Addis Ababa University
Addis Ababa Institute of Technology (AAiT)
School of Electrical and Computer Engineering

Design of Lift and its Control Using FPGA for Machine Tower of Assela Malt Factory

A thesis submitted to the School of Graduate Studies in partial fulfilment of the requirement for the degree of Master of Science in Electrical and Computer Engineering

By

Bekele Solomon Alemu

Advisor:

Dr. Mengesha Mamo.

June, 2020

Addis Ababa University
Addis Ababa Institute of Technology (AAiT)
School of Electrical and Computer Engineering

Approval by Board of Examiners

Dr. Yalemzewud Negash _____

School Dean Signature

Dr. Mengesha Mamo _____

Advisor`s Name Signature

Dr. Libsework Negash _____

Internal Examiner Signature

Ms. Yalemzerf Getnet _____

External Examiner Signature

Dr. Ermias Tesfaye___ _____

Assoc. Graduate Program Director Signature

Declaration

I, the undersigned, declare that this thesis is my original work, has not been presented for a degree in this or any other university, and all sources of materials used for the thesis have been fully acknowledged.

Bekele Solomon _----------
Name signature

Addis Ababa, Ethiopia June, 2020

Place **Date of Submission**

This thesis has been submitted for examination with my approval as a university advisor.

Dr. Mengesha Mamo. _____

Advisor's Name Signature

Dedicated to my cousin

Ruhama Mamuye *whom I*

lost just before her graduation

due to renal failure.

ACKNOWLEDGMENTS

First of all, I want to start expressing a sincere acknowledgement to my Father and Creator Jesus Christ. Next to my god, I want to express my gratitude to my advisor doctor ***Mengesha Mamo*** *for his guidance and supervision on my research. I received motivation, comments and encouragement from him during my graduate studies.*

My other thanks are to my mother, Shega Tafes, who has supported me in all my activity as mother and father in my educational life, and my sister Beza Tilahun. My mother's praying and moral advice has led me to this success.

I want to include my thankful messages to my classmate friends of Birhanu Gizaw and Girmay Gebru for their continual encouragement and advice in the accomplishment of this thesis.

Lastly, I need to mention my wife`s, Wube and friend, Chenenus Gelana`s encouragement. Their directions and pray was my power.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION AND SYMBOLS

HDL	hardware description language
CAD	Computer Aided Design
REM®	Remote Elevator/lift Monitoring
FPGA	Field Programmable Gate Array
I/O	input/output
ASIC	Application Specific Integrated Circuit
VHDL	Very high speed integrated circuit Hardware Description Language
SRAM	**Static Random Access Memory**
LE	**logic elements**
DSP	Digital Signal Processing
FSM	Finite State Machine
ASM	Algorithmic State Machine
LED	light emitting diode
ASM	algorithmic state machine
PMDM	permanent magnet dc motor
PCB	printed circuit board

ABSTRACT

Assela malt factory has different subsection needed for processing of barley. The machine tower is one which has high frightening feature to climb up or to come down over floor stories. Due to this higher officials of the factory couldn't see top floor machines and problems. Workers of the factory has a problem to transport motors and materials in case of problem occurrence for replacement. So, designing a lift for this room/subsection can hinder these mentioned problems.

Most previous known elevators/lifts are designed using conventional PLC technology. PLC provides flexibility, lower cost and security compared to other control techniques. But these technology has series-cycling implementation on rungs and takes long duration to address the command signal. However, the FPGA programing technique can executes programs in parallel, hence, give fast response for the command signal given to it as input.

In this thesis, lift control is designed and simulated using embedded Xilinx FPGA into Matlab Simulink platform. VHDL code could be generated from Simulink controller block of lift model. The Simulink model simulation result demonstrates that the lift is operating within design specification. It is designed that the lift has a linear speed of 1.25m/s, and acceleration of 0.8m/s^2 (-0.8 m/s^2 deceleration).

Five floor level efficient lift logic control system is designed and simulated using XILINX ISE 8.1i stateCAD. By changing the state diagram, VHDL code is generated and used to implement control system on FPGA Spartan-3E. ***State Bench*** *simulator is used to simulate results of proposed control system. The logic based controller for door opening, closing and level indication all are functioning according to design.*

CHAPTER ONE

INTRODUCTION

1.1 Introduction to Lift

Lifts and escalators plays vital role in the vertical transportation of peoples and materials. They are results of the great human endeavor to save energy and time over vertical transportation. They have some differences in their construction. Lifts are used to move loads from one height to another height through straight direction without disturbing the load itself. Escalators, on the other hand is used to transport peoples on inclined direction by using belts/conveyors.

The discoveries of lifts led construction engineers to design very large tall buildings in the world. Lifts plays vital role in both commercial and residential transportation. It also has great application in mining area as explained on [1].

1.2 Motivation / Statement of Problem

Assela malt factory, which was owed by government until February 21, 2018, is the one that receives students from higher education institutions for their industry linkage program. I was one who joined it for the mentioned program on March 3, 2012. After understanding the factory environment several problems of the factory was identified. One problem of the factory was the non-existence of lift in the machine tower. The tallest part of the factory is this room with having much number of motors inside. All the production materials such as sifting machine, temporarily storing silos, balances and barley elevators etc. are found in this section of the factory. The barley to be processed is given to this as intake and the final output products are dispatched through this section. But, the workers in this section have not any means/access to climb other than walking on foot on stories for diagnosis purpose in case of problem occurrence. As evidenced from the section workers, at least three times up/down movement per each shift (3 shift workers per a day) is conducted within the section.

Since the section is high tall and frightening, the management workers do not visit the problems happened at the top part of the section. Hence, the problem occurred at the top is not mentored by management workers. Also they are not delighted to go to the room and to show for other high officials coming to visit the factory. In case of less weight motor failure and replacement, the

technical workers are obligated to carry to remove the damaged one and bring the replacement over the frightening stories. If the motors on movement escaped from the workers hand, it will have a chance to destroy other beneath wealth or human life. And the escaped motor itself will be damaged and will become non-repairable. This leads economic destruction. Lift installation within the room could be the only solution to overcome this problem.

By the time different literatures has been reviewed concerning lift design, one problem is visible designed by use of PLC. The problem was due to PLC nature of information execution process. The input signals given to the controller passes through series cycles of execution. This series cycling of signal reading and execution has the drawback of time delay. If the microcontroller becomes busy to read software programs, it will result unnecessary time delay for the users. So, other solution which overcome this problem was desirable. The solution was replacing PLC by FPGA devices, which uses hardware programs. Hardware description language (HDL), which is used to program the FPGA devices can remove the series cycling time delay of microcontroller.

1.3 Significance of the thesis

Lift installation within the factory could facilitate the seeing of the section by management workers and visitors. It also has great advantage for shift workers to climb and to adjust if problems or disturbances have occurred on intake and dispatch processing system. The other advantage of the lift is for the electrical and mechanical technicians. By the time electrical and mechanical materials become defective, they can transport other spare part by using the lift for replacement.

Using FPGA device instead of PLC/microcontroller has great significant in the response speed of a lift call. This is due to FPGA device do not enter into cycling reading of commands.

1.4 Objectives of the Thesis

1.4.1 General objective

This thesis is intended to study and design speed controller based on FPGA HDL programming for a permanent magnet dc motor (PMDM) drive. It is expected that this control scheme can track the reference speed to result desired position well under parameter uncertainties and load torque disturbance.

1.4.2 Specific objectives

Toward achieving the general objective mentioned**,** the following specific objectives have been accomplished in this thesis:

- Enabling to draw state diagrams using MATLAB and Xilinx software tools.
- Studying permanent magnet dc motor characteristic integration for the lift design.
- Selecting appropriate motor size for the needed torque output.
- Converting high load torque to low torque which the motor torque can compensate using gears.
- Simulating speed and position tracking of lift using pi control of speed and electric current of permanent magnet dc motor on Matlab Simulink software.
- Generating VHDL code from the Simulink control model which is compatible with an FPGA device.
- Designing stateCAD diagrams for HDL code generation for five floor level.

1.5 Scope and limitation of the thesis

The design of lift using FPGA devices (e.g. Spartan 3E, zynq, …) as a means of controller by replacing microcontroller is a deal of modern technology which is covered in this paper. Geared dc motor is used as power source for lift car movement.

Some of the scope and limitation of this thesis are listed below.

* Now a days, it is a common practice to control the lift status remotely. The more advanced lift or escalator controllers have been made recently which could be controlled from different location. But, this thesis could not include the monitoring of it from remote area.
* This thesis do not also include the civil construction of the car material joints, frames and truss influence on the design, except the load.
* The Hall Effect position/speed sensor plays a vital role in the design of lift. Mal-functioning of the sensor results no movement of the lift car if it is at rest on one of the floor. If the sensor become mal-functioning instantly on lift car movement, there will be occurrence of falling danger due to gravitational acceleration. But, the governor interrupt the falling of the lift since

its objective is to measure the speed of the lift car. The Hall Effect sensor problem/failure and its frequency system resonances has not been included in this paper.

* In modern time, lift without governor is not imaginable. Governor increases the safety of passenger by controlling the designed speed. If rope hanging the car wratched/cut, the governor removes direct falling to ground. The study of lift has not included governor regulation.
* Most lifts are installed inside closed rooms where there is so a little wind disturbance inside it. So, the wind effect on velocity of the car has not been considered.
* In high building installation of lifts, rope elongation over duration of time is common. Re labeling and diagnosing is usually done by lift installers. The impact of rope elongation has not been included in this thesis.

1.6 About Assela Malt Factory

Assela malt factory is located 167 km south east of Addis Ababa city and 7km north of Assela town at an altitude of 2430m in Arsi zone of Oromia region. Its main input is barley, supplied from its surrounding farmers. Its immediate output is malt, a raw material for production of Brewery Company. The factory has an annual production capacity of 36,000 tons of barley. There are _ section within the factory, namely machine tower, germination room, power house, kilning, quality and standardization.

The machine tower is the tallest room that encompasses three floors above the ground. Most functional unit of the factory like that of sifting machine, temporary storing silos, intake and dispatching motors are located inside this room. That is why the machine tower is selected for lift design in this thesis. The other sections do not contain so much high tall floors and functional unit.

1.7 Thesis Outline

The thesis is organized into **five** chapters including this introduction. The rest chapter of the thesis is organized as follows:

Chapter II illustrate literature reviews and general description of essential parts of lifts. Classification of lifts and the idea of field programmable gate array (FPGA), which mainly focuses on hardware description programming (HDL) have been included under this chapter. **Chapter III** of this paper, tried to show the mathematical modeling of a lift system powered by the permanent magnet DC motor. There, it present the load behavior when the lift car is in ascending/upward and descending/downward direction movement differences. The equations are derived from two

different journals mentioned there by using some adjustment. There has been some slight difference between the referenced journal`s equations.

Chapter IV on the other hand, describes the PI speed and current simulation result for trapezoidal reference speed in continuous and in discrete time forms. The discrete time integral part of the control will be rearranged into other form for the purpose of having single precision floating point, which is basic for HDL code generation. The last chapter of this thesis has recommendations and conclusions for the whole document mentioned in this thesis. It was covered and totally included in **Chapter V**.

CHAPTER TWO

LITERATURE REVIEW

1.1 Historical background of lifts

Vertical lifts may have been used to build the pyramids in Egypt. However, the first recorded use came in the third century B.C.,i.e,236 B.C. Archimedes, the Greek mathematician, physicist and astronomer, is typically credited with inventing the first known elevator. His device was operated by ropes and pulleys.

The ropes were coiled around a winding drum by a capstan and levers. These early lifts, or hoists, powered by people, animals or water, were primarily used to lift heavy loads, such as water or building materials.

By A.D. 80, gladiators and wild animals rode crude elevators up to the arena level of the Roman Coliseum. Medieval records contain numerous drawings of hoists lifting men and supplies to isolated locations. Among the most famous is the hoist at the monastery of St. Barlaam in Greece. The monastery stood on a pinnacle approximately 61 meters (200 ft) above the ground. Its hoist, which employed a basket or cargo net, was the only means up or down. At an abbey on the French seacoast, a hoist was installed in 1203 that used a large tread wheel. A donkey supplied the lifting power. The load was raised by a rope wound on a large drum.

By the 18th century, machine power was being applied to the development of the lift. In 1743, a counterweighted personal lift was commissioned by Louis XV in France for his personal chambers in Versailles. By 1833, a system using reciprocating rods raised and lowered miners in Germany's Harz Mountains. A belt-driven lift called the "teagle" was installed in an English factory in 1835. The first hydraulic industrial lift powered by water pressure appeared in 1846. As machinery and engineering improved, other powered lifting devices quickly followed.

Despite these advances, one problem continued to trouble the lift/elevator as it had since ancient times. There was no effective way to prevent the hoist from plummeting to earth if the lifting cable

parted. This ever-present danger made lifts a risky proposition. In 1854, Elisha Otis, working as a master mechanic at the Bedstead Manufacturing Company in Yonkers, New York, was given the assignment to design a freight lift to haul the company's products. Otis was then successful in discovering the solution of the problem which deprive plummeting of hoist when the rope is parted. Following his success and opening of his own company in 1853, construction companies started to build tall buildings with more than 6 stories.

On March 23, 1857, the world's first passenger safety lift went into service in a store at Broadway and Broome Street in New York City. The elevator was powered by steam through a series of shafts and belts.

As the safety and efficiency of the early lift s continued to improve, space in buildings' upper floors soon became more desirable, reversing a long-standing trend in commercial and residential leasing. In 1872, there were no less than 2,000 Otis elevators in use.

The company continued to make technological advancements. In 1878, Otis introduced a hydraulic lift that increased speeds to 244 meters (800 ft) per minute. That same year Otis installed its first hydraulic passenger lift at 155 Broadway in New York City. The company also introduced a governor-operated safety device that would bring the car to a gradual stop in an emergency.

In 1889, Otis innovation produced another first with the direct-connected electric lift machine. This worm-gear electric unit was primarily used for carrying freight. As better gearing arrangements were developed, the speed of the geared electric lift increased from 30 to 120 meters (100 to 400 ft) per minute. This brought the electric lift into passenger service in medium height buildings. Although it offered the advantage of a more compact installation, it was not yet fast enough to compete with the steam-powered hydraulic systems in the taller buildings. But major breakthroughs were not far off [2].

In 1903, Otis introduced the design that would become the standard in the lift industry. The gearless traction electric lift could be employed in buildings of any height and operated at much higher speeds than steam-powered elevators.

Different works have been done on the improvements of lift designs. Most of these designs were conducted based on the microcontroller programing concept of if-else statement. The engineers studying on these improvements were successful on clear design of group of lifts, which have great advantage in choosing the closest lift among other lifts to give response to the requests within a short period. Among the studies conducted on lift designs, [3] has presented a three floor lift system using LabVIEW software, [4] explains autoFocus state diagram for the design of 4 floor lift system.

Now a days microcontroller based lift design is on the process to be replaced by hardware description language. The reason for this is that software programing spend more time in series cycling reading of codes to execute the input command. It has no the ability to process parallel execution capability. However, hardware description language can solve the problems mentioned above. Hardware description language (HDL), which is written by hand or generated from other software, has the aim to bring the hardware components arranged as programs form on FPGA devices. Digital systems like that of FPGA (field programmable gate array) are one part that will be included under embedded system which use hardware description language (HDL).

Different several HDL based elevators have been studied. For example [5] shows state CAD graphical hardware description language programming for three floor using Xilinx software. The study done by [6] has revealed that Verilog code based simulation of lift system with the possible occurrence of errors and the need of corrections.

Recently, Remote lift monitoring is another big issue on lift design for security and safety purpose. The REM® system identifies many problems before they occur by detecting failing components and “intermittent anomalies” that might have gone undetected until they caused a service disruption. Intermittent problems can be addressed before they cause a loss of service. If the REM system detects an urgent issue, the system alerts the appropriate dispatching center and mechanics are sent to repair it and restore service. REM service has been continuously advancing in performance and capabilities since its introduction in the mid-1980s. The latest developments allow REM data to be transmitted directly over the Internet.

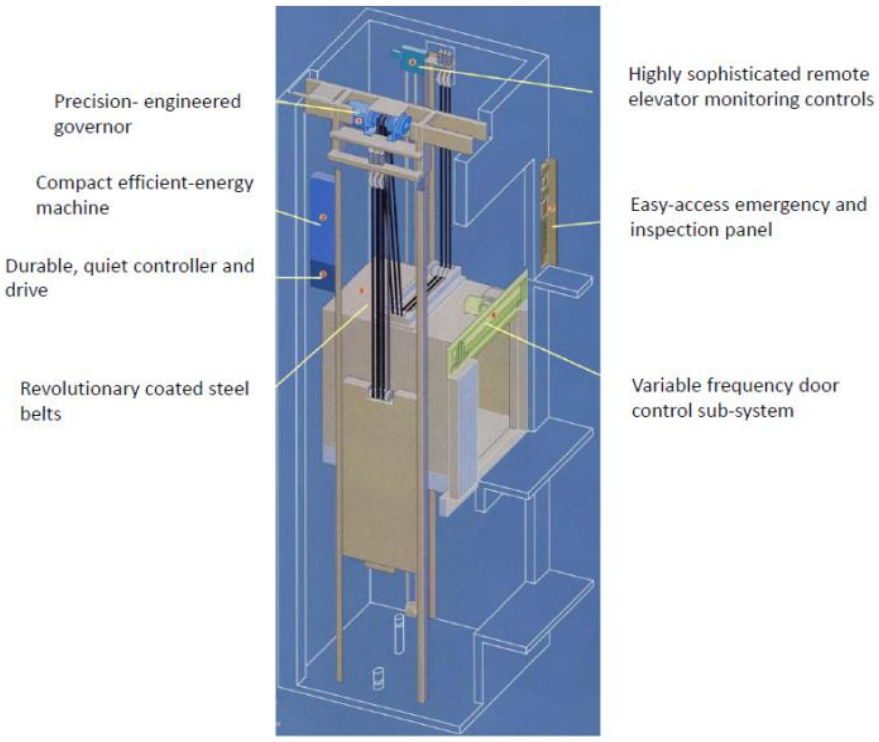


Fig 2 1 Diagram of OTIS GeN2 Lift [7]

Different researches are on process to further improve lift systems till now. All lift systems done till now uses **ropes** for lifting and lowering the car. As previously mentioned Otis was the first person to solve the sudden fall of car if the rope is parted. But, recent researchers are focusing to replace ropes with magnetic material for pulling and lowering the car for too high positions. This will be done with help of magnetic field attracting magnetic car at different position.

2.2 Lift Guide rail, Roller guide and Hoist way

Lift is a permanent lifting equipment serving two or more landing levels, including a car for transportation of passengers and/or goods, running at least partially between rigid guide rails, vertically. All lifts have **guide rails**, in which car or counterweight slides on, for the purpose of directing the lift car and counterweight by help of **roller guides**. The roller guide is used to slide the car and the counterweight upward or downward over the guide rails/vertical bus-bar. In short, the guide rail is used as road while roller guide is used as wheel of car. Since it encircles guide rail/bus-bar, the roller guide never leave the guide rail. Without the roller guide, the lift and the counterweight will suffer from tectonic waves within the hoist ways, disturbing passengers and

goods enclosed in it. It removes a wave like movement of here and there within the hoisting room. The **hoist way** on the other hand, refers to the whole room under the lifting motor including the four vertically rigid bus-bar/guide rail, counterweight, trailing cables and steel ropes.

