thinner than the edges.

4.4.2 Network analysis

4.4.2.1 Mathematical programming

The edge linking process is effectively the same as finding the best path from A to B if it have a good starting point A for a boundary segment, and end point B i.e. set of potential end points. Construct some network from which all possible paths from A to B can be cleaned to find the best path. Figure 4.12 shows the example of how such a network can be constructed from a gradient image that the each edge pixel is transformed to a network node and only pixels with an edgeness value above a certain threshold are part of the network, and arrows are drawn to all neighboring nodes (pixels) that can reached from a certain node.

Main goal is to find the best path here i.e. the best sequence of arrows from A to B. It can be defined in several ways depending up on application. The best path is the shortest path, in which the path should be in highest average edgeness value; the path should be in least curvature or a combination of these and other criteria. By assigning costs to each arrow these criteria are included in a network. If the shortest path is taken to assign distances to each arrow then 1 is assigned to each horizontal or vertical arrow and √2 is assigned to each diagonal arrow. The cost of each path from *A* to *B* is defined as the sum of costs of the arrows that make the path. The best path is the path with lowest cost.

Algorithm: Solving the shortest-route problem

The objective of this algorithm is to find the nth nearest node to A, where *n* is the iteration number. For example, if *n* = 1 then find the nearest node to A, If *n* = 2 then find the second closest node to A, likewise the algorithm is terminated if the nth closest node is desired end node B.

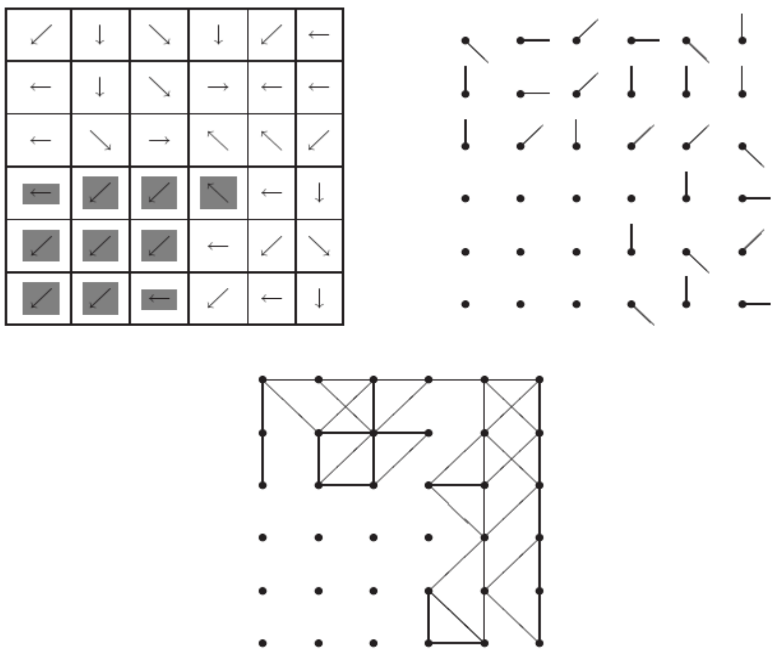


Figure 4.12 creating a network from a gradient image.

Top left: image with arrows indicating the gradient direction. (Actually, the nearest direction in the set of {0, 45, 90, 135, 180, 225, 270, 315} degrees.) Pixels where the gradient (edgeness) value falls below a threshold are marked in grey. Top right: network created from the gradient image. Each pixel has been turned into a node. Lines show the significant gradients, turned 90 degrees so they are oriented along the edge at a node. Only the directions north, northeast, east, and southeast are used; they are unique modulo 180 degrees. Bottom: final network. For each node with an arrow in the top right network, we examine the nodes in the approximately along the edge direction of a node (not more than 45 degrees deviation). If the gradient direction of the new node differs by 45 degrees (modulo 180) or less from the current direction, a connection is drawn.

4.5 Region segmentation

In the previous section, objects were found by locating their boundaries. In this section, we discuss the dual approach of finding the object region instead of its edges. In theory, finding an object by locating its boundary and finding it by establishing the region it covers will give you exactly the same object; the boundary and the region are just different representations of the same object. In practice, however, taking an edge based approach to segmentation may give radically different results than taking a region based approach. The reason for this is that we are bound to using imperfect images and imperfect methods; hence the practical result of locating an object boundary may be different from locating its region.

Region segmentation methods have only two basic operations: merging and splitting and many methods even feature only one of these.

The basic approach to image segmentation using merging is:

1. Obtain an initial (over)segmentation of the image,
2. Merge those adjacent segments that are similar –in some respect– to form single segments,
3. Go to step 2 until no segments that should be merged remain.

The initial segmentation may simply be all pixels, *i.e.*, each pixel is a segment by itself. The heart of the merging approach is the similarity criterion used to decide whether or not two segments should be merged. This criterion may be based on grey value similarity (such as the difference in average grey value, or the maximum or minimum grey value difference between segments), the edge strength of the boundary between the segments, the texture of the segments, or one of many other possibilities.

The basic form of image segmentation using splitting is:

1. Obtain an initial (under)segmentation of the image,
2. Split each segment that is inhomogeneous in some respect (*i.e.,* each segment that is unlikely to *really* be a single segment).
3. Go to step 2 until all segments are homogeneous.

The initial segmentation may be no segmentation at all, *i.e.,* there is only a single segment, which is the entire image. The criterion for in homogeneity of a segment may be the variance of its grey values, the variance of its texture, the occurrence of strong internal edges, or various other criteria. The basic merging and splitting methods seem to be the top-down and bottom-up approach to the same method of segmentation, but there is an intrinsic difference: the merging of two segments is straightforward, but the splitting of a segment requires we establish suitable sub-segments the segments can be split into. In essence, we still have the segmentation problem we started with, except it is now defined on a more local level. To avoid this problem, the basic splitting approach is often enhanced to a combined split and merge approach, where inhomogeneous segments are split into simple geometric forms (usually into four squares) recursively. This of course creates arbitrary segment boundaries (that may not be correlated to realistic boundaries), and merge steps are included into the process to remove incorrect boundaries.

4.5.1 Merging methods

4.5.1.1 Region growing

To merge adjacent pixel segments into one segment many merging methods of segmentation use a method called region growing. Region growing needs a set of starting pixels called seeds. The region growing process consists of picking a seed from the set, investigating all 4-connected neighbors of this seed, and merging suitable neighbors to the seed. The seed is then removed from the seed set, and all merged neighbors are added to the seed set. The region growing process continues until the seed set is empty. The algorithm below implements an example with a single seed, where all connected pixels with the same grey value as the seed are merged.

Algorithm: Region growing

The stack operation in data structure is performed to keep track of the seeds. Two such operations are performed on stack they are push and pop, where push is which puts a pixel or it coordinates on the top of the stack, and pop is which takes a pixel from the top of the stack. In this algorithm, the image is called f, the seed has coordinates (x, y) and grey value $g = f(x, y)$. The region growing is done by setting each merged pixel's grey value to a value h (which must not equal g). The pixel under investigation has coordinates (a, b).

The algorithm runs

1. Push (x, y)
2. As long as the stack is not empty do
    a) pop (a, b)
    b) if $f(a, b) = g$ then
        i. set $f(a, b)$ to h
        ii. push $(a - 1, b)$
        iii. push $(a + 1, b)$
        iv. push $(a, b - 1)$
        v. push $(a, b + 1)$

Final region can be extracted from the image by selecting all pixels with grey value h. Select h to be a value which is not present in the original image in running the algorithm to ensure the correct result. Statement that decides if a pixel should be merged f(a, b) = g can be modified to use a different merging criterion. A simple modification can be allowed for merging of pixels that are in a certain range of grey values. $l < f(a, b) < h$.

4.5.2 Splitting and merge methods

Where region merging is an agglomerative approach, region splitting is divise. We mentioned before that this difference makes that the two approaches are not opposites, but fundamentally different problems; the merging of two segments is straightforward, but the splitting of a segment requires that suitable sub-segments are established to split the original segment into. The problem of *how* to split a segment is of course itself a segmentation problem, and we can treat it as such: any segmentation method (such as the ones treated in this chapter) can be applied to the segment to establish sub-segments. Besides the hierarchical level, there is no intrinsic difference The problem of how to decide if a 316 Segmentation segment need splitting can be solved using the same measures of region homogeneity mentioned in the section on region merging, *e.g.*, a segment needs to be split if

- the grey value variance exceeds a threshold, or
- the variance of a texture measure exceeds a threshold, or
- the histogram entropy (or another histogram measure) exceeds a threshold, or
- high edgeness pixels are present.

When a segment needs splitting, a faster approach than starting a segmentation on the segment is to simply split it into four quadrants. This of course has a high probability of placing segment boundaries at unrealistic positions. This problem is reduced by adding merging operations to the segmentation process; recursively splitting and merging the image segments in a way such as the one used in figure 4.13. Especially if different criteria are used for splitting and merging –for example, edge heuristics for the merging and region homogeneity for splitting– the split and merge segmentation may not converge to a stable result, but instead keep alternating between various solutions.

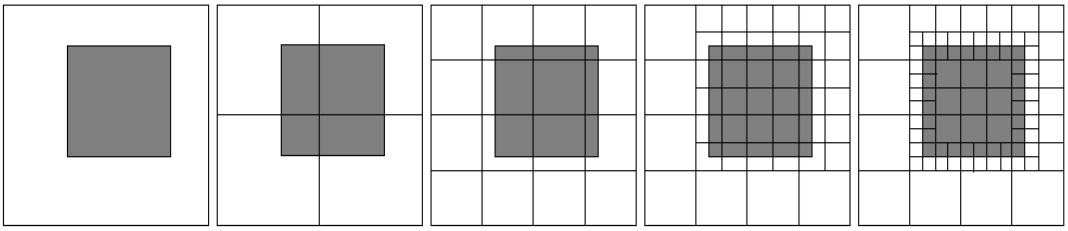


Figure 4.13 Example of split and merge segmentation.

Left: original image. From left to right: recursively splitting each segment into four segments when the grey value variance of a segment is non-zero. If we subsequently recursively merge all adjacent segments with the same (average) grey value, we end up with two segments; exactly the desired segments of object and background.

4.6 Morphological Operations

Morphology is a broad set of image processing operations that process images based on shapes. Morphological operations apply a structuring element to an input image, creating an output image of the same size. In a morphological operation, the value of each pixel in the output image is based on a comparison of the corresponding pixel in the input image with its neighboring image. By choosing the size and shape of the neighborhood, can construct a morphological operation that is sensitive to specific shapes in the input image.

The most basic morphological operations are dilation and erosion. Dilation adds pixels to the boundaries of objects in an image, while erosion removes pixels on object boundaries. The number of pixels added or removed from the objects in an image depends on the size and shape of the structuring element used to process the image. In the morphological dilation and erosion operations, the state of any given pixel in the output image is determined by applying a rule to the corresponding pixel and its neighbors in the input image. The rule used to process the pixels defines the operation as dilation or erosion.

Dilation: The value of the output pixel is the maximum value of all the pixels in the input pixel's neighborhood. In a binary image, if any of the pixels is set to 1, the output pixel is set to 1.

Erosion: The value of the output pixel is the minimum value of all the pixels in the input pixel's neighborhood. In a binary image, if any of the pixels is set to 0, the output pixel is set to 0.

Below figure is the dilation of a binary image. The structuring element defines the neighborhood of the pixels which is in circled. The dilation function applies the appropriate rule to the pixels in the neighborhood and assigns a value to the corresponding pixel in the output image. In figure, the morphological dilation function sets a value of the output pixel to 1 because one of the elements in the neighborhood defined by the structuring element.

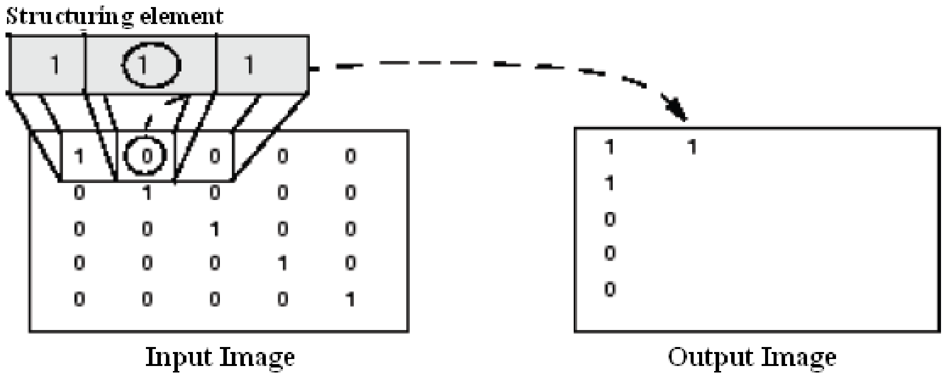


Figure 4.14: Morphology Dilation of Binary Image

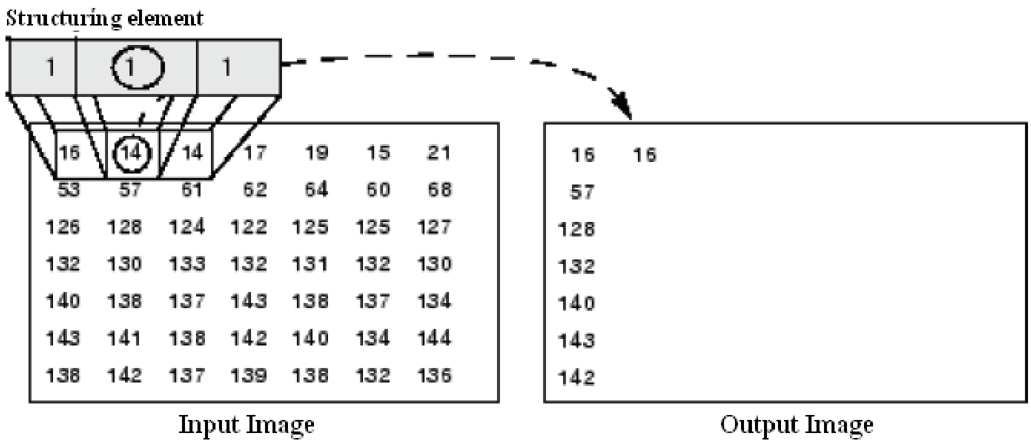


Figure 4.15: Morphology Dilation of Grayscale Image

The above figure is greyscale image. The above figure shows the processing of a particular pixel in the input image. The function applies the rule to the input pixel's neighborhood and uses the highest value of all the pixels in the neighborhood as the value of the corresponding pixel in the output image.

4.6.1 Structuring Elements

An essential part of the dilation and erosion operations is the structuring element used to probe the input image. A structuring element is a matrix consisting of only 0's and 1's that can have any arbitrary shape and size. The pixels with values of 1 define the neighborhood.

Two-dimensional, or flat, structuring elements are typically much smaller than the image being processed. The centre pixel of the structuring element, called the origin, identifies the pixel of interest pixel being processed. The pixels in the structuring element containing 1's define the neighborhood of the structuring element. These pixels are also considered in dilation or erosion processing.

Three-dimensional, or nonflat, structuring elements use 0's and 1's to define the extent of the structuring element in the *x*- and *y*-planes and add height values to define the third dimension.

4.6.1.1 The Origin of a Structuring Element

The morphological functions use this code to get the coordinates of the origin of structuring elements of any size and dimension.

Origin = floor((size(nhood)+1)/2)

For example, the following illustrates a diamond-shaped structuring element.

