DESIGN AND IMPLEMENTATION OF EMBEDDED BASED ELEVATOR CONTROL SYSTEM

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology

in

VLSI Design and Embedded System

By

RAJESH KUMAR PATJOSHI

Roll No: 208EC216

Department of Electronics & Communication Engineering

National Institute of Technology

Rourkela

2010

DESIGN AND IMPLEMENTATION OF EMBEDDED BASED ELEVATOR CONTROL SYSTEM

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology

in

VLSI Design and Embedded System

By

RAJESH KUMAR PATJOSHI

Roll No: 208EC216

Under the Guidance of

Prof. K. K. MAHAPATRA

Department of Electronics & Communication Engineering

National Institute of Technology

Rourkela

2010

NATIONAL INSTITUTE OF TECHNOLOGY

ROURKELA

CERTIFICATE

This is to certify that the thesis report titled "**Design and Implementation of Embedded based Elevator Control System**" submitted by **Mr. Rajesh Kumar Patjoshi** (Roll No: 208EC216) in partial fulfillment of the requirements for the award of Master of Technology Degree in Electronics and Communication Engineering with specialization "**VLSI Design and Embedded System**" during session 2009-2010 at National Institute Of Technology, Rourkela (Deemed University) and is an authentic work carried out by him under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other university / institute for the award of any Degree or Diploma.

Date :

Prof. K. K. Mahapatra
Department of E.C.E
National Institute of Technology,
Rourkela-769008

ACKNOWLEDGEMENTS

It is a pleasure to thank many people who made this thesis possible.

I would like to take this opportunity to express my gratitude and sincere thanks to my supervisor **Prof. K. K. Mahapatra** for his guidance, insight, and support he has provided throughout the course of this work.

I am grateful to our teachers **Prof. S. K. Patra, Prof D. P. Acharya, Prof. G. S. Rath,** and **Prof S. Dash**. From these teachers I learned about the great role of self-learning and the constant drive for understanding emerging technologies, and a passion for knowledge.

My special thanks go to Ph.D. scholars, research scholars, friends, especially to Embedded and VLSI lab friends, and juniors at NIT Rourkela for their encouragement and help throughout the course.

I would like to thank all faculty members and staff of the Department of Electronics and Communication Engineering, N.I.T. Rourkela for their extreme help throughout course.

Finally, I am forever indebted to my parents, brother and sister for their prayer, love, understanding, endless patience and encouragement when it was most required. I truly believe that by giving thanks to all I mentioned above, I am expressing my gratitude to God who helped me in all my need.

Rajesh Kumar Patjoshi

CONTENTS

ABSTRACT

The elevator control system is one of the important aspects in electronics control module in automotive application. In this investigation elevator control system is designed with different control strategies. First the elevator control system is implemented for multi-storage building. This implementation is based on FPGA based Fuzzy logic controller for intelligent control of elevator group system. This proposed approach is based on algorithm which is developed to reduce the amount of computation required by focusing only on relevant rules and ignoring those which are irrelevant to the condition for better performance of the group of elevator system. Here only two inputs are considered i.e. elevator car distance and number of stops. Based on these data, fuzzy controller can calculate the Performance Index (PI) of each elevator car, the car which has maximum PI gives the answer to the hall calls. This would facilitate reducing the Average Waiting Time (AWT) of the passenger.

In the second level, the dispatching algorithm is implemented for multi-storage building. Here six types of dispatching algorithms are considered. Based on the traffic situation and condition, one algorithm out of six is operated, that facilitates reducing the Average Waiting Time of the passenger and also reduces the power consumption of the elevator system.

The hardware part of the work comprises a simple D. C. Motor, which can control the up and down movement of the elevator car. This D. C. Motor is controlled through the MC9S12DP256B microcontroller. Here four floor elevator systems have been considered and every floor has two switches, one switch is used for up movement and another switch is used for down movement. Based on the switch pressed, the elevator car can move either in upward or downward direction. Here two sensors are used in every floor. One sensor is used for detecting the elevator car when elevator car reached to its destination floor. This sensor detects the car and stops the D.C. Motor. At the same time, another sensor is used for opening and closing the door.

Finally, a novel fuzzy based PID controller algorithm is implemented using MC9S12dp256B microcontroller. This algorithm is mainly used for maintaining the constant speed of D.C. Motor with different load conditions.

LIST OF FIGURES

LIST OF TABLES

CHAPTER – 1

INTRODUCTION

1.1 OBJECTIVE

An elevator system is a vertical transport vehicle that efficiently moves people or goods between floors of a building. They are generally powered by electric motors. The most popular elevator is the rope elevator. In the rope elevator, the car is raised and lowered by transaction with steel rope. Elevators also have electromagnetic brakes that engage, when the car comes to a stop. The electromagnetic actually keeps the brakes in the open position. Instead of closing them with the design, the brakes will automatically clamp shut if the elevator loses power. Elevators also have automatic braking systems near the top and the bottom of the elevator shaft.

Many modern elevators are controlled by a computer. The computers job is to process all of the relevant information about the elevator and turn the motor correct amount to move the elevator car in correct position. In order to do this the computer needs to know at least three things those are

i) where people want to go
ii) where each floor is
iii) where the elevator car is

Finding out where people want to go is very easy. The buttons in the elevator car and the buttons in each floor are all wired to the computer, when anyone presses these buttons, the computer logs this request.

1.2 HISTORY OF ELEVATOR

The first reference elevator was invented by Archimedes in 312. From some literacy source, elevator were developed as cable on a hemp rope and powered by hand or by through animals. This type of elevator was installed in the Sinai Monastory of Egypt. In the 17^{th} century, the very small type elevators were placed in the building of England and France. In 1793, Lvan Kuliben created an elevator with the screw lifting mechanism for the winter place of Saint Petersburg. In 1816, an elevator was established in the main building of Sub-moscow village called Arkhamgelskoye. In the middle 1800's, there were many type of curd elevators that

carried freight. Most of them ran hydraulically. The first hydraulic elevators used a plunger below the car to raise or lower the elevator. A pump applied water pressure to a plunger, or steel column, inside a vertical cylinder. In 1852, Elisha Otis introduced the safety elevator, which prevented the fall of the cab, if the cable broke. In 1857 March 23rd, the first Otis passenger elevator was installed in New York City. The first electric elevator was built by Werner Von Siemens in 1880.

In 1874, J.W. Meaker patented a method which permitted elevator doors to open and close safely. In 1882, when hydraulic power was a well established technology, a company later named the London Hydraulic Power Company was formed. In 1929, Clarence Conrad Crispen, with Inclinator Company of America, created the first residential elevator.

1.3 ELEVATOR SYSTEM OVERVIEW

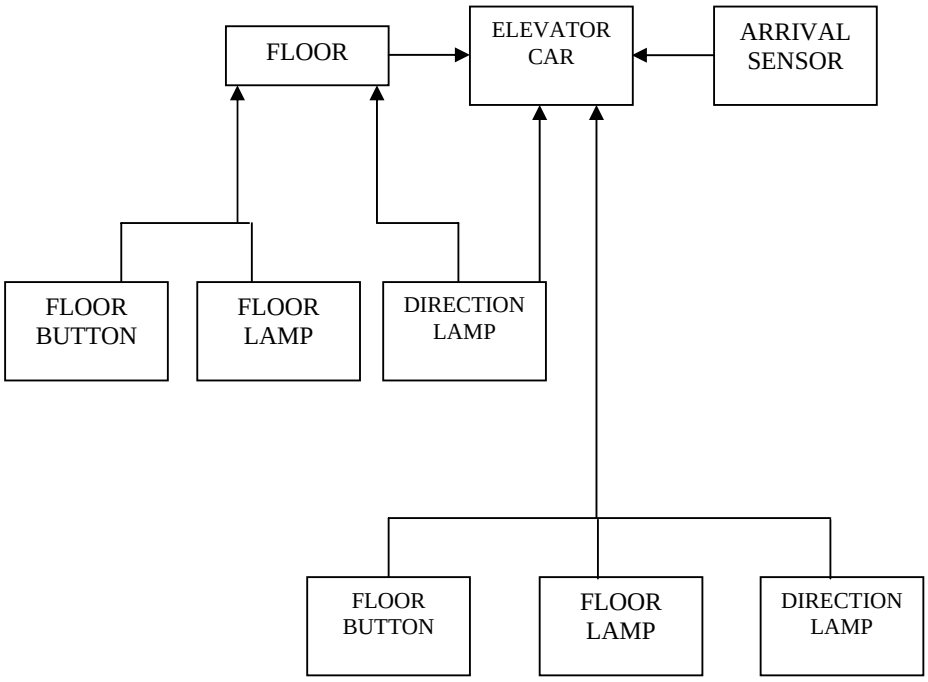


Figure 1.1 : Elevator System overview

Figure 1.1 shows the elevator system overview. This figure consists of floor where passenger wants to visit. Elevator car moves it either upward or

downward direction. The arrival sensor detected the arrival of the elevator to the respective floor. Floor button is used to take the elevator to the respective floor. Floor lamp shows the indication of floor and direction lamp shows the direction of elevator movement, whether it is upward or downward direction. Elevator button is used for moving the elevator car either in upward and downward direction. Based on the elevator switch pressed, the elevator car is moved either in upward and downward direction. D.C. Motor is another important component of elevator system. Based on the switch pressed, the D.C. Motor either moves in forward and reverse direction to move the elevator in either upward or downward direction. Door of the elevator system is one of the important factors of elevator system. When elevator car stops in particular floor, the door of the elevator is opened for passenger to be come out and come in to the elevator car. Arrival sensor is used in every floor, for detecting the elevator car. When a particular car is reached to the particular floor, this arrival sensor detects the elevator car and stops that car.

1.4 DESCRIPTION

When User presses an elevator button, the elevator button sensor sends the elevator button request to the system, identifying the destination floor the user wishes to visit. When any new request comes, this new request is added to the list of floors to visit. If the elevator is stationary, the system determines in which direction the system should move in order to service the next request. The system commands the elevator door to close, when user presses the elevator door closed button. When the door has closed, the system commands the motor to start moving the elevator, either in up and down direction, based on switch pressed.

When the elevator moves between floors, the arrival sensor detects that the elevator is approaching a floor and notifies the system to stop the elevator and open the door of the elevator system. Figures 1.2 shows the elevator dispatching strategy.

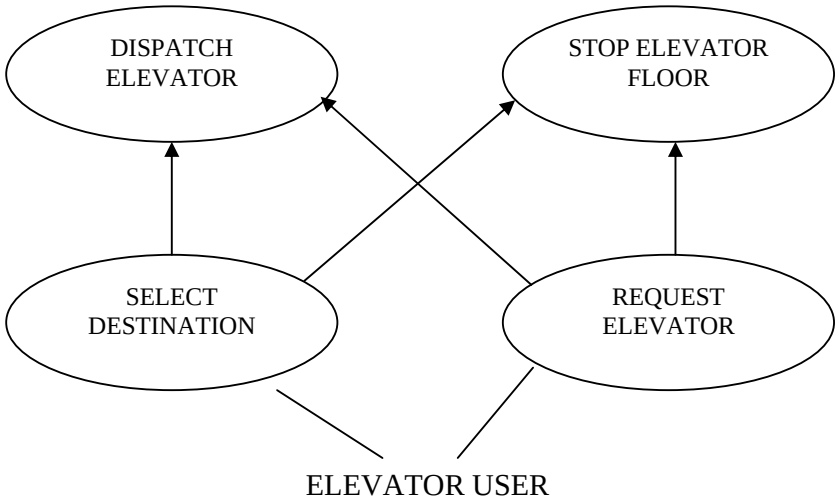


Figure 1.2 : Elevator Dispatching Strategy

1.5 THESIS OVERVIEW

Chapter 1 of this thesis presents an introduction to the Elevator control system techniques. The scope and objectives of this work and outline of this project is also presented in this thesis.

Chapter 2 of this thesis highlights the overview of FPGA based fuzzy logic controller for intelligent control of elevator group system algorithm.

Chapter 3 presents brief description about the Dispatching algorithms .These algorithms reduce the average waiting time of passenger up to certain value and also reduce the power consumption of the elevator system.

Chapter 4 deals with the implementation of embedded based elevator positioning control system with HCS-12 (MC9S12DP256B) Microcontroller.

Chapter-5 presents conclusion part of the various study conducted in this work. It also contains a subtopic relating suggestions for the future work.

1.6 OVERVIEW OF FPGA BASED FUZZY LOGIC CONTROLLER FOR INTELLIGENT CONTROL OF ELEVATOR GROUP SYSTEM

Here the fuzzy logic controller is implemented (FLC) on a field programmable gate array (FPGA) system for intelligent control of elevator system. This design is based on the algorithm which is developed to reduce the amount of computation required by focusing only on relevant rules and ignoring those which are irrelevant to the condition for better performance of the group of elevator system.

Here the simulation was carried out by considering the two inputs i.e. elevator car distance and elevator number of stops. The elevator distance is calculated by considering number of factors such as Pc the hall call floor position, Pe the car position, Ph the highest floor position, and P_l is the lowest floor position. Based on this data car distance of every car is calculated.

After that the value of car distance along with number stop of each car is applied to the fuzzy controller for calculation of performance Index (PI) of each car and the car which has maximum (PI) gives the answer to the hall calls. This would facilitate reducing the average waiting time (AWT) of the passenger.

1.7 OVERVIEW OF DISPATCHING ALGORITHMS

Dispatching algorithms is the most important aspects in elevator control system, this algorithms can reduce the average waiting time of passenger up to certain value and also reduce the power consumption of the elevator system. Here we implement six types of dispatching algorithms, these are

(i) Collective up Algorithms (CU)
(ii) Collective Down Algorithms (CD)
(iii) Selective up Algorithms (SU)
(iv) Selective Down (SD)
(v) Selective - Collective UP (S-C-U)
(vi) Collective – Selective Down (C-S -D)

Based on the traffic amount and traffic percentage any one of the algorithm is selected on particular time instance, so this makes the reduction of average waiting time and power consumption of elevator system.

1.8 OVERVIEW OF EMBEDDED BASED ELEVATOR POSITIONING CONTROL SYSTEM

Here we implement an embedded based elevator positioning control system (PCS) on a HCS-12(MC9s12dp256b) microcontroller system for intelligent control of elevator system. The search for an intelligent group controller that can satisfy multicriteria requirements of an elevator group control system has become a great challenge for researchers. This proposed approach is based on MCU control module, DC motor driver module, display module and key module. MCU controls the speed and direction of DC motor by inputing the PWM signal to its driver circuit. Display module shows the real time information of elevator running status. The elevator's running path is set by key ,based on key pressed the elevator either moves in upward or downward direction. Two infrared sensors are used in this project, One for detecting the elevator car in the particular floor and another for opening and closing the elevator door. This project is also implemented with FUZZY PID controller for providing intelligence to the elevator car in different load condition for smooth running of elevator.

1.9 FUZZY LOGIC FOR EMBEDDED SYSTEM

Intelligent systems are becoming increasingly distributed in terms of both their applications and their implementation. While large systems will remain important e.g. for commerce and industry, smaller embedded intelligent system have also started to appear in the home and workplace. Fuzzy logic extends from the traditional crisp boundary of Arisolelian logic (true or false) to include the concept of partial truth, having truth value between completely true and completely false. Motorola's HC68HC12 (HCS-12) microcontroller incorporates several fuzzy logic primitives directly in its instruction set. The instruction set contains the fuzzy logic operations of trapezoidal membership rule evaluation, and weighted average

defuzzification. The microcontroller also includes other instructions that are helpful in fuzzy logic application such as MIN / MAX instruction. Motorola's HC12 allows the development of low level application that can utilize the unique features of fuzzy logic.

1.10 PROBLEM BEING ADDRESSED

Here main problem is being addressed when we design the VHDL code for fuzzy logic control for elevator control system. In the fuzzy logic code design, the division operator is not synthesizable in the rule evaluation and defuzification process hence some modification is required in the algorithms i.e. separate VHDL program for division operator is written in the total process.

Another problem is being addressed in the time of hardware design with (HCS-12) MC9SI2dp256B microcontroller. The main problem arises when we design the different interfacing circuit with MC9SI2dp256B microcontroller. Total experimental set-up is performed using (HCS-12) (MC9SI2dp256B) microcontroller board, D.C. Motor device circuit, key interfacing circuit, sensor interfacing circuit and display circuit.