Almost all lifts, now a days, have these three components i.e. ***hoist way***, ***guide rails*** and ***roller guide***.

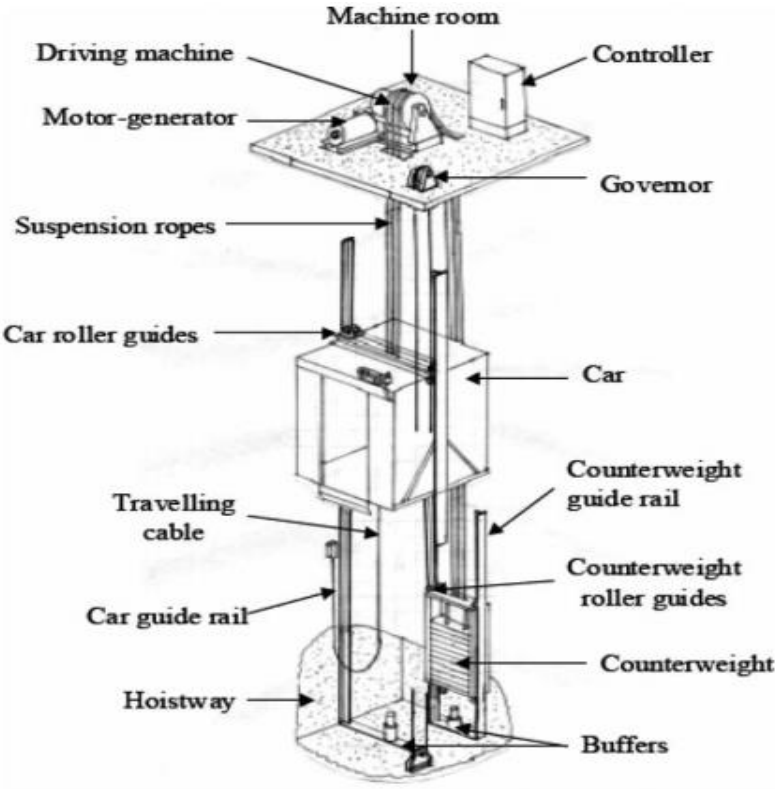


Fig 2 2 Guide rail (roller guide) and Hoist way [8]

1.2 Classification of lifts

1. Based on **power source of transportation,** lift are of two types itself. These are **hydraulic type**, which uses fluid pumped into or out of the cylinder to raise or lower, and **electric motor type** which use electric motors, whether DC or AC as power source with a support of suspension ropes, to raise or lower.

   Hydraulic lift- Hydraulic elevators pump hydraulic fluid to a cylinder/plunger assembly that results in vertically moving the piston that raises the elevator/lift cab/car up and lowers it back down. The conventional cylinder/plunger assembly resides in a well hole directly beneath the

elevator. In older time, these assemblies were installed with little or no protection from the subsurface elements. Without the ability to inspect the assembly beneath the surface, the possible damage to the cylinder that would ultimately result in a hydraulic fluid leak goes unnoticed. When this condition occurs, it typically results in the contamination of the soil and possibly groundwater. Hydraulic elevators need to be provided within a controlled environment requiring additional heating and in some instances air conditioning. Remote piping running underground is another potential problem for temperature control as well as potential leakage. Hydraulic lift divided in to the following groups:

- **Conventional hydraulic lift**.-they use an underground cylinder, are quite common for low level buildings with 2–5 floors (sometimes but seldom up to 6–8 floors), and have speeds of up to 200 feet/minute (1 meter/second).
- **Holeless hydraulic lift**-were developed in the 1970s, and use a pair of above ground cylinders, which makes it practical for environmentally or cost sensitive buildings with 2, 3, or 4 floors.
- **Roped hydraulic lift**-use both above ground cylinders and rope system, allowing the lift to travel further than the piston has to move.

The low mechanical complexity of hydraulic elevators in comparison to suspension lifts makes them ideal for low rise, low traffic installations. They are less energy efficient as the pump works against gravity to push the car and its passengers upwards, this energy is lost when the car descends on its own weight. The high current draw of the pump when starting up also places higher demands on a building's electrical system. There are also environmental concerns should either the lifting cylinder leak fluid into the ground. The modern generation of low cost, machine room-less traction/suspension lifts made possible by advances in miniaturization or reduce to the traction motor and control systems challenges [9].

Suspension systems driven by **electric motors** are the ideal choice for applications having minimal space and great heights. Because these systems do not require infrastructure to build beneath the floor of the lift and are limited in height only by the ability of the electric drive to overcome static forces like gravity and dynamic forces such as friction and air drag. For these reasons, this thesis will focus on the suspension system driven by an electric motor.

2. Based on **building type**, lifts can be classified as
    a. Hospital lift
    b. Residential /Domestic lift.
    c. Agricultural lift.
    d. Industrial lift.
    e. Commercial lift.
    f. Parking buildings lift.
3. Based on their **purpose of transportation**. These are

Passenger Lift

- ✓ Designed to move the people between a building's floors.
- ✓ Passenger elevators capacity is related to the available floor space.
- ✓ Usually, for eight floors or less buildings, hydraulic or electric power source are used with speeds up to 1m/s (hydraulic) and up to 2.5m/s (electric). But for buildings up to ten floors, electric and gearless Lift are used with speeds up to 2.5m/s, and for ten floors above, speeds begin at 2.5m/s up to 10m/s.

Freight Lift

- ✓ Designed to carry goods rather than passengers.
- ✓ Generally, it requires to display a written notice in the car how passengers uses the lift.
- ✓ It is typically larger and capable of carrying heavier loads than a passenger elevator, generally from 2,300 to 4,500 kg.

Vehicle Lift

- ✓ It is installed where ramps are considered space-in conservative for smaller buildings(usually in apartment building where frequent access is not an issue)
- ✓ The car platforms are raised and lowered hydraulically and are connected to chain steel gears. The platform also can rotate about its vertical axis (up to 180 degrees) to ease driver access and accommodate building plans.

Aircraft

- ✓ It carry aircraft between the flight deck and the hangar deck for operations or repairs.
- ✓ It is designed for much greater capacity than any other lift ever build, up to 200,000 pounds of aircraft and equipment.

- ✓ Smaller elevators lift munitions to the flight deck from magazines deep inside the ship.

Dumbwaiter

- ✓ Often used for the moving of small items such as dishes in a 2-story kitchen or books in a multistory rack assembly.
- ✓ Modern dumbwaiters are generally driven by a small electric motor with a counterweight and their capacity is limited to about 340 kg.
- ✓ Dumbwaiters are used extensively in the restaurant business (hence the name) and may also be used as book lifts in libraries, or to transport mail or similar items in an office tower.
- ✓ Dumbwaiters, especially older ones, may also be hand operated using a roped pulley

Paternoster

- ✓ It is a constantly moving chain boxes.
- ✓ A similar concept moves only a small platform, which the rider mounts while using a handhold and was once seen in multi-story industrial plants.

In suspension type lifts, the motors will be positioned at the top or at one side of hoist way (look the figure given below). Motor driven lifts are functional with the help of ropes to handle car load. The most common and efficient motor arrangement is the top positioned motor. It has an advantage of regenerative power utilization based on the counterweight and car load directional movement. It also occupies less space than side positioned motors. But, the load on the four bus-bar is the summation of counterweight, rope and car load. Side positioned motor arrangement, the old fashioned and now replaced by the suspension type arrangement, occupies the most necessary extra places. It also do not enable us to get regenerative power. It rather consumes electric power due to absence of counterweight.

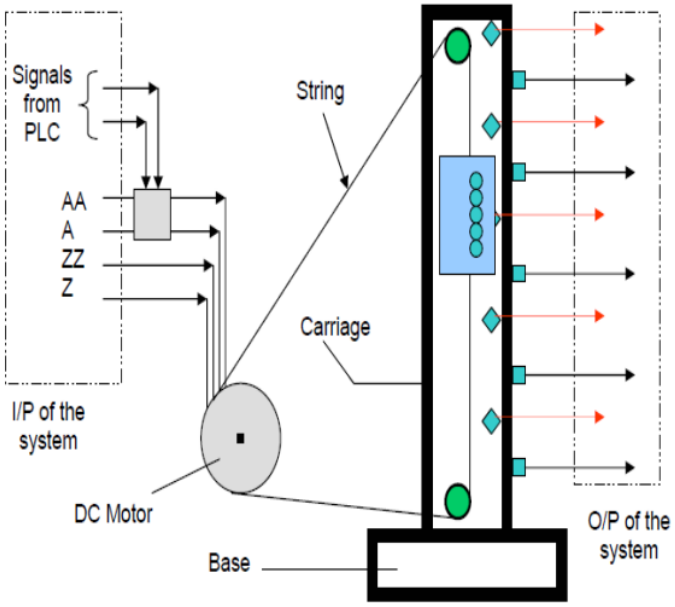


Fig 2 3 To one side motor positioned lift system [9]

2.3 Construction of suspension Elevator

An electrical suspension type passenger lift design has four essential components. These are

i. The lift/elevator cab/car
ii. The drive system/machine.
iii. The counterweight.
iv. Control system

All these components must be properly selected and designed based on the lift standard of the country by lift designer/engineer.

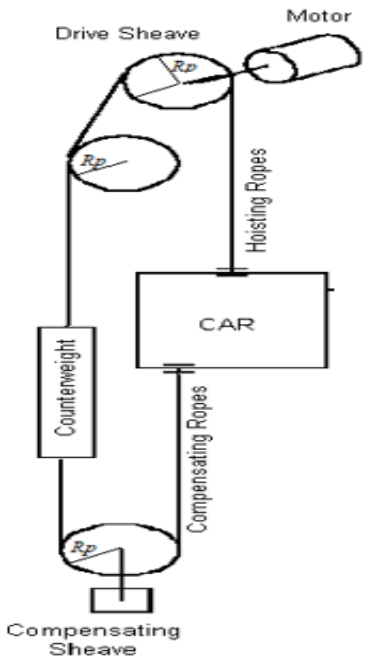


Fig 2 4 The three components of lift (drive, counterweight &car)

2.3.1 Car/cabin

The car is the main part of lifting system in which passengers or goods are transported up/down wards. This action of movement is accomplished by the support of steel ropes and electric drive placed above the hoist room. The weight of the car is balanced by counterweights to enable minimum energy usage of drive machine. The lift must fit within the given space requirements of the building. It must considers the load to be transported on the basis of daily traffic. Two or more lifts will be placed if there is high daily traffic. But, the initial economy of building is considered by institution managers` with its income generation.

The drive selection is dependent on the load to be transported within the car. Possible restrictions on the weight carried within the lift may be determined from the size of the motor and the other components within the lift system.

2.3.1.1 Lift car/cabin insider load

The main objective of the lift designer is the transportable load inside the car. There will not be lift designing without the consideration of load. In other words, the maximum size/mass of the load has a determinant factor in the choice of lift car material selection, motor size and electric power usage, number and strength of steel rope, drive and deflector sheave groove and in some extent the counterweight mass.

Lift/elevator companies have different standards in which the load mass has a relation with empty car mass. But, in text books there will be deviation with standards. This is due to the text book/journal objectives of familiarizing students with concept of lift designing.

Different lift companies have different load-lift car load relations. For example, in [10] the car mass is 100kg while load mass is 400 kg. Let's see the standards by which one lift company use and see the differences.

Table 2 1 results from the British Colombia study [11]

Lift model No	Rated load(kg)	Car mass(kg)	Ratio (R_m)	Speed (m/s)
A	907	1813	0.50	1.25
B	680	1613	0.42	1.25
C	816	1804	0.45	1
D	1588	3270	0.49	1.5
E	1011	2023	0.50	1
F	1590	2900	0.55	1.5

From the above table, the ratio R_m indicates the rated load to empty car mass.

2.3.1.2 Standard car size:

In designing of lift car, weight and car area relational standards has been used by lift producing companies. This is used to simplify the work and to focus on other part of designing. To prevent overloading of the car by persons, the available area of the car shall be limited and related to the nominal/rated load of the elevator. The following table shows the standard car sizes in relation with the lift nominal loads.

Table 2 2 lift car specification [12]

Load(Q)	Persons	CW(mm)	CD(mm)
320	4	1000	900
		900	1000
400	5	1100	1000
		1000	1100
480	6	1200	1100
		1100	1200
630	8	1400	1100

		1100	1400
		1200	1300

The following definitions for the car dimensions are very important:

Car Width (CW): The horizontal dimensions between the inner surfaces of the car walls measured parallel to the front entrance and at 1m above the car floor.

Car Height (CH): The inside vertical distance between the entrance threshold and the constructional roof of the car. Light fittings and false ceilings are accommodated within this dimension.

Car Depth (CD): The horizontal dimensions between the inner surfaces of the car walls measured at right angles to the car width and at 1m above the car floor.

2.3.2 Counterweight

When designing lifts, it is essential and economical to use counterweight suspended to the opposite side of the car using steel rope. The maximum stress load on the motor will be the load difference between counterweight and car. Hence, it helps to reduce the electric power consumption by the motor. Counterweight is usually made up of heavy, minimum place occupying material like ceramic rock with special protecting and holding of metals. The counterweight mass has a smooth relation with an empty car and rated load mass. This relation differs from company to company or from text book/journal to text book/journal. But, their usage is close similar.

i. The first relation of the counterweight mass to car with rated load mass is that the counterweight mass is equal the empty car mass. That is, $M_{ec} = M_{CW}$.

ii. The second relation is that the counterweight mass is equal to empty car mass plus half of the rated load mass. Mathematically, $M_{CW} = M_{ec} + 0.5M_L$.

Both these structural arrangement have their advantages and disadvantages. The first structural arrangement will not be economical for large rated load mass. This is due to the rated load influence on the motor to be balanced or reduced by the counterweight side mass will be minimum. In other words the load difference between car and counterweight become high which leads to higher motor size and power source usage. The larger the mass of the load the more electric

consumption the motor needs to raise up. This problem is corrected if the second structural arrangement for the large load mass has been used.

The advantage of the first structural arrangement is that the mass exerted by the empty car over the drive sheave through steel rope has no influence on the driving motor. Because they are in equilibrium. The opposite is true for the second structural arrangement.

2.3.3 Machine/drive system

Driving machine, this is the power unit of the elevator, and usually located at the top of the hoisting room. The Driving machine is used to refer the collection of components that raise or lower the car. These include the drive motor, brake, speed reduction unit (gearbox), sheaves and encoders.

Generally, there are ***two standard types*** of driving machines provided for elevators. These are;

2.3.3.1 Geared Machine: - A geared driving machine is one that utilizes a geared-reduction unit between the motor and the drive sheave. The main advantage of this design is that a less powerful motor can be used to drive it. This is due to the capability of the gear to reduce the load torque to less desired value. A geared system, usually designed to run at 350 feet per minute(1.78m/s) or less (though they can go faster), sacrifices speed to its gearless counterpart. Geared systems are often used in slower-moving passenger and freight elevators with less floor numbers, specifically, less than 10 floors.

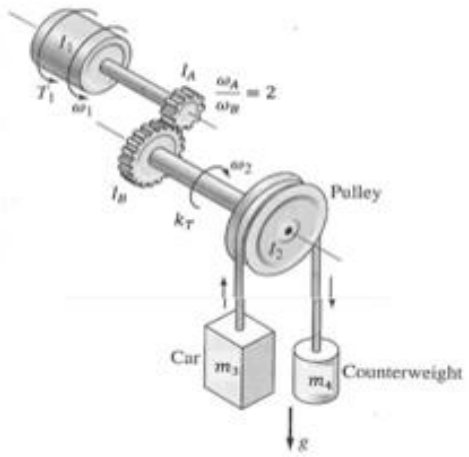


Fig 2 5 geared motor [13]

2.3.3.2 Gearless Machine:-A gearless driving machine is a direct-drive system in which there is no reduction gear between the motor and the drive (or hoisting) sheave. That is, the drive sheave is connected directly to the motor and brake. Gearless designs are used in the world's tallest structures. They are efficient and used for driving speeds greater than 500 feet per minute (2.5 m/s) with greater number of floors, i.e. above 10 floors.

Fig 2 6 gearless dc motor [14]

An Electrical Motor (whether it is geared or gearless) is used to raise and lower the lift cab and counter weight. The direction of motor rotation and speed (revolutions per minute) are directed and supervised by devices located within the lift controller. The motor is powered either from AC or dc power source.

In modern lift technology the most available and acceptable motor types as traction motors are variable voltage and variable frequency induction motor, SCR-DC motor, PMSM and DC Pulse Width Modulation (PWM) drives. They are used in gearless or geared application.

2.3.3.3 Permanent magnet dc motor preference

Suspension type elevators/lifts uses different types of motors for raising and lowering using ropes. The motors can be ac motors or dc motors based on the power source. Now a days, an induction motor is a preferred motor for the usage of lift design. This is due to its nature of rug, low cost etc.… but, it couldn't be directly fed up by ac source, rather it needs ac-dc and dc-ac convertors. These convertors are used for conducive functioning of induction motors but has great cost expenditure. Permanent magnet dc motor type is usually selected as driving motor for its low cost and for its listed behaviors and advantages.

I. ***Wave interference***: A problem with RFI increases if grounding is inadequate. Modern AC drives use power devices known as Insulated Gate Bipolar Transistors, or IGBTs, which has the ability of minimizing annoying audible noise, by using switching frequencies beyond the human hearing range. Nevertheless, IGBTs present a high RFI.

II. **Need of expertise**: AC applications require specialized expertise from both motor and control suppliers, along with good cooperation and coordination between the two, but not the dc motor.

III. **Starting delay**: AC drives have an inherent delay in starting to build magnetic flux through the motor. There is no such delay in the case of dc motor.

IV. **Economical**: Ac drives uses both convertors and invertors. But, dc motors uses only convertor. As a result dc motors are economically cheap.

V. **Simplicity**: the other benefit of using PMDC motor is its simplicity of modeling in Simulink environment. After developing control strategies by using PMDC one can design for other motor types like that of VV/VF induction motor and PMSM.

Permanent-Magnet Motors have performance advantages over DC excited-synchronous motors and, are becoming more common in fractional horsepower applications because they are smaller, lighter, more efficient and reliable. Large industrial motors originally used wound field or rotor magnets. Permanent-magnets have traditionally been used only on smaller motors because of the difficulty in finding a material capable of retaining a high-strength field. Recent improvements in material technologies have made it possible to create high-intensity permanent-magnets, allowing the development of compact, high-power motors without the extra real estate of field coils and excitation means.

2.3.3.4 H-bridge DC Motor

In lift design bidirectional control of a DC motor using an H-bridge circuit is mandatory. The H-bridge named for its schematic appearance, is able to move current in either direction through the motor winding. H-bridge topology was chosen for the movement of the lift car upward (counterweight downward) and lift car downward (counterweight upward) direction based on the control command signal. It is called an "H-bridge because it looks like an H letter. An H-bridge is an electronic circuit which enables electric motors to be run forward or backward action. It is available as integrated circuits or can be built from separate components for specific design. In this system, the DC motor with H-bridge driver circuit is used to advance the lift car to the next position. The motor can provide for both directions: clockwise (CW) direction and

counterclockwise (CCW) direction. This circuit uses four transistors for forward and reverse directions. Its operation is as follows.