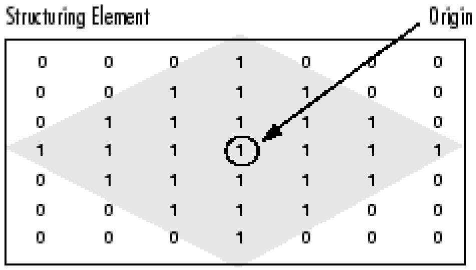


Figure 4.16: Origin of a Diamond-Shaped Structuring Element

4.6.2 Dilation

Dilation: "grow" or "thicken" an object in a binary image

- Extent of thickening controlled by a structuring element
- Dilation of image *A* and structuring element *B*: A ⊕ B

  $A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \varnothing\}$

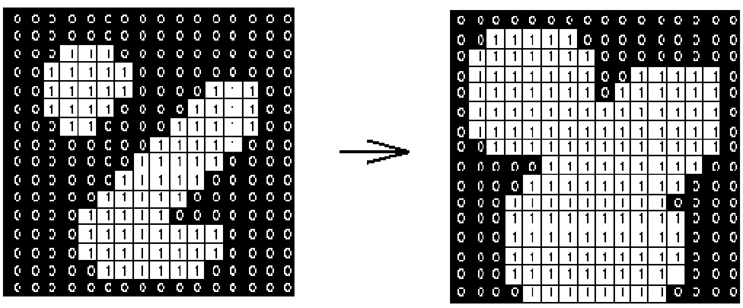


Figure 4.17 Dilation

4.6.3 Erosion

Erosion: “shrink” or “thin” an object in a binary image

- Extent of shrinking controlled by a structuring element
- Erosion of image *A* and structuring element *B*: $A \ominus B$

  $A \ominus B = \{z \mid (B)z \cap A \neq \emptyset\}$

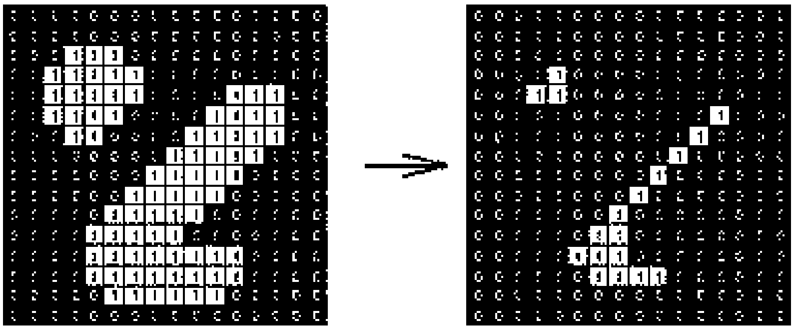


Figure 4.18: Erosion

4.6.4 Combining Dilation and Erosion

Dilation and erosion are often used in combination to implement image processing operations. For example, the definition of a morphological opening of an image is erosion followed by dilation, using the same structuring element for both operations. The related operation, morphological closing of an image, is the reverse: it consists of dilation followed by erosion with the same structuring element.

4.6.4.1 Opening

Opening generally smoothes the contour of an object and eliminate thin protrusions. The opening of a set *A* by structuring element *B* is defined as

$$A \circ B = (A \ominus B) \oplus B$$

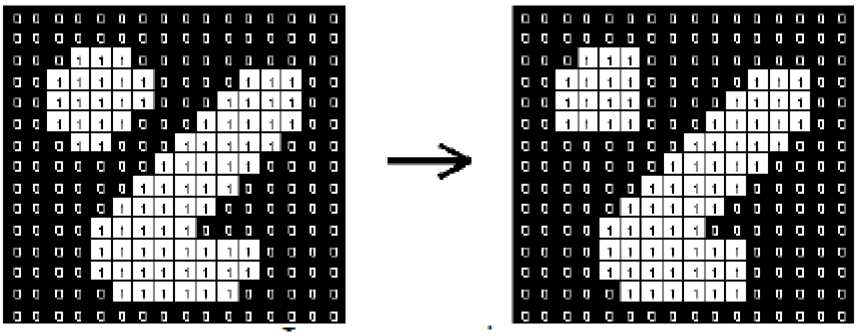


Figure 4.19: Opening

4.6.4.2 Closing

Closing also tends to smooth sections of contours but fusing narrow breaks and long, thin gulfs and eliminating small holes and filling gaps in the contour. Similarly, the closing of a set *A* by structuring elem ent *B* is defined as

$$A \bullet B = (A \oplus B) \ominus B$$

Therefore, the closing of *A* by *B* is the dilation of *A* by *B*, followed by the erosion of the result by *B*.

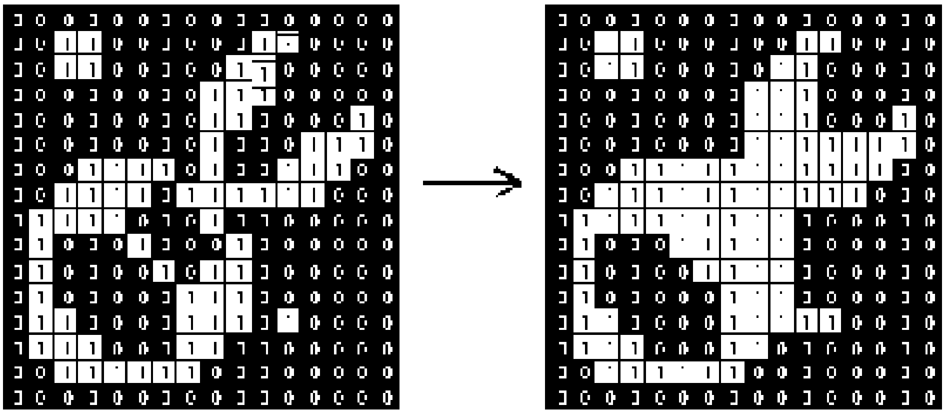


Figure 4.20: Closing

4.7 Median Filtering

Some smoothing filters i.e. Low pass filters reduce noise. However, some assumption is that the neighboring pixels represent additional samples of the same value as the reference pixel, i.e. they represent the same feature. But it is blurring of features results.

There is some linear process for convolution techniques to implement weighting kernels as a neighbourhood function. But there are nonlinear neighborhood operations that can be performed for the purpose of noise reduction.

One such method is known as median filtering. In median filtering, the neighboring pixels are ranked according to brightness (intensity) and the median value becomes the new value for the central pixel. Median filters can do an excellent job of rejecting certain types of noise, in particular, “shot” or impulse noise in which some individual pixels have extreme values. In the median filtering operation, the pixel values in the neighborhood window are ranked according to intensity, and the middle value (the median) becomes the output value for the pixel under evaluation as shown in figure 4.21.

91	**55**	**90**
77	**68**	**95**
115	**151**	**210**

Figure 4.21: 3X3 Original Image

From the above figure the Median Filtering using the full 3X3 neibhorhood output is

55 68 77 90 **91** 95 115 151 210

The median value is 91.

The original image is added with salt and pepper noise and then filtered with median filter. After applying Median filter the salt and pepper noise is removed with less blurring of edges

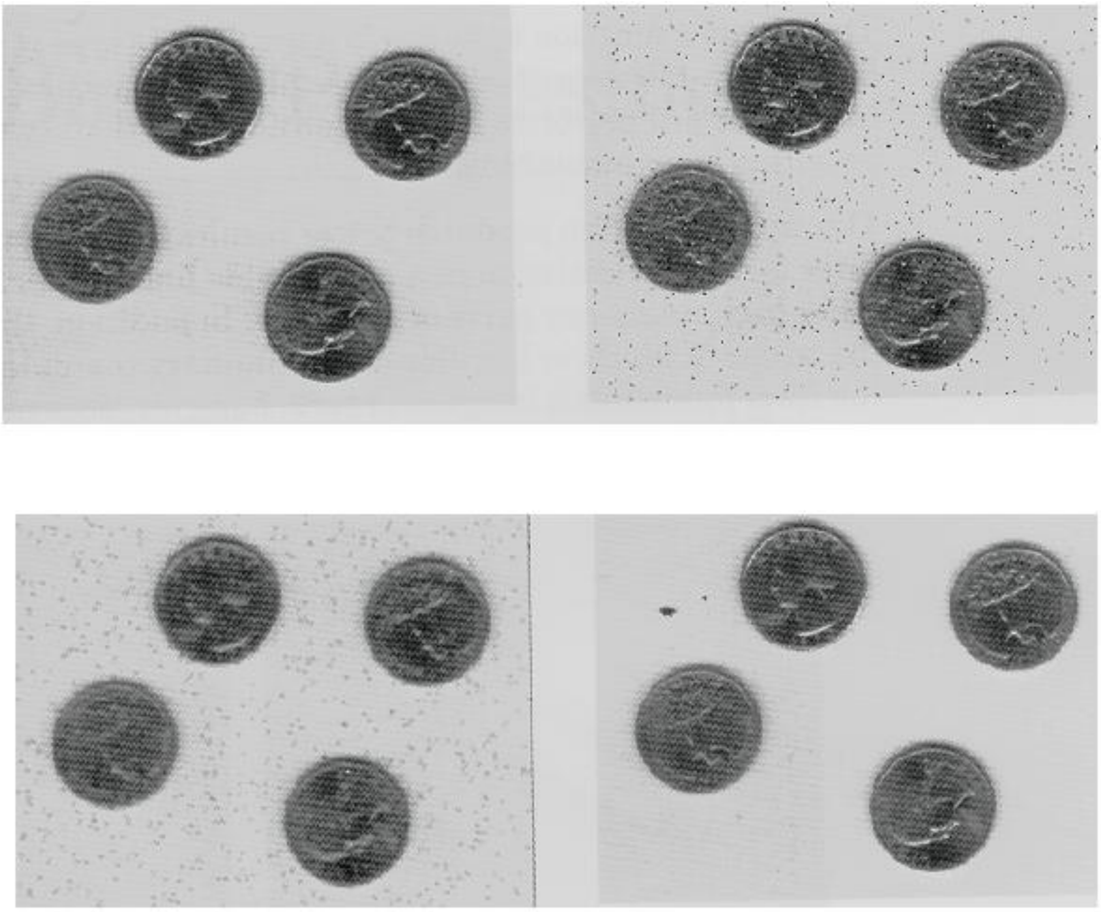


Figure 4.22: Example of Median filter

4.8 Optical Character Recognition

The optical character recognition is a recognition method in which the input is an image and the output is string of character. OCR is a process which separates the different characters from each other taken from an image. Template matching is one of the approaches of OCR. The cropped image is compared with the template data stored in database. OCR automatically identifies and recognizes the characters without any indirect input. The characters on the number plate have uniform fonts then the OCR for number plate recognition is less complex as compared to other methods.

4.8.1 Preprocessing:

Before preparing the template for each of the characters for further use, we need to do some processing on the images. The following are the operations that are performed Binarization, Inversion of intensity of the characters. Finding the connected component that represents the character. Finding the smallest rectangle enclosing this connected component. Normalization of the image to size 15 X 15. - Storing the intensity values using the below mentioned algorithm for each of the characters.

4.8.2 Creating the template:

In order to create the template for each character we do the following operation. For every white pixel we insert the value 1 and for every black pixel 0. We do this for all the training samples for each character and calculate the weights to get the template.

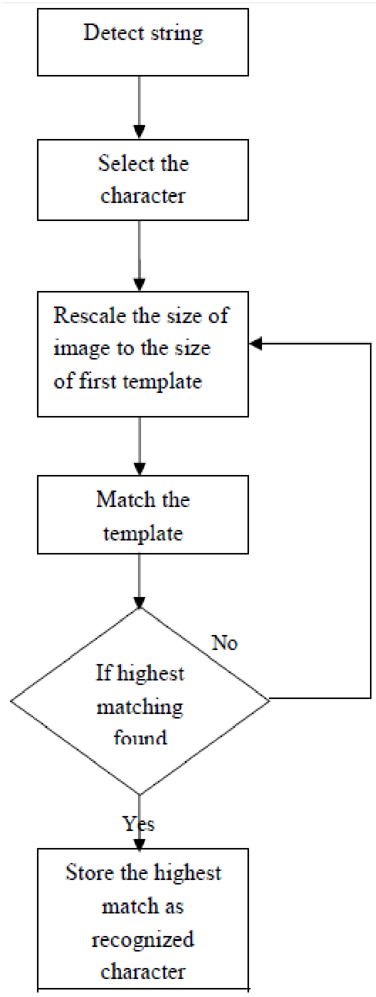


Figure 4.23: Template matching flow chart

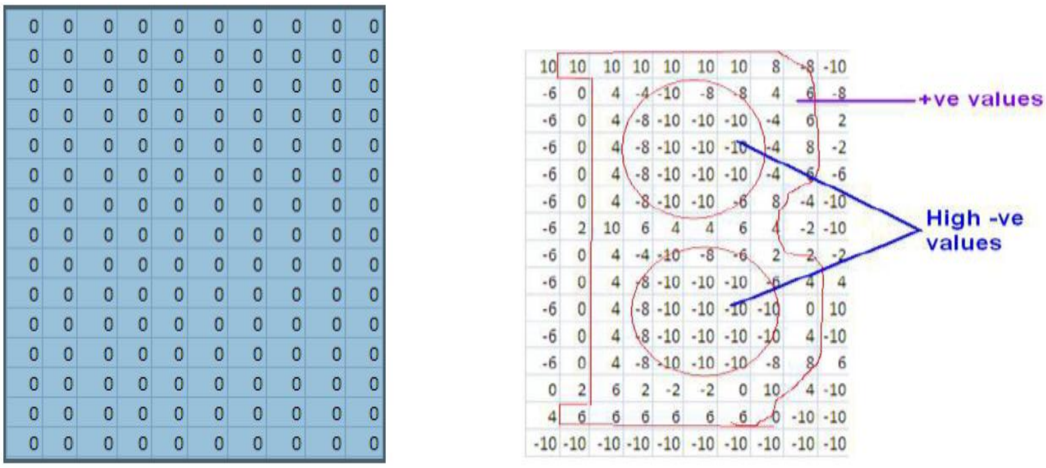


Figure 4.24: Template

left image shows the empty template right image shows Template after one sample of 'B'.

4.8.3 Character Recognition:

Preprocessing the image obtained after segmentation is Grayscale. Follow the preprocessing steps used for the training of the characters. Calculate the score for each of the characters then calculate the matching score of the segmented character from the templates of the character stored by the following algorithm. Compare the pixel values of the matrix of segmented character and the template matrix, and for every match add 1 to the matching score and for every miss-match decrement 1. This is done for all 225 pixels. The match score is generated for every template and the one which gives the highest score is taken to be the recognized character.

Figure 4.25: Car image

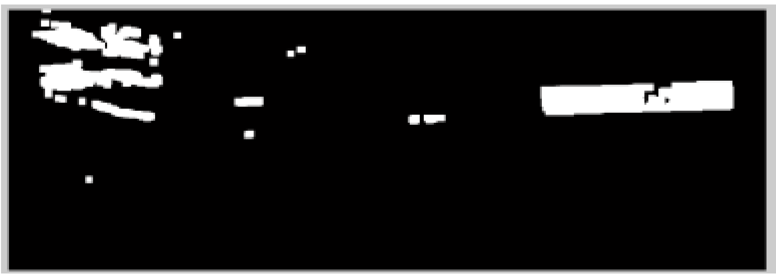


Figure 4.26: After Morpholofical operations and removal of noise

Figure 4.27: After segmentation

Figure 4.28: recognizing the character

The output of OCR on the segmented license plate shown above is:

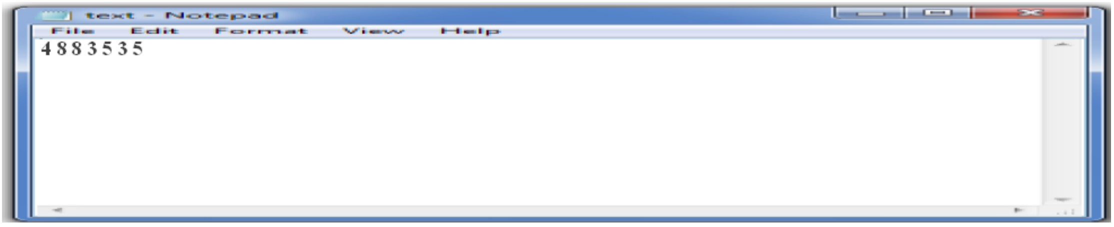


Figure 4.29: Output after OCR

CHAPTER 5
SERIAL COMMUNICATION

5.1 INTRODUCTION

Serial communication is defined as the transmission or reception of one bit of data at a time. Now a day's computer generally addresses data in byte format or some multiple thereof. A byte consists of 8 bits. A bit is basically either logic one or zero. Each character on this page is actually defined internally as one byte of data. The serial port communication is used to convert each single byte of data to a stream of ones and zeros otherwise to convert a stream of ones and zeros to bytes of data. The serial port consists of an electronic chip named as Universal Asynchronous Receiver/Transmitter (UART) that actually do the all these conversions.