CHAPTER – 2

FPGA BASED FUZZY LOGIC CONTROLLER
FOR
INTELLIGENT CONTROL
OF
ELEVATOR GROUP SYSTEM

2.1 INTRODUCTION

In control engineering analysis and design, FPGA based Fuzzy logic control (FLC) for elevator group system has been attractive because it offers a compromise between special purpose ASIC hardware and general purpose processors. This purposed FPGA based Fuzzy Logic Controller for elevator group system can reduce the design development cycle, simplify the design complexity, and also improve the control performance that simplifies implementation and reduces the hardware costs.

Many researchers have reported about Fuzzy Logic Controller for group control of elevator system. They described the validation of five dispatching algorithms for elevator system that were implemented on spartan-3 FPGA based board in an integrated approach reducing the area and improving performance. The overall system is composed of several LCS, which implement the dispatching algorithm. The EGCS-based Fuzzy Logic (FEGCS) runs on a PC and under different traffic situation determines the best algorithms to be run in each LCS in order to reduce the average waiting time of passenger and also reduce the power consumption.

Elevator group controller based Fuzzy Logic Framework with self tuning scheme for reducing the average waiting time (AWT) of passenger is presented in the literature Development of a self-tuning fuzzy logic controller for intelligent control of elevator system.

Fuzzy Controller described in the above literature, has evaluated six set of rules and each rule set consists of a different number of rules (between 12 and 14). Hence fuzzy computation time is large for response to hall call even if it is a self tuning fuzzy controller.

In the present investigation, we present a mamdani's inference technique with the algorithm which is developed to reduce the amount of computation required by focusing only on the relevant rule and ignoring those which are irrelevant to the condition, which means in the fuzzification process for any single crisp value of

the input, only two adjacent fuzzy values are significant. By ignoring the insignificant fuzzy values the number of fuzzy output signals can be reduced from five to two.

That means the fuzzification block has five output i.e. five membership functions for each input. From the five membership function only two membership functions are significant for any particular input.

For the two inputs, the number of output signal can be reduced from twenty five to only four. In this technique we access the content of the FAM table through a small window and only four adjoining rules can be viewed through this window at a time. Therefore, instead of accessing 25 rules, the inference engine has to access four rules during every computation.

In this work, simulation was carried out by considering two input i.e., elevator car distance and number of stop. These two input values are determine using number of factor that means when we compute the car distance, the position of the car and its direction must be known. Thus car distance can be calculated by comparing the position of the car and its direction of travel with respect to the hall calls floor. In the proposed technique, four possible position and direction of the elevator car for up hall call floor and down hall call are considered.

Calculation of the number of stops depends on the number of hall calls and the car calls of a elevator car before it reaches to a respective hall call floor.

This algorithm is simulated using VHDL and implemented in FPGA. After it is compared with the fuzzy rule i.e. mamdani's inference technique which uses an inference engine which triggers all 25 rules during every calculation. Then the average waiting time of passenger is calculated for both the techniques, and it is found that AWT is minimum in the reduced rule technique algorithm in comparison to the actual mamdani's inference technique.

2.2 FPGA ARCHITECTURE - AN OVERVIEW

FPGA - is an acronym for Field Programmable Gate Array. It belongs to a class of user programmable digital devices called Programmable Logic Devices (PLD's). A programmable logic device is an integrated circuit that enables the user to configure it in many ways, enabling the implementation of various digital logic functions, of varying sizes and complexities. PLD's can be classified into various categories

1. Simple programmable logic devices (SPLD)
    (a) Programmable logic array (PLA) : A programmable logic array is an integrated circuit that contains two levels of programmable logic ; an AND plane and an OR plane.
    (b) Programmable array logic (PAL): A PAL is an integrated circuit that contains a fixed OR plane followed by a programmable AND plane.
2. Complex Programmable Logic Device (CPLD)
3. Field Programmable Gate Array (FPGA)

2.2.1 FPGA Architecture

The typical FPGA consists of the following components:

1. Programmable Logic blocks
2. Interconnection Resources
3. Input output blocks

The general schematic of an FPGA is as shown in the figure

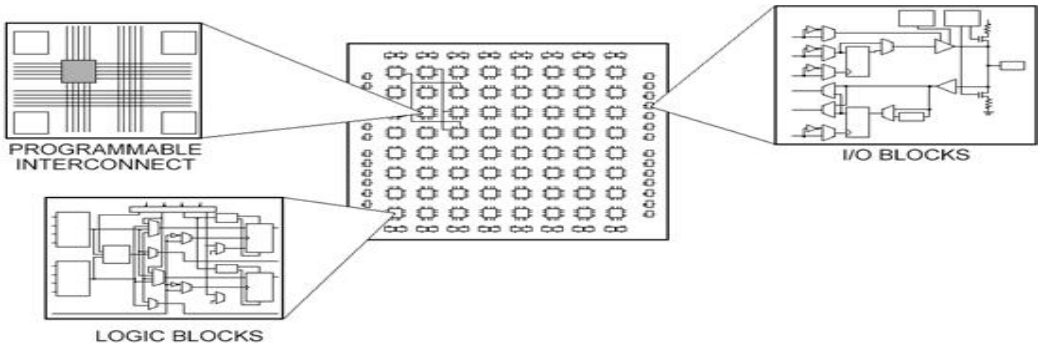


Fig 2.1: FPGA Schematic

2.2.2 Programmable Logic Block

The programmable logic block in a typical FPGA consists of Configurable Logic Blocks (CLB). The CLB can be realized in many ways; one of them being the Look Up Table (LUT) based CLB. The LUT is a one bit wide memory location. The memory address lines are the inputs to the LUT and the one bit output is the LUT output. Thus the LUT with K-inputs acts as a 2k by 1 bit memory and the user can directly implement any k input function by programming the functions truth table into the LUT [8].

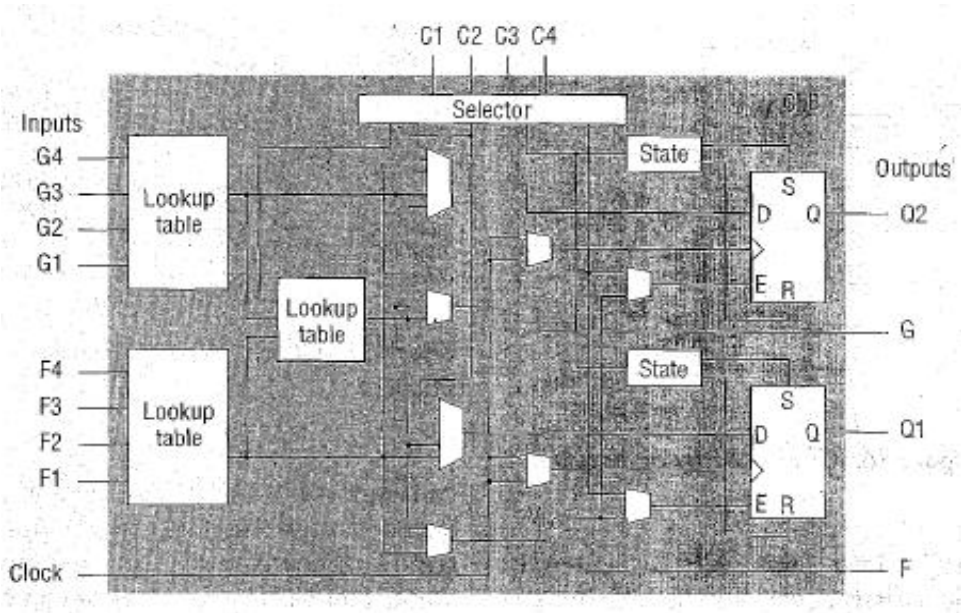


Fig 2.2: Xilinx FPGA-CLB Schematic

Above diagram shows a generalized CLB that can be used for implementing any logic function of up to nine inputs; two separate four input logic functions and many other possibilities. The CLB also has a D-Flip Flop that can be used to implement sequential logic functions. The CLB has also got features that support the integration of entire systems. It has also got certain specialized circuitry that enables it perform arithmetic operations like addition, multiplication etc. in a fast and efficient manner. Users can also configure the LUT in the CLB as read/write RAM locations. Some FPGA also allow configuration of their LUT‘s as Dual port

RAM‘s; with one write and two read inputs. The chips also include very wide AND planes around the periphery of the CLB to facilitate implementation of wide decoders. Some of the modern FPGA also include entire micro controllers on the chip; enabling easier implementation of complicated logic functions on a single chip. This is especially suited for control applications [8].

2.2.3 Interconnect Resources

The other most important feature that decides the performance of the FPGA and its suitability for control applications is its interconnect resources. This is because the interconnection resources allow the implementation of an entire digital system by providing a means of connecting various individual circuits (subsystems) that have been implemented on different CLB‘s in an FPGA. The interconnect resources in an typical FPGA can be classified as :-

1. General Purpose Interconnects: Signal between CLBs and Input Output Blocks (IOBs) can be routed through switch matrices as they travel along the horizontal and vertical interconnect lines.
2. Direct Interconnects: Adjacent CLBs are interconnected directly.
3. Long Lines : Long lines provide for high fan out, low-skew distribution of signals that

   must travel relatively long distances. They span the entire length or width of the interconnect area. They are typically used for clock signals.

FPGA interconnects are normally unsegmented i.e. each wiring segment spans only one logic block before it terminates in a switch box. A switch box is a switching matrix that contains programmable interconnections to all the wiring segments that terminate inside it. By turning on some of the programmable switches within a switch box, longer paths can be constructed [6]. Figure 1.3 shows a typical FPGA interconnection scheme.

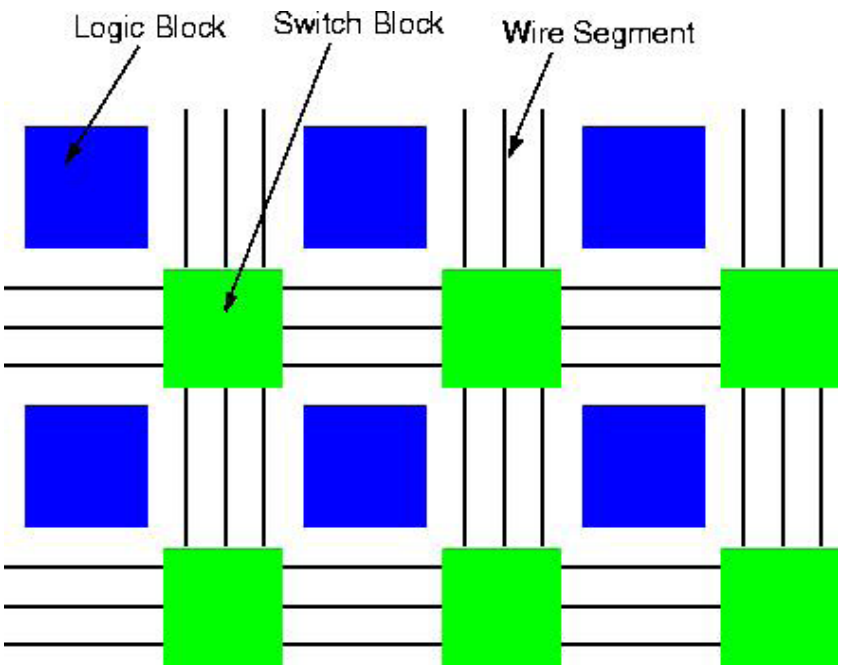


Fig 2.3: FPGA Interconnection schematic

2.2.4 Input Output Blocks (IOB)

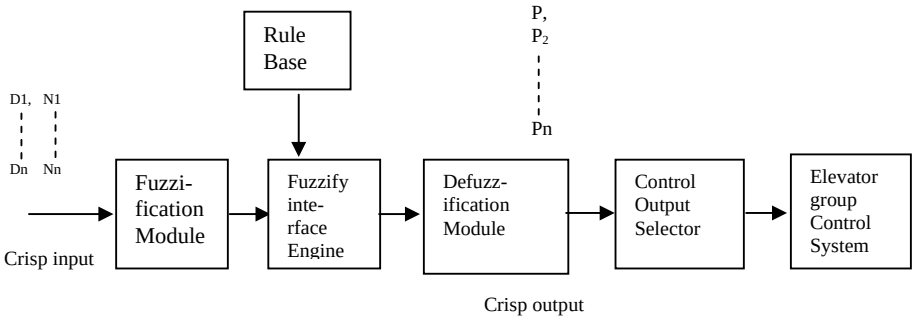
The IOB provides the interface between the FPGA and the real world signals. The IOB consists broadly of I/O pads. The I/O pads connect to one of the pins on the IC package so that the external signals can be input to or output from the array of logic cells. It also consists of tristate buffers, which enable the signals to be input to and output from the logic array. Flip flops are provided so that the input and the output values can be stored within the IOB. Each IOB has also got a variety of other features like re programmability of the input threshold to respond to either TTL or CMOS logic levels. It also incorporates slew rate control of the output signal and includes internal pull up resistors to avoid floating inputs [9].

The FPGA can be a fine grained or a coarse grained device. A fine grained FPGA consists of a large number of small width programmable logic resources that can be used to implement a variety of functions. A typical example of such an FPGA would be the Atmel AT40K. A coarse grained FPGA like the Xilinx Virtex series consists of a smaller number of more powerful logic blocks like LUT‘s and flip flop‘s. Modern FPGA‘s also come with features like Low Voltage

Differential Signaling (LVDS) and also support programmability of the input threshold to respond to LVTTL,LVCMOS etc. They also provide Discretely Controlled Impedance (DCI) features. Most FPGA also include Peripheral Component Interconnect (PCI) support; by which they can be interconnected to a general purpose computer or form a part of a larger development board. FPGA‘s are also JTAG compliant i.e. they support the IEEE 1149.1-1990 boundary scan architecture; which enables test data to be serially loaded into the device and test results to be serially read out. JTAG can also be used for loading configuration bit streams into the FPGA. Another important feature that FPGA‘s possess is that of In System Programming (ISP) that enables the FPGA to be programmed while it is a part of the end target system. This eliminates the necessity of physical removal of the chip from the system and easy programmability.

2.3. DESIGN METHODOLOGY OF PROPOSED FPGA BASED FUZZY LOGIC CONTROL FOR ELEVATOR GROUP SYSTEM

A conventional FLC is designed on a simple concept by reducing the number of rules that facilitates reduction of computation time of Fuzzy Logic Controller for better performance of elevator group control system. Basically the FLC is divided into four components. Figure-2.4 shows the diagram of the Basic Structure of Proposed FPGA based Fuzzy Logic Controller for intelligent control of Elevator System. It consists of Fuzzification Module, Rule Base, Fuzzy Interface Engine, Defuzzification Module.



D : Car Distance N : No. of Stops

Fig. 2.4 : Basic Structure of the proposed Fuzzy Logic Controller for Elevator Group Control System

2.4 FUZZIFICATION DESIGN MODULE FOR THE PROPOSED FPGA BASED FUZZY LOGIC CONTROLLER FOR ELEVATOR GROUP CONTROL SYSTEM

The fuzzification block has five outputs, one for each fuzzy value defined in the inputs Universe of Discourse. However, the fuzzification process entails that, for any single crisp value of the input X_i, only two adjacent fuzzy values are significant (with non-zero-membership values). By ignoring the insignificant fuzzy values, the number of output signals can also be reduced from five to two. The possible combinations of significant fuzzy values for an arbitrary input are:

$$B_i^1 \text{ and } B_i^2 \text{; and } B_i^2 \text{ and } B_i^3 \text{; } B_i^3 \text{ and } B_i^4 \text{; and } B_i^4 \text{ and } B_i^5$$

It is found that using just three variables, ADR_i, B_i_A and B_i_B, all the combinations can be sufficiently represented for any value of x_i as shown by the following statements:

$$ADR_i = \text{“00”} : B_i\_A = B_i^1,\ B_i\_B = B_i^2$$

$$ADR_i = \text{“01”} : B_i\_A = B_i^2,\ B_i\_B = B_i^3$$

$$ADR_i = \text{“10”} : B_i\_A = B_i^3,\ B_i\_B = B_i^4$$

$$ADR_i = \text{“11”} : B_i\_A = B_i^4,\ B_i\_B = B_i^5$$

Figure-2.5 illustrates how these conditions correspond with the universe of discourse.