A positive voltage will be applied across the motor and the motor will rotate clockwise direction when the transistor switch 1 and switch 4 are ON and the other transistors are OFF. The voltage across the motor will be negative, allowing counterclockwise operation of the motor when the transistor switch 2 and switch 3 are ON and the other transistors are OFF. In DC motor, clockwise direction for up condition and counter clockwise direction is for Down-ward condition [15]. The source [16] further explains the Simulink model of H-bridge for the dc motor.

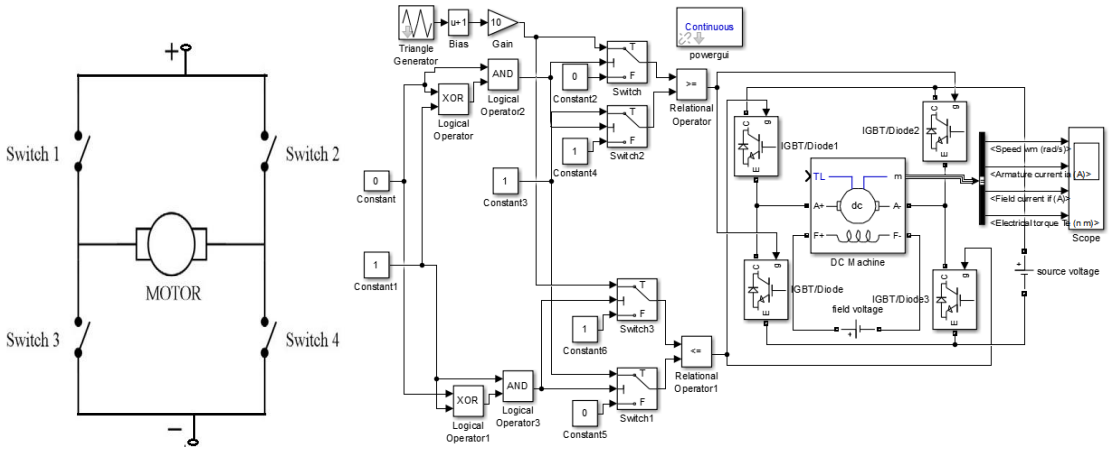


Fig 2 7 H-bridge DC-motor a) model Circuit b) Simulink model [16]

Regenerative power

Motors have the capability of regenerating energy when operating in the overhauling (negative-load) condition (i.e., driven by the load- **Fig 2.8a & d**). Examples of overhauling conditions in traction elevators are "full load down" and "empty car up". This will happen if counterweight mass is greater than cabin/car mass when it is empty. Regenerative braking involves sending the regenerated energy from the lift hoist motor back to the electrical power source from the building.

When the motor is working as motoring and regenerative mode it said that it is functioning in fourth quadrant. It can be explained as [10] based on figures.

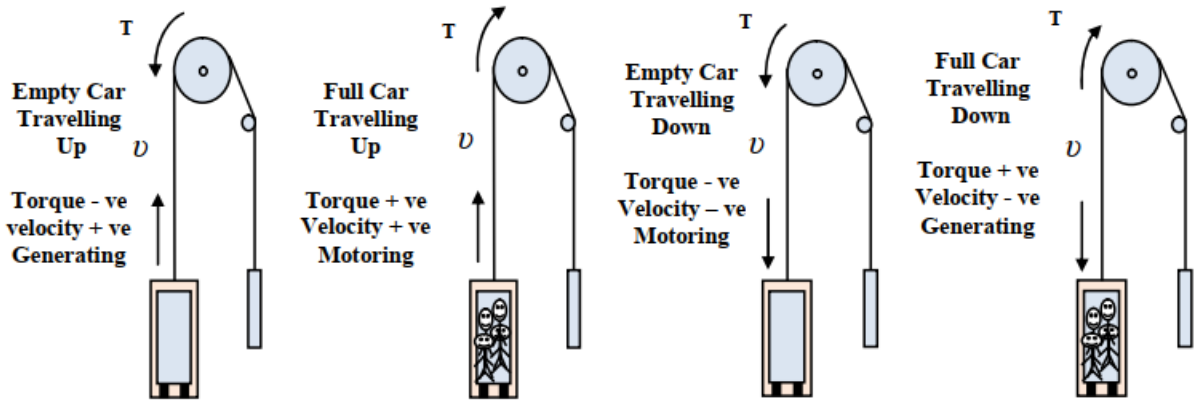


Fig 2 8 fourth quadrant operation of lift [10]

When an empty car is raising up since the mass of the counterweight is greater than the cabin the torque produced is negative. The counterweight mass enables the generation of electricity which will be given back to the electric grid using special technology or may be lost as heat through resistor of the armature. (The generated electricity dissipation via armature resistor has bad influence on the motor. It will lead to happening of high heat within the motor inside part. Therefore, there must be other resistor to consume the generated electricity with special technology). Hence, the motor in ***fig 2.8a*** is acting as generator.

In case of Fig 2 8*b*, when full load lift car is going upward, the motor consume electricity from the grid for raising of the car. Since lift car mass is greater than the counterweight mass, the motor is behaving as motor (motoring mode). The torque produced is positive.

From Fig 2 8*b*, an empty car is traveling to downward direction. Because of mass of counterweight greatness from the lift car, the motor uses electricity from the grid to result the movement. The electromagnetic torque produced from such movement is negative while the mode is motoring.

Fig 2 8 *d* is the counter part of *fig (2.8a)* with slight difference of directional movement and electromagnetic torque direction exchange. Full load lift car mass is greater than the counterweight mass. So, the downward movement of the car can be accomplished with the help of the gravitational attraction force. Since the electromagnetic torque is dependent on mass difference of lift car and counterweight it is positive. i.e.

$$T_m \cong (W_c - W_{cw}) * R_p$$

Where

W_c is car weight,

w_{cw} is counterweight and

R_p is pulley radius.

All the above concept can be summarized as

Table 2 3 Regenerative power

Lift car mass status	Full load lift car		Empty lift car	
Speed, ω	+(up)(b)	-(down)(d)	+(up)(a)	-(down)(c)
Torque,T_m	+	+	-	-
Electric power flow	+	-	-	+

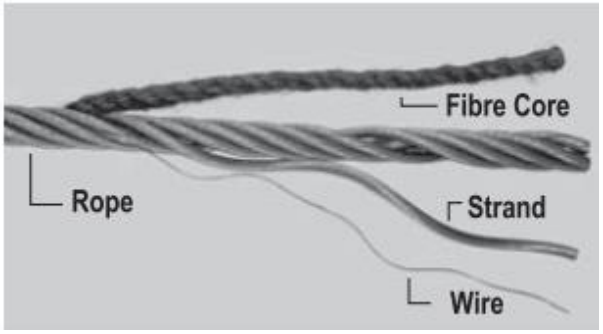
Note that the above data is applicable if counterweight mass is greater than an empty lift car mass. The negative (-) sign in "electric power flow" indicates that the motor is in generating mode.

2.3.4 Steel Rope

Wire rope is an intricate device, a composite structure, made up of a number of individual units, called wires, which are designed and produced in such a manner that they always exist and work in some precise relationship with one another. This precise relationship of individual wires which possess certain defined physical and mechanical characteristics, ensures that the composite wire rope has strength, flexibility and durability essential for safe hoisting applications.

A wire rope has three main elements: these are

a) Wire
b) Strand and
c) Core.

a) 

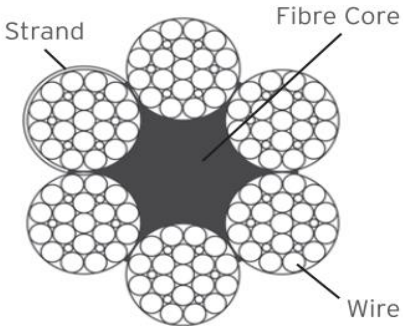
b) 

Fig 2 9 steel rope a) longitudinal view b) cross-sectional view [17]

The basic component of a steel rope is the wire which form strand when defined number of wires are spun helically around a central wire as explained in [17]. A number of such strands are then helically spun together over a core to form a steel rope. The way the wires are spun to form the strands and the way these strands are spun around the core greatly contribute to the overall performance characteristics of the steel rope. Generally, the composition breakdown in terms of cross-sectional area is approximately 48-52% steel 46-49% hemp/fiber and 2-3% air.

The fiber core provides increased flexibility to the wire rope and in some cases act as reservoir of lubricant. This flexibility is desirable for the bending of the rope over the drive and deflecting sheaves. A steel rope is constructed by having a number of strands encircling the core in "s" format. For example 8x19S denotes that it is composed of 8 strands and each strand has 19 wires spun together. The one shown on the above ***fig (***Fig 2 9***b)*** can be described as 6x19s.

A lift steel rope surface may have bright finish (without no additional metallic coating) or in galvanized form to resist corrosion. Lift ropes are generally procured in bright finish and seldom as galvanized. Galvanized ropes are produced based on demand by lift installers.

According [18], lift models is dependent on the lifting height and velocity. As real combinations of the two parameters, the three structures are:

- Passenger/freight elevators with small heights and low velocities of lifting, the so-called elevators/lift with small lifting velocities.
- Passenger/freight elevators with great heights and medium velocities of lifting, the so-called elevators with high lifting velocities.

- Passenger elevators and exploitation facilities in mining with great heights and lifting velocities, the so called express elevators.

In the same way as lift models with height and velocities, the selection of rope must account these two variables by additionally including the load mass. The low height, velocity and load may not need more attention on the construction, tensile capability and elasticity of ropes as express lift of high load and height. High number of strands over the core with great tensile ability is chosen for building having very tall height. Ropes with great number of strands increases the surface area contacts when passing over the sheaves.

Rope tensile and stretching/elongation capacity

The tensile grade of a steel rope refers to the tensile designation of its constituent wires. Steel ropes usually has two tensile grade, the outer wire and the inner wire. The outer wire has a lower tensile grade than the inner wire. Example, 1180N/1770N or 1370N/1770 N. Rope tensile capacity is used to select the right rope, which has the capacity to suspend/hang the car and the counterweight with almost negligible bending influence/affection of ropes over sheaves. The length of the rope from the drive sheave to the ground floor at the car side and from the drive sheave to the counterweight side, and the size of the load plays an important role for rope selection. As the rope length increases there will be an increment of **stretching of the rope**. This stretching is common immediately after installation of the lift ropes and gradually fades away with cycle of usage. The lighter the load the higher is the time needed for stabilization. If the extension of the rope stretch is recoverable, meaning the rope length become same when applied load is removed, the stretch is called **elastic stretch** and the stretched length can be calculated as

$$elastic\ stretch(mm) = \frac{WxL}{ExA}$$

$$Where \quad W - \ applied\ load(N)$$

$$L - \ rope\ length(mm)$$

$$E - \ modulus\ of\ Elasticity(N/mm\text{^}2)$$

$$A - Metalic\ area\ of\ rope(mm\text{^}2)$$

But, in the case of non-elastic stretch of the rope, the operator of the lift is subjected to redesign after some duration of usage to bring the car to the correct leveling by removing/cutting the stretched part of the length. This continual repairing is not needed by all operators. So, the more

rigid with slight flexible stretching capacity rope is selected with hindering ability of tripping hazards.

Rope lengthening for tall building lifts is a common problem. It needs relabeling after some duration usage of the lift. Or the ropes will be replaced by new one if the elongation is high or rope appear to wear out. Since lift design for this paper has a height of 0 – 40m (small - medium height), the effect of elongation and dynamic mass of ropes has been omitted.

Lift rope lubrication

Lift ropes are lubricated during manufacture in order to prevent corrosion and abrasion between the wires. However, the quantity of lubricant applied should only be enough to ensure that lift operate with sufficient traction and without slippage. As lubricants also tend to bind dust and abraded particles, this initial lubrication is hardly ever sufficient to be effective over the entire service life of the rope. So, it is advisable to occasionally re-lubricate lift ropes [19].

2.3.5 Sheave

The powered pulley connected to either the lift drive motor's output shaft (gearless) or to the output side of the mechanical speed reduction unit (geared) is called the **driving sheave**. The circumference of the sheave has a series of **"U" or "V" shaped grooves** cut into it (as shown in *Fig 2.10 A*), in which the hoist ropes sit or pass over. The friction loads created as the suspension ropes pass over the grooved surface of the sheave causes motion to be transmitted from the drive motor to the lift car or counterweight.

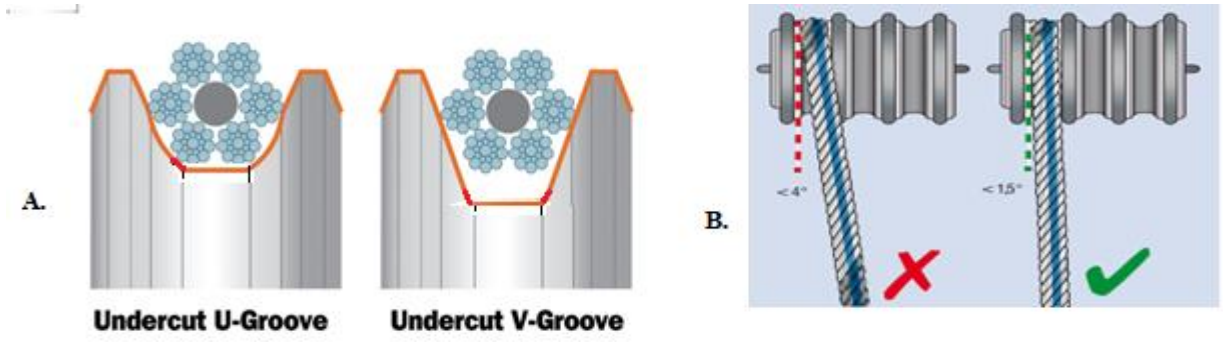


Fig 2 10 A. "U" or "V" shape groove B. suspension format on "U" grooved sheave [19]

As one can understand from the above figure "V-groove" has a high sticky resistive power on the sheave. As the motor rotate the steel rope will enter into the groove in sticky way in one edge while it separates in another edge. This high resistance of the groove to the rope may result short period service of ropes. Continual usage of v-groove will bring abrasion if the suspended load is high. Therefore, it is advisable to use the "U-groove" for higher loads and also to lengthen the life service of the ropes by reducing the stickiness of ropes into grooves with high resistance. Abrasion may occur if improper suspension method of installation as indicated on *fig 2.10 B* with sign "X" is used. So, the right proper method of installation (*fig 2.10 B* with sign "✔") must be followed.

2.3.5.1 Friction coefficient of grooves

As much as possible, the friction coefficient of the groove are minimum in order to hinder the wear and tear of the steel rope without reaching their expected age. The friction coefficient is usually less due to the lubrication of ropes after some duration of services.

2.3.5.2 Deflector Sheave:-Pulley used to offset or direct the vertical drop or location of the steel hoist ropes running between the lift car and its counterweight. Where the horizontal distance between the hitch point for the car and the counterweight is larger than the diameter of the drive sheave, one or more deflector sheaves are used to guide the hoist ropes.

These devices are grooved sheaves that lead lift suspension ropes off the drive sheave down to the car top and counterweight. The number and size of deflector sheaves will be a function of the elevator's size, machine placement and roping arrangement.

Many installations having carrying capacities of 1,136 kg or less, are provided with drive sheaves of sufficient diameter that do not require the use of deflector sheaves in a typical overhead arrangement. Look the figure below.

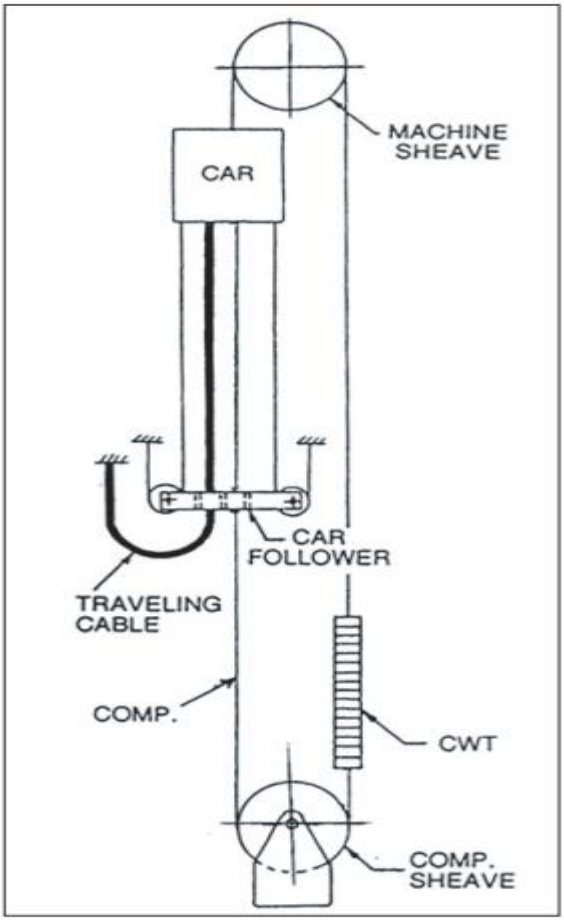


Fig 2 11 lift without deflector sheave [20]

2.2.6 Traveling/trailing cables: a cable made up of electric conductors, which provides electrical connection between a lift car and motor controller means. In most cases, their weight is minimum hence not considered as that of lift ropes.

2.2.7 Lift Control System:

-is the system responsible for coordinating all aspects of lift service such as travel, speed, and accelerating, decelerating, door opening speed and delay, leveling and hall lantern signals. It accepts inputs (e.g. button signals) and produces outputs (elevator cars moving, doors opening, etc.).

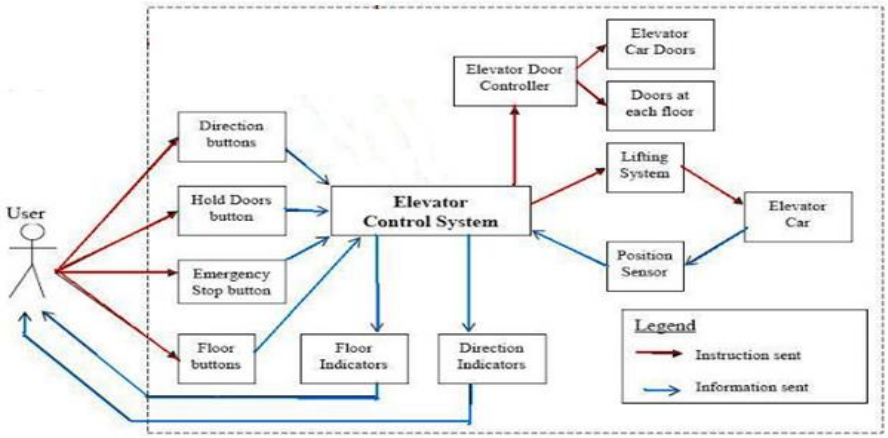


Fig 2 12 user interaction with the lift control [21]

Aims of the control system

The main aims of the lift control system are:

- To bring the lift car to the correct floor.
- To minimize travel time.
- To maximize passenger comfort by providing a smooth ride.
- To accelerate, decelerate and travel within safe speed limits.