The serial port cable is having many pins. Electrically speaking, whenever the serial port transmit a logical one (1) a negative type voltage is get activated on the transmission pin. Whenever the serial port transmits a logical zero (0) a positive voltage is get enabled. When no data is sent, then the serial ports transmit the pins voltage is negative voltage (1) and it is said to be it is in a mark state. The serial port can also interested to keep the transmit pin at a positive voltage (0) and it is said to be that it is in BREAK state or MARK state.

5.2 RS232:

For showing pin out of the RS 232 port, two pins are present. They are used for flow control. These two pins are one is RTS request to send, and other one is CTS, clear to send.

With DTE/DCE communication (it is communication between a computer and a modem device) RTS is an output of the DTE and input for the DCE. CTS are the response signal coming from the DCE. Before sending a character, DTE asks permission for its RTS output. None of information will be sent until the DCE gives permission by using the CTS line

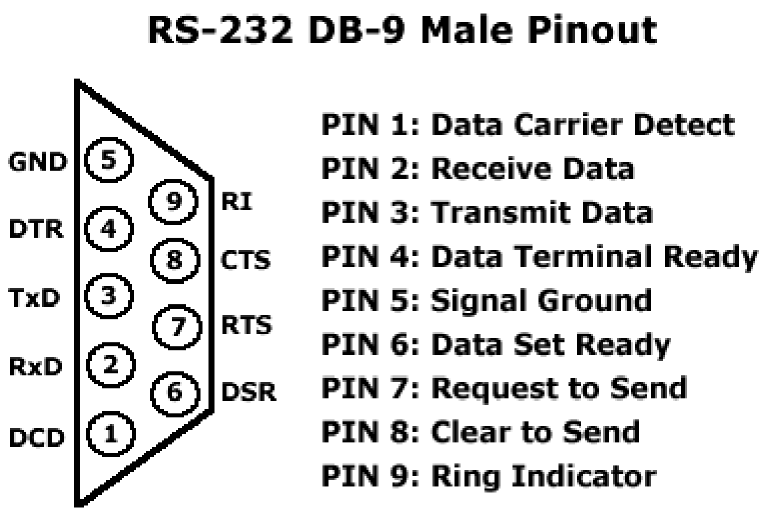


Figure 5.1Pin diagram of RS 232

For future controlling of the information flow, both devices are having the ability to indicate their status to the other side. For this reason the DTR data terminal ready and DSR data set ready signals are present. The two pins DTE uses the DTR signal, that it is ready for to receive the information, whereas the DCE uses the DSR signal for the same reason.

The pin diagram of RS232 is shown in above figure 5.1. The last flow control signal present in the DTE/DCE communication. It is the CD carrier detect. It is not used for flow controlling directly, but mainly an indicating the ability of modem device to communicate link between two modems devices.

5.3 Universal Asynchronous Receiver/Transmitter:

A Universal Asynchronous Receiver/Transmitter is known as UART could be a variety of constituent that translate the {data} or data is within the style of parallel and serial communication. UARTs are in the main employed in combination of communication standards like EIA, RS-232, RS-422 or RS-485. The universal name indicates that the information format and transmission speeds are get configurable and people are the particular electrical communication levels and these ways (differential communication etc.) are usually handled by employing a special driver circuit connected outwardly to the UART connection.

Normally a UART could be a part of associate an used for serial communications against computer or electronic equipment primarily based interface. UARTs are currently unremarkably employed in microcontrollers. A dual UART or it\'s conjointly known as DUART, which mixes with combination of 2 UARTs placed into one chip.

5.3.1 Transmitting and Receiving Of Serial Data Communication:

The UART receives bytes of data transmits the individual bits in an order. At the receiver side, a second UART re-collect the bits into complete bytes. Each UART should contain a register that is that the basic methodology of conversion between serial and parallel forms. A Serial transmission data is of digital data (bits) through one wire or another medium is for a lot of and a lot of price effective than the parallel transmission through the multiple wires.

Normally the UART doesn't generate or receive the external signals directly and it's used between totally different things of equipments. Separate interfaced devices are used to convert the signal levels of the UART from and to the external levels of communication. The external signals are is also of various forms. Those are RS-232, RS-422 and RS-485 from the EIA is that the examples of standards for voltage signaling. Some sort’s communication schemes don't use electrical wires. The opposite kind of signaling schemes are used to modulation of a carrier signal. Examples are modulation of audio signals. The resulting Communication is also "full duplex" (both causing and receiving at the same time) or "half duplex" (devices act sending and receiving).

5.3.2 Character framing:

Character Frame format for UART is shown in the below figure 5.2 the idle, no data state is having high-voltage, or powered. This line is held for top to point out that the line and transmitter both are not damaged. Each character within the data is sent as a bit format like logic low as said to be start bit, and a configurable number of knowledge or data bits (usually 8, but legacy systems can use 5, 6, 7 or 9), For an optional bit one or more logic1 stop bits. The beginning bit signals receiver that a new character is coming as input. Following five to eight bits are looking on the code set got employed, it represents the character. The below following data bits could also be a parity. The further one or two bits are always in the form of mark or logic1 condition and named as the stop bit(s). They signal receiver that the character is going too completed. Since the start bit is as logic low (0) condition and the stop bit is as in logic high (1) condition. There are always at least two conformed signal changes between these characters.

Figure 5.2 Frame Format of UART

Obviously, if any problem get occurs, if a receiver detects a line that it is low for more than one character time. Then it is called as a "break". It is normal to detect breaks to disable a UART or enable to an alternative channel. Sometimes remote devices are arranged to reset or shut down when it receives a break. Quality guaranteed UARTs can detect and create breaks.

CHAPTER 6

Field Programmable Gate Array (FPGA)

6.1 Field-Programmable Gate Array:

The FPGA is an integrated circuit which contains several (64 to over 10,000) identical logic cells that may be viewed as normal elements. Every logic cell will severally take on anyone of a restricted set of personalities. The all individual cells are interconnected with matrix of wires and programmable switches. User's style is enforced by specifying the easy logic perform for every cell and by selection of closing the switches within the interconnected of matrix. The logic array cells and interconnects type a fabric of basic building blocks for logic circuits. These complicated designs are created by combining of these basic blocks to make the specified circuit. Field Programmable means it performance is outlined by a users program instead of by the manufacturer of the device.

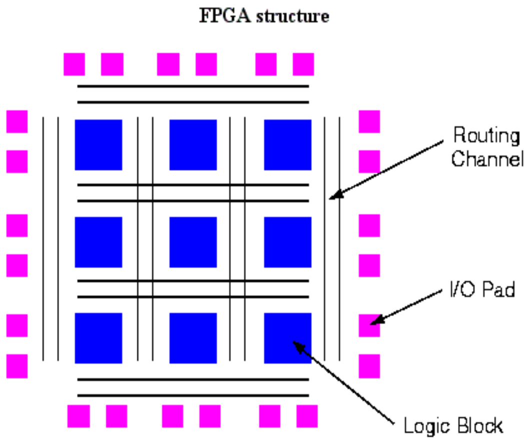


Figure 6.1 FPGA structure

A typical integrated circuit performs a selected perform defined at the time of manufacture. In contrast, the FPGA's perform is outlined by a program written by somebody apart from the device manufacturer.

Depending on the actual device, the program is either burned in permanently or semi-permanently as a part of a board assembly method, or is loaded from an external memory every time the device is high-powered up. This user programmability offers the user access to complicated integrated designs without the high engineering prices related to application specific integrated circuits.

Figure 6.1 shows the FPGA structure. The logic cell design varies between completely different device families. Generally speaking, every logic cell combines a number of binary inputs (typically between three and 10) to at least one or 2 outputs according to a Boolean logic perform lay out in the user program. In most of structures, the user additionally has the choice of registering the combinatorial output of the cell, so clocked logic is simply implemented. The cell's combinatorial logic is also physically implemented as a little look-up table memory (LUT) or as a group of multiplexers and gates. LUT devices tend to be somewhat a lot of versatile and supply a lot of inputs per cell than multiplexer cells at the expense of propagation delay.

6.1.1 FPGA Vs Microcontrollers:

Micro controllers are supported CPU design. As all CPUs, they execute instructions in a serial manner. FPGAs are programmable logic and run in a parallel fashion. Micro controllers have on-chip peripherals that also execute in parallel with their CPU. However they're still a lot of less configurable than programmable logic.

6.2 XILINX SPARTAN 3AN

6.2.1 On-Chip Features:

- 50,000-gate Xilinx Spartan 3AN FPGA in a 144-TQG (XC3S50AN-4TQG144C).

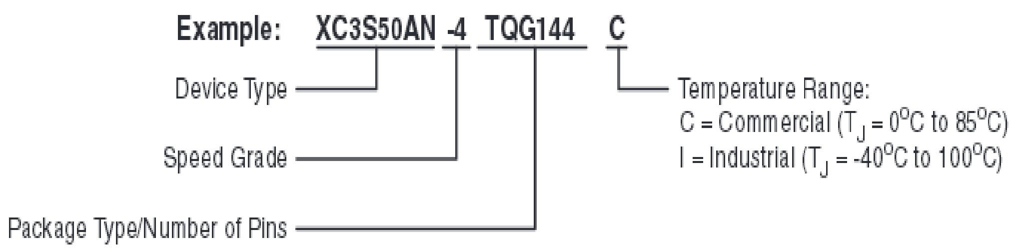


Figure 6.2: XC3S50AN-4TQG144C Numbering Format.

- 1,584 logic cell equivalents.
- 54K-bit block RAMs.
- Three 18x18 hardware multipliers.

- Two Digital Clock Managers (DCMs).Up to 108 user-defined I/O signals.

6.2.2 On-Board Features:

- 8 Nos. Slide Switches for digital inputs 8nos. of point LEDs for Digital outputs 2x16 Character digital display interface.
- 5x1 Matrix keypad interface RS232 port.
- 12-Bit SPI ADC 12-Bit SPI DAC Buzzer.
- 50 mhz quartz oscillator clock supply.
- 40-pin I/O connective for interface external peripherals modules JTAG port for transfer user program through cable.
- 9V AC/DC power input through adapter On-board 5V, 3.3V and 1.2V regulators.

6.2.3 Architectural Overview

The Spartan-3AN FPGA architecture is compatible with that of the Spartan-3A FPGA. The architecture consists of five fundamental programmable functional elements:

6.2.3.1 Configurable Logic Blocks (CLBs):

contain flexible Look-Up Tables (LUTs) that implement logic plus storage elements used as flip-flops or latches.

6.2.3.2 Input/Output Blocks (IOBs):

control the flow of data between the I/O pins and the internal logic of the device. IOBs support bidirectional data flow plus 3 state operation. They support a variety of signal standards, including several high-performance differential standards. Double Data-Rate (DDR) registers are included.

6.2.3.3 Block RAM:

provides data storage in the form of 18-Kbit dual-port blocks.

6.2.3.4 Multiplier Blocks:

accept two 18-bit binary numbers as inputs and calculate the product.

6.2.3.5 Digital Clock Manager (DCM) Blocks:

provide self-calibrating, fully digital solutions for distributing, delaying, multiplying, dividing, and phase-shifting clock signals.

These elements are organized as shown in Figure 6.3. A dual ring of staggered IOBs surrounds a regular array of CLBs. Each device has two columns of block RAM except for the XC3S50AN, which has one column. Each RAM column consists of several 18-Kbit RAM blocks. Each block RAM is associated with a dedicated multiplier. The DCMs are positioned in the center with two at the top and two at the bottom of the device. The XC3S50AN has DCMs only at the top.

The Spartan-3AN FPGA features a rich network of traces that interconnect all five functional elements, transmitting signals among them. Each functional element has an associated switch matrix that permits multiple connections to the routing.

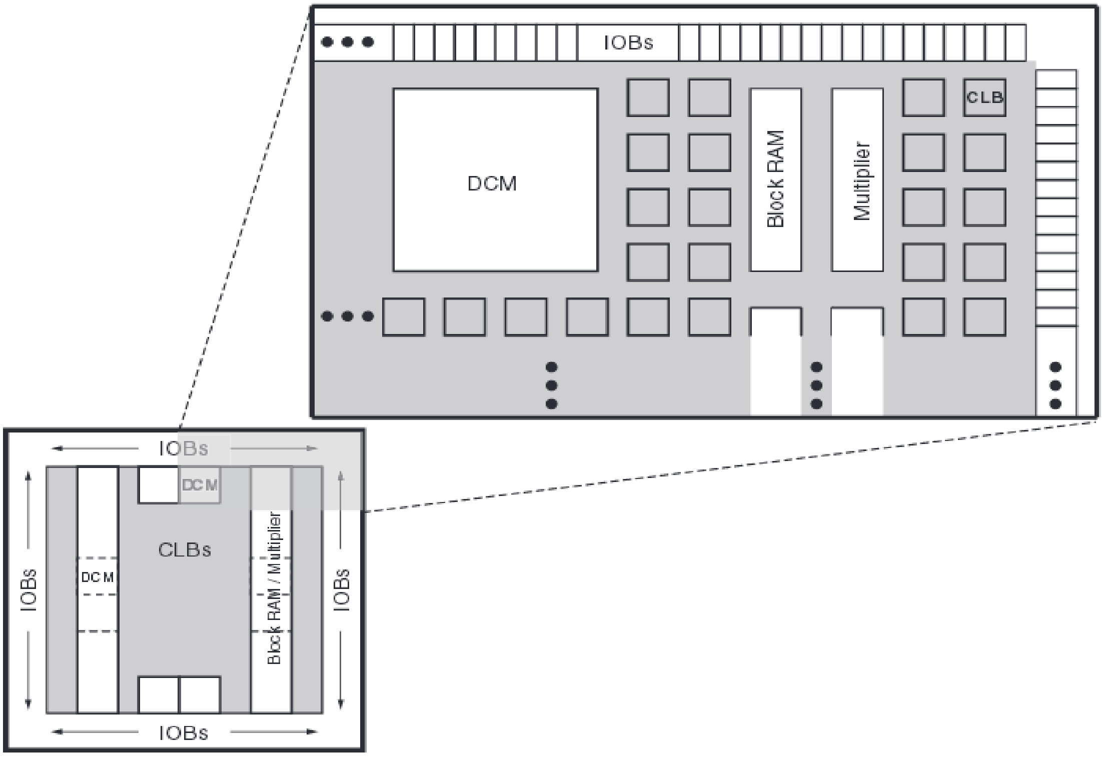


Figure6.3: Digital Clock Manager Blocks

6.2.4 Configuration

Spartan-3AN FPGAs are programmed by loading configuration data into robust, reprogrammable, static CMOS configuration latches (CCLs) that collectively control all functional elements and routing resources. The FPGA’s configuration data is stored on-chip in nonvolatile Flash memory, or externally in a PROM or some other nonvolatile medium, either on or off the board. After applying power, the configuration data is written to the FPGA using any of seven different modes.