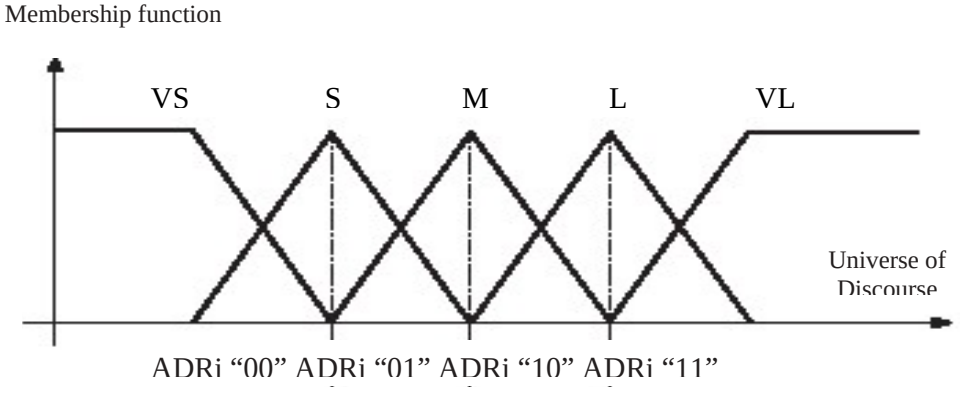


Fig. 2.5 : Input Fuzzy set

2.4.1. Mini FAM Table

The FAM table of the FLC design is shown in table-I. Here we describe an algorithm which is developed to reduce the amount of computation required by focusing only on the relevant rules and ignoring those which are irrelevant to the condition. It is known that for every set of inputs, only four fuzzy values (two for each input) are relevant at any one time.

An easier way of explaining the technique is to imagine the entire FAM table to be covered from view. Access to the content of the FAM table is only allowed through a small window and only four adjoining rules can be viewed through this window at a time. Therefore, instead of having to access 25 rules, the inference engine only has to access four rules during every computation. The window can move around the FAM table and its position is identified by an index *j* defined as:

ADR1 = “00” & ADR2 = “00” → *j* = 0
ADR1 = ”00” & ADR2 = “01” → *j* = 1
ADR1 = “00” & ADR2 = “10” → *j* = 2
ADR1 = “00” & ADR2 = “11” → *j* = 3
ADR1 = “01” & ADR2 = “00” → *j* = 4
ADR1 = “01” & ADR2 = “01” → *j* = 5
ADR1 = “11” & ADR2 = “11” → *j* = 15

There are sixteen ‘window positions’ altogether and the first six are shown in Fig.2.6. The shaded blocks are the rules which are considered relevant for the input conditions corresponding to the index j. When the window technique is applied to the FAM table in Table I, it is observed that a number of the mini-FAM tables are identical (e.g. j = 1 and j = 4). Out of the 16 mini-FAM tables, there are only seven unique tables which are shown in Fig. 2.7. If WIN is the index for the new set of tables, then the tables can be arranged using the following

IF j=0 THEN WIN=“0000”
IF j=1 OR j=4 THEN WIN=”0001”
IF j=2 OR j=5 OR j=8 THEN WIN=”0010”
IF j=11 OR j=14 THEN WIN=”0101”
IF j=15 THEN WIN=”0110”

Ele_car_dist / Ele_no_sto	VS	S	M	L	VL
VS	PVB	PB	P	PS	Z
S	PB	P	PS	Z	NS
M	P	PS	Z	NS	N
L	PS	Z	NS	N	NB
VL	Z	NS	N	NB	NVB

NVB	Negative Very Big
NB	Negative Big
N	Negative
NS	Negative Small
Z	Zero
PS	Positive Small
P	Positive
PB	Positive Big
PVB	Positive Very Big

Table-2.1 (FAM table of the FLC design)

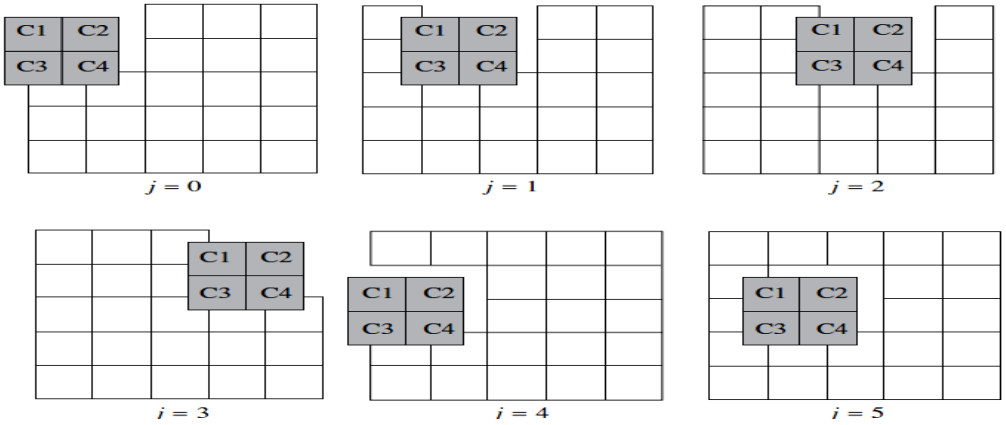


Fig.2.6 : Mini FAM Table

This algorithm requires a considerable number of IF-THEN operations and is not necessarily an efficient way to implement the design into hardware. By observing the pattern in the original FAM table, it can be shown that the mini-FAM tables are identical when the sum of ADR1 and ADR2 is the same. Therefore, instead of using numerous IF-THEN operations, the arrangement of the mini-FAM tables is achieved using a single addition.

WIN <= (“00”&ADR1) + ADR2;

PVB	PB
PB	P

WIN "0000"

PB	P
P	PS

WIN "0001"

P	PS
PS	Z

WIN "0010"

PS	Z
Z	NS

WIN "0011"

Z	NS
NS	N

WIN "0100"

NS	N
N	NB

WIN "0101"

N	NB
NB	NVB

WIN "0110"

Fig. 2.7 Mini – FAM Tables for the FLC Design

function of the statement ("00"&ADR1) is to expand the value of ADR1 from 2 bits to 4 bits such that it is compatible with the 4-bit signal WIN. The variables inside the mini-FAM table are subsequently processed in the section of the code that is marked 'Mini-Fuzzy Inference Engine'. In the original code, the inference engine contains 25 MIN-operations. The modified code consists of only four MIN-operations, which is a notable reduction.

2.4.2. Defuzzification algorithm

The function of the defuzzificaition is to convert the fuzzy output value of the control system into the corresponding crisp value of the membership function shown in fig.2.8. This is achieved using the weighted average defuzzification method. This defuzzification operation requires several multipliers and a divider. Behavioral modeling in VHDL supports multiplication and division but these operations are complicated to realize in the synthesis and implementation stages.

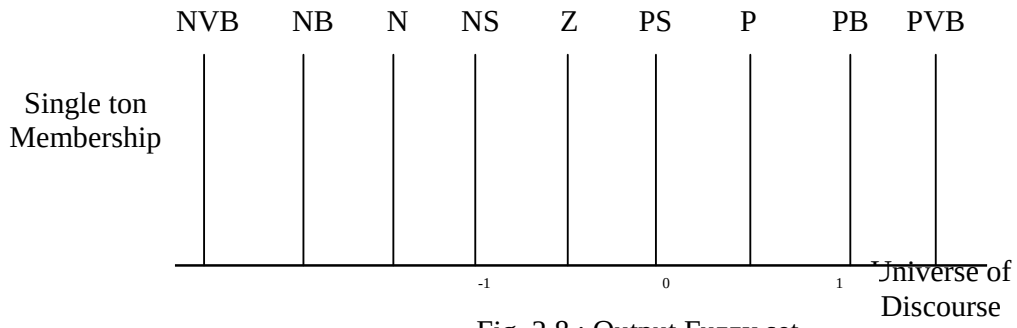


Fig. 2.8 : Output Fuzzy set

In the proposed approach, only four significant rules consequents are considered. Therefore, the number of rule-consequents to be aggregated is reduced but the allocation of correct weightings for the significant output values becomes slightly more complicated. From the tables in Fig.4 it is obvious that regardless of the **WIN** value, the consequents C2 and C3 always point to the same fuzzy value (e.g. when **WIN** = "0000": C1→PVB, C2→**PB**, C3→**PB**, C4→P). This implies that only C2 and C3 have to be aggregated, hence

DA = C1

DB = max[C2, C3]

DC = C4

Where DA, DB and DC represent the membership function of the output fuzzy values. Thus the modified output fuzzy set is shown in figure 2.9.

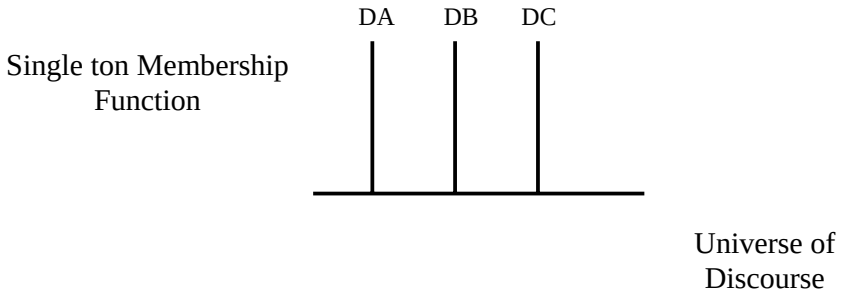


Fig.2.9 : Modified output fuzzy set

2.5 DESIGNED METHODOLOGY FOR ELEVATOR GROUP CONTROLS

2.5.1. Value calculation of input variables

Determination of the values of the two input variables for fuzzy evaluation must not ignore important factors that influence its results. As in computing the car distance, the position and the direction of travel of the car must be known.

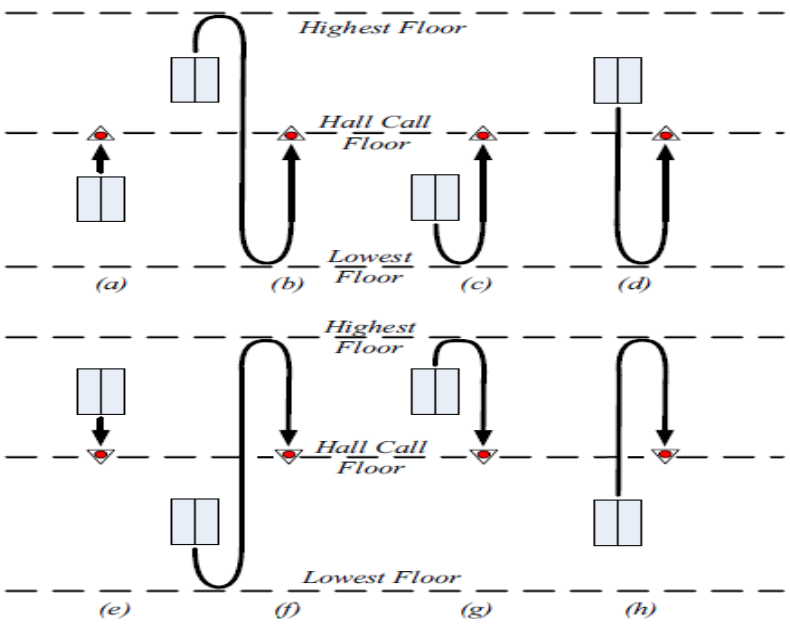


Fig 2.10. The different Position taken by a car to reach the hall call floor.(a)–(d) for up hall call, whereas (e) – (h) for down hall call.

The car distance can be calculated by comparing the position of the car and its direction of travel with those of the hall call. As shown in Fig. 2.10, for a car to arrive at the hall call floor, there are four possible positions for the up direction and another four for the down direction. First consider the four for up hall call. A scan be seen from Fig. 2.10, besides the information mentioned above, position (b)–(d) require additional data for the computation, the highest floor and the lowest floor served by the car for position(b), and only the lowest floor for position (c) and (d). From these, three equations can be derived to find the distance traveled in each case, presented in the following:

For position (a),

$$d = P_c - P_e. \qquad (2.1)$$

For position (b),

$$d = (P_h - P_e) + (P_h - P_l) + (P_c - P_l) \qquad (2.2)$$

For position(c) and (d),

$$d = (P_e - P_l) + (P_c - P_l) \qquad (2.3)$$

Where d is the distance traveled, $P_{c,}$ the hall call floor position, $P_{e,}$ the car position, $P_{h,}$ the highest floor position and $P_{l,}$ the lowest floor position.

The difference between routes (c) and (d) is the different position of the car relative to the hall call floor position. Regardless of the position, (3) applies to both routes. In the case of the down hall call, the same equations can be rearranged to consider the down direction of the hall call.

For position (e), by rearranging (2.1), we obtain

$$d = P_e - P_c. \qquad (2.4)$$

For position (f), by rearranging (2.22), we obtain

$$d = (P_e - P_l) + (P_h - P_l) + (P_h - P_c) \qquad (2.5)$$

For position (g) and (h), by rearranging (2.3), we obtain

$$d = (P_h - P_e) + (P_h - P_c) \qquad (2.6)$$

Calculation for the number of stops depends on the number of hall calls and the number of car calls a car has to serve before it can reach a hall call floor. All calls, including hall calls and car calls, in between the present position of the car and the hall call floor being considered for assignment, reflect the stops to be made during the journey of the car. By counting the number of these calls, we are actually counting the number of stops. Besides the number of stops, the floor number that corresponds to each stop is also recorded for use later in the computation.

2.5.2. Elevator dispatching Strategy

Figure-2.11 shows the process of selecting the most suitable car to answer a hall call through the FLC system. Here I consider the up-down call button system when a hall call is registered. Relevant data from all cars in the group are needed for computation of the value of the input variables. The data required from every car are, its present position in the building, motion status, speed direction of travel.

Here we use two input variables for fuzzy evaluation they are the car distance and number of stops. Their definitions are as follows: -

1) **Car distance**: The distance travelled by car to move from its present position to a hall call floor when a hall call is registered.

2) **Number of stops** : The number of stops of floors a car has to stop at to load or to unload passengers before a hall call floor is reached.

For every car in the group, the values of the two input variables are calculated based on the data supplied by each car controller. The calculated values of each car are then processed by the FLC and from fuzzy evaluation, a performance index (PI) is assigned to each car. The PI denotes the suitability of the car to answer a hall call at a particular instant, the car with the highest PI value is considered the car most suitable to attend to the hall call.