1) Controller

The controller uses Inputs to get outputs. The inputs which are feed to the controllers are

A- Sensors. B- Buttons (push buttons)

Sensor: Sensors have important application in the controlling of many hardware components. There are four types of sensors on the lift design. These are ***Magnetic and/or photo electric sensor*** (used for sensing position of the car), ***infrared sensor*** (used to sense peoples entering to or leaving the car), **weight sensor** (to sense overload condition) and ***velocity transducer*** (to detect the speed of the car). In this technique the speed and direction of the DC motor are controlled by FPGA hardware by inputting PWM signals to its H-bridge drive circuits.

2.3 Performance Measurement of lift

The lift industry has established standards by which the desired performance of a given lift installation can be measured. These measures vary depending on the type of lift configuration that is being installed. The performance measurements include

1. Acceleration (m/s^2)
2. Floor to floor time
3. Speed(m/s)
4. Number of Floors
5. Leveling error

The other performance measurements are:-

1 **Ride quality**:- With respect to ride quality, a "very smooth" ride is one in which there are no discernible bumps or vibration during the ride. Although this is a somewhat subjective definition, recent developments in the instrumentation used in the lift industry have led to the application of 3-dimensional accelerometers that provide a more objective measure of ride quality.

2 ***Fliqht Time***: defined as the time [sec] measured between the start of the doors closing at any given floor and the time at which the doors reach the 3/4 open position at the next adjacent landing. (The car is stopped and level at the floor when the doors are 3/4 open and the passengers can begin to move in and out of the car.)

3 ***Round Trip Time***: the average time required to deliver a normal passenger to the top floor and return to the first floor ready for new passengers. Round trip time assumes some number of stops being made in the run (for example 10 stops for a 15 floor building).

4 ***The Governor*** The governor is a device actuated by the centrifugal force of whirling weights opposed by gravity. It is used in lifts as a standard safety measure to set an emergency mechanical brake that brings the car to a stop when the car exceeds a safe speed. A set of redundant safety features (both mechanical and electrical) are used in the lift industry to ensure that cars do not run away. These safety measures depend on local and national codes and vary from country to country and even state to state.

2.4 Field-Programmable Gate Arrays (FPGA)

"Circuit components that can be reconfigured by software commands may be able to reduce the rapidly-growing cost of electronics in future cars." Prof Ju¨rgen Becker, senior member IEEE.

2.4.1 Introduction

Electronics revolutionized the 20th century and continues to make an impact in the 21st century. The birth and subsequent growth of the computer industry, the creation of mobile telephony and the general digitization of television and radio services has largely been responsible for the recent growth. In the 1970s and 1980s, aggregate electronic systems like microprocessors and memories with large number of I/O units on printed circuit has been developed. As levels of integration grew, manufacturing working PCBs became more complex, largely due to increased component complexity in terms of the increase in the number of transistors and I/O pins but also the development of multi-layer boards with up to as many as 20 separate layers. Thus, the probability of incorrectly connecting components also grew, particularly as the possibility of successfully designing and testing a working system before production was coming under increasingly limited time pressure.

The problem was becoming more intense due to the difficulty that system descriptions were evolving as boards were being developed. Pressure to develop systems to meet evolving standards, or that could change after the board construction due to system alterations or changes in the design specification, meant that the concept of having a 'fully specified' design in terms of physical system construction and development on processor software code, was becoming increasingly unlikely. Whilst the use of programmable processors such as microcontrollers and microprocessors gave freedom to the designer to make alterations in order to correct or modify the system after production, this was limited as changes to the interconnections of the components on the PCB, was only limited to I/O connectivity of the processors themselves. Thus, the attraction of using programmability interconnection or 'glue logic' offered considerable potential and so the concept of field-programmable logic (FPL) specifically field-programmable gate array (FPGA) technology, was borne.

2.4.2 What is FPGA?

An FPGA is a semiconductor device on which the function can be defined after manufacturing.

Product features and functions of FPGA can be programmed, adapt to new standards, and reconfigure hardware for specific applications even after the product has been installed in the field — hence the term field programmable. And gate arrays are two-dimensional arrays of logic gates. If you get enough of these things put together, you can make those simple calculations add up to do something meaningful. [22]

In less technical terms, an FPGA allows flexibility in designs and is a way to change how parts of a system work without introducing a large amount of cost and risk of delays into the design schedule. A simple example is a rear-view camera designed for a car. If the camera system takes 250 milliseconds from the time the image sensor sees the image until the image frame actually appears on the display, and a change in government regulations requires that this delay or latency be no more than 100 milliseconds, you could find ways to adjust the image signal processing pipeline in an FPGA to comply with the new latency requirements. This would be almost difficult to do with a microprocessor-based system. In this example, a company can gain a big advantage using an FPGA because it doesn't have to redesign parts or buy all new processors.

In the early days, FPGA circuits were very large and you couldn't fit many onto a single chip. All a designer could do was build an interface with an FPGA and customers could reprogram that interface to do something different (for example, changing an interface that operates on a keyboard input to one that handles input from a touchscreen device). Soon, however, designers realized they could build entire subsystems out of FPGAs, which meant designers were no longer restricted to only using ASICs (application specific integrated circuit) to implement these subsystems. Because circuit components keep getting smaller and smaller, designers can put many more devices on the same chip — allowing for more sophisticated functionality and faster arithmetic, which in turn leads to faster computation and less power consumption. ASICs typically take months to fabricate and cost hundreds of thousands to millions of dollars to obtain the first device; FPGAs are configured in less than a second (and can often be reconfigured if a mistake is made) and cost anywhere from a few dollars to a few thousand dollars.

Modern FPGAs consist of mixes of configurable static random access memory (SRAM), high-speed input/output pins (I/Os), logic blocks, and routing. More specifically, an FPGA contains programmable logic elements called, naturally, logic elements (LEs), as well as a hierarchy of reconfigurable interconnects that allow the LEs to be physically connected to one another. You

can configure LEs to do complex functions or simply perform basic logic gates, such as AND, NOT and OR. Most FPGAs also contain memory blocks.

Look the below figure of basic FPGA structure, which shows a programmable routing fabric that allows blocks to be programmably interconnected. The array is surrounded by programmable input/output blocks, labeled I/O in the figure, that connect the chip to the outside world. The "programmable" term in FPGA indicates an ability to program a function into the chip after silicon fabrication is complete. This customization is made possible by the programming technology, which is a method that can cause a change in the behavior of the pre-fabricated chip after fabrication, in the "field," where system users create designs.

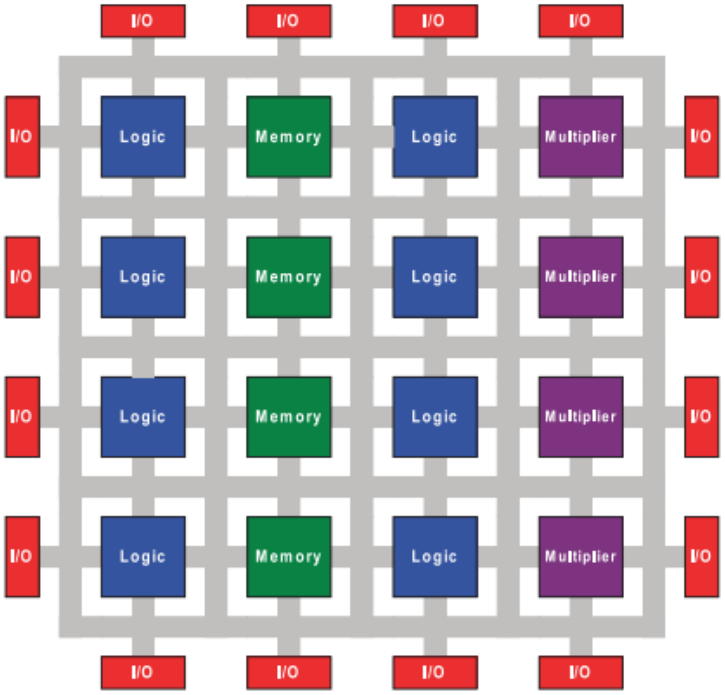


Fig 2 13 Basic FPGA Architecture [22]

2.4.3 Building blocks of FPGA

The building block of any digital circuits [23] are a wire, a logic gate, and a register (see Figure 3.2). A **register** remembers a piece of information until it is told to remember something else. A **logic gate** performs simple logic operations on signals, and a **wire** connects these other pieces.

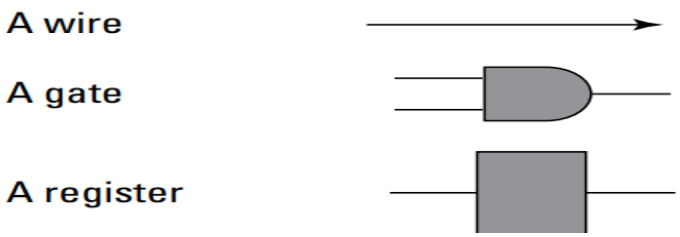


Fig 2 14 the building blocks of a digital system [23]

2.4.3.1 Logic gates

Logic gates perform the core functionality of digital circuits, which means they perform simple logic on *inputs* — electrical pulses that all computer uses to represent "0"s and "1"s. On their own, these simple operations don't do much, but when thousands or even millions of these are arranged together they can do something really powerful. Computer's CPU is made up of billions of these logic gates.

In order to perform as functional circuits, logic gates use a type of arithmetic called **Boolean** algebra. Boolean algebra was first introduced in 1854 by George Boole. The basic Boolean logic gates are AND, OR and NOT.

- ✓ **And** (conjunction) is denoted as x AND y, which yields a result of true if both x and y are true, and false otherwise.
- ✓ **Or** (disjunction) is denoted as x OR y, which yields true if either x or y is true, and false if neither x nor y is true.
- ✓ **Not** (negation) is denoted as NOT x, which yields true if x is false and false if x is true.

2.4.3.2 Wire

The wire is used to connect all the registers and logic gates. These elements must be connected for the entire system to do what human want, from simple tasks, such as adding 1 + 2, all the way to more complex tasks, such converting pulses from blue LEDs reading a Blu-ray disc into crisp, high-definition images on television screen. Any digital system can be constructed from the right amount of logic gates and registers with wires to connect them all.

2.4.3.3 Registers

Registers are simple devices that store pieces of data for use in the future. They are short-term spot for placing data where quick access is needed; example a phone number written before dialing.

Registers keep whatever information is given to them until they're told to forget it and keep new information. This behavior is useful for lifts for storing push button data if two or more command is ordered by passengers from different floors.

2.4.4 The programming technique of FPGA

There are three commonly used software programs used for the design of FPGA. These are the **Xilinx software, Modelsim** and the **Altera software**. They have similar features with a slight differences. All software have two sub-type hardware description programming languages. These are **VHDL** (very high speed integrated circuit hardware description language) and **Verilog**. These sub-type coding style/language is very similar. But, the language of Verilog has some similarity to other coding style of high level programming software like C, C++ etc. Even if the programming languages are different, they can result similar desired behavior on output of FPGA object. One can use his/her preferences which is suitable/easy to him/her.

Procedure of FPGA programming

The steps to programming an FPGA include identifying any blocks of the design that are actually needed to design, choosing a hardware description language (HDL), writing the code in a text editor, synthesizing the design, placing and routing the design, then loading the design onto the FPGA itself. After the design is loaded onto the FPGA, it may require a cycle of *debugging* to fix errors in its functionality. In Figure,

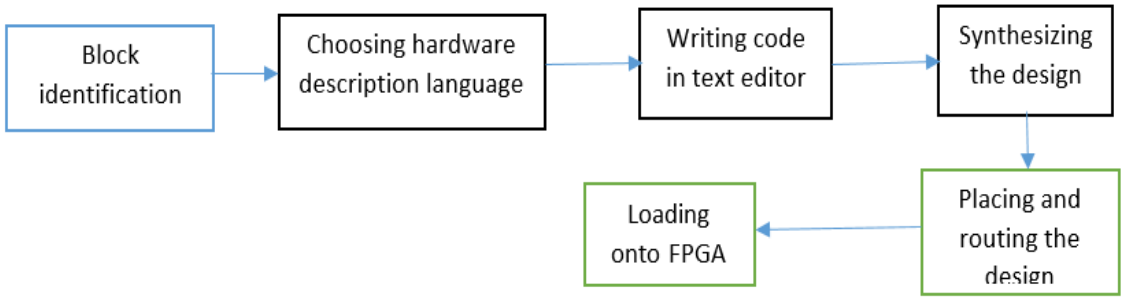


Fig 2 15 FPGA programming procedure

A ***bug*** is an inexplicable defect in computer software that produces an incorrect or unexpected result. The term *debugging* refers to eliminating defects until the whole design runs according to its required functionality.

After writing the HDL design, the next step is to compile the HDL design. In FPGA programming, a synthesis tool takes the HDL design as input and converts it into a network of gates, registers, and wires configured to implement the functions the HDL describes. Then additional processes select which particular gates, registers, and wires to use in the FPGA and create a programming file that will configure the FPGA when it powers up.

So, the HDL code gets mapped directly into the physical hardware elements available on the selected FPGA device. In microprocessor programming, program logic gets mapped into a list of processor instructions that the processor must execute. So it is quite a different — and wonderful — feature that one can convert his/her logic directly to silicon gates for execution.

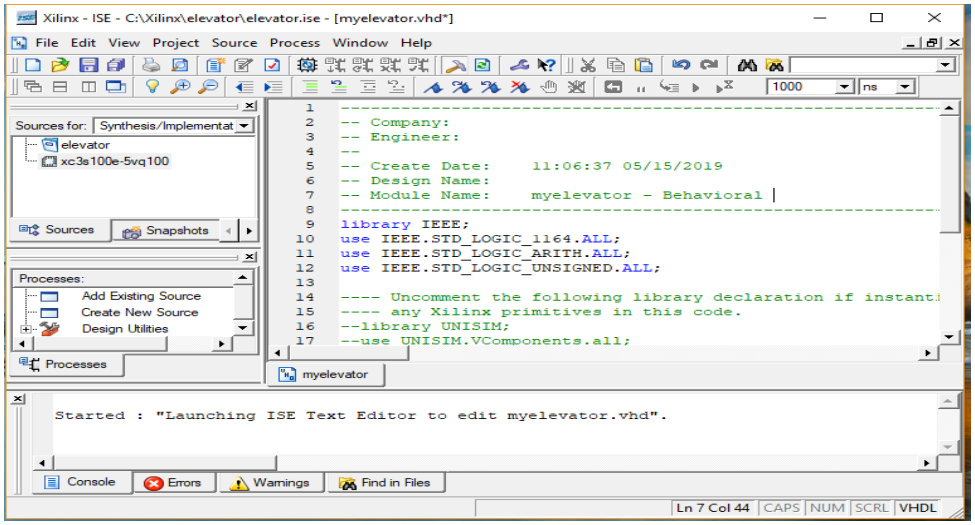


Fig 2 16 Xilinx software feature used for FPGA (VHDL)

2.4.5 Application area of FPGA

2.4.5.1 FPGA for motor control

Motors and motor control are commonplace in any industrial design. In any factory or industrial complex, there will be a variety of widely used different machines with one thing in common, i.e. motors. Most motor control systems are designed with microcontroller technology. However, microcontrollers can fall short of the performance demands of sophisticated motor-control algorithms such as direct torque control (DTC) or sensor-less field oriented control (SFOC), for

example. DSPs have been used in the past to get around that problem, but are usually unable to cost-effectively match an FPGA when it comes to high performance.

2.4.5.2 Automotive Driver Assistance Cameras

One of the big areas of growth in the automobile industry is the explosion of technology-driven features. Even lower priced automobiles come equipped with fancy gadgets like navigation systems, video entertainment systems, and cameras.

Driver assistance and backup cameras are some of the most important safety innovations and help make cars safer than ever.

Forward camera systems are made up of high-speed video processing, complex sensor fusion, and real-time data analysis that enable the automobile to perform corrective action in cases like when the driver nods off and veers into another lane. Forward cameras do their job by integrating with different sensors such as radar and laser sensors. Each type of sensor is different in how it provides data, posing a design challenge for multiple architectures.

Note that Traditional DSP processors or microcontrollers don't have the power to do real-time video processing and analytics at the same time [23].

2.4.6 MATLAB and FPGA

MATLAB® and Simulink® for Model-Based Design provide signal, image, and video processing engineers with a development platform that spans design, modeling, simulation, code generation, and implementation. Engineers who use Model-Based Design to target FPGAs or ASICs can design and simulate systems with MATLAB, Simulink, and State flow® and then generate bit-true, cycle-accurate, synthesizable Verilog® and VHDL code using HDL Coder™ as stated in [24].

HDL describes electronics circuits in terms of the circuit's operation, design, and tests to verify its operation by means of simulation. At the first step of code conversion process, the new design ideas and algorithms are represented in terms of mathematical models and are tested in MATLAB/Simulink floating point data types. However, implementation of control algorithms in FPGAs and ASICs require fixed-point data type conversion to reduce hardware

resources. This conversion process often introduces quantization errors. As a consequence, a signal scaling and word-length optimization becomes a difficult aspect of implementing an algorithm on an FPGA. The real HDL code generation process starts by modeling the algorithm in MATLAB Simulink using a HDL Coder library of more than 200 blocks or State flow. The components and block set supported in HDL Coder can be found by typing "hdllib" in the command window. Code conversion and verification process in MATLAB Simulink HDL Coder involves creating subsystem and converting that subsystem to single fixed. Fixed point conversion can be very challenging and time-consuming, typically demanding 25 to 50 percent of the total design and implementation time.

Procedural HDL conversion from Simulink

The first process in designing Simulink model is to design all mathematical modeling in the continuous time domain. Real world models are require continuous time while HDL code uses refined and fixed point discrete time model. Once the physical mathematical model is created, the next step is deciding which part of the model is to be converted to HDL code. After deciding the desirable part of the model for HDL code generation, cut this part and allow it to be saved by another Simulink model new name. On the parent model, bring "model" block from Simulink library and double click over it. After double click, click on "browse" to relate the cut part Simulink model name. After relating and accepting, all the input and output port terminals will be seen with its name as the cut model, being enclosed in the "model". Then, connect this model reference to the parent model as the previous continuous time model. What is the necessity of model reference block on the parent model? It is used for simulation of parent Simulink model having two or more subsystems with different configuration parameter.

The controller subsystem is desired to be controlled from FPGA devices. For this purpose, the above procedure is followed for the controller subsystem.

Double clicking the model reference will show the insider controller subsystem blocks. Since this model is my target, change the "configuration parameter solver to fixed-step and discrete (not continuous states). Then after, click on 'analyses' of the tool bar of the Simulink model. From the given option choose "control design" then "model discretizer". Giving the sample time and clicking the "discretize" toolbar will change the Simulink model from continuous time to discrete time model. After discretizing, cancel toolbar of discretizing and click again "analysis" of the

Simulink-model to get 'fixed point tool'. Choosing 'fixed point tool' can lead to the procedure of fixed point conversion from double floating point. (Mat lab has a procedural conversion methods in its 'help' documentation explained in Appendix B.)

In most cases, the output result of the Simulink model (s-model) with "fixed point" data system result won't show similar result as the original continuous one. It also refuses to generate HDL (Verilog or VHDL) codes. Therefore, it is appropriate to use FPGA blocks other than conversion.

Once the Simulink model is created and converted to fixed point data type subsystem, HDL Workflow Advisor guides in a step-by-step process to generate code from the model. Right clicking on the subsystem to be converted and choosing 'HDL workflow advisor' let for starting of HDL code generation. Moreover, it helps to check various other parameters and setting that is required for optimal code generation and verification. The generated code bit stream could be downloaded into the FPGA device to function accordingly.