- Configure from internal SPI Flash memory.
    - Completely self-contained
    - Reduced board space
    - Easy-to-use configuration interface
- Master Serial from a Xilinx Platform Flash PROM.
- Serial Peripheral Interface (SPI) from an external industry-standard SPI serial Flash.
- Byte Peripheral Interface (BPI) Up from an industry-standard x8 or x8/x16 parallel NOR Flash.
- Slave Serial, typically downloaded from a processor.
- Slave Parallel, typically downloaded from a processor.
- Boundary-Scan (JTAG), typically downloaded from a processor or system tester.

The MultiBoot feature stores multiple configuration files in the on-chip Flash, providing extended life with field upgrades. MultiBoot also supports multiple system solutions with a single board to minimize inventory and simplify the addition of new features, even in the field. Flexibility is maintained to do additional MultiBoot configurations via the external configuration method. The Spartan-3AN device authentication protocol prevents cloning. Design cloning, unauthorized overbuilding and complete reverse engineering have driven device security requirements to higher and higher levels. Authentication moves the security from bit stream protection to the next generation of design-level security protecting both the design and embedded microcode. The authentication algorithm is entirely user defined, implemented using FPGA logic. Every product, generation, or design can have a different algorithm and functionality to enhance security.

6.3 General Block Diagram:

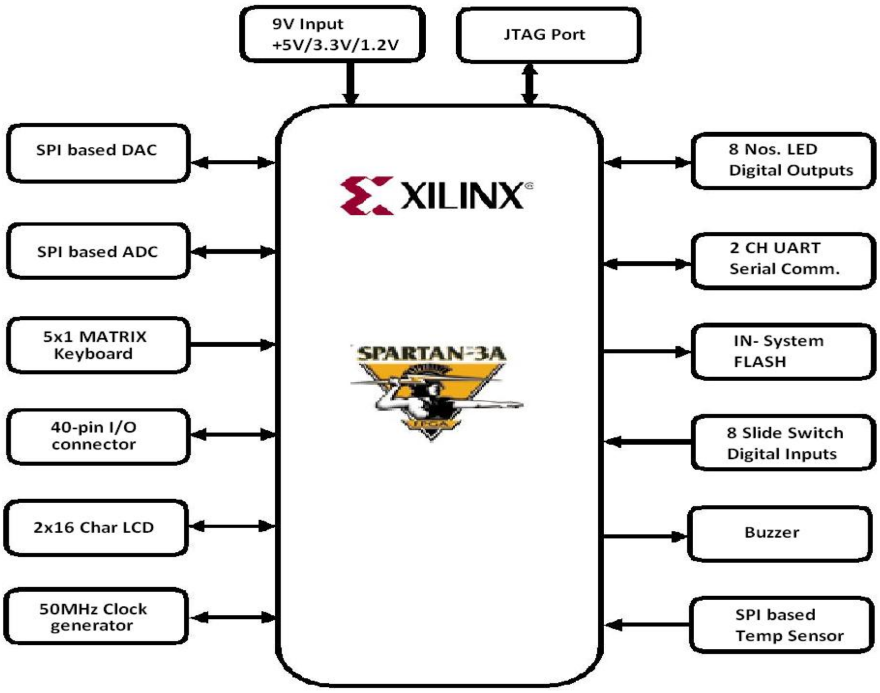


Figure 6.4: Block Diagram

6.3.1 40pin – Box Connector Details:

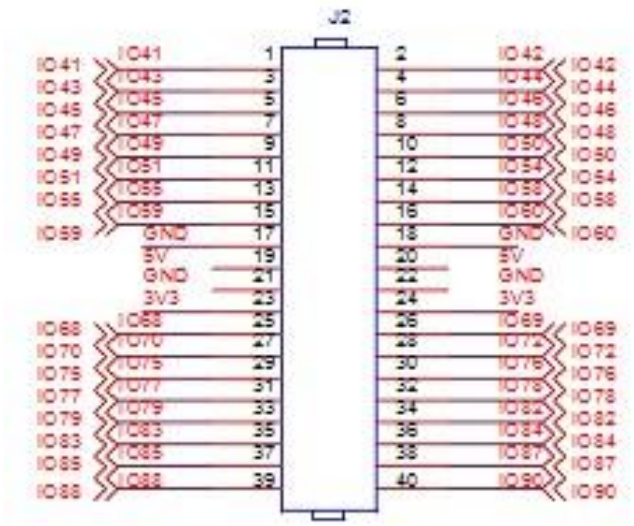


Figure6.5: 40 pins Connector

6.3.2 JTAG Connector:

JTAG IS Joint Test Action Group Dedicated pins are 4 per device. Not available as a user-I/O pin. Every package has four dedicated JTAG pins. These pins are powered by VCCAUX. The Pins are TDI, TMS, TCK, TDO.

Figure 6.6: JTAG Connector

6.3.3 JTAG Programmer:

The FPGA SP3 Kit contains a JTAG programming and debugging chain. JTAG cable is included as a part of the kit and connects to the JTAG header. DB-25 parallel port connector to 6 pin female header connector. The JTAG cable connected directly to the port of a pc and to a standard 6 pin JTAG programming header within the kit, it can program a devices that have a JTAG voltage of 1.8V or greater.

Table 6.1: JTAG Programmer

	6-Pin Header	JTAG Signals	SPARTAN3 FPGA Lines	JTAG Cable – III
JTAG	1	TMS	P1	
	2	TDI	P2	
	3	TDO	P107	
	4	TCK	P109	
	5	GND		
	6	VCC		

6.3.4. Power Supply:

The outer power will be AC current or Direct current, with a voltage range between (9V, 1A output) at 230V AC input. The SPARTAN 3AN board generates +5V using an LM7805 voltage regulator, that provides supply to the all peripheral devices. Individual On/Off Switch (SW1) for maintaining power to the board.

Table 6.2: Power supply

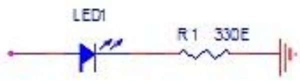
ON/OFF SW2		Power +5V **ON** - External through ***Adaptor***
		Power +0V ***OFF***

The multiple voltages supplied on the Spartan-3AN evaluation Kit are, 3.3V and 1.2V regulators. Similarly, the 3.3V regulator give's all the VCCO voltage provides inputs to the FPGA's I /O banks and powers most of the elements lies on the board. The FPGA interface on the board is controlled by 3.3V. Ultimately, a 1.2V regulator gives power to the FPGA's VCCINT voltage inputs, which control the FPGA's core logic. The board uses 3 distinct regulators to get the required voltages.

6.3.5. Light Emitting Diodes:

Light Emitting Diodes (LEDs) are the most used elements, usually for displaying pin's digital states.

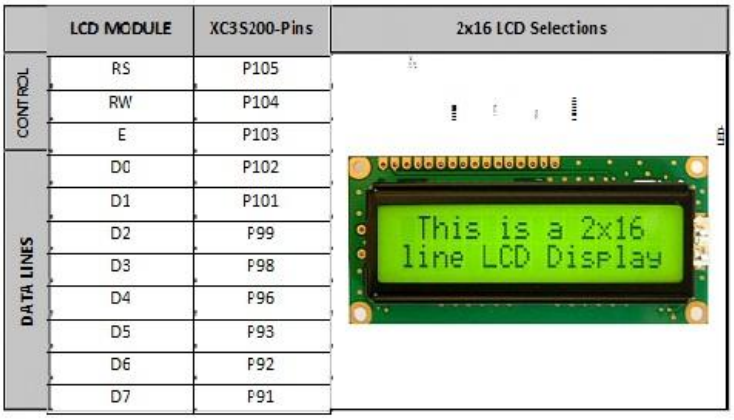
Table 6.3: LED Connection Table

	Point LEDs	XC3S200 - Pins	LED Selection
DIGITAL OUTPUTS	LED-LD1	P7	 Make Pin High – LED ON Make Pin Low – LED OFF
	LED-LD2	P8	
	LED-LD3	P10	
	LED-LD4	P11	
	LED-LD5	P12	
	LED-LD6	P13	
	LED-LD7	P15	
	LED-LD8	P16	

The FPGASP3 KIT has 8 point LEDs, connected with port pins (details tabulated above table); the cathode of every led connects to ground via a 330Ω resistance. To lightweight a personal led, drive the associated FPGA control signal to High state.

6.3.6. 2x16 Char LCD Display:

The 2x16 character LCD interface with support of each modes 4-bit and 8-bit interface. In eight-bit interface lines required to make 8-bit interfaces; 8 data bits (D0 – D7), 8 control lines, address bit (RS), read/write bit (R/W) and control signal (E). The digital display controller is a standard KS0070B or equivalent, it is a very well-known interface for smaller character based LCDs.



	LCD MODULE	XC3S200-Pins	2x16 LCD Selections
CONTROL	RS	P105	
	RW	P104	
	E	P103	
DATA LINES	D0	P102	
	D1	P101	
	D2	P99	
	D3	P98	
	D4	P96	
	D5	P93	
	D6	P92	
	D7	P91	

Figure 6.7: LCD Interface

6.3.7 Buzzer Interface:

Buzzer is connected to 5V continuous supply through FPGA's I/O pins (P5), to enable buzzer, place jumper JP7 is at E label mark position. Here Spartan3 FPGA pins (Buzzer – P5), make port pins to high, buzzer get enabled.

Table 6.4: Buzzer Interface Table

	5V Buzzer	XC3S50AN-pins	RELAY Power Select
	Buzzer	P64	J5 To Enable Buzzer

6.3.8 RS-232 Communication (USART):

USART stands for Universal Synchronous Asynchronous Receiver Transmitter. FPGA SP3AN Kit presents an RS232 port that may be driven by the Spartan-3AN FPGA. A set of RS232 signals is employed on the 3AN kit to implement RxD and TxD type of interfaces.

RS-232 communication enables point-to-point data transfer. It’s mainly employed in data possession applications, for the transfer of information between the microcontroller/FPGA and a pc. The voltage levels of a FPGA and computer aren't directly compatible with RS-232, here a level transition buffer like MAX3232 be used.

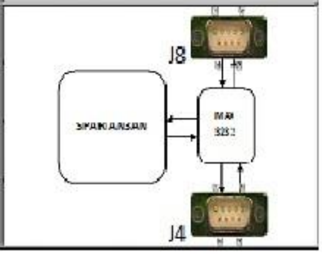
	UART DB-9 Connector	SPARTAN3AN FPGA Lines	Serial Port Section
UART	TXD0	P3	
	TXD1	P4	
	RXD0	P5	
	RXD1	P6	

Figure 6.8: RS-232 Communication

6.3.9 Clock Source:

The FPGASP3 Kit includes a dedicated 50 mhz series clock initiator source and an optional socket for another clock oscillator supply voltage.

	U18	Signal	SPARTAN3 FPGA Lines	Crystal Oscillator
	50MHz	Clock	P57	

Figure 6.9: Clock Source

6.3.10 In-System Flash Memory

Each Spartan-3AN FPGA contains abundant integrated SPI serial Flash memory, used primarily to store the FPGA's configuration bit stream. However, the Flash memory array is large enough to store at least two MultiBoot FPGA configuration bit streams or nonvolatile data required by the FPGA application, such as code-shadowed Micro Blaze processor applications.

After configuration, the FPGA design has full access to the in-system Flash memory via an internal SPI interface; the control logic is implemented with FPGA logic. Additionally, the FPGA application itself can store nonvolatile data or provide live, in-system Flash updates. The Spartan-3AN device in-system Flash memory supports leading-edge serial Flash features.

- Small page size (264 or 528 bytes) simplifies nonvolatile data storage
- Randomly accessible, byte addressable
- Up to 66 MHz serial data transfers
- SRAM page buffers
    - Read Flash data while programming another Flash page
    - EEPROM-like byte write functionality
    - Two buffers in most devices, one in XC3S50AN
- Page, Block, and Sector Erase
- Sector-based data protection and security features
    - Sector Protect: Write- and erase-protect a sector (changeable)
    - Sector Lockdown: Sector data is unchangeable (permanent)
- 128-byte Security Register
    - Separate from FPGA's unique Device DNA identifier
    - 64-byte factory-programmed identifier unique to the in-system Flash memory
    - 64-byte one-time programmable, user-programmable field
- 100,000 Program/Erase cycles
- 20-year data retention
- Comprehensive programming support
    - In-system prototype programming via JTAG using Xilinx Platform Cable USB and IMPACT software

- Product programming support using BPM Microsystems programmers with appropriate programming adapter
- Design examples demonstrating in-system programming from a Spartan-3AN FPGA application

6.4 Power Supply:

The power provides circuit is constructed using filters, rectifiers, and so voltage regulators as shown within the figure 6.10.

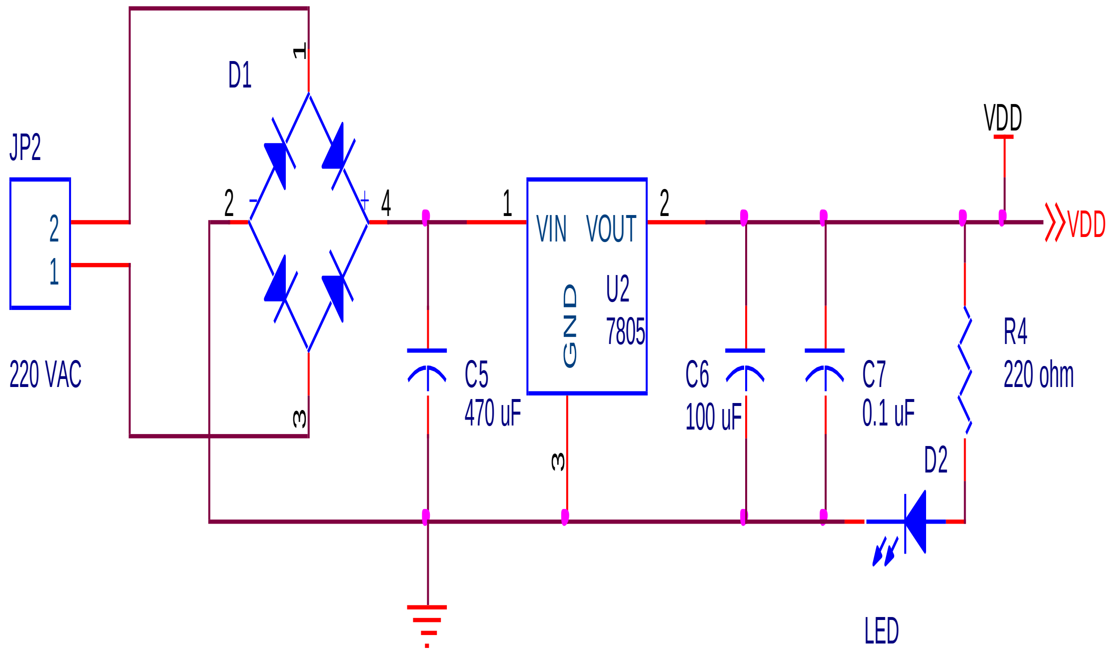


Figure 6.10: Power Supply Circuit

Beginning of an AC voltage, a study DC voltage is obtained by rectifying the AC voltage, then sort out to a DC level, and finally, regulation to obtain a desired fixed DC voltage. The regulation is usually obtained from an IC transformer Unit that takes a DC voltage and provides a somewhat lower DC voltage that remains constant even if the input DC voltage changes, or the output Load connected to the DC voltage may changes.

CHAPTER 7
HARDWARE MODULES

7.1 Pressure Sensor:

A piezoelectric sensor has shown in figure 7.1 is a tool that uses the piezo effect to measure pressure, acceleration, strain or force by changing them to an electrical charge. Here a simple pressure detector is employed to guard door or window. It generates a loud beep once someone tries to interrupt the door or window. The alarm stops mechanically after 3 minutes. The circuit uses a piezoelectric because the pressure detector.