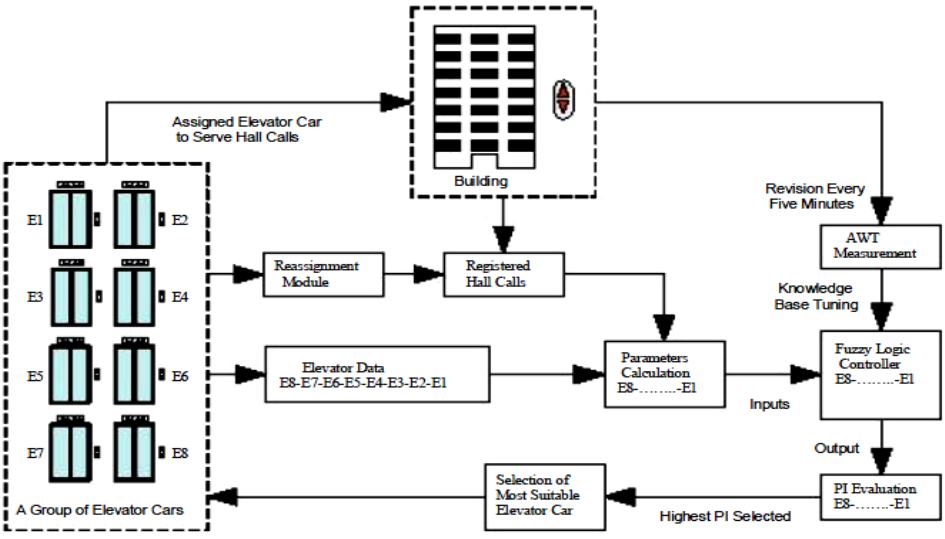


Fig. 2.11: The process flow for the elevator dispatching strategy with the use of a fuzzy logic controller

2.6 SIMULATION RESULT AND FPGA IMPLEMENTATION

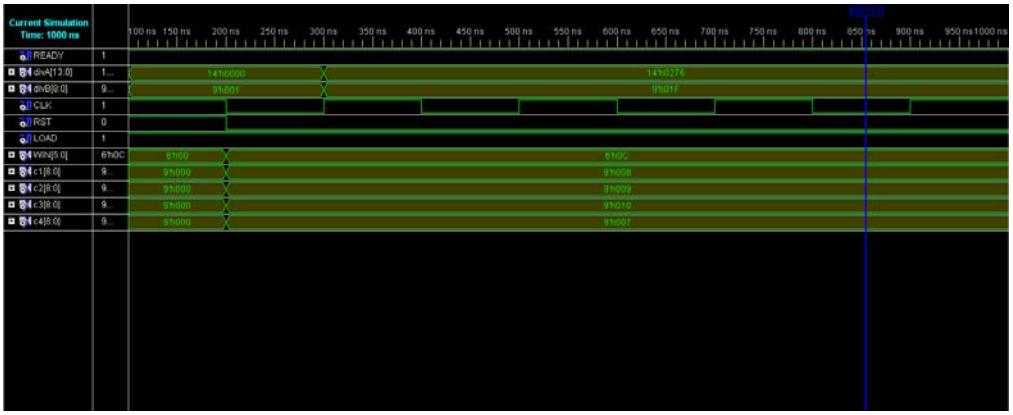
We consider here two inputs i.e. elevator car distance and number of stops. These two input values are determined using considering number of factors, mainly Hall Call floor position of the elevator car in the building, direction of travel, number of hall calls and number of car calls.

In the simulation a hall call floor is considered, from that hall call floor one can move to upwards direction to reach higher destination floor of the building and other can move to downward direction to reach lower destination floor of the building.

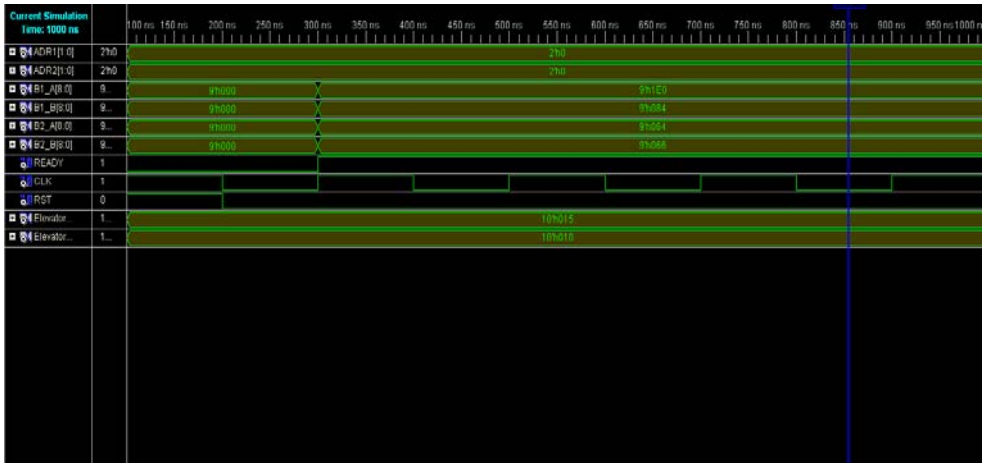
Here different data inputs are considered such as highest floor number, lowest floor number, hall call floor number, position of each elevator car, direction, number of hall calls and car calls.

These data are provided to the interface engine. Based on these data, interface engine calculate the elevator car distance and number of stops to the respective hall-call floor.

After that this data is fed to the fuzzy controller for calculation of performance index (PI) of each car and the controller evaluates the largest PI which denotes the suitable car to answer a hall call at a particular instance. Corresponding simulation result of fuzzification and defuzzification is shown in figure 2.12. Based on the data given to the input of fuzzy controller, elevator car-1 has the highest PI to response the hall calls for upward direction hall call and elevator 5 has the highest PI to response the hall calls for downward direction hall call, which is shown in simulation result in figure 2.13.



(a) Simulation result of Fuzzification



(b)

Figure 2.12 : (a) Shows the Simulation result of Fuzzification (b) Shows the Simulation result of Defuzzification

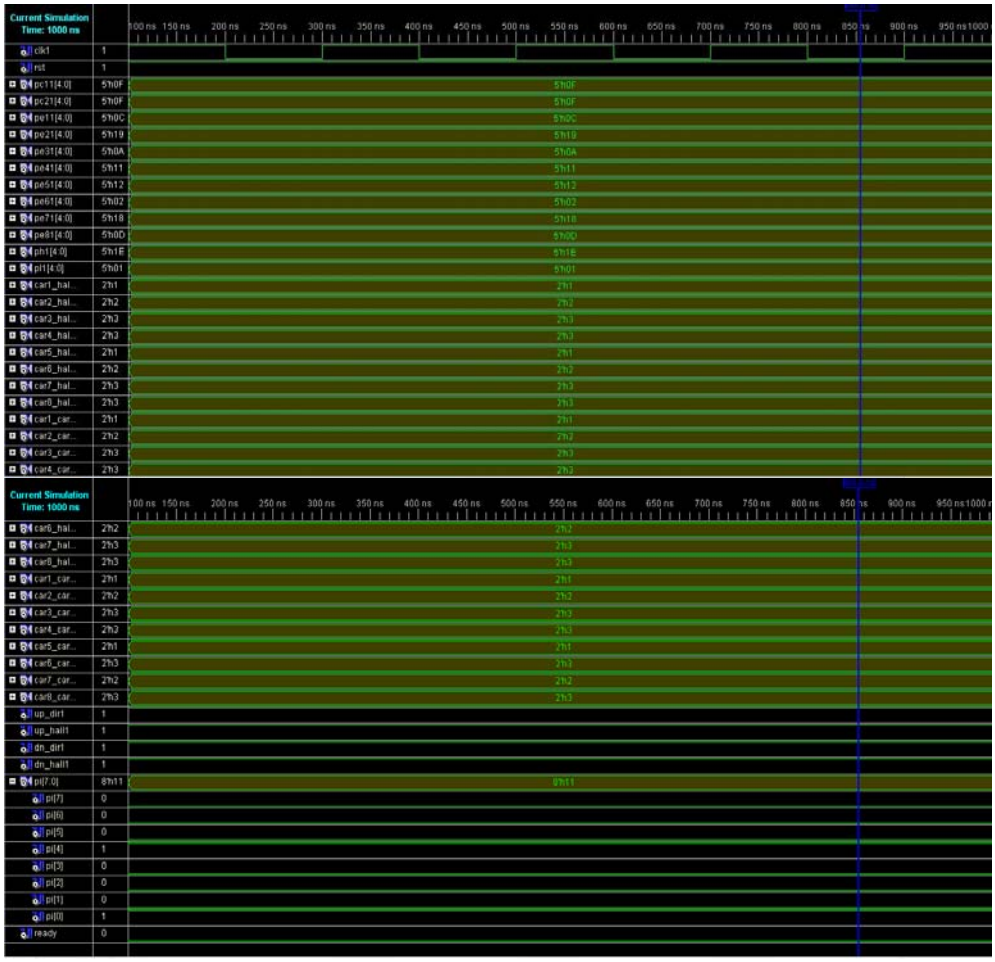


Figure 2.13 : Simulation Result for PI evaluation

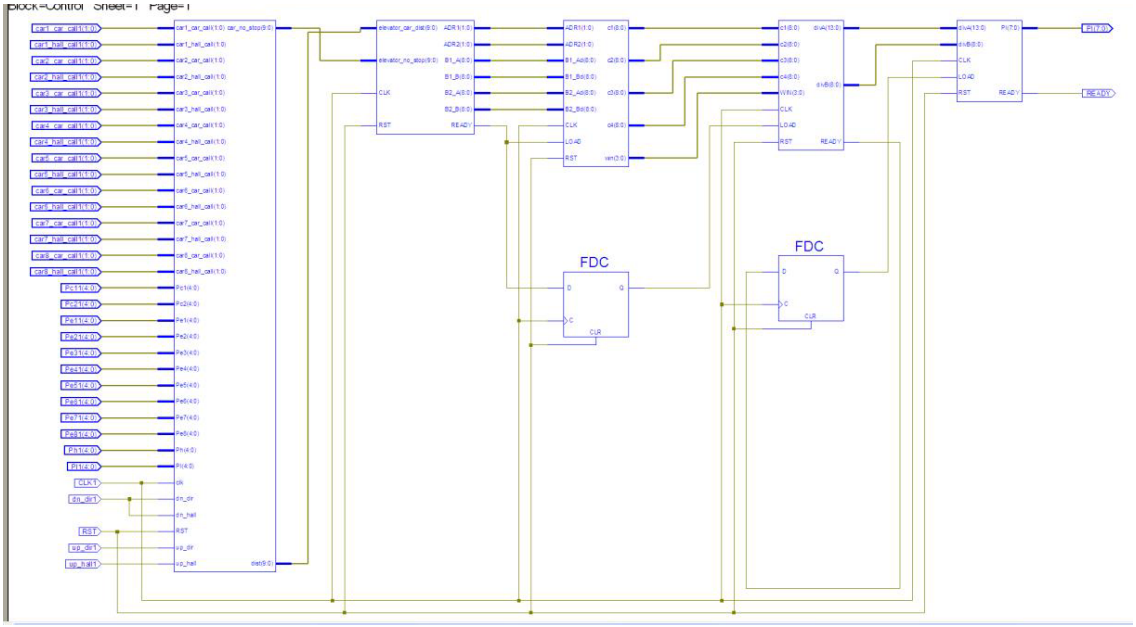
In this work, we consider eight number of elevator cars. These eight numbers of cars are distributed in different position of the building with different directions that means some car move in upward direction and some car move in downward direction.

A hall call floor from those floor two hall calls is considered. One hall is present in upper part of the building and another hall is present in lower part of the building.

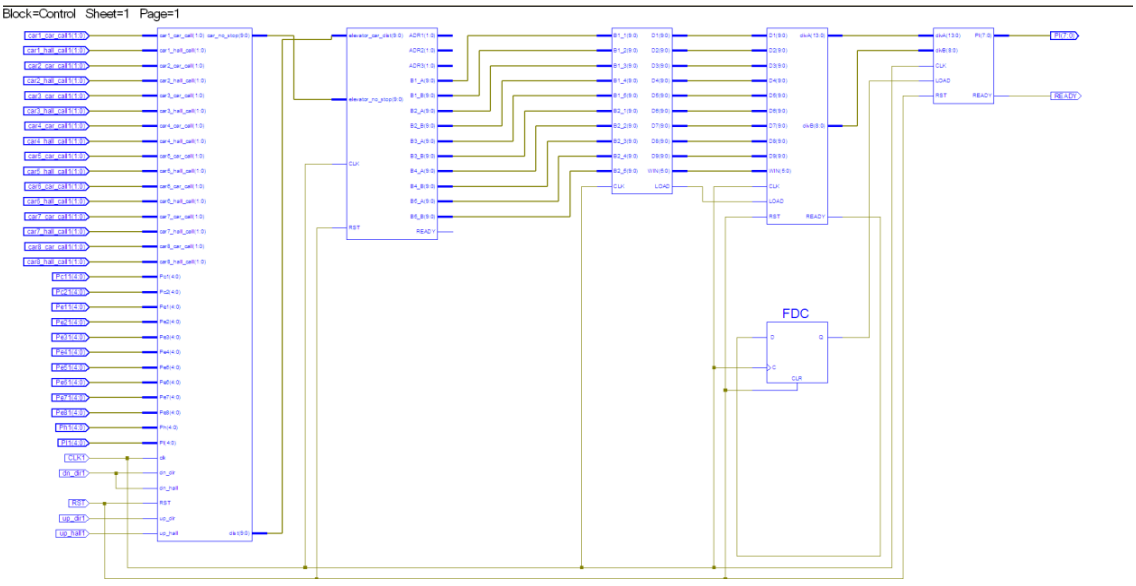
Based on the position, direction and number of stops one elevator car will give the response with in minimum time for the upward direction hall call floor and another elevator car will give response within minimum time for the downward direction hall calls. These would facilitate reducing the average waiting time (AWT) of the passenger.

Two types of fuzzy algorithms are considered here for elevator control system. One algorithm is mamdani's type inference technique and this uses an inference engine which triggers only four rules during every calculation. Another type algorithm is mamdani's inference technique and it uses an inference engine which triggers all 25 rules during every calculation.

Both the fuzzy algorithm used for elevator control system is implemented using VHDL code. The VHDL codes are synthesized for converting into Register Transfer Logic (RTL) views of the fuzzy logic control (FLC) architecture which is shown in figure. This figure 2.14 shows the RTL view of both the algorithm.



(a)



(b)

Figure 2.14: (a) RTL view of the FLC Architecture for Reduce rule mamdani's inference Fuzzy algorithm**.** (b) RTL view of the FLC Architecture for Simple mamdani's inference Fuzzy algorithm.

2.6.1 FPGA Implementation:

Both the algorithm of elevator control system is implemented using FPGA Spartan 3E (XC3S500E) with FG 320 package and a speed grade of -4. The result is implemented through FPGA using Chipscope pro is shown in figure 2.15.

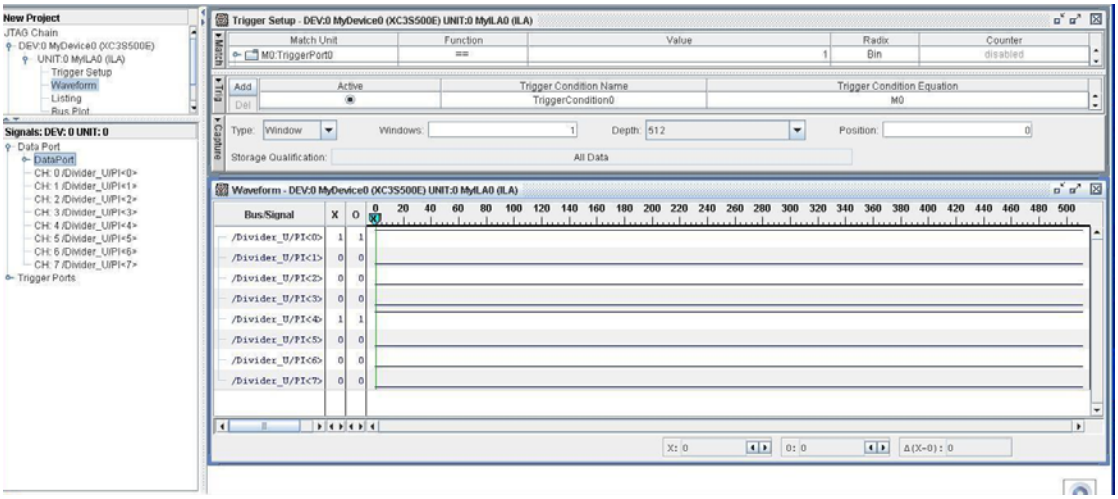


Figure 2.15: Result of FPGA implementation using Chipscope pro

Table 2.2 shows the device utilization summary of both algorithms. From the device utilization summary I found that fuzzy rule using reduced rule technique takes lesser space in FPGA and operating frequency is higher in comparison to actual mamdani's inference Fuzzy algorithm..