2.4.7 Finite state machine (FSM)

An FSM is used to model a system that transmits among a finite number of internal states. The transitions depend on the current state and external input. Unlike a regular sequential circuit, the state transitions of an FSM do not exhibit a simple, repetitive pattern. Its next state logic is usually constructed from scratch and is sometimes known as "random" logic. This is different from the next state logic of a regular sequential circuit which is composed mostly of "structured" components, such as increments and shifters.

2.4.7.1 Mealy and Moore FSM

An FSM is known as a Moore machine if the output is only a function of state, and is known as a Mealy machine if the output is a function of a state and external input. Both types of output may exist in complex FSM. The Moore and Mealy outputs are similar but not identical. Understanding their subtle differences is the key for control design [25].

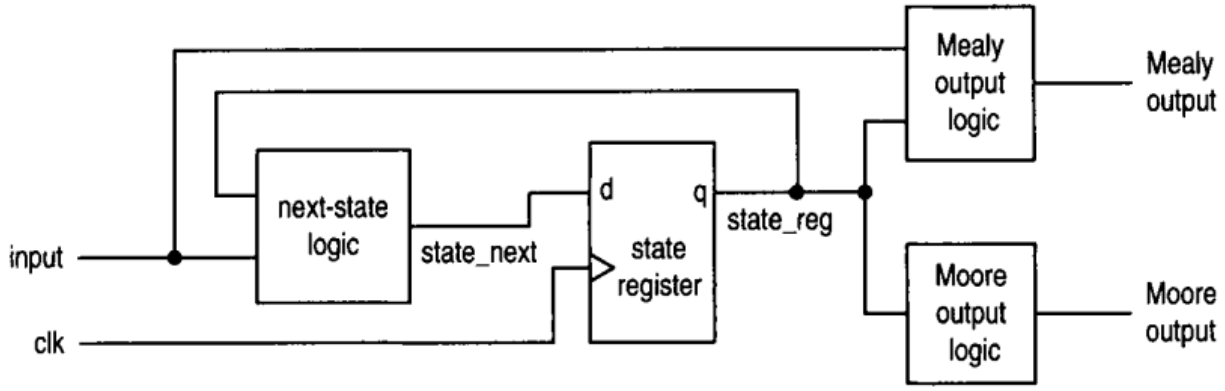


Fig 2 17 block diagram of a synchronous FSM [25]

An FSM is usually represented by state diagram or ASM (algorithmic state machine) chart, both having input, output, states and transitions. The ASM chart representation is somewhat like flow chart and is more descriptive for applications with complex transition conditions and actions. [25] Presents a detail explanation of FSM with examples from simple to complex. FSM structures is not my target. So, my attention directs to objective i.e. lift design.

Both MATLAB and Xilinx software have tools which enable us to draw graphically state diagrams. They can also generate HDL code from their graphical representations. There is some name differences between them. State flow is used for MATLAB graphical representation while State CAD is used to draw state diagrams on Xilinx software (available on Xilinx ISE 8.1i or earlier version).

2.4.7.2 Hardware in the loop (HIL)

In the model based design verification process, HIL is the final testing method of hardware implementation. Several testing methods are there before reaching on HIL testing part. These testing methods are MIL (Model in the loop), SIL (Software in the Loop) and PIL (Processor In the loop).

1. The first step is to develop a model of the actual Plant (hardware) in any simulation environment, for example Simulink, which captures most of the important features of the hardware system. After the Plant model is created, developing the Controller model and verifying if the Controller can control the Plant (which is the model of the motor in this case) as per the requirement, is the following step. This step is called Model-in-Loop (MIL) and the

Controller logic is tested on the simulated model of the plant. If the Controller works as desired, the input and output of the Controller should be recorded for the later stage of verification.

2. Once the model is verified (i.e., MIL in the previous step is successful), the next stage is Software-in-Loop(SIL), which is generation of code only from the Controller model and replace the Controller block with this code. Then, run the simulation with the Controller block (which contains the C code) and the Plant, which is still the software model (similar to the first step). This step will give a clue whether the control logic i.e., the Controller model can be converted to code and if it is hardware implementable. The input-output should be logged here and match with what have been achieved in the previous step. If there exist a huge difference in them, going back to MIL stage and make necessary changes must be followed. If the model tested for SIL and the performance is acceptable, the next step will be Processor-In-Loop (PIL).
3. In this Processor-In-Loop (PIL) step, the Controller model code will be loaded to an embedded processor/FPGA board and run a close loop simulation with the simulated Plant. The Controller Subsystem will be replaced with a PIL block which contain the Controller code running on the FPGA board. This step will show whether the processor/FPGA board has the ability of running the developed Control logic. If there exist glitches, then go back to code, SIL or MIL and rectify them is necessary.
4. Once the plant model has been verified using PIL, now the plant model will be replaced with the original hardware. DC motor whose speed controller is being designed and the controller in FPGA/processor, is now interfaced to the DC motor by connecting the inputs and outputs/states at the right points of sensors/transducers).

It is common to use **speed-goat device** as c-code or HDL code controller. Speed-goat is a controller device which accepts both HDL code and c-code generated from Simulink. When speed-goat is used as HDL code source of controller, the controller should be converted from c-code to HDL code.

Co-simulation

Co-simulation involves the simulation of different subsystems in a distributed manner with different solvers on Simulink. Hence, the modeling is done on the subsystem level without having the coupled problem among the subsystems of the model. During the simulation the subsystems will exchange data. Co-simulation can be considered as the joint simulation of the already well-

established tools and semantics; when they are simulated with their suitable solvers with the subsystem abilities to generate HDL codes. Co-simulation proves its advantage in validation of multi-domain and cyber physical system by offering a flexible solution which allows consideration of multiple domains with different time steps, at the same time. As the calculation load is shared among simulators, co-simulation also enables the possibility of large scale system assessment.

2.4.8 What is state CAD?

Definition: - a state CAD is a graphical representation of the program, found in Xilinx ISE 8.1i. It cannot be found on the new version of Xilinx software.

One can design a simple lift using state cad of Xilinx ISE 8.1i. Using this tools further simplifies the code writing study, since it is possible to generate the code from the graphical drawing. It also has a test bench simulation to show the simulation results. **Chapter 4** illustrates this simulation result.

CHAPTER THREE

MATHEMATICAL MODELING OF LIFT CONTROL (METHODOLOGY)

3.1 Load - the first thing to consider before designing a lift is to know what amount of load is intended to be transported via the lift. Different lifts have different capacity based on their designated objectives. For example, tall buildings will use a group of lifts in order to hinder delay and to accommodate loads. A building with ten stories should use two or more lifts if the building is used for commercial purpose. However, without considering its purpose (load capacity) and shorter time delay, it will bring failure, high electric power consumption or human injuries.

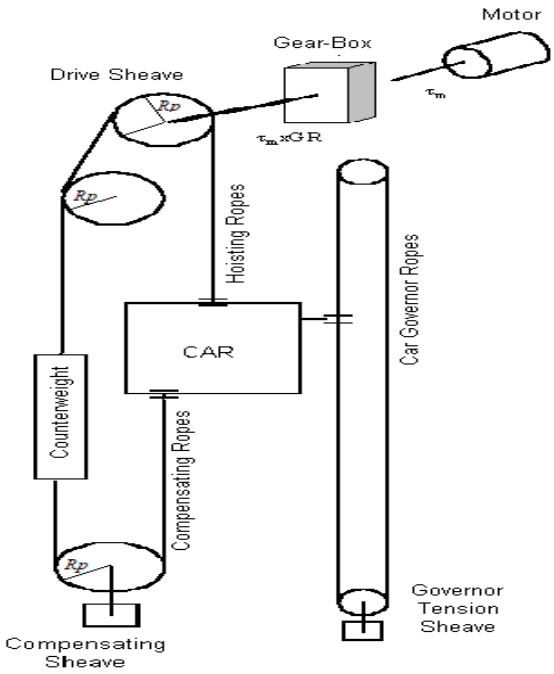


Fig 3 1 overall lift diagrams *[26]*

The maximum load to be transported by the lift in this design is six (6) person each having 80kg using the Table 2 2. The lift is intended to transport persons and some replaceable motors each having less mass size from the mentioned inside the factory. So, the total maximum weight

$$W_L = MXa = 6x80kgx9.8\,m/s^2 = \mathbf{4704}\,kg\,m/s^2 = 4704N \;\ldots\ldots\ldots\ldots (3.1)$$

3.2 Car- the other part that directly influence the motor specification is the load of the car`s materials. Car is the constructed combination of rigid or hard substances in which the load is enclosed in. It is made up of steel materials or aluminum. The car will contain door, light, floor call buttons and emergency push button.

The space occupied by one person is elliptical in shape with width of 550 mm and length of 400 mm, considering man's shoulder area as explained by [27].

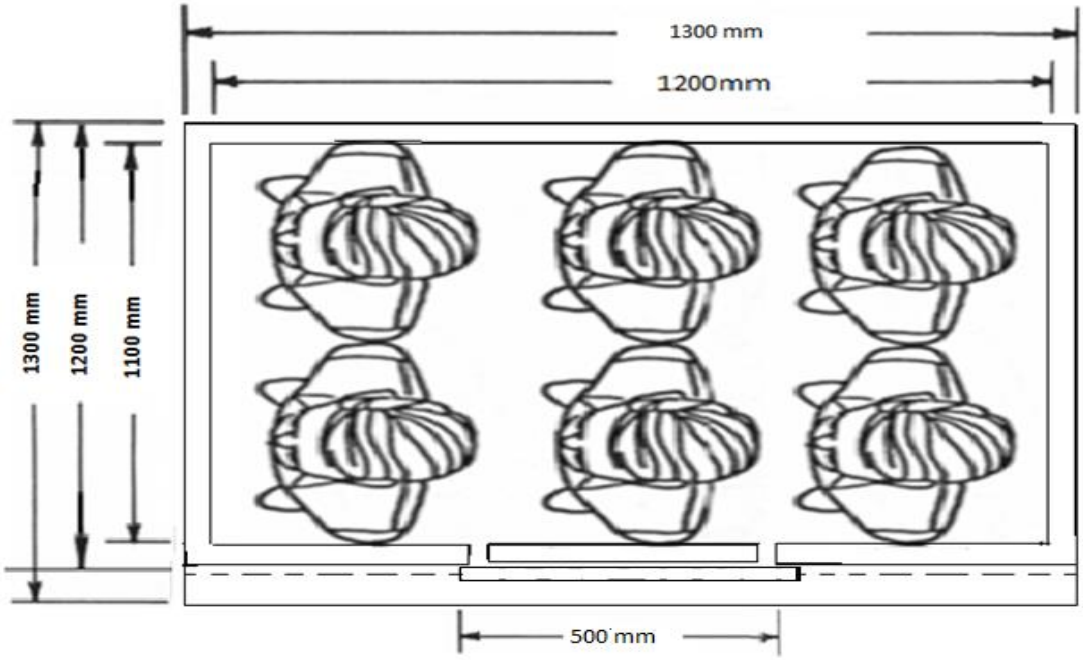


Fig 3 2 six passengers in the lift (top view) [27]

One can calculate the mass of the car by using the volume of the car and the densities of the materials it is composed of. But, this activity is time consuming and tedious. This is due to the material used to build the car is not be one type. The car is mostly made from aluminum, bronze,

steel and others. The calculation of the mass of car being built from same material has been explained in simplest way from the following figure.

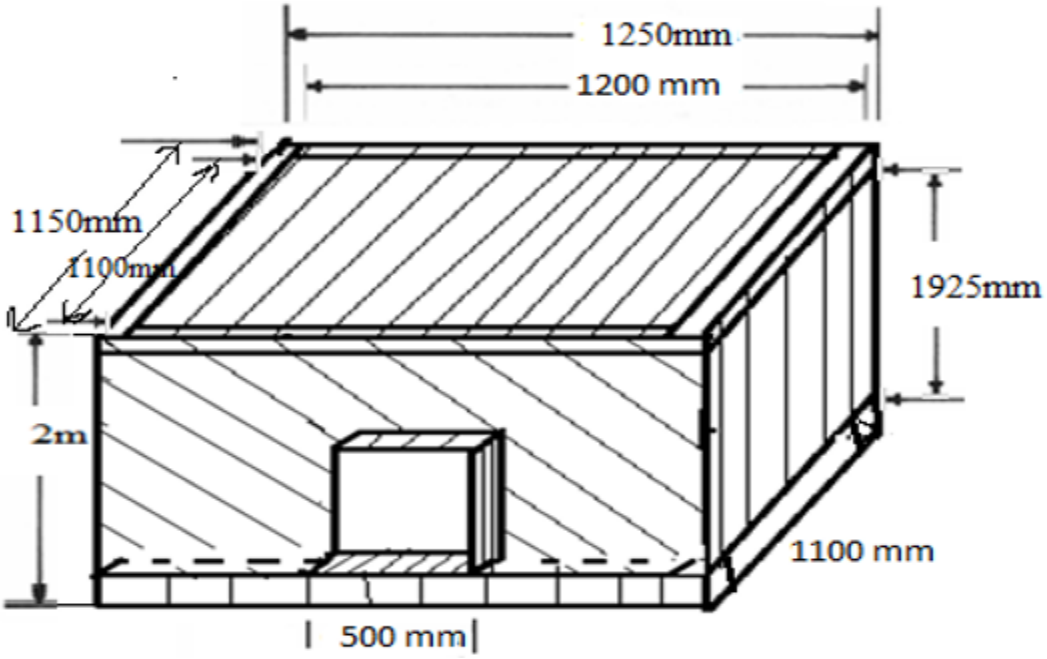


Fig 3 3 three dimensional view of the car

Total volume of the car

$$V_T = lxwxh = 1250mmx1150mmx2000mm = 2875000000mm^3 = 2.875m^3 \dots\dots\dots (3.2)$$

Where l=length, w=width and h=height of the car.

The inside volume of the car

$$V_I = lxwxh = 1200mmx1100mmx1925mm = 2541000000mm^3 = 2.541m^3 \dots\dots\dots (3.3)$$

The car material volume is the difference between the total outside volume and the inside volume of the car. This can be obtained as

$$V_C = V_T - V_I = 2.875m^3 - 2.541m^3 = 0.334m^3 \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.4)$$

Assuming that the car is made from aluminum, the mass of the car can be calculated by using the densities of it. The density of aluminum $\rho_{Al} = 2691kg/m^3$, mass of the car,

$$M_C = \rho_{Al} x V_C = \frac{2691kg}{m^3} x\ 0.334m^3 = 898.794kg \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.5)$$

In other way, in order to reduce the mentioned problems, it is preferable to use the ratios of rated load mass to car mass used by lift companies. Different companies use slightly different rated load mass to car mass ratio. The reference given by **[11]** of **chapter 2** shows the different rated load to car mass ratios.

The table shows that the ratio of the rated load mass to the car mass, denoted by R_m . Taking the lift model ratio of "F" from the table, which is$R_m = \frac{1590}{2900} \approx 0.55$ for this paper with rated load mass of 480kg (six passenger each having 80kg), then car mass will be,

$$R_m = \frac{M_L}{M_C} = \frac{480kg}{M_C} = 0.55 \ \ ====> M_C = \frac{480kg}{0.55} \approx 873kg \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.6)$$

Comparing equation (3.5) and equation (3.6), the mass of the car has a slight difference. Hence, for the paper design analysis equation (3.6) result has been used.

When using the full rated load mass, the overall load to be transported up and down will be the sum of the load and weight of the car. Mathematically,

$$W_{CT} = W_{Car} + W_{Load} = 873x9.8N + 4704N = (873 + 480)x9.8\ N \approx 13259.4N \ldots\ldots (3.7)$$

3.3 Counter-Weight: - as it is clearly indicated on chapter 2, the first relation of counterweight mass to car mass has been chosen. i.e. $M_{cw} = M_c$.

Steel rope: **-** is the other part, which has little load effect on motor for small height building. On tall building the weight of the rope increases exponentially to its length. This is the reason that almost all tall buildings use other types of ropes like fiber ropes and others.

For medium height building, the mass of the steel can be described mathematically as,

$$M_{steel} = mass\ per\ length * total\ length \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.8)$$

However, the weight of the steel rope has been neglected since it has dynamic effect on the load based on its position as explained below.

⇒ When the car/cabin is on the ground floor: most of the rope Wight will be to car-side hence increases car-side weight.

⇒ When the car/cabin is around the third floor:-most the rope weight will distribute equally to the car-side and counterweight.

⇒ When the car/cabin is on the fifth floor:-almost all rope weight will have a position to counterweight-side.

Sheave groove selection

Selecting ropes involves determination of the number of grooves in which steel ropes are passing over it. There are two type shape of grooves available on the driving and deflector sheave, as explained on chapter 2. U-type groove has a good benefit than V-type groove since it has less friction, which helps for long durability of the steel rope.

3.4 Tension force on the rope

Free body diagram are usually used to calculate the tension on the rope in simple way. Ignoring the compensating rope tensions found under the counterweight and the car because of their free movement with low friction on the compensating drive sheave and, assuming that the car with its full load passengers accelerating upward, the acceleration of the car can be solved from the bottom diagram/figure.

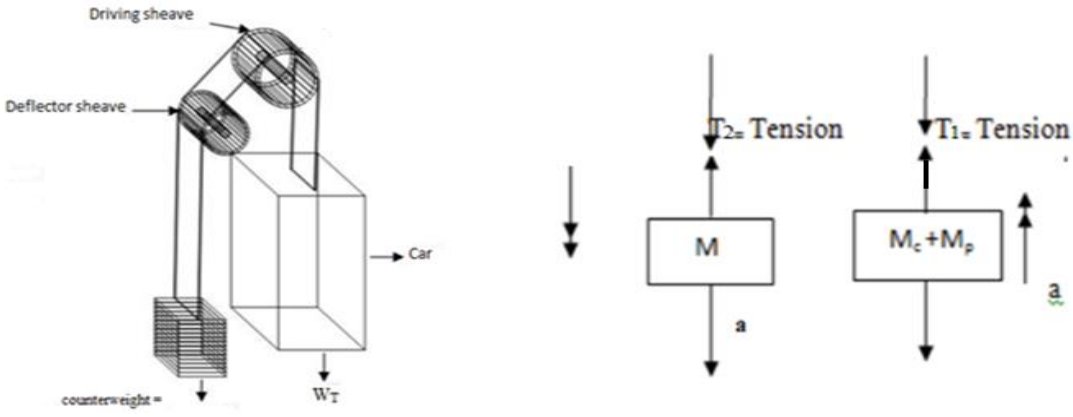


Fig 3 4 tension (car side & counterweight side) [1]

Lift has accelerating region, constant speed region and decelerating region. The lift accelerating region is when the lift starts from its resting floors. After a short period of run, it enters to its constant speed until deceleration starts. This has been explained on subtopics of the ***reference speed*** below.