Piezo buzzer develops the piezoelectric property of the electrical crystals. The piezo effect is also direct piezo effect during the electrical charge develops as a result of the automatic agent like reverse or indirect piezo effect (Converse piezoelectric effect) during which a mechanical force like pressure develops because of the appliance of an electrical field.

Figure 7.1 Piezo electric sensors

A typical example of direct piezo effect is that the generation of measurable quantity of piezo effect once the Lead Zirconate Titanate crystals are collapsed by mechanical or heat stress. The Lead Zirconate Titanate crystals are additionally show indirect piezoelectric effect by showing pressure once an electrical potential is applied.

7.1.1 Operation:

Operation of pressure detector is extremely simple and straightforward. Here we've 2 plates that's one is input plate and also the alternative is output plate, whenever pressure is applied as shown in figure 7.2 then these two plates inherit contact we get voltage as a output then this output is send to the FPGA successively it shows the weight of the vehicles is shown within the display as per our project. Consequently toll tax is calculated. It's created of a piezoelectric crystal. Depending on however the piezoelectric objects are cut, then three main modes of operations are distinguished. Those are namely transverse, longitudinal, and shear.

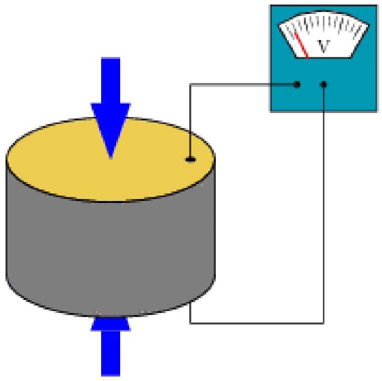


Figure 7.2 Pressure sensor operations

7.1.2 Transverse Effect:

A force is applied on a neutral axis and also the charges are generated on another direction perpendicular to the line of force. The amount of charge depends on the geometrical dimensions of the respective piezoelectric component.

7.1.3 Longitudinal Effect:

The amount of charge created is directly proportional to the applied force and is independent of size and it is in the form piezoelectric component. Development as many parts that are automatically in series and electrically in parallel with the only way to increase the charge output.

7.1.4 Application:

Vibration sensors may be used to collect otherwise wasted energy from automatic vibrations. This can be accomplished by using piezoelectric materials to convert mechanical strain into usable electricity.

7.1.5 Flow Chart for Pressure Sensor Module:

The figure 7.3 shows the entire process that takes place using the pressure sensor.

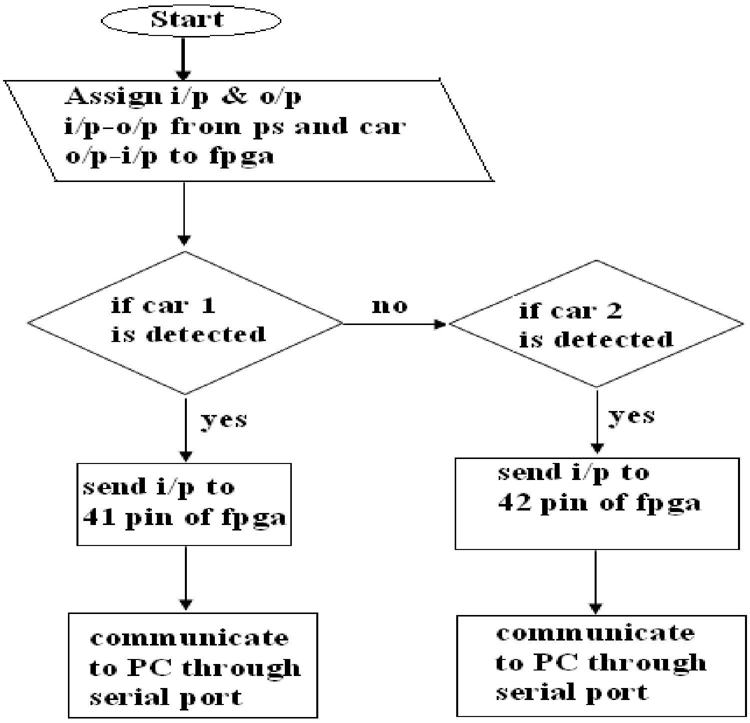


Figure 7.3: Flow chart for pressure sensor module

7.2 Gas Sensor:

The flammable Gas and Smoke sensors will notice the presence of flammable gas and smoke at concentrations from 300 to 10,000 ppm. Attributable to its simple analog voltage interface, the detector needs one analog input pin from the FPGA.

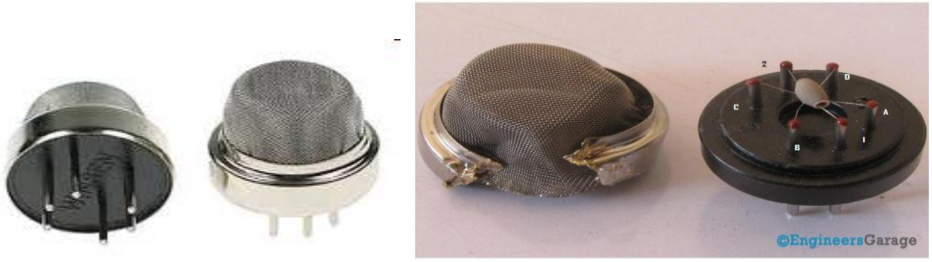


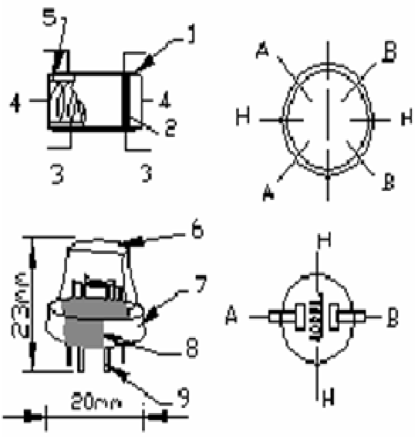
Figure 7.4 Gas Sensor (type-MQ2)

The product will notice the pressure of the smoke and send the output within the form of analog signals. Our range will perform at temperature starting from -20 to 50 degree Centigrade and consume but 150 mA at 5V.

Sensitive material of MQ-2 gas detector within the figure 7.4 is SnO_2 (Tin dioxide), which with lower conductivity in clean air. Once the target flammable gas exist, the sensor's conductivity is higher along with the gas concentration rising. MQ-2 gas detector has high sensitivity to LPG, gas and hydrogen, additionally might be accustomed methane.

7.2.1 Structure and Configuration:

Structure and configuration of MQ-2 gas detector is shown as figure 7.5, detector composed by small AL_2O_3 ceramic tube, Tin dioxide (SnO_2) sensitive layer, measure electrode and heater are fixed into a crust created by plastic and stainless-steel net. The heater provides necessary work conditions for work of sensitive elements. The enclosed MQ-2 has 6 pin, four of them are used to fetch signals, and alternative 2 are used for providing heating current.



	Parts	Materials
1	Gas sensing layer	SnO2
2	Electrode	Au
3	Electrode line	Pt
4	Heater coil	Ni-Cr alloy
5	Tubular ceramic	Al2O3
6	Anti-explosion network	Stainless steel gauze (SUS316 100-mesh)
7	Clamp ring	Copper plating Ni
8	Resin base	Bakelite
9	Tube Pin	Copper plating Ni

Figure 7.5 Structure and Configuration of M0Q2 sensor

7.2.2 Characteristic Configuration:

- Good sensitivity to flammable gas in wide selection
- High sensitivity to LPG, propane and hydrogen
- Long life and low price
- Simple drive circuit

7.2.3 Sensitivity Characteristics:

Figure 7.6 shows the standard sensitivity characteristics of the MQ-2, ordinate suggests that resistance ratio of the sensor (Rs/Ro), abscissa is concentration of gases. Rs suggest that resistance in several gases, Ro suggests that resistance of sensor in 1000ppm hydrogen. All tests are under normal test conditions.

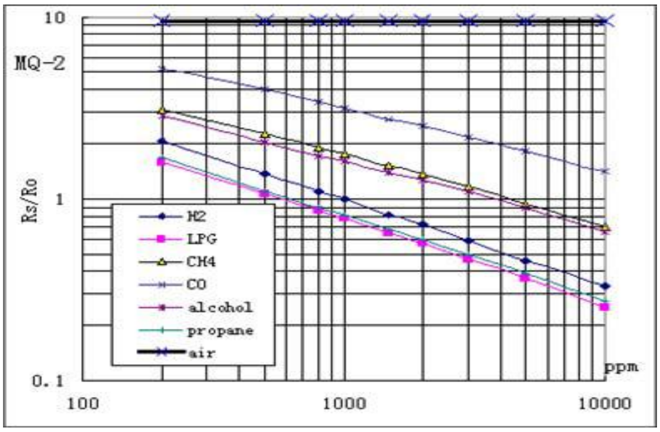


Figure 7.6 Sensitivity Characteristics of MQ2 sensor

7.2.4 Applications:

- Domestic gas discharge detector
- Industrial flammable gas detector
- Portable gas detector

7.2.5 Flow Chart for Gas Sensor Module:

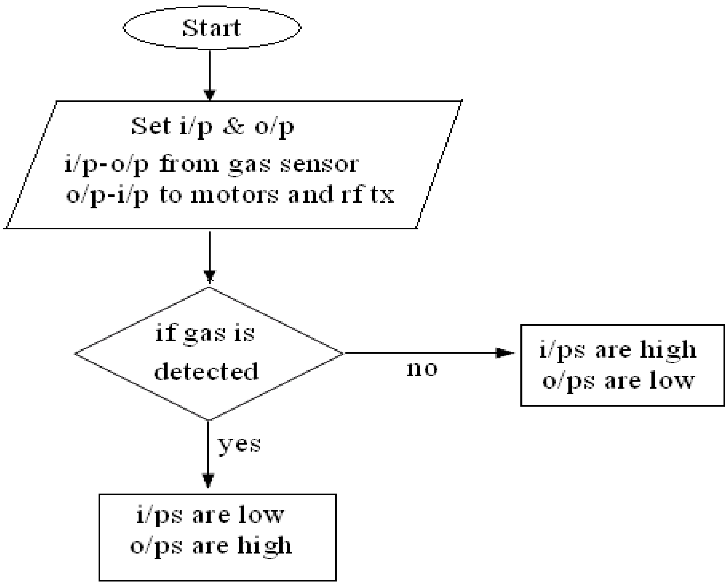


Figure 7.7: Flow chart for the gas sensor module

7.3 LM 324:

LM324 may be a 14pin IC consisting of 4 independent operational amplifiers (op-amps) paid during a single package. Op-amps are high gain electronic voltage electronic equipment with differential input and typically a single- concluded output. The output voltage is {many times repeatedly persistently again and again over and over} over the voltage distinction between input terminals of an op-amp. These op-amps are operated by one power provide and thus the requirement for a dual supply is eliminated. They will be used as amplifiers, comparators, oscillators, rectifiers etc. The figure 7.8 shows the pin diagram and pin details of LM 324.

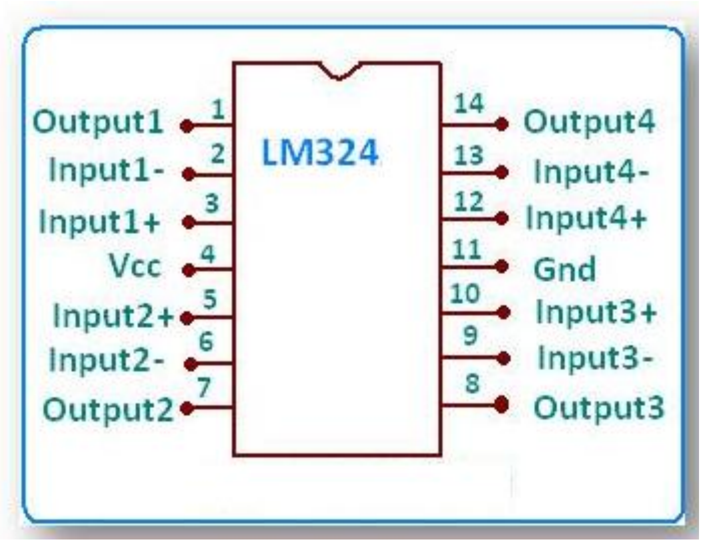


Figure 7.8 pin diagram and pin details of LM 324

7.3.1 Operation:

The LM324 series consists of independent, high gain, internally frequency compensated operational amplifiers that are designed in particular to control from one power provide to wide variety of voltages. Operation from divide power supplies in moreover possible and also the low power supply current drain is becomes independent of the magnitude of the power supply voltage. Application areas are include transducer amplifiers, DC gain blocks and every one of the conventional op amp circuits that currently is a lot of easily implemented in single power supply systems.

7.3.2 Advantages:

- Eliminates need for would like dual
- Four internally compensated op amps during a single package
- Allows directly sensing close to GND and VOUT automatically goes to GND.
- It is similar with all forms of logic
- Power drain is appropriate for battery operations.

7.4 LM 358:

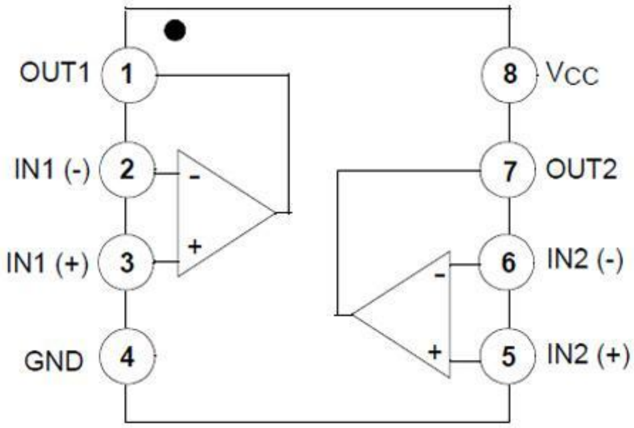


Figure 7.9 Pin details of LM 358

LM 358 series consists of two independent temporary, high gain, internally frequency compensated operational amplifiers which are designed particularly to control from one power provide over a better range of voltages. Operation from split power provides is additionally to possible and the lower power supply current drain is independent of the magnitude of the power supply voltage. The figure 7.9 shows the pin diagram and pin details of LM 358.

7.4.1 Advantages:

- Two internally compensated op amps.
- Eliminates would like for dual supplies.
- Allows for direct sensing close to sensing close to GND and Vout also goes to GND.
- It is similar for the all kinds of logic.
- Power drain is suitable for battery operation.

7.5 Wireless Communication Modules:

A general RF communication diagram is shown in figure 7.10. Since most of the encoders/decoders/microcontrollers are TTL compatible and mostly inputs by the user are going to be given in TTL logic level. Thus, this TTL input is to be converted into serial information input using AN encoder or a microcontroller. This serial information will be directly read using the RF Transmitter, that then performs ask (in some cases FSK) modulation on it and transmit the information through the antenna. Within the receiver side, the RF Receiver receives the modulated signal through the antenna, performs all types of process, filtering, reception, etc and provides out a serial data. This serial information is then converted to a TTL level logic data, that is that the same data that the user has input.