Device utilized in FPGA	**Reduce rule mamdani's inference Fuzzy algorithm.**	**Actual mamdani's inference Fuzzy algorithm.**
No. of slices	9 %	20 %
No. of slices flip flop	3%	3 %
Number of 4 inputs LUTs	8%	18 %
Number of bounded IOBs	20%	20 %
Number of GCLKs	4%	4 %
Minimum period	17.485 ns	36.056 ns
Maximum frequency	57.192 MHz	27.735 MHz
Total memory uses	172596 Kilobytes	206964 Kilobytes

Table 2.2: Comparison result of device utilization summary of both the algorithm

2.6.2 Average Waiting Time

Average Waiting Time is the most important parameter, that is used in measuring the performance of the elevator control system. It is defined as time gap between pressing of hall call button and reaching of assigned car at concerned floor. Here I calculate the average waiting time (AWT) of passenger for both the algorithm and compare them. Average waiting time is calculated using the formula.

$$AWT = T_{resp,\ car} + \sum_{i=1}^{N} T_{drive} + \sum_{i=1}^{N.1} T_{stop}$$

$$T_{rep,\ car} = T_{\text{delay time of controller}} \times \sum_{i=1}^{M} D_N \times \sum_{i=1}^{M} D_k \times \text{No. of car call} \times \text{No. of Hall Call}$$

$$T_{stop\ (i)} = T_{speed\text{-}down} + T_{get\text{-}on\ /\ off\ (i)} + T_{speed\text{-}up}$$

$$T_{drive} = D_i \times T_{travel\text{-}floor}$$

Where,

$T_{resp,\ car}$ is the car response time.

$T_{\text{delay time of controller}}$ is the execution time of fuzzy controller.

D_N is the distance between hall call floor to controller.

D_k is the distance between controller and car.

T_{drive} means floor where there are calls near the floor.

T_{stop} means floor where hall calls and car calls are assigned.

$T_{travel\text{-}floor}$ is the time required to travel between two floor.

D_i is the distance travel of each car from its respective hall call floor, when hall call occurs.

$T_{speed\text{-}down}$ Time duration for elevator car to come in stop position from maximum speed.

$T_{get\text{-}on\ /\ off\ (i)}$ is the time duration of passenger get-on and get off from the elevator car.

$T_{speed\text{-}up}$ Time duration for elevator car to reach in maximum speed from starting condition.

2.6.3 Parameter for calculation of Average Waiting Time of passenger :

1) Number of floor 30
2) The time required to start and stop the car takes 3S.
3) The time required to travel between two floor takes 5S.
4) Number of elevator car 8 Nos.
5) Floor height is 4mtr.
6) Number of passenger per floor in 5S is 40
7) Number of stop to reach hall call floor is 2.
8) Elevator car distance from hall call floor is 12mtr.
9) Get in and get off time of the passenger in elevator car is 5S.
10) Execution time of fuzzy controller is 17.485ns for case-1 and 36.056ns for case-2.
11) Total number of car call : 10
12) Total number of car call : 30

Based on these data the average waiting time of the passenger becomes 30.428 sec for the algorithm reduce rule technique, whereas this time was taken to be 54.69 sec for actual mamdani's inference algorithm. The average waiting time is calculated based on the average traffic flow in particular time instance and computation time of the fuzzy controller. Table2 .3 shows the comparison result of average waiting time of both the algorithms.

AWT	***Time***
Average Waiting Time calculation using reduce rule mamdani's inference Fuzzy algorithm.	30.428 Sec
Average Waiting Time calculation using actual mamdani's inference Fuzzy algorithm.	54.69 Sec

Table-2.3 Comparison result of average waiting Time of both the algorithms

CHAPTER - 3

FPGA IMPLEMENTATION OF DISPATCHING ALGORITHM FOR ELEVATOR CONTROL SYSTEM

3.1 INTRODUCTION

Waiting for elevator car is a major situation with this we all are familiar. When we press a button and wait for an elevator, we may have to wait for a long time, if there are too many passengers or not enough elevator car is present in particular situation. Important thing is that, how much time we wait, which depends on the dispatching strategy of the elevator car in particular time instance. Efficiencies of multiple elevators installed in an office building may increase if a central dispatcher is used to group passengers going to the same floor to the same elevator. Figure 3.1 shows the four elevator dispatching strategy. Here each elevator has a position, direction and speed. It has also set of buttons to indicate where passengers want to get off. Dispatching algorithm for elevator is generally designed primarily for the different periods that is for morning and evening rush hours.

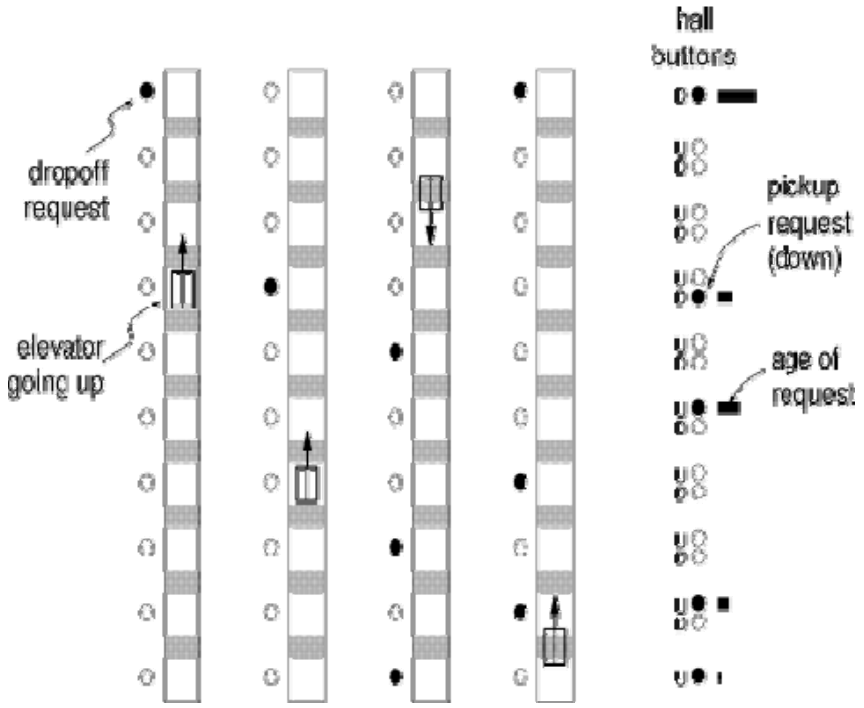


Figure 3.1 Four elevator dispatching strategy

The performance of elevator dispatching algorithms is measured in several ways, all with respect to the how much passenger entering the system. The average waiting time is how long the passenger wait before getting on an elevator and

the average system time is required for passenger to reach its destination floor. Another encountered statistics of the passenger is that whose waiting time exceeds 60 seconds.

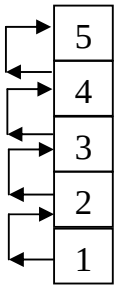
The most important thing is every elevator formulated some action based in the dispatching algorithm. First, each elevator made its own decisions independently and second, the numbers of constraints were placed on the decision. An elevator carrying the passenger could not pass the floor, if any passenger wanted to get off the floor, if any passenger wanted to get off his respective floor, nor could it reverse the direction until all the passengers wanting to go in current direction to their respective floor.

3.2 HARDWARE IMPLEMENTATION OF THE DISPATCHING ALGORITHMS

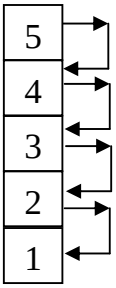
Here six types of dispatching algorithms are considered. Based on traffic condition elevator control system can choose one algorithm from six, and operate in this algorithm for some period until the traffic condition is again changed. This would facilitate reducing the average waiting time (AWT) of the passenger and also reduce the power consumption of elevator system. Figure 3.2 shows the six type of dispatching algorithms.

3.2.1 Collective-up Algorithm(CU)

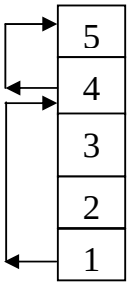
Collective-up algorithm is one of the important aspects of elevator control system. It is used mainly when traffic situation is very low. In this case collective principle is used when elevator car moves in upward direction. The car stops in the every floor based on the hall calls and car call during it upward direction journey. This is done by taking the nearest hall-call in place of the order in which the calls were given. Figure shows the sequence of how the elevator car can visit the floors in up trip for twenty floors building.



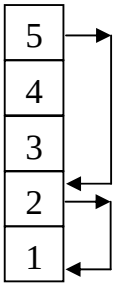
Collective-UP Algorithm



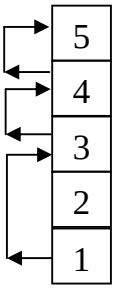
Collective-Down Algorithm



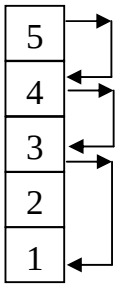
Selective-UP Algorithm



Selective-Down Algorithm



Selective / Collective -UP Algorithm



Collective / Selective Down Algorithm

Figure 3.2 shows the six type of dispatching algorithm

3.2.2 Collective down Algorithm (CD)

Here we use collective principle when elevator car moves downward direction. The car stops in the every floor based on the hall calls and car call during it downward direction journey. This is done by taking the nearest hall-call in place of the order in which the calls were given. Figure shows the sequence of how the elevator car can visit the floors in a down trip for twenty floors building.

3.2.3 Selective down Algorithm (SD)

Selective down algorithm is used during Down Peak mode, elevator cars in a group are sent away from the lobby towards the highest floor served, after which they commence running down the floors in response to hall calls placed by passengers wishing to leave the building. This allows the elevator system to provide maximum passenger handling capacity for people leaving the building. The commencement of Selective down algorithm may be triggered by a time clock, by the arrival of a certain number of fully loaded cars at the lobby within a given time period, or by a switch manually operated by a building attendant.

3.2.4 Selective-up Algorithm (SU)

Selective up algorithm is used during Up Peak mode, elevator cars in a group are recalled to the lobby to provide expeditious service to passengers arriving at the building, most typically in the morning as people arrive for work or at the conclusion of a lunch-time period. Elevators are dispatched one-by-one when they reach a pre-determined passenger load, or when they have had their doors opened for a certain period of time. The next elevator to be dispatched usually has its hall lantern or a "this car leaves next" sign illuminated to encourage passengers to make maximum use of the available elevator system capacity. . The commencement of Selective up algorithm may be triggered by a time clock, by the departure of a certain number of fully loaded cars leaving the lobby within a given time period, or by a switch manually operated by a building attendant.

3.2.5 Selective-collective Up- Algorithm (SCU)

Selective-collective-up is most important algorithm in multi-storage building particularly in morning time, when all the passenger want to go to different floor of the building from ground floor. This algorithm first operates in selective mode that means elevator first moves upward direction selectively and stops at particular floor. Then it moves to collective mode by reaching top floor with halting at every floor.

3.2.6 Collective – Selective Down- Algorithm (CSD)

Collective-selective-down is most important algorithm in multi-storage building particularly in evening time, when all the passenger want to come to ground floor from different floor of the building .This algorithm first operates in collective mode that means elevator first moves downward direction collectively and stops at all the floors up to some extent. Then it moves to selective mode by directly going to ground floor without halting in between floors even if pressing of hall cal and car call.

3.3 SIMULATION RESULT

Figure 3.3 shows the simulation result for collective up algorithm. From the figure it is shown that based on the hall-call, the elevator car can attend the respective floor collectively i.e. one after another.

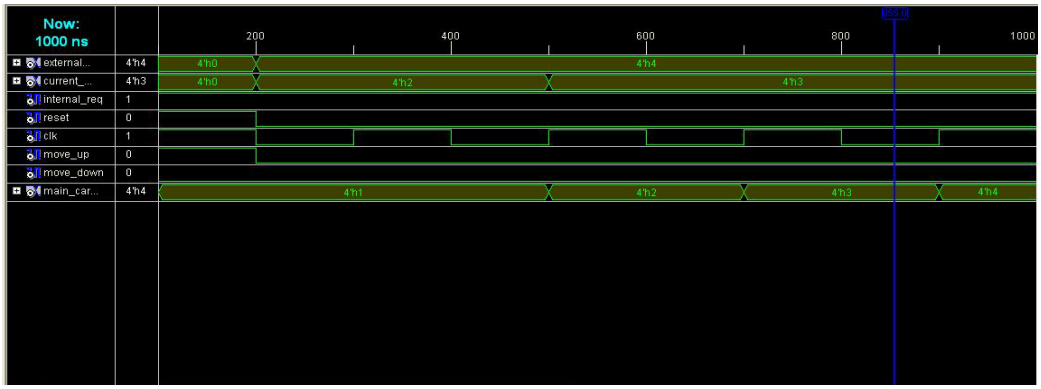


Figure 3.3 Simulation result of collective-up algorithm

Figure 3.4 shows the simulation result of collective down algorithms. From the figure it is shown that based on the hall-call the elevator car attends the respective floor collectively.

It is shown that from figure 3.4 that hall call comes from the floor ‘3’ and floor ‘2’, because of that elevator car directly attends the floor 3 and floor 2 from floor 5.It skips the floor 4, because there is no hall call coming from the floor 4. After that it comes to the ground floor.

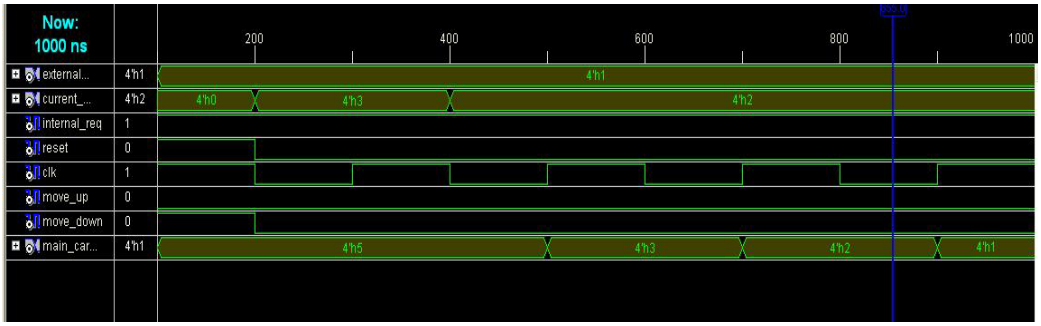


Figure 3.4 Simulation result of collective-down algorithm

Figure 3.5 shows the simulation result of selective down algorithms. From the figure it is seen that, hall call and car call are coming from the floor number 2, 3 and 4 .Based on the algorithms the elevator car attends the lowest hall call floor that is floor number 2, after that it comes to ground floor.

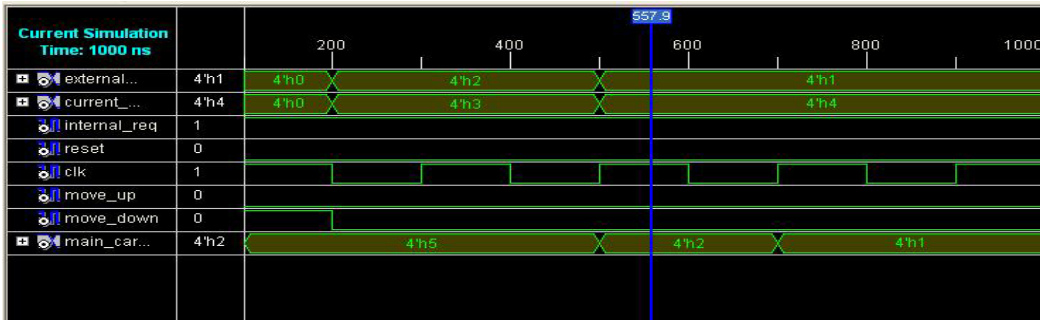


Figure 3.5 Simulation result of selective-down algorithm

Figure 3.6 shows the simulation result of selective up algorithms. From the figure it is seen that, hall call and car calls are coming from floor number 2, 3 and 4. Based on the algorithms, the elevator car attends the highest call that is the floor number 4, after that it comes to the floor number 5.

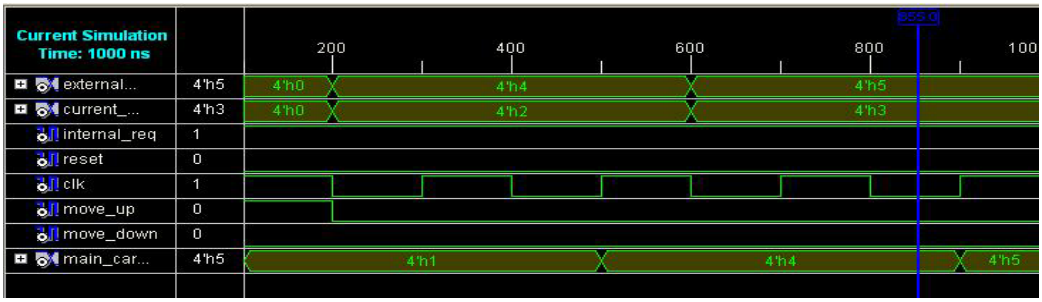


Figure 3.6 Simulation result of selective-up algorithm

Figure 3.7 shows the simulation result of selective-collective up algorithms. From the figure it is seen that hall call and car call are coming from the floor number 1, 2, 3 and 4. In response to that the elevator car moves selectively i.e. from floor number '1' to floor number '3'. After that it moves collectively to floor number '4' and '5'.