For the car moving upward from rest at full load, the sum of forces acting on the car, using newton's 2nd law

$$\sum_{i=0}^{N} F_i = ma,$$

$$T_1 - (M_c + M_P)g = (M_c + M_P)a$$

$$T_1 = (M_c + M_P)g + (M_c + M_P)a$$

$$T_1 = (M_c + M_P)(g + a) \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.9)$$

On the counterweight side, when the lift moves upward

$$T_2 - M_{cw}g = -M_{cw}a$$

$$T_2 = M_{cw}g - M_{cw}a = M_{cw}(g - a) \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.10)$$

The acceleration ranges of a lift when car start from rest is usually between $0.1 - 2m/s^2$. Consider an acceleration of the car to be $0.8\ m/s^2$ at its full load. The above two equations can be solved as

$$T_1 = (M_c + M_P)(g + a) = (873 + 480)(9.81 + 0.8) = 14{,}355.33N \dots\dots\dots (3.11)$$

$$T_2 = M_{cw}g - M_{cw}a = M_{cw}(g - a) = 873(9.81 - 0.8) = 7{,}865..73N \dots\dots\dots (3.12)$$

The above results of equation conclude that the tension force at the opposite position is different in magnitude. The tensional stress on the motor will be their difference. i.e.

$$T_{stress} = T_1 - T_2 = 14{,}355.33N - 7{,}865.73N = 6{,}489.6N \dots\dots\dots\dots\dots (3.13)$$

The tension force of rope calculated above in equation (3.11) is used to choose the right rope capable of suspending the load. Therefore, the rope with tension force greater than the calculated

could be used. The other benefit of calculating tension force is for the choice of the right driving and deflector sheave stress capacity.

From the **Fig 3 4**, the tension forces of the counterweight side and car side sum up at the two sheaves. Since the driving sheave is found at elevated point on the figure, the stress on the sheave approximately will be

$$T_1 + T_2 = 14{,}355.33N + 7{,}865.73N = 22{,}221.06N \ldots\ldots\ldots\ldots\ldots (3.14)$$

As a result, the sheave to be installed should have equal or greater than the mentioned stress force. Two or more number of ropes passing over two or more number of grooves has the tensile capacity to suspend the two-sided weights. Consider 5 steel ropes in U-shaped groove of the sheave. One of the rope passing over the driving sheave will be enough to suspend the whole load. But, the other 4 ropes are used as security to hinder falling if abrasion is occurred. If two of the ropes are used for holding the counterweight and car load mass, the tension force must be distributed equally to both ropes. The load stress force over one of the two ropes will be

$$T_{12} = \frac{T_1 + T_2}{2} = \frac{14{,}355.33N + 7{,}865.73N}{2} = \frac{22{,}221.06N}{2} = 11{,}110.53N \ldots\ldots (3.15)$$

Lift motor speed

It is known that most real world motors have high speed: they couldn't be used directly attached with the driving pulley. If it is used, passengers will fear due to their sensing of that they are being fired as bullet when moving up or they are falling from the building because of the high speed. For this reason, most lift company uses motors with less speed (gearless motor). But, for high speed motors, it is desirable to use gears for lift design. Although usage of gear between motor and driving sheave can lead efficiency reduction problems, they play an important role in converting torque and speed to the desired values.

3.5 Gear

When a gearbox is fitted between the motor and the drive, it has an effect on the torque, speed and inertia. In general, a gearbox is used to match the drive to the load. Most gearboxes are used as reduction gearbox (i.e., to reduce the rotational speed). Thus, the speed will be reduced, and the torque will be increased.

This is very useful in practice; the rotational speed of most motors is too high to directly drive the load, and the torque is too low. The gearbox *matches* the drive to the load. The motor shaft is sometimes referred to as the high speed shaft and the load shaft is sometimes referred to as the low speed shaft.

Inertia of the translational masses referred to the motor [28]

A motor fitted with a flywheel driving a translational load through a gearbox and driving sheave is shown below.

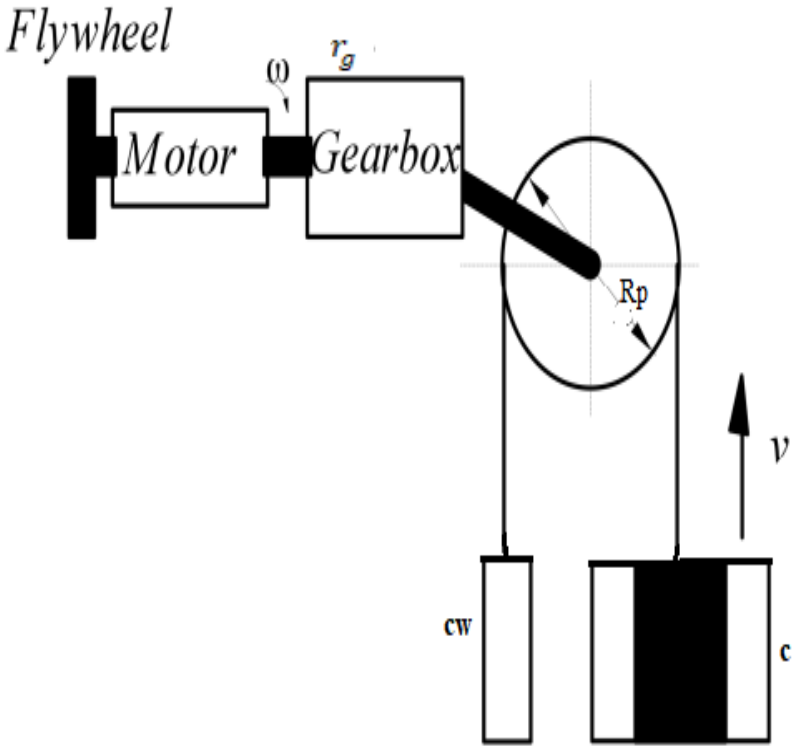


Fig 3 5 geared lift system [28]

To convert from translational quantities to rotational quantities referred to the motor, I need to find the relationship between the linear motion and the rotational motion. If the car travels by *x* meters, this will cause the sheave to rotate by:

$$\frac{x}{2\pi Rp} revolutions = \frac{2x}{2\pi Rp} radians = \frac{x}{\pi Rp} radians \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.16)$$

Where Rp is the radius of the drive sheave.

When this rotation is referred to the motor through the gearbox, it will be amplified by the reduction ratio of the gearbox, r_g. Thus, the final rotation in radians will be:

$$x\left(\frac{r_g}{Rp}\right) radians. \quad where\ , r_g = gear\ ratio$$

Thus, the factor, $\left(\frac{r_g}{Rp}\right)$, (or its inverse, $\left(\frac{Rp}{r_g}\right)$), can be used to convert between translational quantities at the sheave, and rotational quantities at the motor (or vice versa). For inertia, translational masses are multiplied by the square of this factor to convert them from translational masses at the sheave, to rotational masses at the motor side. Thus, the inertia of the car, counterweight and passenger load, can be reflected at the motor side (high speed shaft) as follows :(Note that the masses of the trailing electric cable, rope, door and driving pulleys have been omitted due to their low inertia when referred to the motor shaft.)

$$J_{shaft,T} = (M_C + M_P + M_{CW}) * \left(\frac{Rp}{r_g}\right)^2 = (873 + 480 + 676.5)\left(\frac{Rp}{r_g}\right)^2 \ldots\ldots\ldots\ldots\ldots (3.17)$$

Alternatively, if the sheave diameter and the gearbox reduction ratio are not known (or have not been selected yet), but the linear speed of the lift and the rotational speed of the motor have been decided, another ratio can be used i.e.

$$\frac{Rp}{r_g} = \frac{v}{\frac{n}{60} * 2\pi} = \frac{v}{n} * \frac{60}{2\pi} = 9.55 * \frac{v}{n} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.18)$$

Where,

V is the linear lift speed in ms^{-1}

n is the motor speed in rpm.

3.6 Efficiency of whole installation

The system installation efficiency are of two types. These are the gearbox efficiency and shaft efficiency. It is denoted by η, is the product of the two efficiencies.

$$\eta = \eta_{shaft} x \eta_{gear} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.19)$$

The efficiency has inversely proportional to inertia and torque. That means,

$$J_{shaft,T} = M_T * \left(\frac{Rp}{r_g}\right)^2 * \frac{1}{\eta} \ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.20)$$

$$\tau = M\left(\frac{Rp}{r_g}\right) * \frac{1}{\eta} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.21)$$

Note that the forward gearbox efficiency is used in this case. However, when the system is overhauling (Quadrant II, braked lowering) and the motor is acting as a generator, the power flow is from the shaft to the motor, and thus the reverse efficiency is used in that case.

Table 3 1 lift Parameters used for the design are listed below

Parameters	**Value**
weight of lift car	873 kg
Total weight of lift	1353 kg
Maximum no. of passengers loaded	6 (80 kg/person)
Speed of lift	1.25 m/sec
Acceleration, a_{max}	$0.8 m/s^2$
sheave radius, r	0.1 m
Motor torque	98 Nm
Motor speed	1432 RPM
Gear ratio	12:1
Pulley mass	1 kg
$Efficiency, \eta$	75 %

3.7 Electrical mathematical modeling**:**

- Dc permanent magnet motor with two pole has the circuit diagram as given in the below figure.

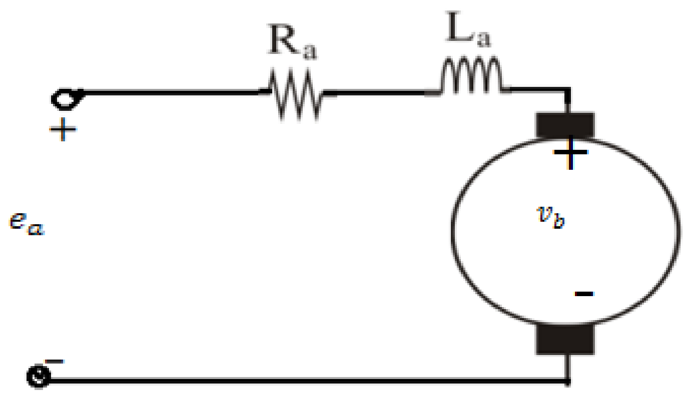


Fig 3 6 dc motor circuit diagram [29]

Using KVL equation around the loop,

$$e_a = R_a i_a + L_a \frac{di_a}{dt} + v_b \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.22)$$

The torque obtained from the motor is,

$$T_m = K_t i_a \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.23)$$

The electro-motive force developed in the armature's winding

$$v_b = K_b \omega_m \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.24)$$

The DC Motor block assumes that there are no electromagnetic losses. This means that mechanical power is equal to the electrical power dissipated by the back EMF in the armature. Equating these two terms gives:

$$p_{mec} = T_m \omega_m = v_b i_a$$

Using 3.23 and 3.24 given above

$$K_t i_a \omega_m = K_b \omega_m i_a$$

$$K_t = k_b \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.25)$$

Therefore, the variable, Kt, has been used in place of, kb, in the following equation.

Ignoring the effect of the inductor for dc case and applying KVL equations, I can get another useful equation. This is,

$$e_a = R_a i_a + v_b$$

$$i_a = \frac{e_a - v_b}{R_a} = \frac{e_a - K_t\omega_m}{R_a} \quad \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.26)$$

Substitute 3.26 into 3.23, I get

$$T_m = \frac{K_t}{R_a}(e_a - K_t\omega_m) \;\; or \;\; e_a = \frac{R_aT_m}{K_t} + K_t\omega_m \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.27)$$

Where

K_t is the motor torque constant and k_b back EMF constant.

ω_m is the angular speed of the motor's shaft.

e_a is the voltage difference between the armature terminals

R_a is the resistance of the armature circuit.

i_a is the current flowing through the armature circuit.

L_a is the inductance of the armature circuit.

3.8 Mechanical Mathematical Modeling

In order to derive the motion equation that describes the elevator's mechanical system it is assumed that the mass of the drive belt is ignored due to its material composition and length. In practice, the drive belt or drive cable is constructed from steel and its significant mass contributes to the load torque in a non-linear fashion depending upon the position of the car. The motion equation of the entire system from the motor's perspective described in [29], is:

$$\boldsymbol{T_m = J_m\frac{d\omega_m}{dt} + B\omega_m + T_L} \quad \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (\mathbf{3.28})$$

Where

T_m is the motor's moment of inertia

J_m is the angular speed of the rotor

B is the friction coefficient of the motor

T_L is the load torque placed on the motor's shaft

The load torque T_L that is placed on the drive pulley, which is mounted over the motor's shaft, referred to the motor shaft, is expressed as,

$$T_L = \frac{R_P}{r_g} F_L + \frac{3J_P}{{r_g}^2} * \frac{d\omega_m}{dt} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.29)$$

Where

$\boldsymbol{R_P}$ $is\ the\ radius\ of\ the\ drive\ pulley$

$\boldsymbol{J_P}$ $is\ the\ drive\ pulley\ moment\ of\ inertia$

$\boldsymbol{F_L}$ $is\ the\ force\ exerted\ on\ the\ drive\ pulley$

The drive pulley moment of inertia can be calculated by using [30] by the formula of

$$J_P = M_p {R_p}^2 \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.30)$$

Where

$M_p\ and\ R_p$: - are mass and radius of the driving pulley respectively.

If the lift car is moving upwards & using [28], the load force on the motor will be,

$$F_L = \left[g(M_C - M_{CW}) + (M_C + M_{CW}) \frac{dU_C}{dt} \right] \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.31)$$

Where

$\boldsymbol{g}$ $\boldsymbol{is}\ the\ gravitational\ constant$

$\boldsymbol{M_C}$ is the mass of the car side load

$\boldsymbol{M_{CW}}$ $is\ the\ counter\ weight\ mass$

$\boldsymbol{U_C}$ $is\ the\ linear/translational\ speed\ of\ the\ car$

$$a = \frac{du_c}{dt} = \frac{R_P}{r_g} * \frac{d\omega_m}{dt} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.32)$$

Using (3.32), equation (3.31) and equation (3.30) into equation (3.29), I get

$$T_L = \frac{R_P}{r_g} \left[g(M_C - M_{CW}) + (M_C + M_{CW}) \frac{dU_C}{dt} \right] + 3M_P \left(\frac{Rp}{r_g} \right)^2 \frac{d\omega_m}{dt}$$

$$= \frac{R_p}{r_g} g(M_C - M_{cw}) + \left(\frac{Rp}{r_g} \right)^2 (M_C + M_{CW}) \frac{d\omega_m}{dt} + 3M_P \left(\frac{Rp}{r_g} \right)^2 \frac{d\omega_m}{dt}$$

$$T_L = \frac{R_p}{r_g} g(M_C - M_{cw}) + \left(\frac{Rp}{r_g} \right)^2 (M_C + M_{CW}) \frac{d\omega_m}{dt} + 3M_P \left(\frac{Rp}{r_g} \right)^2 \frac{d\omega_m}{dt} \dots\dots\dots\dots\dots (3.33a)$$

When the lift car accelerates moving downwards, the counter-weight's mass should be considered while the e lift car's mass should be ignored. The equation for load torque when it is moving downwards has the form:

$$T_L = \frac{R_p}{r_g} g(M_{Cw} - M_c) + (M_C + M_{CW}) \left(\frac{Rp}{r_g}\right)^2 \frac{d\omega_m}{dt} + 3M_P \left(\frac{Rp}{r_g}\right)^2 \frac{d\omega_m}{dt} \dots\dots\dots (3.33b)$$

Considering the shaft efficiency for both i.e. equation (3.33a) and equation (3.33b), referred to the motor side

$$T_L = \frac{R_p}{\eta r_g} g(M_C - M_{cw}) + (M_C + M_{CW}) \frac{1}{\eta} \left(\frac{Rp}{r_g}\right)^2 * \frac{d\omega_m}{dt} + \frac{3}{\eta} M_P \left(\frac{Rp}{r_g}\right)^2 \frac{d\omega_m}{dt} \dots\dots\dots\dots (3.34a)$$

$$T_L = \frac{R_p}{\eta r_g} g(M_{Cw} - M_c) + \frac{1}{\eta} \left(\frac{Rp}{r_g}\right)^2 (M_C + M_{CW}) \frac{d\omega_m}{dt} + \frac{3}{\eta} M_P \left(\frac{Rp}{r_g}\right)^2 \frac{d\omega_m}{dt} \dots\dots\dots (3.34b)$$

Substituting the load torque equation for the lift car **moving upwards and downward** into the motor's motion equation (**Eqn. 3.28**), the motion equation of the entire system is obtained as:

$$T_m = J_m \frac{d\omega_m}{dt} + B\omega_m + \frac{R_p}{\eta r_g} g(M_C - M_{cw}) + \frac{1}{\eta} \left(\frac{Rp}{r_g}\right)^2 (M_C + M_{CW}) \frac{d\omega_m}{dt} + \frac{3}{\eta} M_P \left(\frac{Rp}{r_g}\right)^2 \frac{d\omega_m}{dt} \dots (3.35a)$$

$$T_m = J_m \frac{d\omega_m}{dt} + B\omega_m + \frac{R_p}{\eta r_g} g(M_{Cw} - M_c) + \frac{1}{\eta} \left(\frac{Rp}{r_g}\right)^2 (M_C + M_{CW}) \frac{d\omega_m}{dt} + \frac{3}{\eta} M_P \left(\frac{Rp}{r_g}\right)^2 \frac{d\omega_m}{dt} \dots (3.35b)$$

Further simplifying the two equations, I get

For upward

$$T_m = \left(J_m + \frac{1}{\eta} \left(\frac{Rp}{r_g}\right)^2 (M_C + M_{CW}) + \frac{3}{\eta} M_P \left(\frac{Rp}{r_g}\right)^2\right) \frac{d\omega_m}{dt} + \frac{R_p}{\eta r_g} g(M_C - M_{cw}) + B\omega_m$$

$$T_m = J_T \frac{d\omega_m}{dt} + T_{LG} + B\omega_m \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.36a)$$

For downward

$$T_m = \left(J_m + \frac{1}{\eta} \left(\frac{Rp}{r_g}\right)^2 (M_C + M_{CW}) + \frac{3}{\eta} M_P \left(\frac{Rp}{r_g}\right)^2\right) \frac{d\omega_m}{dt} + \frac{R_p}{\eta r_g} g(M_{Cw} - M_c) + B\omega_m$$

$$T_m = J_T \frac{d\omega_m}{dt} + T_{LG} + B\omega_m \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.36b)$$

Where the equivalent moment of inertia is:

$$J_T = J_m + \frac{1}{\eta}\left(\frac{Rp}{r_g}\right)^2 (M_C + M_{CW}) + \frac{3}{\eta} * M_p \left(\frac{Rp}{r_g}\right)^2 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.37a, b)$$

And the load torque due to gravity is:

$$T_{Lg} = \frac{R_p}{\eta r_g} g(M_C - M_{cw}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.38a)$$

$$T_{Lg} = \frac{R_p}{\eta r_g} g(M_{Cw} - M_c) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.38b)$$

General assumption

⇒ The three equal magnitude and radius of the pulleys' inertia referred to the motor will have negligible effect when compared to the other masses. That means, using table 4.1 information,

$$J_{3p} = \frac{3}{\eta} * M_p \left(\frac{Rp}{r_g}\right)^2 = \frac{3}{0.6} * 1 * \left(\frac{0.1}{12}\right)^2 = \frac{0.000208333}{0.6} kg.m^2 = 0.00034722 kg.m^2$$

Due to its minimal effect on model ignoring it won`t affect the calculation.

⇒ The viscous friction constant (*B*) of most motor is minimum. As a result, it is possible to neglect it in the calculation of total motor torque for simplified analysis; but, it must be used later in the Simulink analysis in order to increase the simulation result appear close to real.