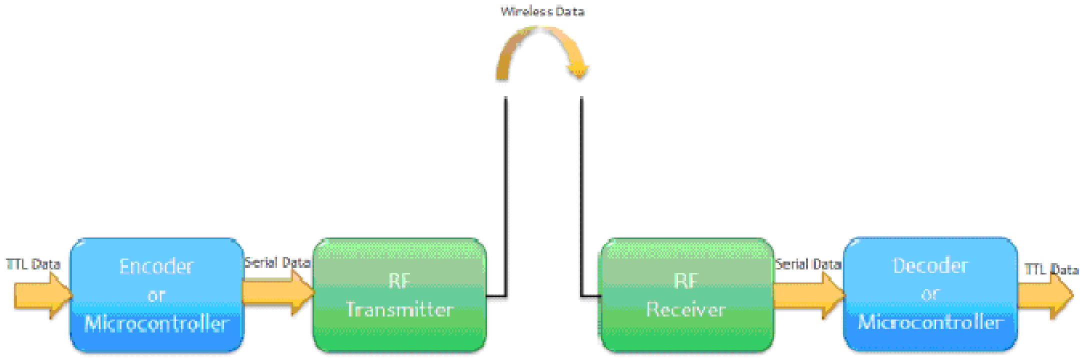


Figure 7.10 RF communication block diagram

7.5.1 RF Transmitter:

An RF transmitter generates frequency waves in its circuits, and also generates carrier signal, it adds the information part by modulating the carrier signal. This composite signal (carrier plus information) is then fed to AN antenna. The antenna makes a corresponding signal into the atmosphere, by altering the electrical and Magnetic fields at constant frequency. The pin diagram of RF transmitter is shown in figure 7.11 and also the pin details are given in table 7.1.

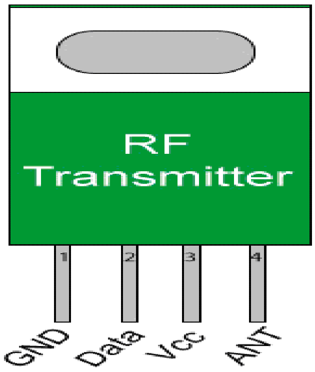


Figure 7.11 Pin diagram of RF transmitter

Table 7.1 Pin details of RF transmitter

Pin No.	Name	Function
1	GND	Ground (0V)
2	DATA	Serial Data Input
3	Vcc	Supply Voltage (5V)
4	ANT	Antenna Output

The following are its features.

- Range in Standard Conditions: 100 Meters
- Transmitter Frequency variation: 433.92 MHz
- Supply Voltage: 3V ~ 6V
- Output Power: 4 ~ 12 Dbm
- Low Power Consumption
- Easy For Application

7.5.2 RF Receiver:

An RF receiver receives the signal from its own antenna in an atmosphere. The receiver antenna is usually quite easy and also the signal level is often of a few micro volts. This gets eliminate the discarded signals and amplifies only the required ones. The receiver circuits then strip the information the signal from the carrier part. The pin diagram of RF receiver is shown in figure 7.12 and the pin details are given in table 7.2.

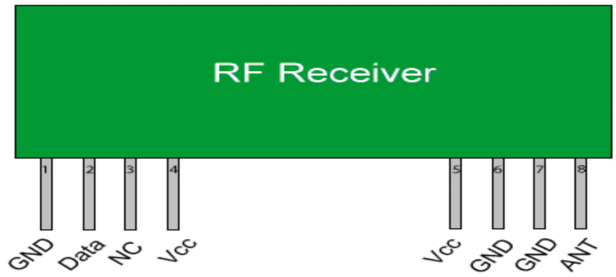


Figure 7.12 pin diagram of RF receiver

Table 7.2 Pin details of RF receiver

Pin No.	Name	Function
1	GND	Ground (0V)
2	DATA	Serial Data Output
3	NC	No Connection
4	Vcc	Supply Voltage (5V)
5	Vcc	Supply Voltage (5V)
6	GND	Ground (0V)
7	GND	Ground (0V)
8	ANT	Antenna Input

The following are its features

- Range in open space (Standard Conditions): 100 Meters
- Receiver Frequency Range: 433 mhz
- Sensitivity: 105 Dbm
- Current: 3.5 mA
- Frequency Range: 1MHz
- Low Power Consumption
- Easy For Application
- operating Voltage: 5V

7.6 HT12E Encoder:

The HT12E Encoder ICs are in series of CMOS LSIs for remote control system applications. They're capable of encoding 12 little bit of information that consists of N address bits and 12-N information bits. Every address/data input is externally binary programmable if bonded out. The programmed addresses/data are transmitted together with the header bits via an RF transmission medium Transmission is enabled by applying a low signal to the TE pin. The figure 7.13 gives the pin details of the encoder and the figure 7.14 shows the interfacing between HT-12E and RF Transmitter.

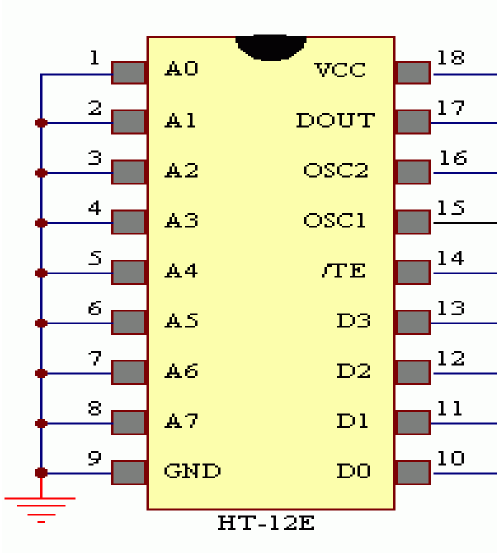


Figure 7.13 pin details of HT12E

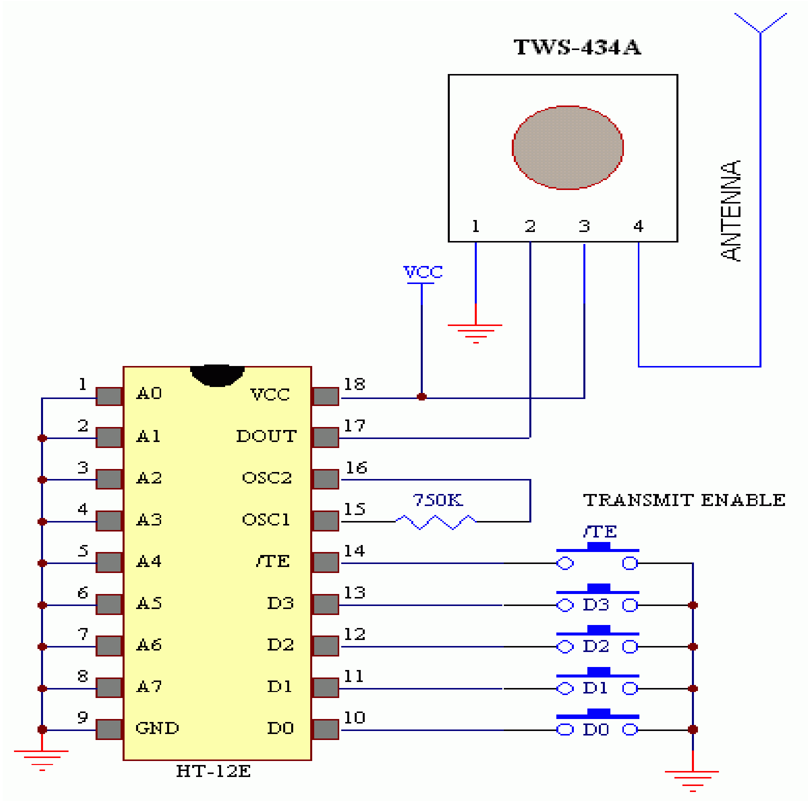


Figure7.14: Interfacing between HT-12E & RF Transmitter

The following are its features.

- Voltage: 2.4V ~ 12V
- Low Power and High Noise Immunity.
- CMOS Technology.
- Low Standby Current and Minimum Transmission Word.
- Built-in oscillator wants only 5-hitter resistor.
- Easy Interface with and RF or an Infrared transmission medium.
- Minimal External elements.

7.7 HT12D Decoder:

The HT12D Decoder ICs are in series of CMOS LSIs for remote control system applications. These ICs are paired with one another. For correct operation, a combine of encoder/decoder with constant range of address and data format ought to be selected. The Decoder receives the serial address and information from its resultant decoder, transmitted by a carrier using AN RF transmission medium and provides output to the output pins once after the information. The figure 7.15 gives the pin details of the decoder and figure 7.16 shows the interfacing between HT-12D and RF Receiver.

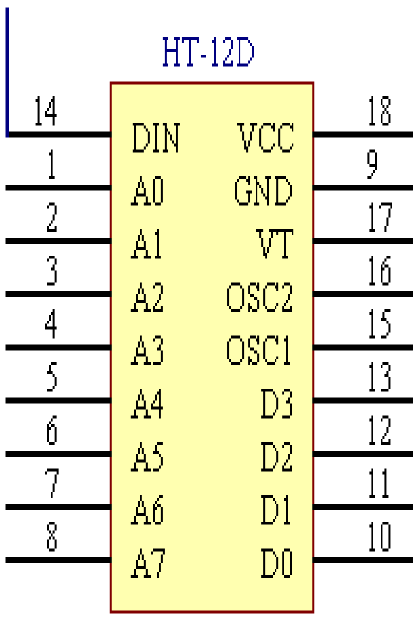


Figure 7.15 Pin details of HT12D

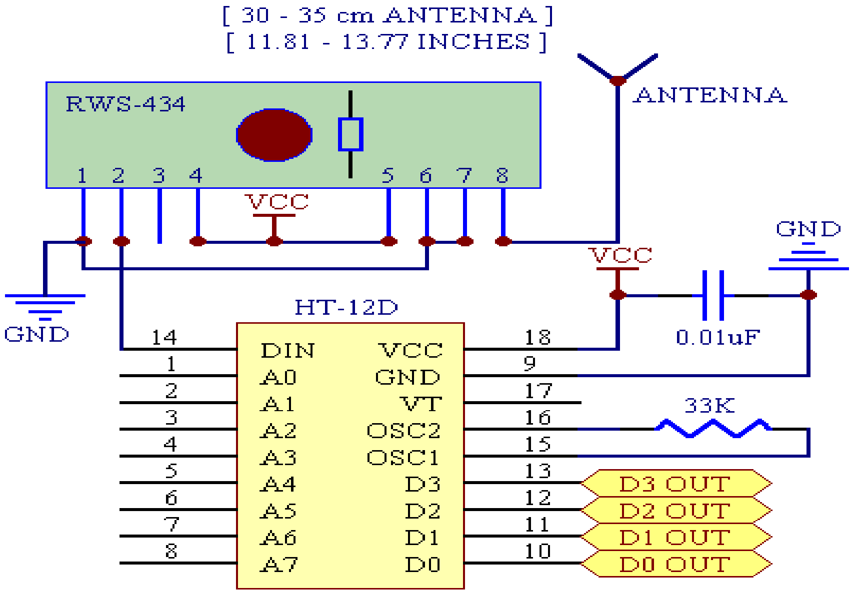


Figure7.16 : Interfacing between HT-12D & RF Receiver

The following are its features.

- 18 PIN DIP.
- Operating Voltage: 2.4V ~ 12.0V.
- Low Power and High Noise exception.
- CMOS Technology.
- Low Stand by Current.
- Ternary address setting.
- Capable of decoding 12 bits of data.
- 8 ~ 12 Address Pins and 0 ~ 4 information Pins.
- Received information are checked two times, built in oscillator wants only 5-hitter resistor.
- TV goes high during a valid transmission.
- Easy Interface with an RF of IR transmission medium.
- External elements.

7.8 DC Motor:

The steel will forms the body of the motor in addition to AN axle, a nylon end cap and 2 battery leads. If the battery leads of the motor are connected to a flashlight battery, the axle can spin. If leads are reversed, it'll spin within the opposite direction. Here are 2 different views of the same motor. The nylon finish cap is held in place by 2 tabs that are a part of the steel will.

Figure 7.17 parts of a DC motor

By bending the tabs back, the end cap will be made free and remove it. Within the end cap the motor brushes are present. These brushes are transferred the power from the battery to the commutator because the motor spins. The figure 7.17 shows all the parts of the DC motor.The axle holds the armature and also the commutator. The armature could be a set of electromagnets, during this case 3. The in this during this motor could be a set of thin metal plates stacked along, with thin copper wire twisted around every of the 3 poles of the armature. The 2 ends of every wire are get connected onto a terminal, and then every of the 3 terminals is wired to at least one plate of the commutator. The inside structure of the motor is shown in figure 7.18.

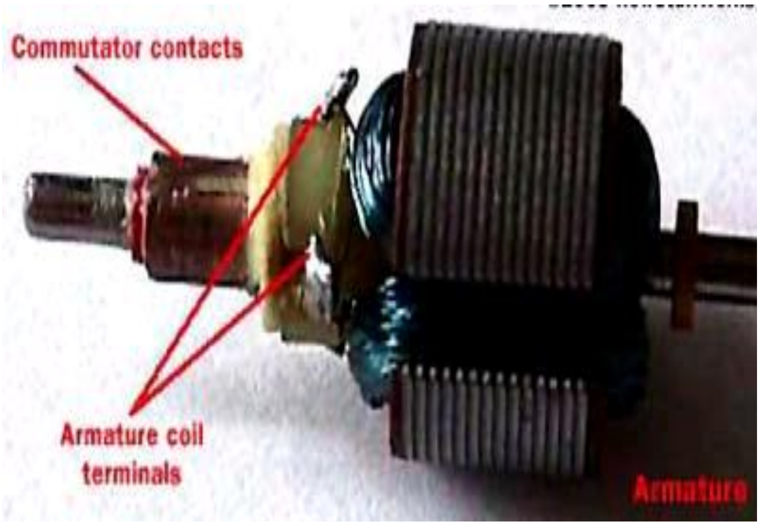


Figure7.18: Inside the motor

The final piece of any DC motor is magnet. The field magnet in this motor is made by itself and with 2 curved permanent magnets. One end of each magnet rests against a slot cut into the will, and other one is holding clip presses against both of the magnets.

7.8.1 Armature, Commutator & Brushes:

The figure 7.19 shows the armature connection of the motor. The armature takes the place of the spike in an electrical motor. The armature is an electromagnet created by coiled thin wire around 2 or more poles of a metal core. The armature has an axle, and also the commutator is connected to the axle. In the diagram to the right, we be able to see 3 completely different views of the same armature: front, side and end-on. In the end-on read, the winding is eliminated to create the switch more obvious. You’ll be able to see that the switch is simply a pair of plates connected to the axle.

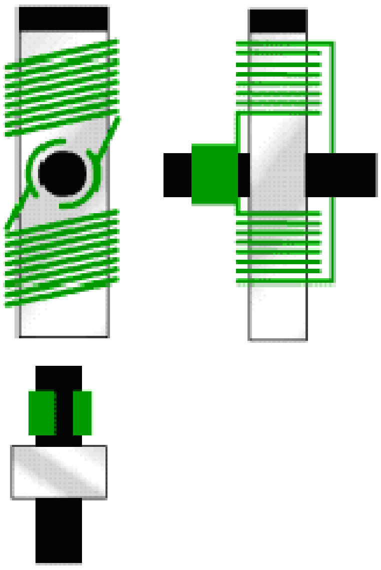


Figure 7.19 Armature

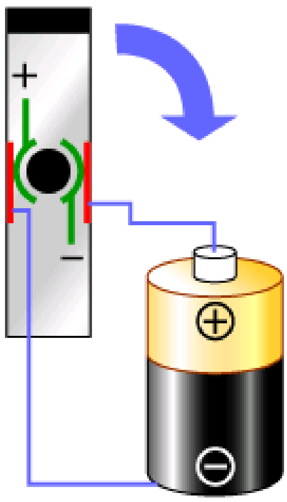


Figure 7.20 commutator and brushes

These plates offer the 2 connections for the coil of the magnet. The "flipping the electrical field" a part of an electrical motor is accomplished by 2 parts: the commutator and also the brushes. The figure 7.20 shows however the commutator and brushes work along to let current flow to the electromagnet, and additionally to flip the direction, the electrons are flowing at the right moment. The contacts of the commutator are connected to the axle of the magnet, so that they spin with the magnet.