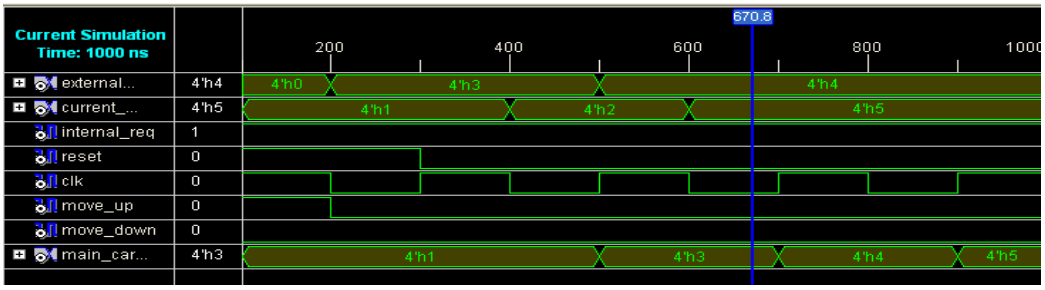


Figure 3.7 Simulation result of selective-collective-up algorithm

Figure 3.8 shows the simulation result of collective-selective down algorithms. From the figure it is seen that hall calls and car calls are coming from the floor number 1, 2, 3 and 4. In response to that the elevator car comes in collectively to attend the floor number 4 and 5, after that it moves selectively to floor number '1'.

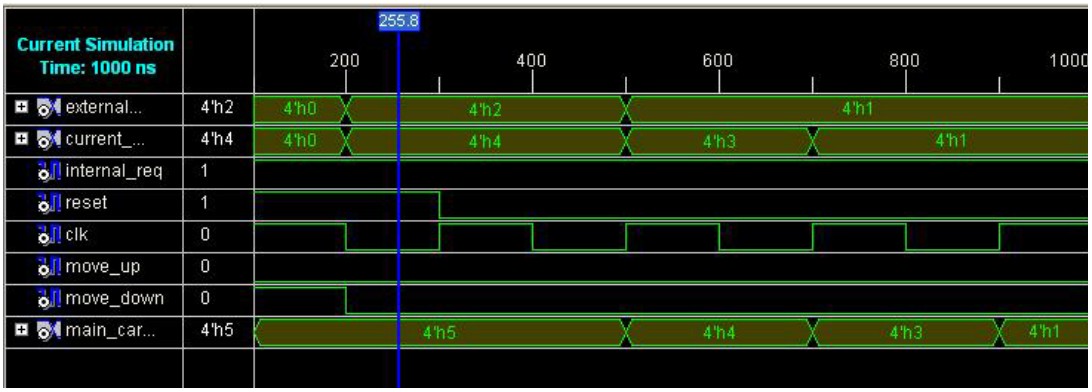


Figure 3.8 Simulation result of collective selective-down algorithm

3.4 FPGA IMPLEMENTATION

The design of dispatching algorithm for controlling the elevator system is implementing using VHDL. Synthesis process has been performed using xilinx tools. The technology mapping chosen for implementation of this dispatching algorithm is spartan 3E (SC3S500E) FPGA with FG320 package and speed grade of – 4.

Device Utilized FPGA	**Collective Up**	**Collective Down**	**Selective Up**	**Selective Down**	**Collective / Selective UP**	**Collective / Selective Down**
Number of Slices	30 out of 4656 1%	29 out of 4656 1%	13 out of 4656 1%	13 out of 4656 1%	25 out of 4656 1%	25 out of 4656 1%
Number of Slices Flip flop	11 out of 9312 1%	11 out of 9312 1%	9 out of 9312 1%	10 out of 9312 1%	10 out of 9312 1%	11 out of 9312 1%
Number of four input LUTs	60 out of 9312 1%	58 out of 9312 1%	24 out of 9312 1%	24 out of 9312 1%	46 out of 9312 1%	47 out of 9312 1%
Number of IOs	17	17	17	17	17	17
Number of bounded IOBs	15 out of 232 6%	15 out of 232 6%	15 out of 232 6%	15 out of 232 6%	15 out of 232 6%	15 out of 232 6%
Number of GCLKs	1 out of 24 4%	1 out of 24 4%	1 out of 24 4%	1 out of 24 4%	1 out of 24 4%	1 out of 24 4%
Minimum Period	6.683 ns	6.569 ns	4.512 ns	4.653 ns	5.987 ns	5.934 ns
Maximum Frequency	149.633 MHz	145.583 MHz	221.631 MHz	222.835 MHz	167.029 MHz	168.934 MHz
Total Memory Uses	160308 kilobyte	160308 kilobyte	160308 kilobyte	160308 kilobyte	160308 kilobyte	160308 kilobyte
Power	0.08313 W	0.08314 W	0.08313 W	0.08313 W	0.08351 W	0.08405 W

Table 3.1 The comparison result of device utilization summary of all six types of dispatching algorithms.

CHAPTER – 4

IMPLEMENTATION OF ELEVATOR CONTROL SYSTEM USING HCS-12 (MC9S12DP256B)

4.1 INTRODUCTION

The main aim of this thesis work is the implementation of the elevator control system using HCS-12 i.e. MC9S12dp256B microcontroller. To implement this a prototype elevator set-up is considered which consists of one D.C .Motor,four level switches,sensor and display module.

This D.C. Motor is connected with gear set up, one wheel is attached with this gear for moving the elevator rope in both upward and downward direction. One end of the rope is connected with gear wheel system and another end of the rope is connected with the car module. One optical encoder is attached with the wheel of the D.C. Motor for indicating the speed of the D.C. Motor. This also provides the feedback signal to the fuzzy PID controller for maintaining the constant speed of D.C. Motor in different load condition.

In our proposed four floor elevator system we use two switches in every floor, one for moving the elevator car in upward direction and another for moving the elevator in downward direction. Two number of sensors are used in every floor ,one for doing stop the elevator car in respective floor and another for opening and closing the elevator door for get in and get out the passenger in the respective floor. This project is also implemented with FUZZY PID controller for providing intelligence to the elevator car in different load condition for smooth running of elevator. In this technique the speed and direction of the DC motor are controlled by MCU by inputting PWM signals to its H-bridge drive circuits. The elevator stops when the information input by the key and the feed-back signals of the infrared detection circuits are the same. The elevator's upward and downward directions are displayed by Display Module.

4.2 ABOUT MC9S12DP256B MICROCONTROLLER

MC9S12DP256B is a 16-bit microcontroller provided by Free-scale consisting a 16-bit central processing unit (STAR 12 CPU), 256K bytes of flash EEPROM, 12K bytes of RAM, 4Kbytes of EEPROM, an Enhanced capture timer, an 8 channel pulse width modulator (PWM), two 8 channel, 10 bit analog to digital converters (ADC), two asynchronous serial communications interfaces (SCI), three

serial peripheral interfaces (SPI), Five CAN 2.0 A/B software compatible modules (MSCAN 12), and an inter IC bus.

4.2.1 Features

- 16 bit STAR 12 CPU

  It has a high speed 16 bit processing unit, and wider internal registers (up to 20 bits) for extended math instruction. It offers an extensive set of indexed addressing capabilities.

- Multiplexed external bus.
- Memory

  It consists of 256K bytes of flash EEPROM, 4Kbytes of EEPROM, and 12Kbytes of RAM.
- Two 8 channel analog to digital converters, 8/10 bit resolution.
- 8 channel enhanced capture timer

  Each channel can configure as input capture, output compare, or pulse accumulator.
- 8 PWM channels with programmable period and duty cycle.
- Five CAN 2.0A/B software compatible modules having a maximum baud rate of 1Mbps.
- Inter IC bus (I2C).
- Serial interfaces

  Two asynchronous serial communications interface (SCI)

  Three synchronous peripheral interface(SPI).
- Byte Data Link Controller (BDLC).
- System Integration Module (SIM)

  Includes Clock and Reset Generation (CRG) module, Multiplexed External Bus Interface (MEBI), Module Mapping Control (MMC), Interrupt control, Background debug mode (BDM)
- Available in two packages, 112-pin LQFP package or 80-pin QFP package.
- Maximum speed of 50MHz equivalent to 25MHz bus speed.
- Generate 2.25 to 2.75V Digital supply voltage using an internal voltage regulator.
- Analog and I/O supply voltage 4.75 to 5.25V.

- 0.25 micron CMOS technology.

4.2.2 CPU (Star 12)

The STAR 12 CPU is a high speed 16 bit processor having a full 16 bit data paths and wider internal registers (up to 20 bits) for high speed extended math instruction. It has an instruction pipe to increase the execution speed. It supports an extensive set of indexed addressing mode.

Accumulator: A and B registers are the 8 bit general purpose accumulators used by the CPU to hold the operands and results for its processing. In 16 bit operation this will be treated as accumulator D.

Index registers: IX and IY are the index registers for keeping effective address in indexed addressing mode.

Stack pointer: Stack pointer points to the last location of the stack used. It can also be used as a pointer in indexed addressing mode. CPU supports an automatic program which stores the system contexts during subroutine call and interrupts.

Program counter: This 16 bit register points to the next instruction to be executed. It can be used in all indexed addressing mode except auto increment/ decrement.

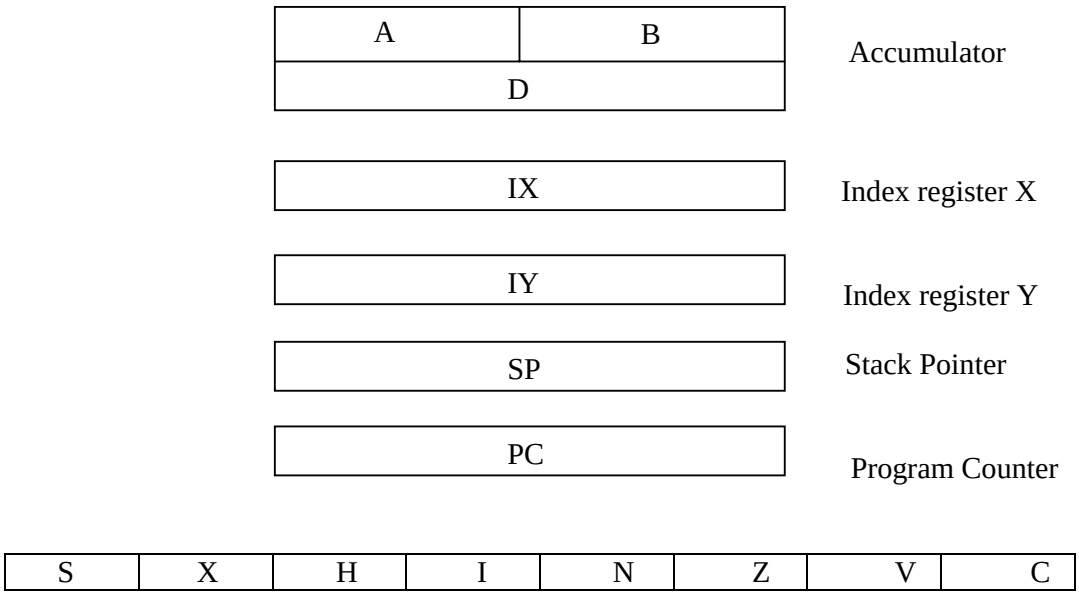


Fig 4.1 Register set for CPU 12

Condition code register: There are five status flags in the conditional code register, they are carry/ borrow flag (C), overflow flag (V), zero flag (Z), negative flag (N), and half carry flag (H). This register also contains two interrupt masking bits, X and I. S bit corresponds to stop disable bit.

The data types supported by this CPU are;

Bit data

8 bit and 16 bit signed and unsigned integers

16 bit unsigned fractions

16 bit address.

Addressing modes supported by the CPU are;

Direct addressing mode

Immediate addressing mode

Inherent addressing mode

Extended addressing mode

Relative addressing mode

Indexed addressing mode

The indexed addressing mode includes, 5 bit offset indexed addressing mode, auto pre-decrement, auto pre-increment, auto post-decrement, auto post-increment, accumulator offset, 9 bit offset, indexed 16 bit offset, indexed indirect 16 bit offset, and Indexed indirect accumulator D offset.

4.2.3 Operating Modes

Input BKGD & Bit MODC	MODB	MODA	Mode description
0	0	0	Special single chip, BDM allowed and active. BDM allowed in all other modes but a serial command is required to activate
0	0	1	Emulation expanded narrow, BDM allowed
0	1	0	Special test (Expanded wide), BDM allowed
0	1	1	Emulation Expanded wide, BDM allowed
1	0	0	Normal single chip, BDM allowed
1	0	1	Normal expanded narrow, BDM allowed
1	1	0	Peripheral; BDM allowed but bus operations would cause conflicts must not be used.

1	1	1	Normal expanded wide, BDM allowed

Table 4.1 Operating modes of STAR 12 CPU

The operating mode in which the microcontroller works, after reset determined by the MODA, MODB, and MODC pin status during reset. The register MODE which contains MODA, MODB, MODC bits, shows the current operating mode during operation.

4.3 : ELEVATOR POSITIONING CONTROL SYSTEM

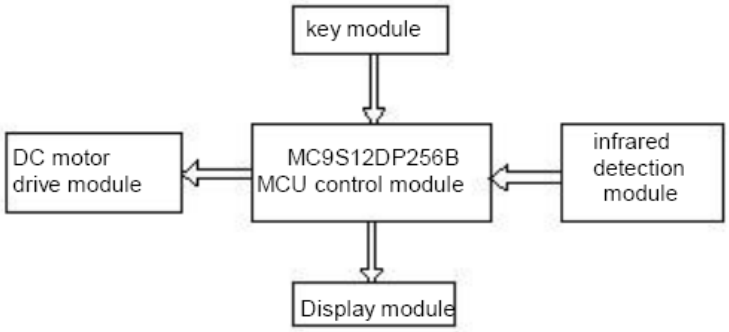


Figure 4.2 : Elevator system model structure diagram

The basic structure diagram of elevator system positioning control system model is shown in Figure 4.2. The model includes MCU control module, DC motor driver module, infrared detection module, display module and key module. The elevator's running path is set up by keys, the real-time information of the elevator's location is set up by keys and the real-time information of the elevator's location is detected by infrared detection circuits and is fed back to the MCU. The speed and direction of the DC motor are controlled through MCU by inputting timing pulse signals to its drive i.e. H-bridge drive circuits. The elevator stops when the information input by the key and the feedback signals of the infrared detection circuits are the same. The elevator's real-time running status is displayed through the display system.

4.4: SYSTEM HARDWARE DESIGN

4.4.1 DC motor drive module design

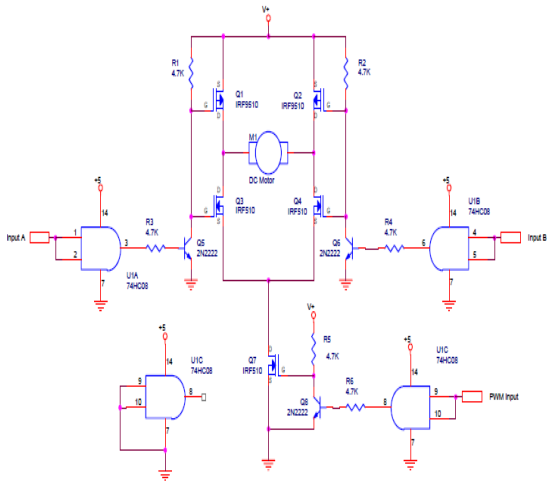


Figure 4.3 shows DC motor driver circuit

DC motor driver circuit is shown in figure 4.3. The PWM input of the DC motor driver circuit is connected with the output port of PWM channel of microcontroller and another two port of DC motor driver circuit is connected with the MCU output port, which makes the DC motor rotate positively or reversely. DC motor rotational speed is directly proportional to the frequency of input pulse, which means the higher frequency of the input pulse, the faster DC motor rotate speed, and vice versa.