And for the inertia, Equating equation (3.20) and equation (3.37a, b), results

$$J_T = J_m + J_{shaft,T} + (J_{3p} \approx 0)$$

$$J_T = J_m + \frac{1}{\eta}\left(\frac{Rp}{r_g}\right)^2 (M_C + M_{CW}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.39)$$

If flywheel is attached to the motor shaft, it will increase the total inertia of the system. The flywheel is usually used to reduce the acceleration of lift car. When flywheel is used, equation (3.39) can be re-written as

$$J_T = J_F + J_m + \frac{1}{\eta}\left(\frac{Rp}{r_g}\right)^2 (M_C + M_{CW}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.40)$$

For the acceleration calculation, use equation (3.36a, upward direction acceleration).

$$T_m = J_T \frac{d\omega_m}{dt} + T_{LG}$$

$$\frac{d\omega_m}{dt} = \alpha = \frac{T_m - T_{LG}}{J_T} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.41)$$

Based on the information given by **Table 3 1**

$$T_{LG} = \frac{R_p}{\eta r_g} g(M_C - M_{cw}) = \frac{0.1}{0.75 * 12} * 9.81 * (1353 - 873)Nm = 52.32Nm \approx 52.3Nm$$

From equation (3.20) and (3.41), $J_{shaft,T}$ will be

$$J_{shaft,T} = \frac{1}{\eta}\left(\frac{Rp}{r_g}\right)^2 (M_C + M_{CW}) = \frac{1}{0.75} * \left(\frac{0.1}{12}\right)^2 * (1353 + 873)kg.m^2 = 0.20611kg.m^2$$

Assume the motor has $J_m = 0.1kg.m^2$ and linear/translational acceleration at full load $a = 0.8\,m/s^2$

Now considering no **flywheel inertia** and calculating the angular acceleration,

$$\frac{d\omega_m}{dt} = \alpha = \frac{T_m - T_{LG}}{J_T} = \frac{T_m - 52.3Nm}{(0.1 + 0.20611)kg.m^2} = \frac{0.8ms^{-2}}{Rp/r_g} = 96\ rad/s^2$$

$$T_m = (0.1 + 0.20611)kg.m^2 * 96\ rad/s^2 + 52.3Nm = 81.687Nm \dots\dots (3.42)$$

Equation (3.42) show that the maximum torque needed from the motor at its starting. But, the net torque needed from the motor is less than its starting torque. If a **flywheel inertia** has been used, (take $J_f = 0.1\ kg.m^2$) . i.e.

$$\alpha = \frac{T_m - T_{LG}}{J_T} = \frac{81.687Nm - 52.3Nm}{(0.1 + 0.1 + 0.20611)kg..m^2} = 72.361\ rad/s^2$$

$$\boldsymbol{a} = \frac{Rp}{r_g}\alpha = \frac{0.1m}{12} * 72.361\,rad/s^2 = 0.603\,m/s^2$$

The acceleration given above is the full load acceleration (the 1st without & 2nd with flywheel inertia).

Now, for an empty car traveling to the upward direction and using equation (3.20), the inertia of the shaft will be

$$J_{shaft,T} = \frac{1}{0.75} * (873kg + 873kg + 0)\left(\frac{0.1m}{12}\right)^2 = 0.16166 \approx 0.1616kg.m^2$$

$$J_T = J_m + J_f + J_{shaft,T} = (0.1 + 0.1 + 0.1616)kg.m^2 = 0.3616kg.m^2$$

$$\alpha = \frac{T_m - T_{LG}}{J_T} = \frac{81.687Nm - 52.3Nm}{0.3616kg.m^2} = 81.269\,rad/s^2$$

$$\boldsymbol{a} = \frac{Rp}{r_g}\alpha = \frac{0.1m}{12} * 81.269\,rad/s^2 = 0.67724\,m/s^2 \approx 0.677\,m/s^2$$

3.9 Electro-mechanical modeling

Now, relate electrical equations with mechanical equations by using the above equation of (3.23) and (3.36a, b). i.e

$$T_m = K_t i_a(t) = J_T \frac{d\omega_m}{dt} + T_{LG} + B\omega_m$$

$$K_t i_a(t) = \begin{cases} \left\{\left(J_m + \frac{1}{\eta}\left(\frac{Rp}{r_g}\right)^2 (M_C + M_{CW}) + \frac{3}{\eta}M_P\left(\frac{Rp}{r_g}\right)^2\right)\frac{d\omega_m}{dt} + \frac{R_p}{\eta r_g}g(M_C - M_{cw}) + B\omega_m\,, for\ upward\ movement \\ \left\{\left(J_m + \frac{1}{\eta}\left(\frac{Rp}{r_g}\right)^2 (M_C + M_{CW}) + \frac{3}{\eta}M_P\left(\frac{Rp}{r_g}\right)^2\right)\frac{d\omega_m}{dt} + \frac{R_p}{\eta r_g}g(M_{Cw} - M_c) + B\omega_m\,, for\ downward\ movement \end{cases} \quad \ldots (3.43)$$

This relation has been used for Simulink analysis on the next chapter.

3.10 The electrical motor power specification

The electric power supplied to the dc motor should be enough to pull up and pull down the lift by rotating clockwise and anticlockwise direction. If enough power is not supplied to the motor, the motor won`t able to rotate due to scarce of electric power (voltage and current). In contrast, if enough electric power source is supplied, the reverse is true i.e. the motor will become functional.

To calculate minimum size of the motor, torque-speed-power relationship will be used.

$$P_m = T_m * \omega_m \ldots \ldots \ldots (3.44)$$

Substituting equation (3.36) into (3.42)

$$P_m = \left(J_T \frac{d\omega_m}{dt} + B\omega_m + T_{Lg}\right) * \omega_m \ldots \ldots \ldots (3.45)$$

This equation describes the maximum mechanical power (P_m) required from the motor. The mechanical power output of the motor should exceed the calculated power of **equation (3.45).** This high rated mechanical output of motor will overcome some residual powers that have been ignored due to their little effect as described the above. Those of residual powers like rope weight, trailing cable weight do not disturb motor from functioning. For the electric power, if the motor has 85% power conversion efficiency from electric to mechanical power: then the electrical power feed to the motor is

$$\boldsymbol{P_{el} = \frac{P_m}{\eta_{em}} = \frac{P_m}{0.85} = 1.176\, P_m \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots. (3.46)}$$

Equation (3.46) describes the maximum electric power supplied to motor.

3.11 Designing of Control System

The digital control system for implementation of position and speed control of the elevator's electric drive is composed of two cascaded loops which are coordinated to work together using information obtained from the motor's speed sensor. After knowing the motor's steady-state and dynamic parameters, the individual control loops are designed to perform a specific task using a specific input. Using MATLAB'S Simulink software package, these separate loops are integrated into one digital control system which is then generated into FPGA bit stream HDL code to be loaded on the FPGA device hardware to function in real time.

Feedback Control of Electric Drive

Feedback control is common in control engineering to precisely and quickly regulate a system based upon the inputs obtained from the system itself and the desired reference input. It has the objective of bringing steady state error to zero with short period and minimum overshoot by applying appropriate controllers.

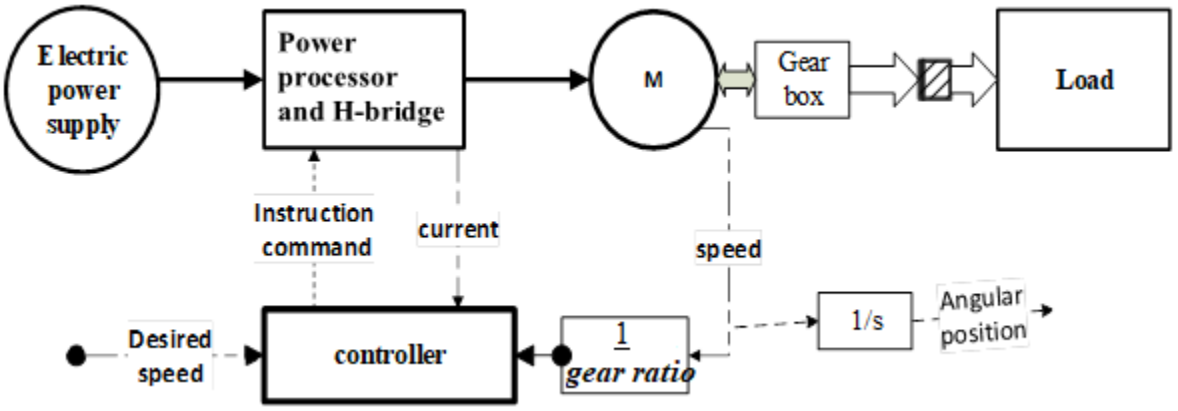


Fig 3 7 Feedback control system of dc motor

Direct Current (DC) power source could be supplied to the motor from rectified ac source via a Power Processor and H-bridge. The speed of the motor is captured by speed sensor and fed back to the controller. The position of the motor is obtained by integrating the speed of the drive over time and is compared with the integrated desired speed reference. The current flow out of the PPU is monitored and reported back to the controller as well. With these two real-time inputs the controller can bring the motor to the desired position. A closer examination of the controller block reveals two nested loops (see Fig. 3.8). The loops contain a Proportional-Integral (PI) controller which is well suited to the task of regulating the electric drive's speed and drive's position.

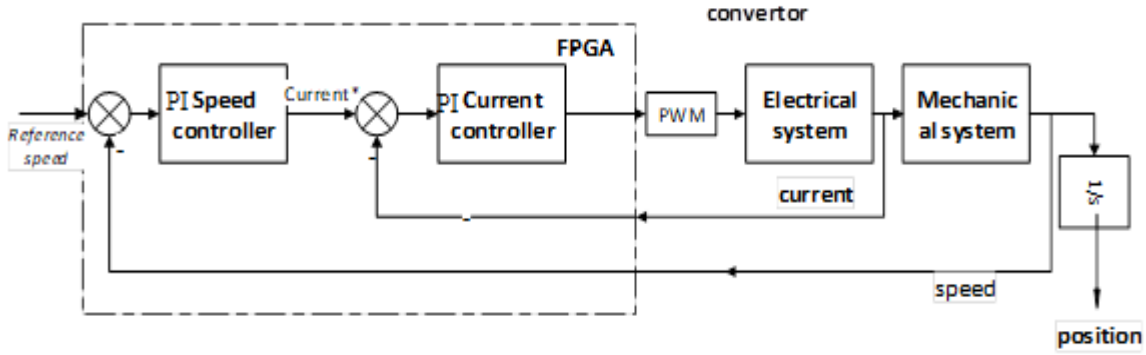


Fig 3 8 Cascaded control system

As shown on **figure 3.8** above, the controller is designed to have HDL code generation to be used on FPGA device. There are huge difference between PI control of fixed point and double point data precision simulation (continuous time and discrete time simulation). The double point data precision simulation is common when designing controller by using MATLAB coding or Simulink modeling. The controller designed by discrete or continuous time blocks must be converted to HDL compatible blocks for FPGA based controller. This is due to FPGA devices, being used as controller, have reduced hardware resources. So, they use single precision data for efficient use of their hardware resources.

The FPGA based Simulink modeling design has complex process. It is not easy as designing of continuous time or discrete time modeling. One can use both modeling methods successively to reach finally on HDL code generation supported Simulink model. The complexity of designing HDL code generation supported modeling is due to the delayed behavior of sample time on the system. So, the controller designed by using continuous time/discrete time blocks won`t be used for FPGA based devices as they are. There must be some necessary adjustment of Simulink blocks on the FPGA based controller from their similar part of the continuous time model.

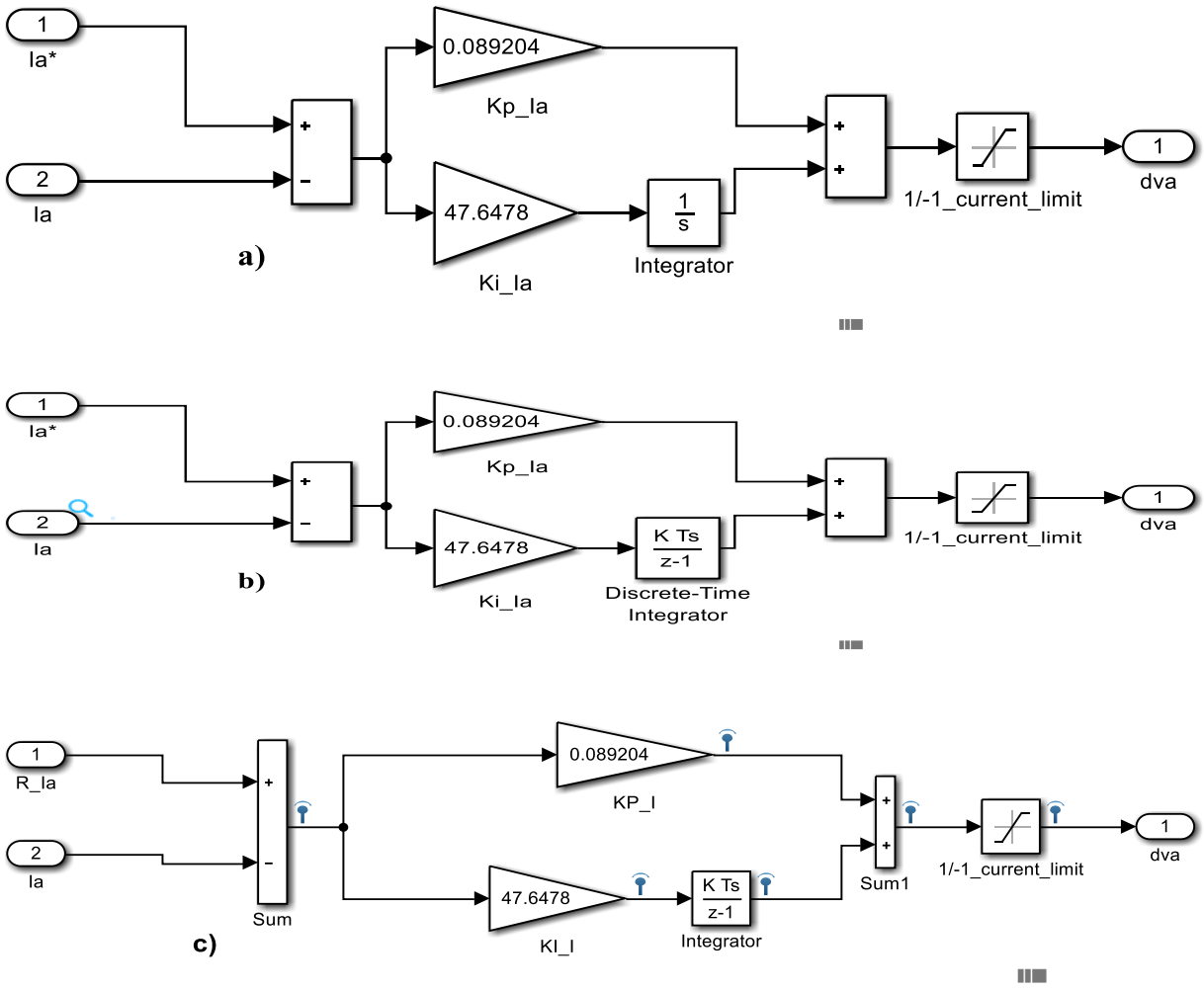
Continuous time PI controller of lift

Once the Simulink model is correctly designed by using the continuous time Simulink blocks, it will simplify the other further simulation in discrete time and HDL code generation enabled Simulink block with some arrangement and adjustment. The discrete time controller converted from continuous time by supplying very small sampling period (small Ts) can behave as the continuous time simulation. But, the problem is that it takes very high time duration for Simulink simulation to finish. Even if the FPGA devices have very high frequency (very low sampling period), its simulation using Simulink will last an hour or a day. So, using very low sampling period is not needed to see simulation result in Simulink.

Current Control Loop

The current control loop is the inner and faster loop of the controller. The current control loop calculates the difference between the reference current and the current being supplied to the motor

by the power processor (from ac to dc converted power) and H-bridge. The reference current value is obtained from the output of the speed controller and the value of the current being supplied to the motor is obtained by sampling the power processor (ac-dc converted power) and H-bridge output in real-time using an analog-to-digital converter. The reference current is the unity output of the speed controller being amplified to the operating range of the motor actual current. The current controller has the following structure.



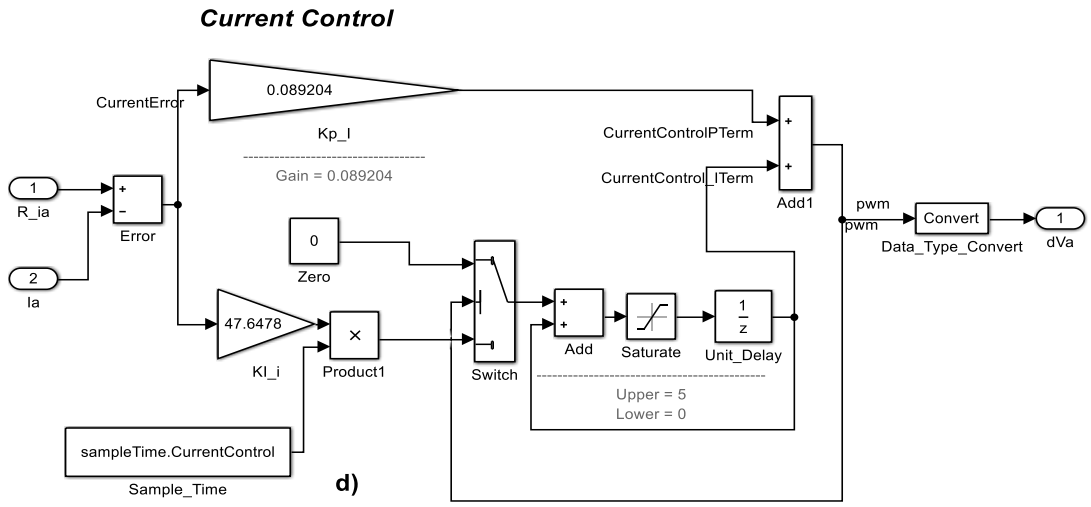
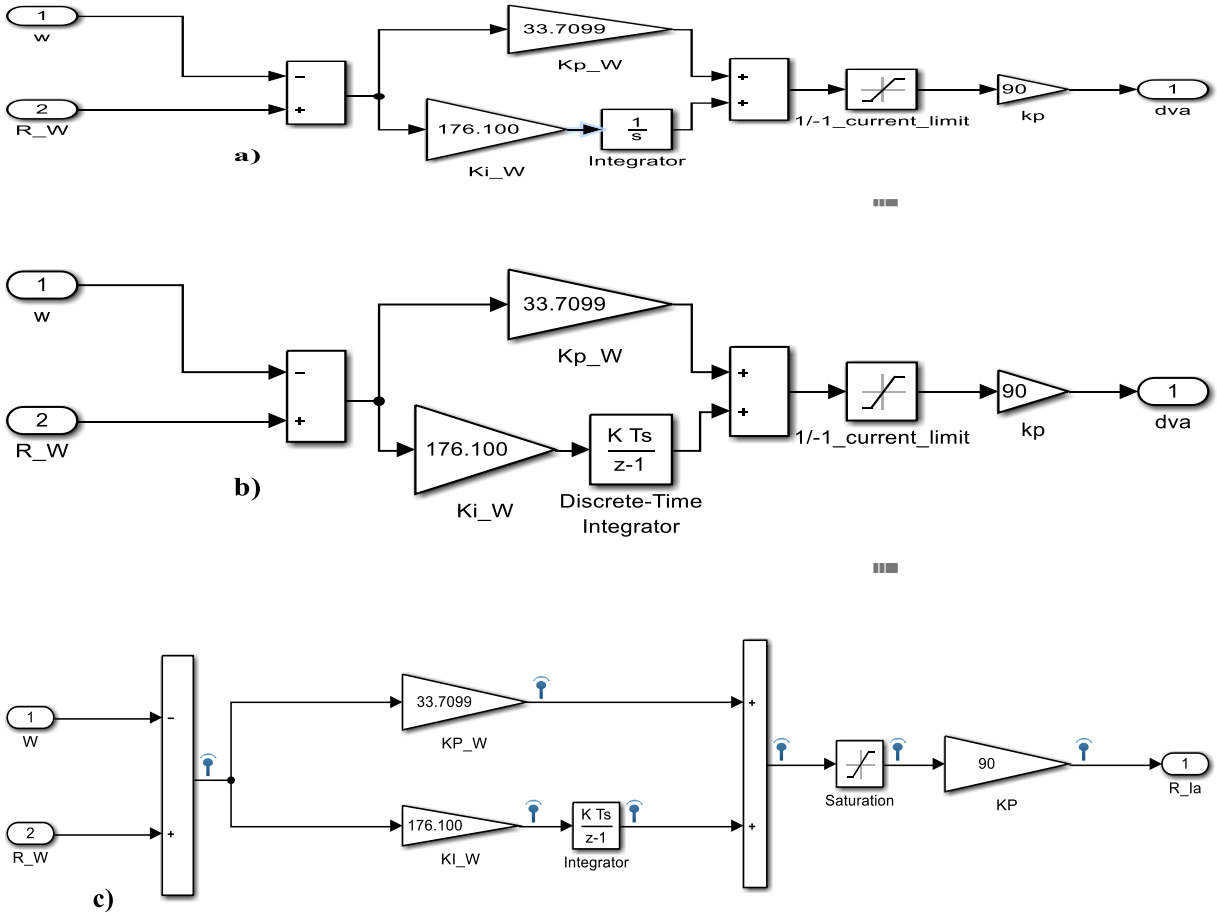