The brushes are simply 2 pieces of springy metal or carbon those are build contact with the switch. Once we place all of those elements together, a whole motor is obtained. The coil winding has been left out in order to see the commutator in action mode. The key issue to note that as the coil passes through the horizontal position. As a result of the flip, the north pole of the magnet is above the shaft so it will repel the field magnet's pole and attract the sphere magnet's South Pole.

When we take apart a little motor, we'll find that it contains constant items described above: two tiny permanent magnets, a commutator, two brushes, and an electromagnet made by winding wire around a piece of metal. However, the rotor can have 3 poles rather than the two poles as shown during this article. There are two good reasons for a motor to have 3 poles:

1) It causes the motor to have better dynamics. During a two-pole motor, if the magnet is at the balance point, then it is perfectly horizontal between the two poles of the field magnet. when the motor starts, you can imagine the coil getting "stuck" there. That never happens during a three-pole motor.
2) Each time the commutator hits the point wherever it flips the field during a two-pole motor, the commutator shorts out the battery for a moment. This shorting is get wastage of energy and drains the battery unnecessarily.
3) A three-pole motor solves this problem as well. It is possible to possess any number of poles, depending on the dimensions of the motor.

7.9 RELAY

The operation of a relay will be understood by explaining the following diagram.

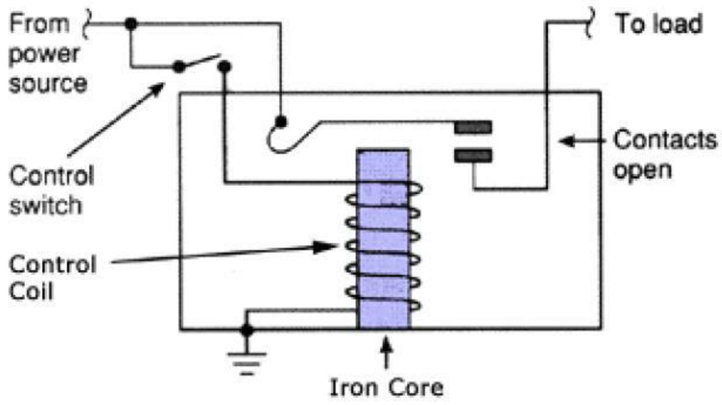


Figure 7.21 Relay Design

The diagram shows an inner part of a relay. An iron core is enclosed by a control coil. The power supply is given to the electromagnet through a control switch and through connected to the load. When current starts flowing through the coil, the electromagnets starts energizing and thus make strong of the magnetic field. Thus the upper contact arm starts to be attracted to the lower fixed arm and it closes to the contacts causing a short circuit for the power to the load. On the other hand, if the relay was already de-energized when the connections were dis connected, then the connection moves oppositely and makes it as an open circuit.

As soon as the coil current is off, the variable armature will be returned back to its initial position. This force will be almost equal to half of the strength of magnetic force. This force is mainly offered by two factors. They are the spring and gravity. These Relays are mainly used for two basic operations. One is low voltage application and the other is high voltage. For low voltage appliances, more preference will be given to reduce the noise of circuit. For high voltage applications, they're mainly designed to scale back a phenomenon it is named as arcing.

7.9.1 Relay Basics:

The basics for all the relays are stable. Take a look at a 4 – pin relay shown below. There are 2 colours are present in diagram. The green colour represents the feedback loop and also the red colour represents the load circuit. A little control coil is connected to the control circuit. A switch is getting connected to the load. This switch is controlled by the coil within the feedback loop. Let us allow us to take the various steps that occur in a relay.

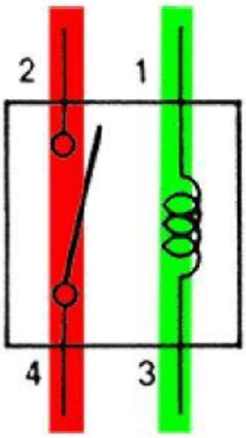


Figure 7.22 Relay operation

7.9.1.1 Energized Relay (ON):

As shown in the circuit, the current flowing through the coils indicated by 4 pins one and three causes a flux to be aroused. This magnetic field causes the short circuit of the pins two and four. So the switch plays a very important role in the relay operating. Because it could be a part of the load circuit, it's used to control a circuit that's connected to it. Thus, once the relay in energized this flow are going to be through the pins two and four.

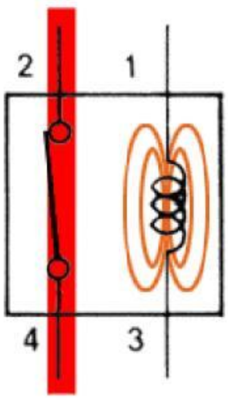
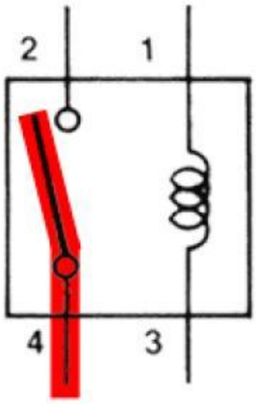


Figure 7.23 Energized Relay (ON)

7.9.1.2 De – Energized Relay (OFF):

As presently as the current flow stops through pins one and three, the switch opens and so the open circuit prevents this flow through pins two and four. So the relay becomes de-energized and so in off position.



\

Figure 7.24 De-Energized Relay (OFF)

Once a voltage is applied to pin one, the electromagnet get activated, and it causes a magnetic field to be developed, which fixed on to shut the pins two and four causing a closed circuit. Once there is no voltage on pin one, there will be no magnetic force and no flux. So the switches get open.

7.9.2 Pole and Throw:

Relays have the exact operation of a switch. So, constant idea is additionally applied to this. A relay is maintained to modify one or more poles. Every pole has get contact that may be thrown in mainly 3 ways. They are

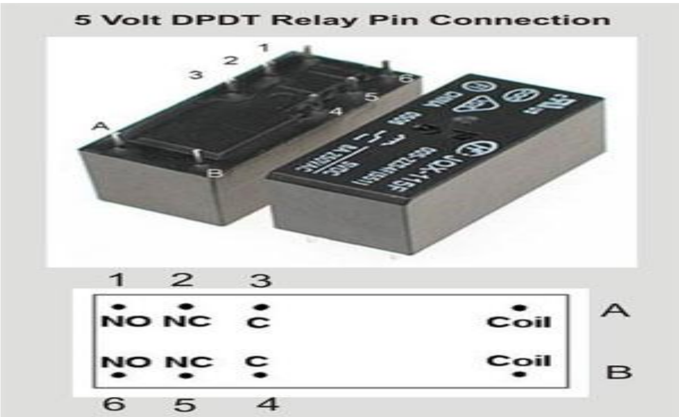


Figure 7.25 : 5 Volt DPDT Relay Pin Connection

Normally Open (NO) – no contact is also referred to as a make contact. It closes the circuit when the relay is got enabled. It disconnects the circuit when the relay is inactive.

Normally Closed (NC) – NC contact is also named as break contact. This is often opposite to the NO contact. When the relay is activated, the circuit got disconnects. When the relay is deactivated, the circuit connects.

Change-Over (CO) / Double-Throw (DT) - this kind of contacts are used to control 2 types of circuits. They are used to control a NO contact and a NC contact with a normal terminal. According to this type they are referred by the names as “break before make and make before break” contacts.

Relays also are named with designations like

Single Pole Single Throw Switch (SPST) -This type of relay has a total of 4 terminals. Out of those 2 terminals will be connected or disconnected. The opposite 2 terminals are required for the coil.

Single Pole Double Throw Switch (SPDT) –This type of a relay has a total of 5 terminals. Out f these 2 are the coil terminals. A standard terminal is additionally included that connects to either of 2 others.

Double Pole Single Throw Switch (DPST) -This relay has containing overall six terminals. These terminals are divided into 2 pairs. So they will act as 2 SPST’s that are motivated by one coil. Out of the six, 2 are coil terminals.

Double Pole Double Throw Switch (DPDT) -This relay is that the biggest of all switches. It is having eight relay terminals. Out of those 2 rows are designed to be modification over terminals and designed to act as two SPDT relays that are motivated by one coil.

7.9.3 Relay Applications:

- Used to notice logic functions. They play a truly crucial role in providing safety critical logic.
- Relays are used to offer the time delay functions. They are used to time the delay open and delay shut of connections.

- Used to control high voltage circuits with the help of low voltage signals. Equally they are used to control high current circuits with the help of low current signals.
- They also used as protecting relays. By this perform all the faults throughout transmission and reception will be detected and isolated.

CHAPTER 8
RESULTS

8.1 Hardware Screenshots

The results obtained using our proposed system is as follows.

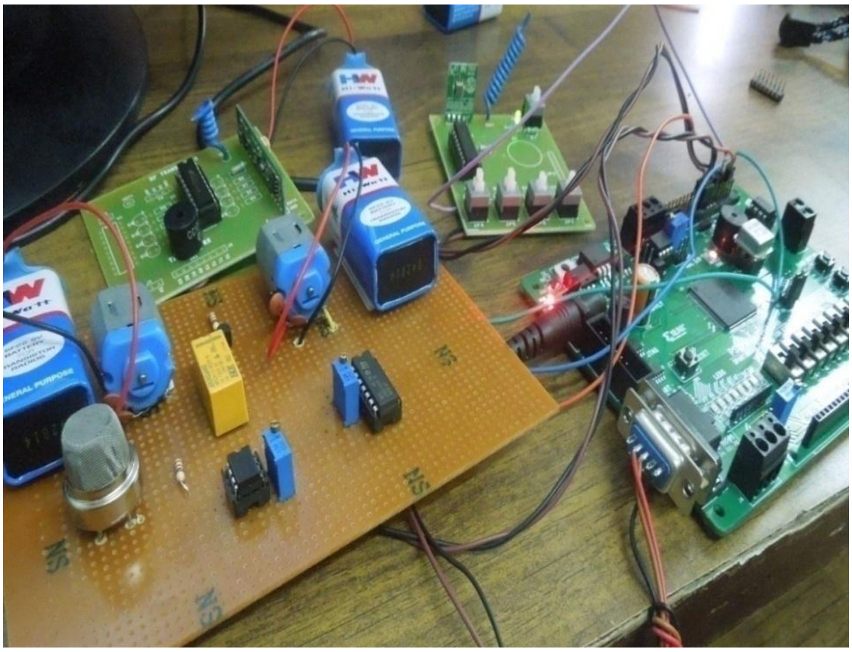


Figure 8.1 Overall Hardware

In the above figure consist of a FPGA, RF transmitter, RF receiver, RFID Reader, gas sensor, pressure device, motors and relay that are connected along.

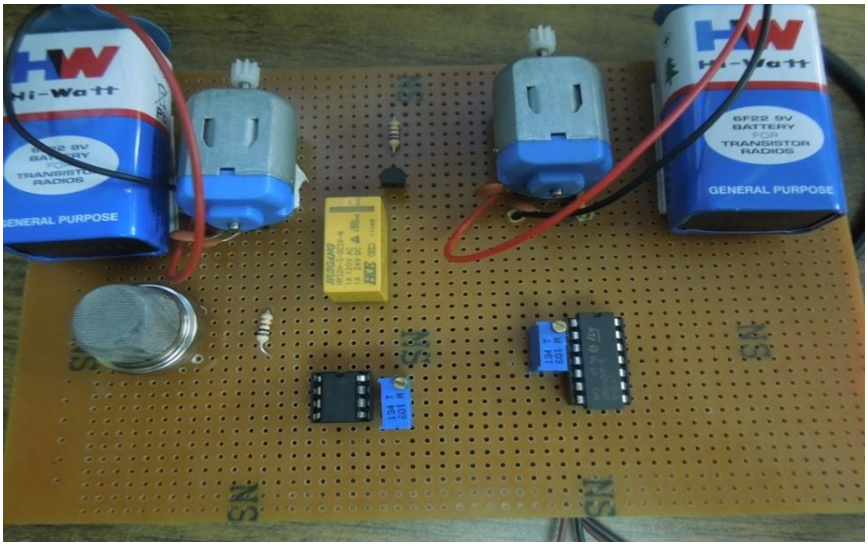


Figure 8.2 Motor and Gas sensor Module

Figur 8.3 RF Transmitter

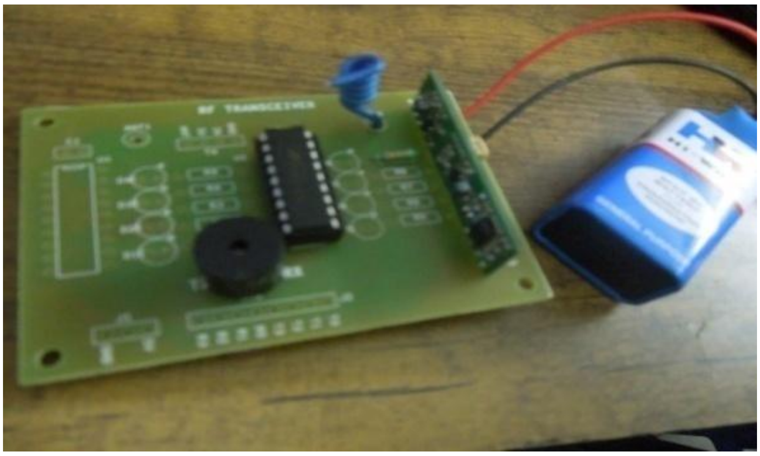


Figure 8.4 RF receiver

The above figure shows the motor and gas module, RF transmitter, RF receiver. These modules work at the same time that is controlled by FPGA. The information is compared with the information in pc database. If the vehicle is found to be stolen then it alters the surroundings and through the RF transmitter it signals the RF receiver that is within the near police station. Equally the same procedure is carried out once a gas is detected by a gas device.