This circuit ties together the input of Q1 to that of Q3, and the input of Q2 to that of Q4. As a result, both Q1 and Q3 or Q2 and Q4 will never be on at the same time, since the N-MOS and P-MOS devices are active for opposite polarity signals (I.e. if ground is applied, the P-MOS will be active while the N-MOS will be off). The addition of the two AND gates allows for a PWM signal to control speed. It is set up such that the PWM signal will only be applied to the N-Channel (Q3 and Q4). The reason for this is because for similar N and P channel MOSFET, the N channel MOSFET will usually have a smaller gate capacitance, and will turn on and off quicker than the P channel MOSFET. It is more efficient to leave one MOSFET active, while toggling the other with a PWM signal. To have the motor always on at full speed, set the PWM input to a steady. For the AND gates, a high speed chip such as the 74F08 is desirable. Here we use the AND gates and tie the gate of Q1 to the

gate of Q3 and the gate of Q2 to the gate of Q4 directly. Turning either A or B high will cause the motor to rotate in one direction or the other. If they are both high and both low at the same time, then the motor will actively brake. If we do not need active braking in the circuit, we can connect wire B to wire A via an inverter, so that B will always be the opposite of A. we then only need to toggle A for direction control, and the PWM line for speed.

When changing the speed of the motor, it is good idea to ramp it, as sudden changes can cause a lot of back EMF and noise to be produced. Ramping means slowly increase the speed, by slowly changing the duty cycle of the PWM signal. Slow are a relative term, and often a ramp time of half a second is more than enough for small motors. This will depend on motors and robot, as we will need to consider the rotational inertia of the motor, and how much current will be drawn for rapid changes in motor speed.

4.4.2 Switch Interfacing module

Figure 4.4 shows the general interfacing between the key module and HCS-12 (MC9S12DP256B) microcontroller. The algorithm used in key interfacing is as follows

1. Port AD accepts an 8 bit combination from the DIP switches.
2. The HCS12 will read the value from Port AD and store it in memory.
3. The HCS12 will perform a couple bit twiddling operations on the value.
4. The result from bit twiddling will be sent to port T.
5. Port T will send the value to the DC motor driver circuit.

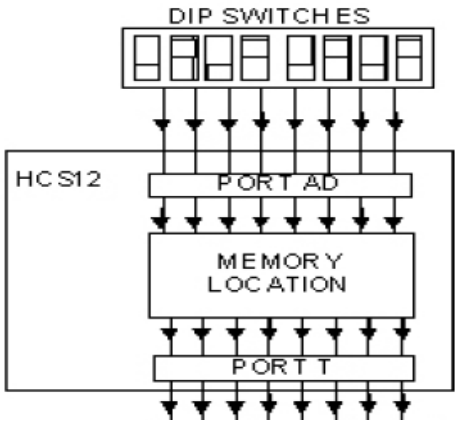


Figure 4.4 shows the general interface between the key module and the HCS12

From the program guidelines, we need to program port Ts registers to serve as general output and PORT AD to serve as general input. We will start with port T since it is easier. Register DDRT is an 8 bit data direction register for port T. Each bit corresponds to the direction of the data line connected to port T. That is one for output and zero for input. We need to configure port T for 8 bit output we will assign 1s to all the DDRT' sbits. Register PTT is the 8 bit transmit register that we will use to send data to the DC motor driver circuit.

4.5 INFRARED DETECTION MODULE DESIGN

4.5.1 Wheel speed sensor

The sensor module consists of an infrared reflector sensor and a code wheel sticker. The infrared light emitted from the sensor will reflect back after hitting the black and white sticker. A photo transistor in the module will receive little or more reflected light depends on reflection surface. The light intensity comes from the black area will be less and it cannot switch on the transistor that will cause a 'Logic 1' output. The reflected light from the white area make the transistor on and the resulting sensor output will be 'Logic 0'. When the wheel finishes a complete round, a total of nine pulses will get at the output of the sensor.

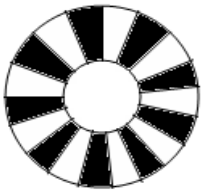


Figure 4.5 : Code wheel sticker

4.5.2 Detecting sensor

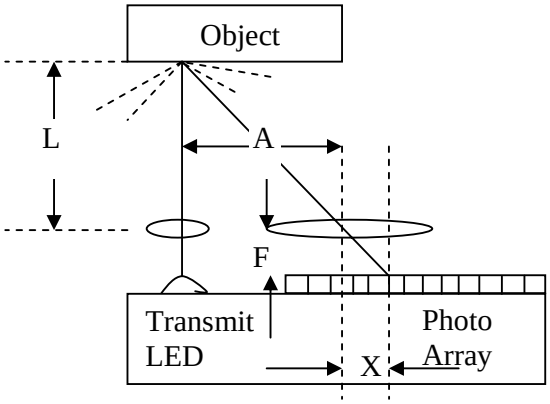


Fig 4.6 Infrared sensor functioning

The infrared distance sensor transmits light to the object in front. The light will pass through a condense lens so that the light intensity is focused to a point. The reflected light will capture through another lens to determine the point of impact. And this can be calculated using the formula:

$$\frac{L}{A} = \frac{F}{X}$$

Figure 4.7 shows the GP2d120xf77 Infrared sensor This sensor can measure a range from 4 to 80cm and the output voltage is ranging from 0.4V to 2.4V when supplied by +5V. An acknowledge period of 32 to 52.9 ms is required before reading the output.

Figure 4.7: Shows the GP2d120xf77 Infrared (IR) Sensor

4.6. OPENING AND CLOSING OF ELEVATOR DOOR SYSTEM

The automatic doors at grocery stores and office buildings are mainly there for convenience and as an aid for handicapped people. The automatic doors in an elevator, on the other hand, are absolutely essential. They are there to keep people from falling down an open shaft.

Elevators use two different sets of doors: doors on the cars and doors opening into the elevator shaft. The doors on the cars are operated by an electric motor, which is hooked up to the elevator computer. You can see how a typical door-opener system works in the Figure 4.8 below. The electric motor turns a wheel, which is attached to a long metal arm. The metal arm is linked to another arm, which is attached to the door. The door can slide back and forth on a metal rail.

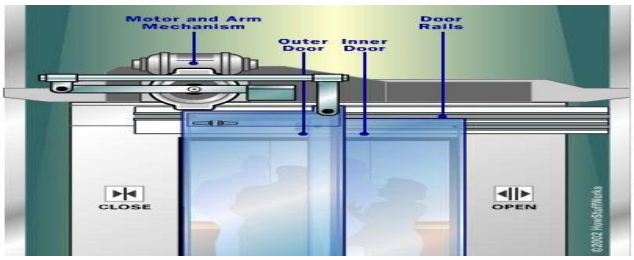


Figure 4.8 shows Opening and closing of elevator door system

When the motor turns the wheel, it rotates the first metal arm, which pulls the second metal arm and the attached door to the left. The door is made of two panels that close in on each other when the door opens and extends out when the door closes. The computer turns the motor to open the doors when the car arrives at a floor and close the doors before the car starts moving again. Many elevators have a motion sensor system that keeps the doors from closing if somebody is between them. Figure 4.9 shows the DC motor driver interfacing module for opening and closing the elevator door. Figure 4.10 shows the result of opening and closing of elevator door system by moving the DC motor in both forward and reverse direction.

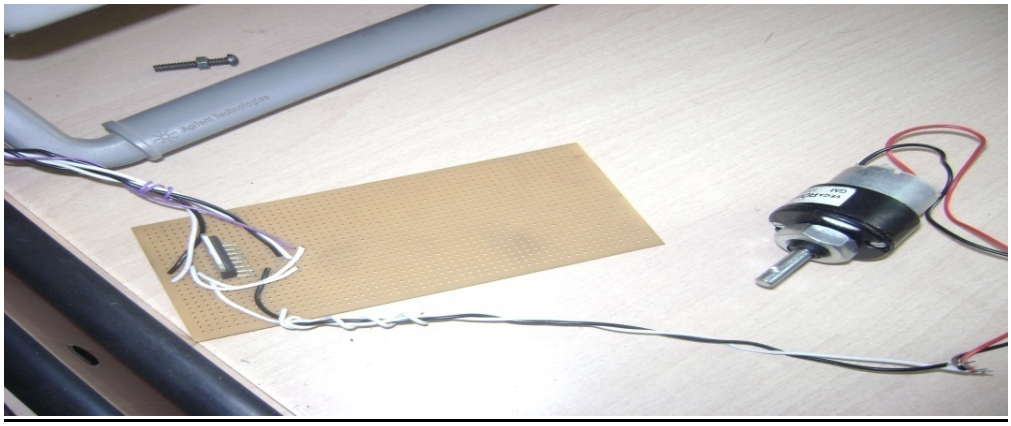


Figure 4.9 shows the driver interfacing module for D.C motor

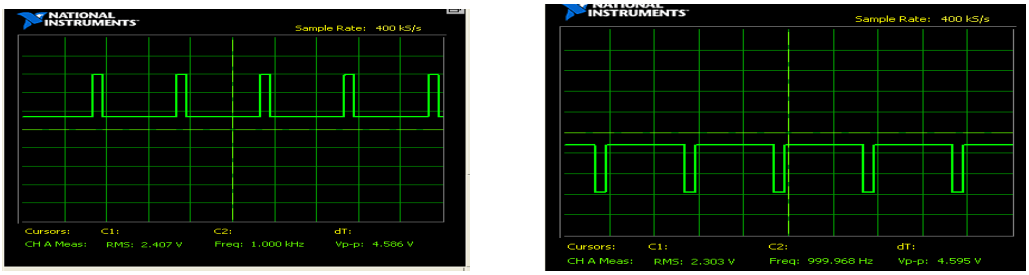


Figure 4.10 : Result of opening and closing of elevator door system

4.7 FUZZY PID CONTROLLER

4.7.1 PID controller

Here we used fuzzy PID controller for controlling the speed of the DC motor with various load condition. The PID controller is the closed loop controller that is shown in Figure 4.11. Here we consider S_d be the desired speed, S_m be the measured speed, and PWM be the PWM value. Define the error is defined as $e(t) = S_D - S_M$ and the resultant PWM signal is base on the $PWM = PWM_P + PWM_I + PWM_D$

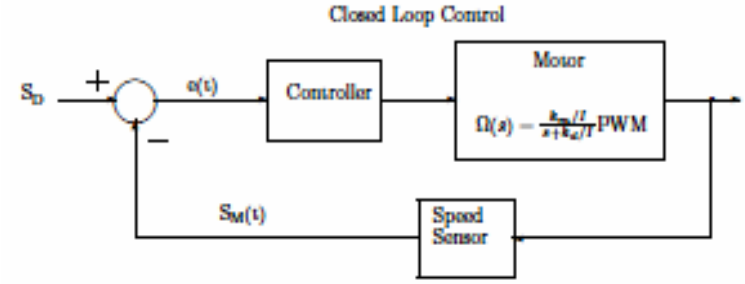


Figure 4.11.Closed loop PID controller

Here we the find k_P, k_I and k_D such that motor responds in a good way. Control theory shows how to find good values for k_P, k_I and k_D, based on the characteristics of the system. Control theory requires one to make a complex mathematical model of the plant to be controlled, and then mathematically analyze the closed-loop control system to find a good controller. Newer control methods have been developed which try to control a system in an intuitive way, like a human.

4.7.2 : Fuzzy logic control algorithms

The control of the rotation speed of DC motors is very complicated when done using traditional control techniques, as it requires a very complex mathematical model. Using Fuzzy logic eliminates the need of mathematical modeling and allows easy realization of a solution. Fuzzy logic defines rules that determine the behavior of the system using word descriptions instead of mathematical equations.

The algorithm consists of three steps:

1. Fuzzification
2. Fuzzy Inference
3. Defuzzification

Fuzzification is the process which determines the degree of membership of the input values to defined fuzzy sets (linguistic variables). In the case of the rotation speed control of DC motors, the input values are:

1. Absolute error in the rotation speed:
Error = Set Speed – Current Speed

2. Differential rotation speed error. This value is obtained by subtracting the previous error value from the current error value:
dError = Error – Last Error

In this application demo, five fuzzy sets are defined for the input values Error and dError:

1. NM: negative medium
2. NS: negative small
3. ZE: zero equal
4. PS: positive small
5. PM: positive medium

The membership function that is shown in figure 4.12 are triangular-shaped and the maximum value is scaled to 400h instead of 1 which is found in other documents describing fuzzy theory. This way the calculation complexity is greatly reduced because the multiplying operation becomes only one addition or subtraction.

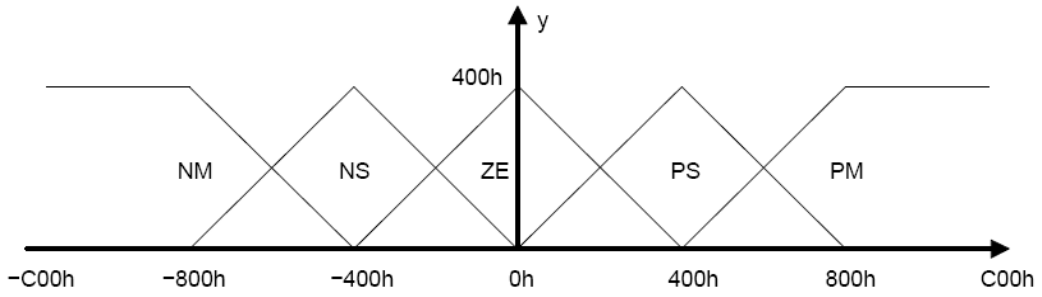


Figure 4.12 Input Membership Function

The result of the fuzzification of an input value is a vector with five elements as there are five fuzzy sets, and the value of each member defines the degree of membership of the input value to a particular fuzzy set (y-value). The vectors for the absolute and differential errors which are the results of the fuzzification are denoted as X1[ ] and X2[ ].

Fuzzy Inference considers the Mamdani max-min relations and it interprets fuzzy value from the fuzzification module, and it's assigned the certain value to the output based on set of fuzzy rule. Table 4.2 shows that fuzzy inference rule

		dError (X2[ ])				
		NM	NS	ZE	PS	PM
Error (X1[ ])	NM	PM	PM	PM	PS	ZE
	NS	PM	PM	PS	ZE	NS
	ZE	PM	PS	ZE	NS	NM
	PS	PS	ZE	NS	NM	NM
	PM	ZE	NS	NM	NM	NM

Table 4.2.Fuzzy Inference Rule Table

The result of the defuzzification has to be a numeric value which determines the duty factor of the PWM signal used to drive the motor. It is obtained by finding the centroid point of the function which is the result of the multiplication of the output membership function and the output vector Y[ ]. The general mathematical formula which is used to obtain the centroid point is shown in equation 4.1. Figure 4.13 shows a graphical representation of the output membership function as used in this application with the coefficients [-10h, -8h, 0h, 8h, 10h].

$$\text{Defuz} = \frac{\sum_{i=1}^{5} Y[i] \times multfact[i]}{\sum_{i=1}^{5} Y[i]} \qquad (4.1)$$

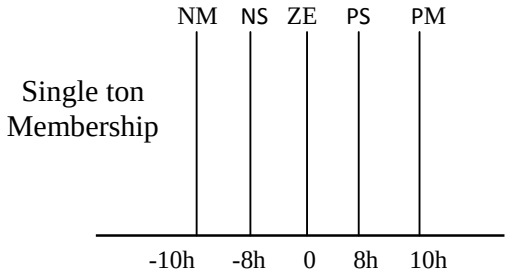


Figure 4.13 : Output membership function

The absolute and differential control loop errors Error and dError are calculated based on target speed (variable Set Speed), current speed (variable Current Speed), and the previous error value (variable Last Error). These error values are then transformed into Fuzzy vectors X1[ ] and X2[ ] using the function Fuzzification. After fuzzification, the fuzzy inference rules are applied and the Fuzzy output vector Y[ ] is generated through calling the Fuzzy Inference function. This output vector is then transformed back into a single control loop output value by calling Defuzzification and it is added to the current PWM duty cycle. In this way the control loop is closed.