8.2 GUI screenshots:

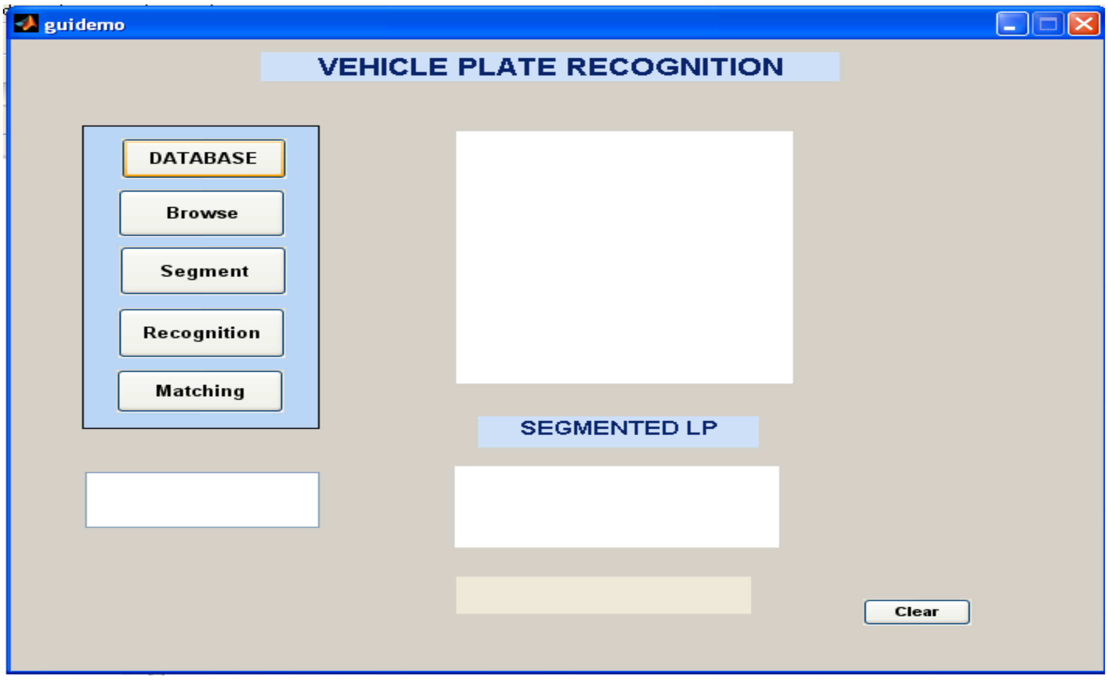


Figure 8.5 Creating DataBase

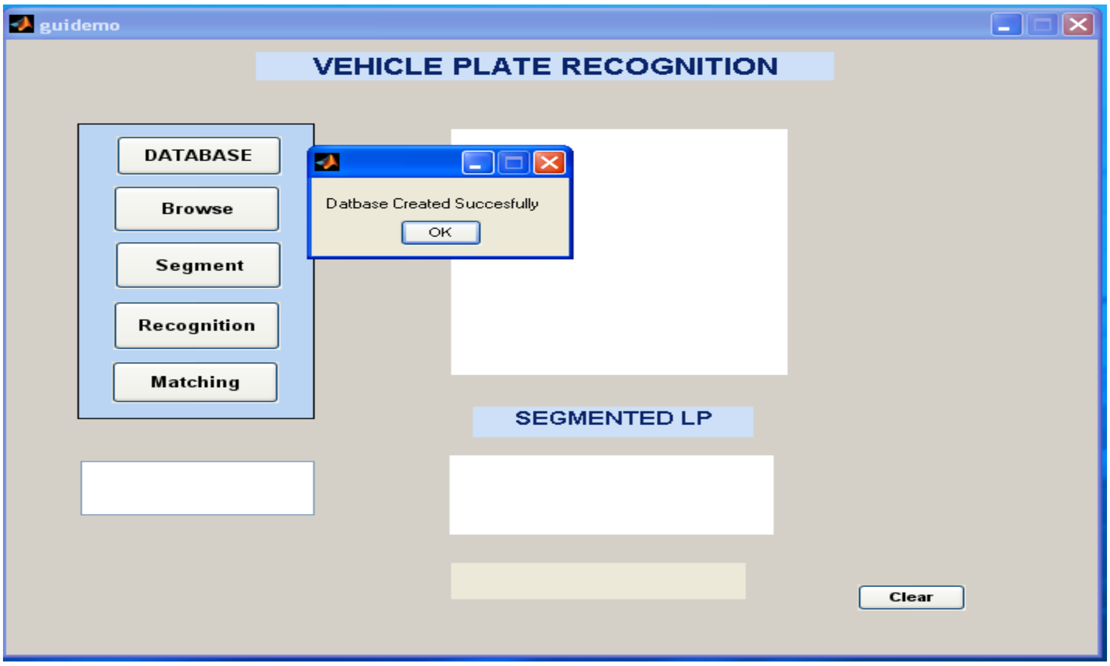


Figure 8.6 DataBase created Successfully

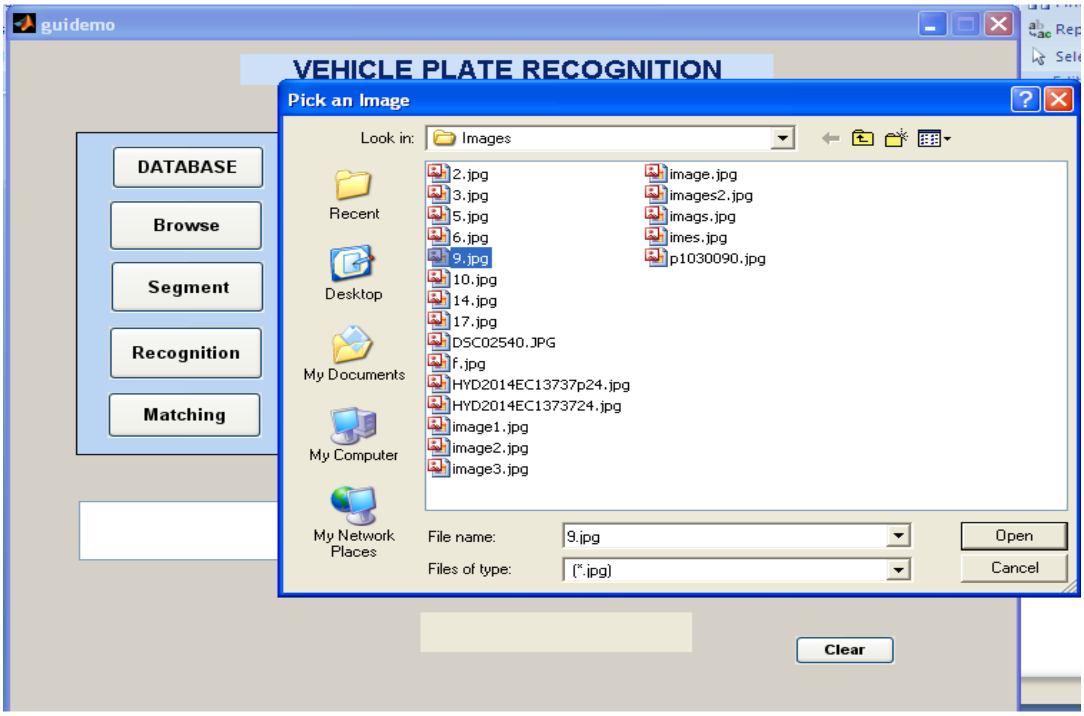


Figure 8.7 Browse an image

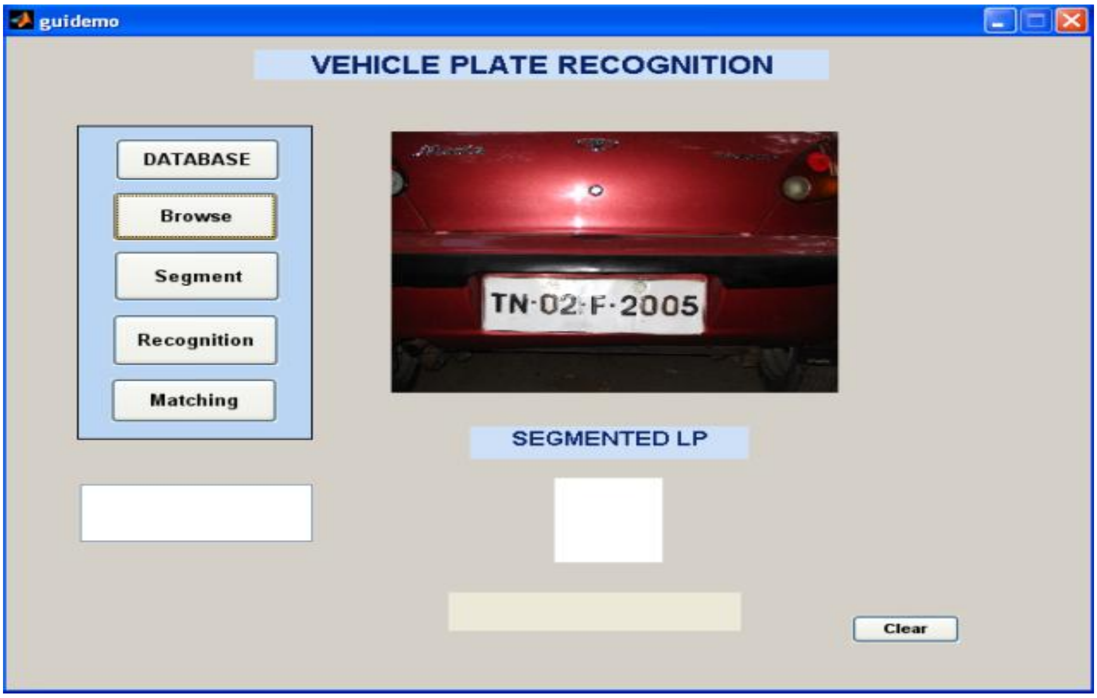


Figure 8.8 Select an image

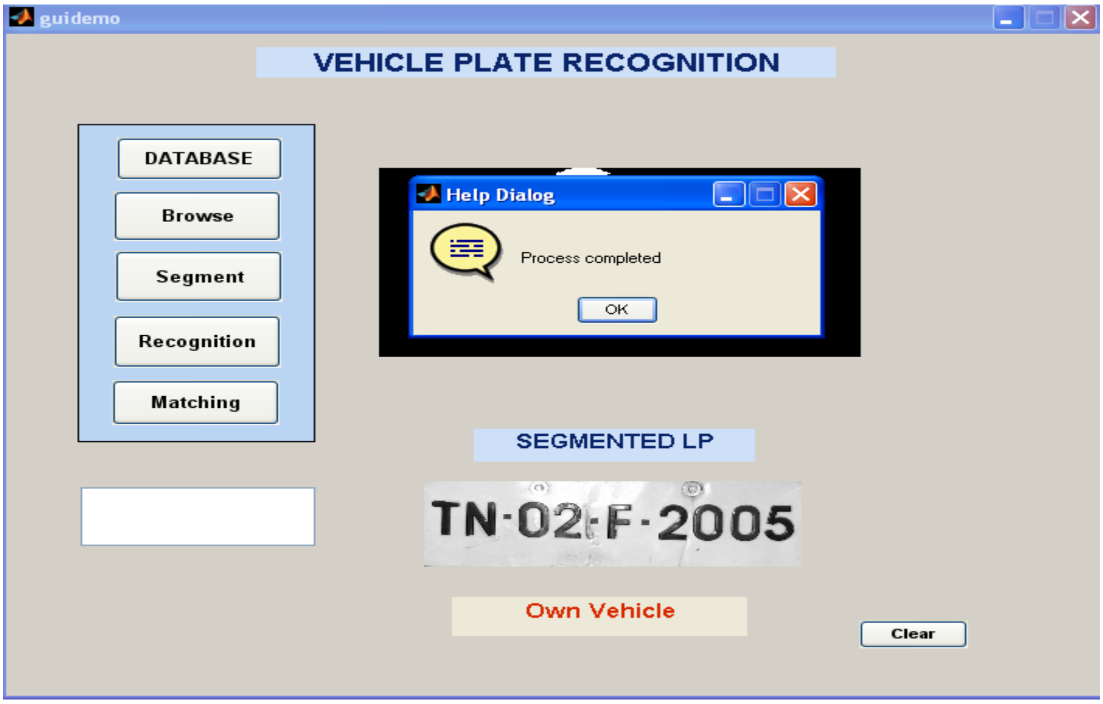


Figure 8.9 Segment the image

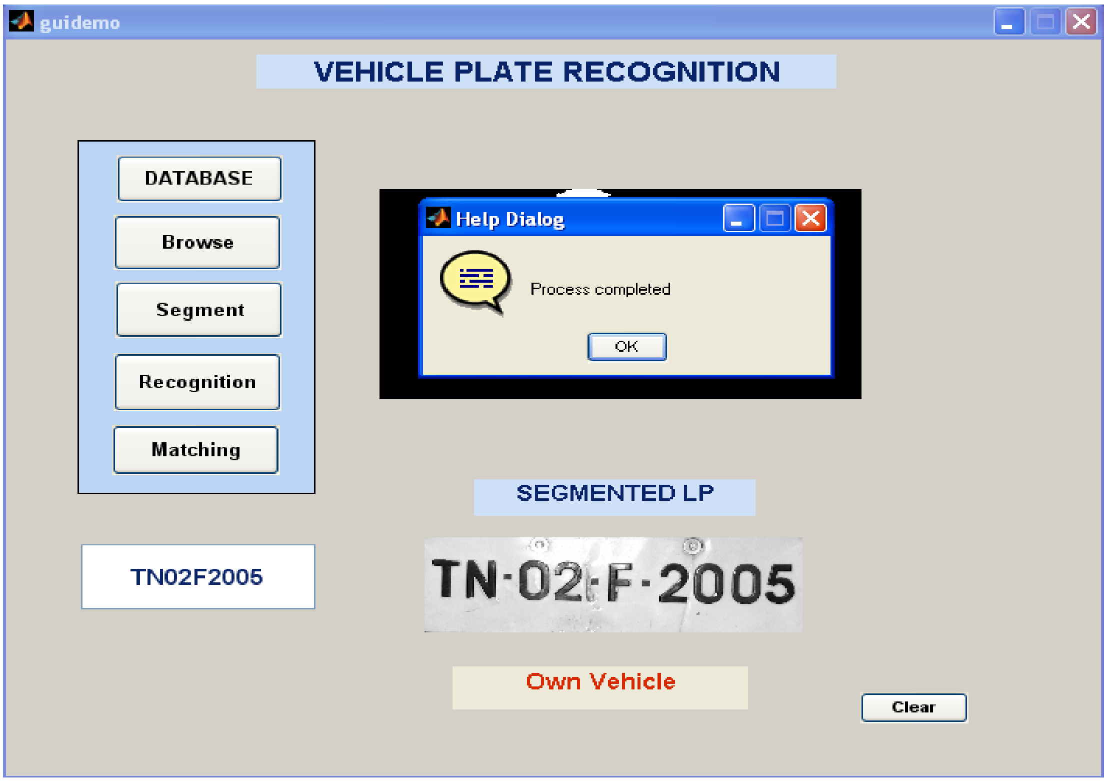


Figure 8.10 Recognize the image & Click on matching

Pressure device is employed to detect the pressure of the vehicle and according to the toll tax is detected from their account mechanically. The program is loaded into the FPGA 1st using Xilinx code. Then using visual basic we tend to implement in system. Visual basic window is opened 1st. After the reader identifies the tag the vehicle details are displayed. When the pressure is given the balance amount is subtracted.

Figure 8.11 Output (Amount and pressure chart)

CHAPTER 9
CONCLUSION

9.1 Conclusion

Our system could be a user friendly toll fee technique which may save time and reduce traffic congestion at toll gates solution resolution for users to succeed in their destination without wastage of your time. It offers the toll authorities the flexibility to line variable pricing for toll services and so a good policy of tax collection can be followed. This manner there's no loss incurred by an individual carrying a vacant vehicle. With the hike of fuel prices in mind, consumption of fuel is additionally considered here because the speed, acceleration and idling is totally eliminated. Here there's no money dealings for the toll gates, thus handling of money is reduced. So difficulties with money handling are eliminated and pricing aid in increased audit control by centralizing user accounts. Info such as vehicle count over the time, date, time etc will be obtained due to the deployment of this technology. This helps in creating decisions regarding the pricing ways for the toll providers. It additionally helps plan and to estimate the travel time that aid in designing decisions.

9.2 Future Enhancement:

In future the speed of the process can be increased more. Similarly add an extra feature like in order to detect the arms that are carried.

CHAPTER 10

REFERENCES

[1]Kin Seong Leong, MunLeng Ng, *Member*, *IEEE*, Alfio R. Grasso, Peter H. Cole, "Synchronization of RFID Readers for Dense RFID Reader Environments", International Symposium on Applications and the Internet Workshops (SAINTW'06), 2005

[2]Manish Buhptani, ShahramMoradpour, "RFID Field Guide - Developing Radio Frequency Identification Systems", Prentice Hall, 2005, pp 7-9, 16-225, 160, 231

[3]Raj Bridgelall, Senior Member, IEEE, " Introducing a Micro-wireless Architecture for Business Activity Sensing ", IEEE International Conference RFID, April 16-17,2008

[4]Sewon Oh, Joosang Park, Yongioon Lee, "RFID-based Middleware System for Automatic Identification", IEEE International Conference on Service Operations and Logistics, and Information, 2005

[5]Shi-Cho Cha Kuan-Ju Huang Hsiang-Meng Chang, " An Efficient and Flexible Way to Protect Privacy in RFID Environment with Licenses ", IEEE International Conference RFID, April 16-17,2008

UrachadaKetprom, ChaichanaMitrpant, PutchapanLowjun, “Closing Digital Gap on RFID Usage for Better Farm Management”, PICMET 200