4.8. RESULT AND ANALYSIS

Figure 4.14 shows the total Experimental setup of Elevator control system. This setup consists of different modules such as DC motor driver circuit, Key module, Infrared module, display module and Opening and closing of elevator door system. The main control unit is MC9S12DP256B microcontroller.

The elevator’s running path is set by keys and the elevator’s running location is detected by the infrared tubes. MCU controls the speed and direction of the DC motor by inputting pulse signals to its drive. Display module displays the real-time information of elevator’s running status. Based on the switch pressed in the particular floor the elevator is moved either in upward or downward direction. Two infrared sensors are used in this project. One sensor is used for detecting the elevator car in the particular floor and another sensor is used for opening and closing the elevator door. Figure 4.15 shows the result of both upward and downward direction flow of elevator car.

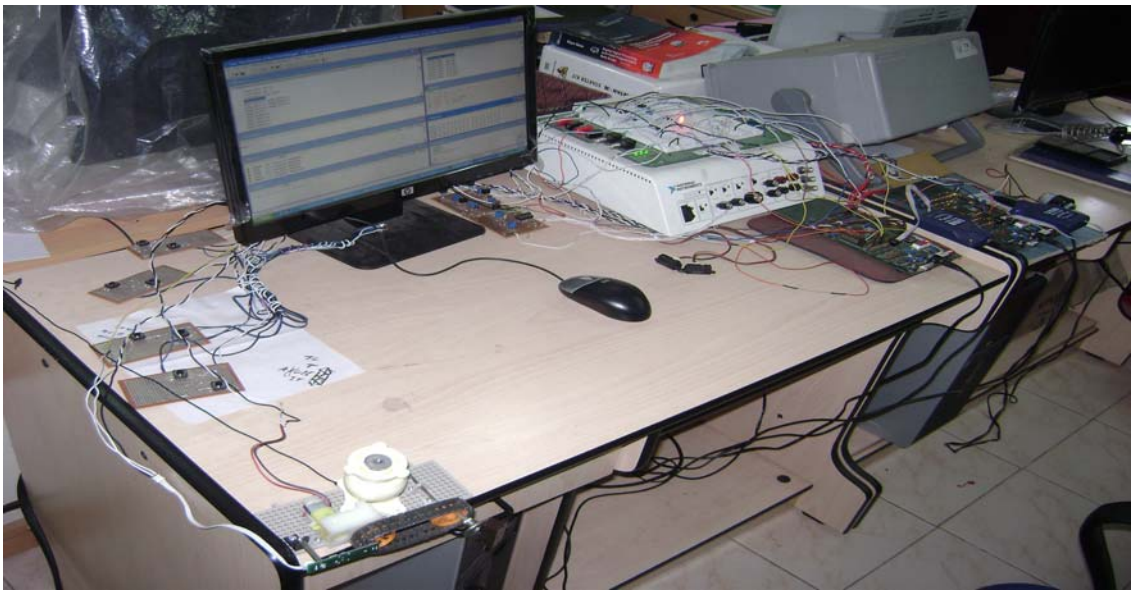
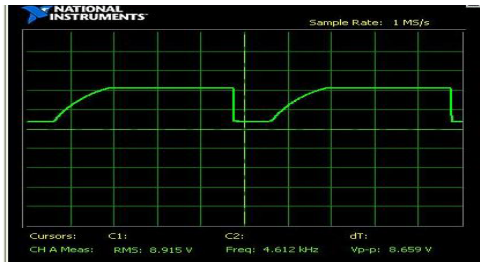
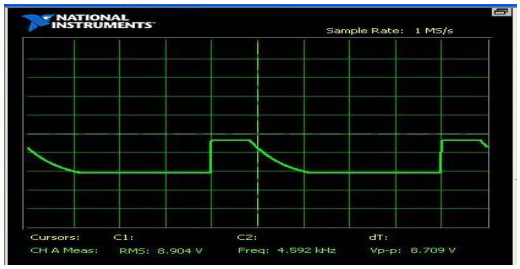


Figure 4.14 Experimental Setup of Elevator control system



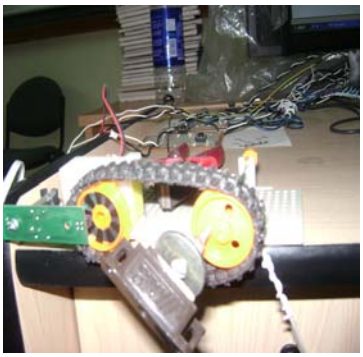
Result of Upward direction flow



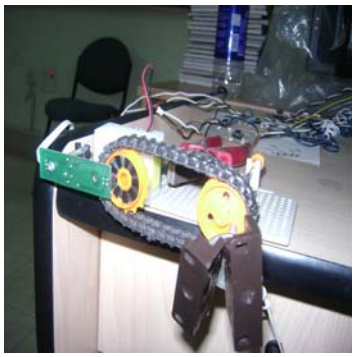
Result of downward direction flow

Figure 4.15 : Result of upward and downward direction flow of Elevator system

The result is also analyzed with the different load condition for smooth running of elevator control system. Here we maintained the constant speed of the DC motor by varying the PWM signal duty cycle. This variation of PWM duty cycle is inputting to the driver circuit of DC motor. When any load change is detected the PID fuzzy controller change the duty cycle of PWM signal and the technique is applied to the DC motor driver circuit for providing constant speed with different load condition. Figure 4.16 and 4.17 shows the different load condition and its corresponding duty cycle variation of PWM signal. Table 4.3 shows the comparison result of speed variation and duty cycle variation of PWM signal with different load factors.

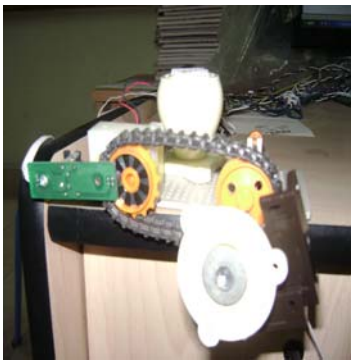


Load-1



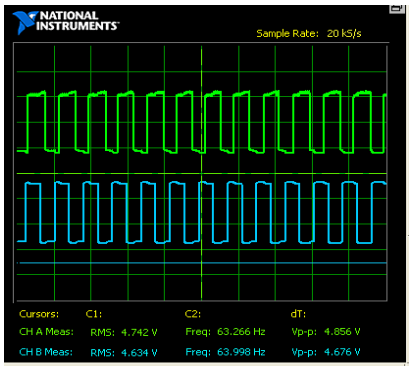
Load-2

Load-3

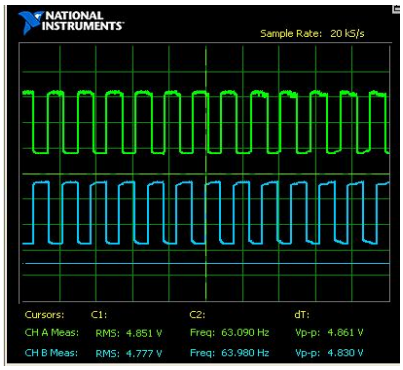


Load-4

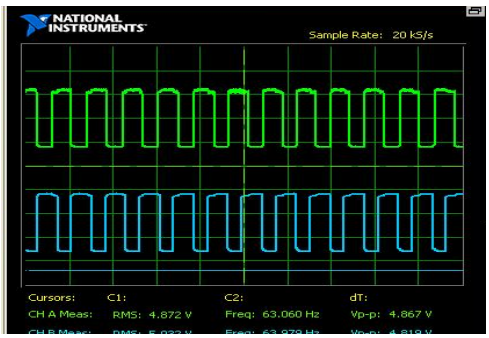
Figure 4.16 shows different load condition



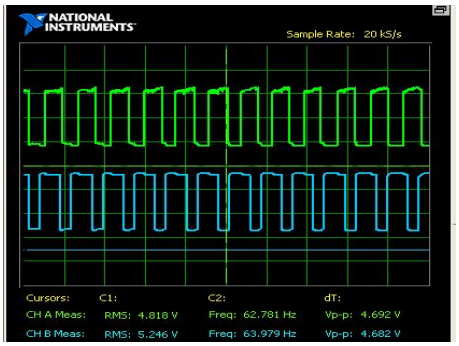
(a) result for no load conditionn



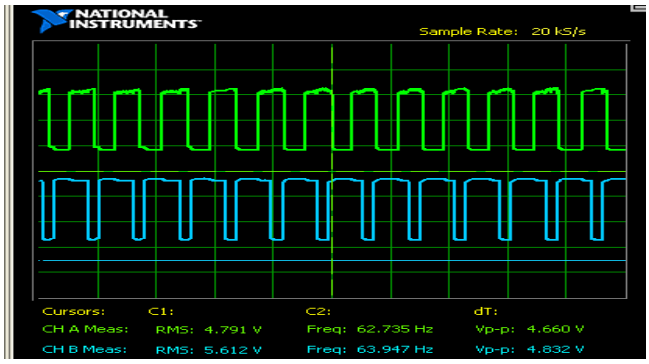
(b) result for load condition 1



(c) result for load condition 2



(d) result for load condition 3



(e) result for load condition 4

Figure 4.17: Duty Cycle variation of PWM signal (a) result for no load condition (b) result for load condition 1, (c) result for load condition 2, (d) result for load condition 3, (e) result for load condition 4.

Load	Speed measured through the optical encoder in terms of second	PWM duty cycle (ON time) variation in term of second
With no load condition	18ms	8ms
Load Factor-1	18.2ms	10ms
Load Factor-2	18.4ms	12ms
Load Factor-3	18.4ms	13ms
Load Factor-4	18.6ms	15ms

Table 4.3: Comparison result of speed variation and duty cycle variation of PWM signal with different load factor.

CHAPTER – 5

CONCLUSION

5.1 CONCLUSION

The first approach is used for the implementation of a fuzzy logic controller for elevator group control system using VHDL with two different fuzzy algorithms. After that the controller for elevator group control system is implemented on a Xilinx Spartan-3E FPGA. The implementation of the fuzzy logic controller is very straight forward by coding each component of the fuzzy inference system in VHDL according to the design specifications. The design of the FLC is highly flexible as the membership functions and rule base can be easily changed with reduced rule techniques. Because of the reduced rule techniques the computation time of the fuzzy controller is reduced and elevator group control system gives faster performance by reducing the average waiting time (AWT) of the passengers up to 30.428 seconds in comparison to the actual mamdani's inference algorithm.

In second approach, six types of dispatching algorithms were implemented for elevator system using VHDL. Elevator control system uses this algorithm in different situation for smooth operation and reduces the average waiting time of passenger and power consumption of elevator system. The behavior of elevator system is improved by choosing the one algorithm out of five based on the possible traffic situation. All the algorithms share the same type of FSM, which control the elevator system operational function correctly.

In third approach, HCS-12 (MC9S12DP256B) microcontroller is choosen as the core control component for elevator control system and DC motor as the implementation component. Based on the key pressed the elevator moves either in upward or downward direction and infrared tubes are used for detecting the location of the elevator, thus acquiring real-time information for opening and closing of the door of elevator**.** To make the elevator more comfortable for passenger, fuzzy PID controller is implemented. This fuzzy PID controller can vary the duty cycle of PWM signal based on load factor of elevator car and maintain the constant speed of the DC motor for smooth running of elevator system.

5.2 SCOPE OF FUTURE WORK

More Improvement can be brought in the design of FPGA based fuzzy logic controller for Elevator controller for elevator control system by considering self tuning mechanism in fuzzy logic controller. In this mechanism shape of the membership function is modified for getting appropriate result for further reducing the average waiting time of the passenger.

In case of dispatching algorithm, more Improvement can be brought by designing FPGA based fuzzy control system for choosing the best dispatching algorithm based on the traffic situation and traffic condition. This makes the more effectively reduce the average waiting time of passenger and reduce the power consumption of elevator system.

In case of elevator positioning control system using HCS-12 (MC9S12DP256B) Microcontroller, more Improvement can be brought by designing the punch card system for opening and closing of the elevator door system. The total system implemented with FPGA is more reliable to the elevator system.

REFERENCE

REFERENCE

1) J. Jamaludin, N.A. Rahim, W.P.Hew, “Development of self tuning fuzzy logic controller for intelligence control of elevator system,” Journal of Engineering Applications of Artificial Intelligence ELSEVIER 22 (2009)1167–1178.

2) Daniel M. Munoz, Carlos H. Llanos, Mauricio Ayala-Rincon, “Distributed Approach to group control of elevator systems using fuzzy logic and FPGA implementation of dispatching algorithms,” Journal of Engineering Applications of Artificial Intelligence ELSEVIER 21 (2008)1309–1320.

3) Akos Becker, Department of Electronics Technology, “Microcontroller based elevator controlling system,” Budapest University of Technology and Economics, Budapest, Hungary , IEEE conference, Jan. 2007.

4) Zhang Yajun, Chen Long,Fan Lingyan,” A Design of Elevator Positioning Control System Model,” IEEE int.Conference Neural Networks & Signal Processing, Zhenjiang,China, IEEE conference, Jun. 2008

5) FREDRICK M. CADY, Software and Hardware Engineering. Assembly and C Programming for the Freescale HCS-12 microcontroller .

6) Daijin Kim, member IEEE “An Implementation of fuzzy Logic Controller on the Reconfigurable FPGA system,” IEEE Transactions on industrial Electronics, Vol.47, No.3, June 2000.

7) Gudwin, R., Gomide, F., Andrade Netto, M., 1998. “A fuzzy elevator group controller with linear context adaptation,” In: Proceedings of FUZZ-IEEE98, WCCI’98—IEEE World Congress on Computational Intelligence, Anchorage. IEEE, Alaska, USA, pp. 481–486.

8) Daniel M. Munoz Carlos H. Lianos, Mauricio Ayala-Rincon Rudi van Els, Renato P. Almeida, “Implementation of Dispatching Algorithms for Elevator Systems using Reconfigurable Architectures,” IEEE Conference 2007 February.

9) Xilinx ISE 10.1 Software manuals www.xilinx.com.

10) Fuzzy Control System Design and Analysis: A Linear Matrix Inequality Approach by Kazuo Tanaka, Hua O. Womg, John Wiley and Sons Publications.

11) Fuzzy Controller by Leonid Rezmik Victoria University of Technology, Melbourne, Australia, Newnes Publication.

12) Crites, R.H., Barto, A.G, "Improving elevator performance using reinforcement learning," In : Touretzky, D. (Ed.), Advances in Neural Information Processing Systems, vol. 8. MIT Press, 1996 Cambridge, MA, pp. 1017–1023.

13) Crites, R.H., Barto, A.G., Elevator group control using multiple reinforcement learning agents. Machine Learning 1998.33 (2–3), 235–262.

14) Fujino, A., Tobita, T., Segawa, K., Yoneda, K., Togawa, A., "An elevator group control system with floor-attribute control method and system optimization using genetic algorithms," IEEE Transactions on Industrial Electronics Jan 1997 vol 44 (4), 546–552.

15) Gudwin, R., Gomide, F., Andrade Netto, M.,"A fuzzy elevator group controller with linear context adaptation". In: Proceedings of FUZZ – IEEE 98, WCCI' 98 - IEEE World Congress on Computational Intelligence, Anchorage. IEEE, Alaska, 1998 USA, pp. 481–486.

16) Homaifar, A., McCormick, E., "Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms". IEEE Transactions on Fuzzy Sets jun 1995 3 (2), 129–139.

17) Huang, S.J., Lee, J.S., "A stable self-organizing fuzzy controller for robotic motion control". IEEE Transactions on Industrial Electronics 2000 47 (2), 421–428.

18) Imasaki, N., Kubo, S., Nakai, S., Yoshitsugu, T., Jun-Ichi, K., Endo, T., "Elevator group control system tuned by a fuzzy neural network applied method". In: Proceedings of the 1995 IEEE International Conference on Fuzzy Systems. IEEE, Yokohama, Japan, pp. 1735–1740.

19) Imrak, C.E., Barney, G.C.,”Applications of neural networks on traffic control”. In: Proceedings of ELEVCON ’1998. IAEE Publications, Zurich, Switzerland, pp. 140–148.

20) Kim, C. B., Seong, K. A., Kwang, H. L., Kim, J. O,“Design and implementation of a fuzzy elevator group control system”. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans 1998 28 (3), 277–287.