Fig 3 9 current controller,a) continous time b)discrete time c)fixed point converted from discrete time(unable for hdl code generation) & d)HDL code generation supported

Even if the **Fig 3 9** on c) and d) have fixed point single data precision, they do not result similar HDL code generation. Fig C) is the one converted from discrete time to fixed point data type by following "fixed point tool …" conversion process under the toolbar of "analysis" on Simulink model**(**explained in **Appendix B**) while **Fig 3 9 d**) contain some adjustment on the integral part with the usage of HDL coder Simulink block. Fig c) couldn't result desired output when whole simulation is running. It rather show delayed output. Hence, it doesn't trace the reference input. It also do not result HDL code generation. The figure on d) on the other hand, can solve the two mentioned problems, i.e. tracking ability and HDL code generation.

Speed Control Loop

Speed control loop dictates at what speed the electric drive should be turning. The speed control loop calculates the difference between the reference speed and the actual speed of the drive which is obtained from a sensor mounted on the motor's shaft. This calculated error in speed is manipulated by the speed loop's PI controller and amplified to produce reference current for further current control manipulation.


1
w
2
R_W
33.7099
Kp_W
176.100
Ki_W
1
s
Integrator
1/-1_current_limit
90
kp
1
dva
a)
1
w
2
R_W
33.7099
Kp_W
176.100
Ki_W
K Ts
z-1
Discrete-Time
Integrator
1/-1_current_limit
90
kp
1
dva
b)
1
W
2
R_W
33.7099
KP_W
176.100
KI_W
K Ts
z-1
Integrator
Saturation
90
KP
1
R_la
c)

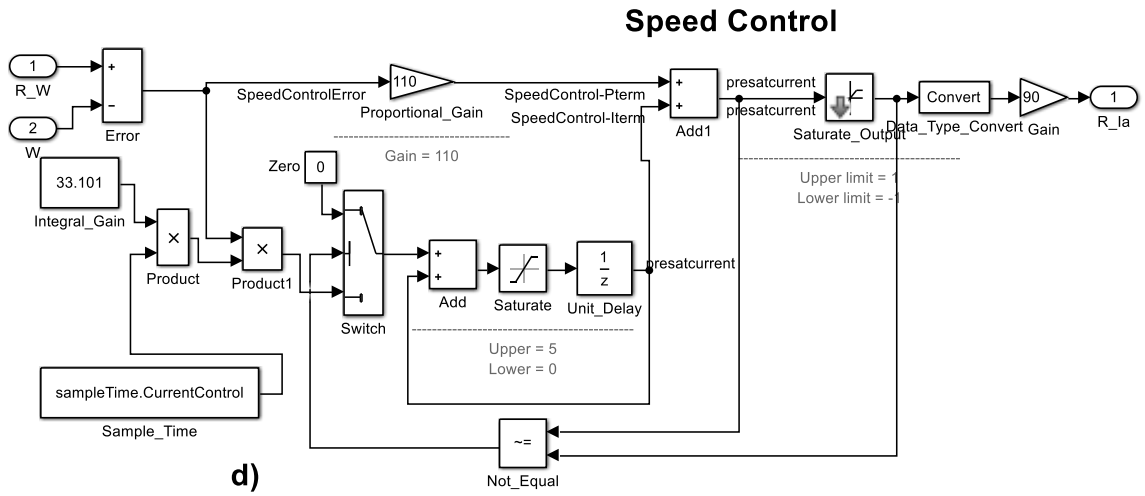


*Fig 3 10 Speed controller,**a**)discrete time **b**)fixed point converted from discrete time(unable for hdl code generation) & **c**)HDL code generation supported*

The speed controller has similar description as explained above. Here, in the same way as current controller, Fig 3 10 c) has less delay time than Fig 3 10 B) with its HDL code generation capability.

Position Control

The reference speed with its speed nature is constructed in order to result desired height measurement. In short, reference speed, being integrated over some duration, should result desired position. Each position is integrated with the triggering input (usually push button) in which it enables one of the speed reference to become active at the pressing of it. These can usually be effective by using the state flow programing of the Simulink part on computer visualization or HDL programing on FPGA devices. They need well organized skill on state flow programing.

Reference speed and position

There are two types of speed reference for lift design in lift design industry. These are "∫ \" shaped speed reference (shown on **Fig 3 11** below) and a trapezoidal speed reference. Having the motor Simulink model and load torque, the lift speed behaviors has been described for the load force interaction with the car surface during the ***three modes of speed reference*** movement.

Generally the reference speed has the following graphical features and modes.

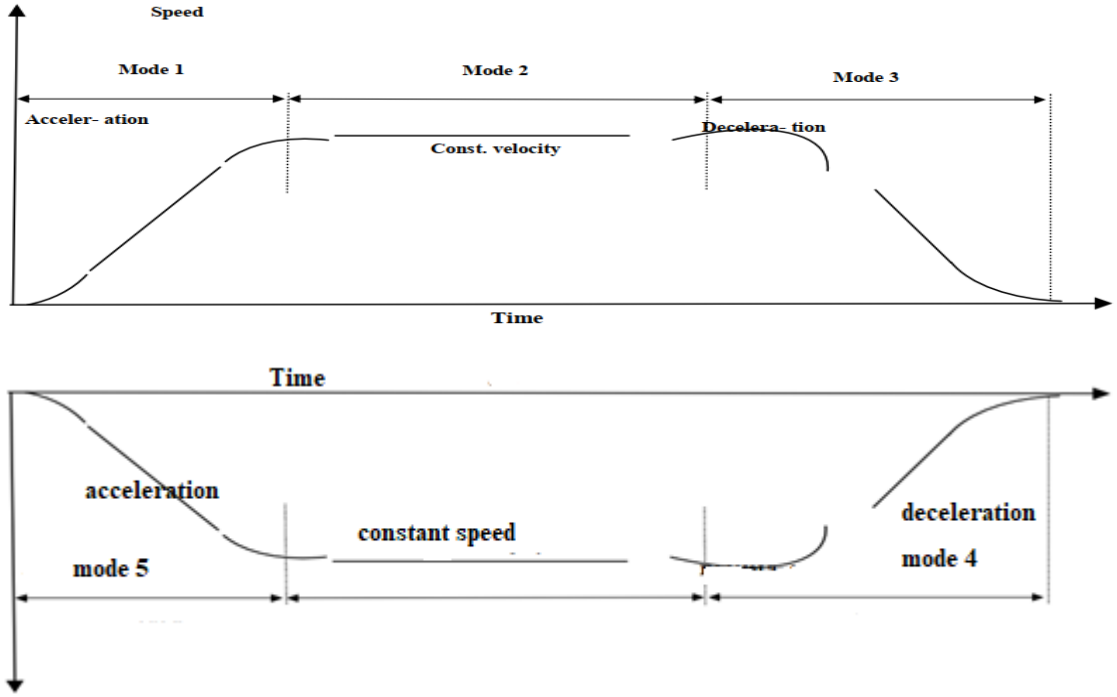


*Fig 3 11 Ideal (typical) lift reference speed (top: **upward**)&(bottom: **downward**)*

Load force effect – reference speed

When passengers enter into the car, they exert a weight/force on the surface of the car. The car surface exerts a normal force backward towards the passengers. The force the car surface exerts back to passengers and the weight that passengers exert on the surface is equal in magnitude but opposite in direction. In short action and reaction force takes place. But, when passengers and car experience motion/accelerates, there will be normal force changes.

Case 1: No acceleration of elevator

If the acceleration of the lift car is zero, then there are two possible scenarios; the lift can be at rest (stationary, zero velocity) or moving with a constant speed (no acceleration if velocity does not change). For this case, the force exerted by person is $-W = -mg$ (negative direction) and car surface force is $FN = +W = +mg$. The force is balanced by action and reaction force.

Case 2: going up & speeding up (acceleration a is positive (up))

The Normal Force is larger to resist person load force when the car is accelerating upward. This can be described by newton's 2nd Law ($\Sigma F=ma$). The overall acceleration of the person is upward

(with the lift car) and can be represented by $= -mg + FN$ i.e. $FN = mg + ma$, which is GREATER than the true weight.

Case 3: going up & slowing down (acceleration a is negative (down))

This similar to the case 2 given above but has directional difference. Here, the acceleration is replaced by deceleration. The whole equation can be represented by Newton's 2nd Law ($\Sigma F = ma = -ma = -mg + FN$). *FN* become $FN = mg - ma$, which is LESS than the true weight.

Case 4: going down & slowing down (acceleration a is positive (up))

This is obtained when negative speed profile is used as reference speed. The car and passenger coming from top position will enter to stopping condition when approaching to the desired destination. Before stopping, the car must enter to acceleration (opposite direction to negative/downward velocity to reduce velocity magnitude) to bring the constant speed down traveling car to zero speed. So, *ma* is positive (upward) with newton's 2nd law equation of $= -mg + FN$. $FN = mg + ma$, which is GREATER than the true weight.

Case 5: going down & speeding up (acceleration a is negative (down))

This the case when car and passenger speed up from rest to downward direction towards a lower floor. The lift acceleration of the lift is negative/downward (increasing the velocity magnitude in the downward direction). The overall acceleration of the person is downward (with the elevator). So *ma* is negative (downward) $-ma = -mg + FN$. (Note that this is the same equation as Case 3.) $FN = mg - ma$, which is LESS than the true weight.

Case 6: freefall (a = -g)

If the lift cable was to break, the whole elevator-scale-person system would all begin to accelerate downward due to the force of gravity. All objects in freefall accelerate downward with the same magnitude (acceleration due to gravity, g).

One can understand that a lift has three modes of operation during one upward movement from one floor to the other floor. These modes are the acceleration time period, a constant speed time and a deceleration time period as explained above in cases of 2, 1 and 3 respectively. It also has the same three modes of movement when lift car is moving down from top floor to the bottom floor as explained above on cases of 5, 1 &4 respectively.

One of the performance measurement of a lift is **jerk**. Jerk is the rate change of acceleration. It is the measure of comfort of the lift to passengers. A good lift has a minimum value or almost around zero units of m/s^3.

Although it has low level **jerk** problem in real world application, the trapezoidal speed reference is usually chosen because of its simplicity for the analysis of lift control. This paper has used the trapezoidal linear speed reference of **1.25m/s** and linear acceleration/deceleration $of\ \pm \mathbf{0.8}\boldsymbol{m/s^2}$.

The factory has five floors within the machine tower. Among the five floors of the factory, two floors have different length from the others. The three floors have similar length from each other (8 m) while the basement floor to ground floor is 4m and the first floor from ground floor is 12m high. So, the simulation is conducted by considering this issues.

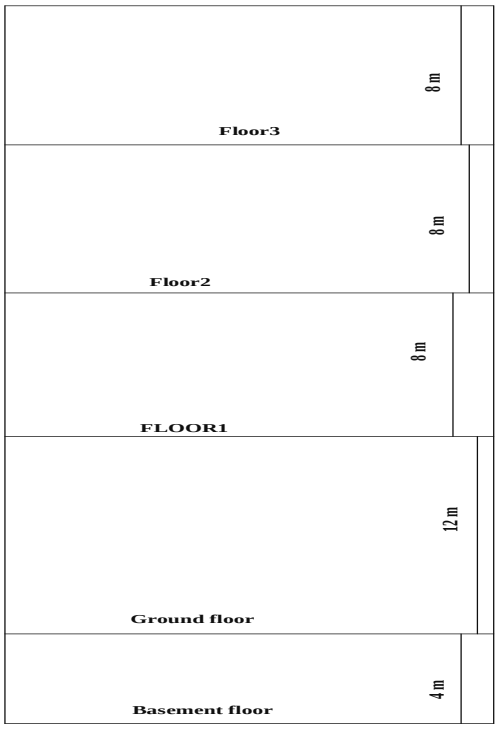


Fig 3 12 floor levels

The reference speed input which desires to bring the lift car to the 3rd floor from basement must differ from the speed reference input, which has the aim to move from basement to ground floor. This difference is not shown on accelerating speed of all speed references rather it is shown on the duration of the constant speed. Different calling input has different reference speed input with

respect to car actual position. If the car is on 3rd floor and there is no other active call beneath of it, it has to move directly towards the called destination without having stops on under floors. Having these in mind, there are several different constant speed duration of speed reference input based on the car actual position and destination. The speed profile of reference speed is generated based on the distances among stories that the lift car must move to reach on the desired floor. The table given below shows the height that lift must move in either direction.

Table 3 2 speed profile based on distance between floors

Floor/Floor	Basement(B)	G	F1	F2	F3
Basement(B)	-	4m	16m	24m	32m
Ground(G)	-4m	-	12m	20m	28m
Floor1(F1)	-16m	-12m	-	8m	16m
Floor2(F2)	-24m	-20m	-8m	-	8m
Floor3(F3)	-32m	-28m	-16m	-8m	-

When a triggering input from each floor is pressed, the **speed pattern/profile** is selected and status of the push button become active until it is canceled by the coming of car to the destination. Based on the constant speed duration that it takes when moving up and down a speed profile is created for different floors.

The difficulty of FPGA connection to Simulink model

As already explained above, FPGA HDL code block containing subsystem is connected to the whole continuous time Simulink model by using 'model' block on the Simulink platform as arranged on Fig 4 2. The model contains two types of solver, one for controller and the other for the whole system. The controller subsystem has the FPGA compatible HDL blocks while the other subsystem has the continuous time solver.

Although real time function of FPGA based device have high speed response, its simulation on Simulink software takes more time than the continuous time simulation. It will spend more than an hour. The simulation response will require more time if the Simulink model contains simscape power systems blocks. Therefore, it will be difficult to show the simulation result using MATLAB Simulink simscape power system blocks. That is the reason why the mathematical modeling of permanent magnet dc motor other than simscape power blocks has been preferred on the next simulation discussed.

CHAPTER FOUR

RESULT AND DISCUSSION

This chapter presents different simulation and their respective functional usage and results. The simulation results has been done using the software of MATLAB Simulink, Xilinx state CAD and the Simulink state flow respectively. These software have been used for their respective purposes and advantages described below.

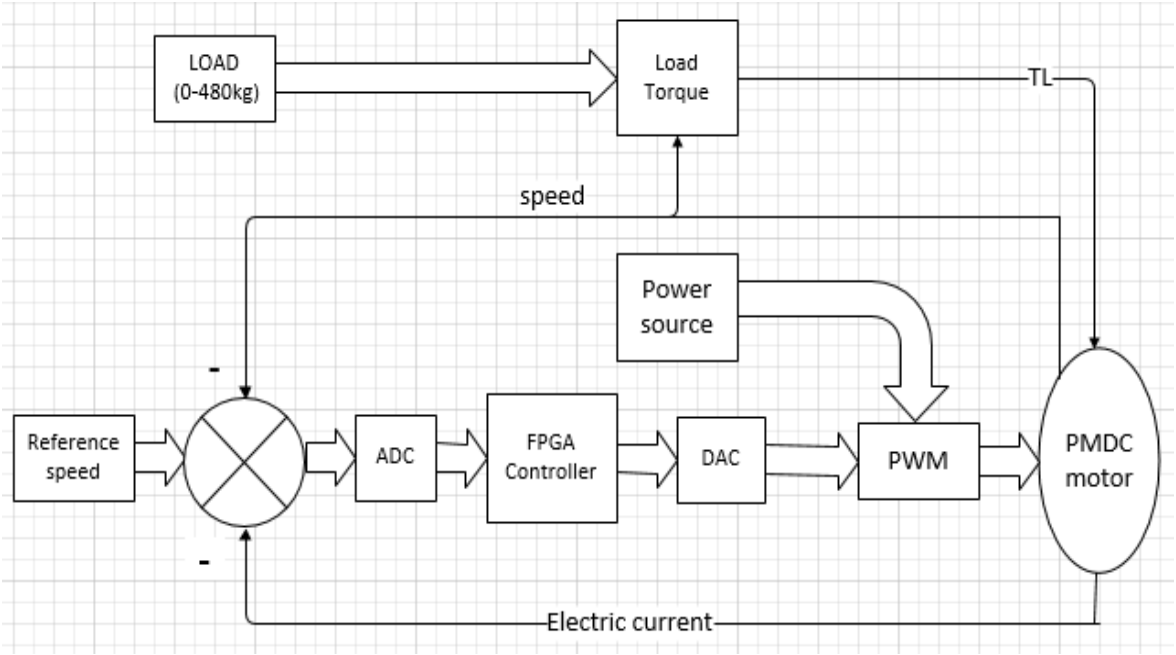


Fig 4 1 block diagram of lift

The mathematical modeling equation of the elevator/lift derived/discussed in chapter three (3) was simulated by using Simulink library blocks. Simulink is chosen due to the following its feature and advantages.

Simulink® is a block diagram environment for multi domain simulation and Model-Based Design. It supports system-level design, simulation, automatic code generation, and continuous test and verification of embedded systems. Simulink provides a graphical editor, customizable block libraries, and solvers for modeling and simulating dynamic systems. It is integrated with

MATLAB® for incorporating MATLAB algorithms into models and export simulation results to MATLAB for further analysis. It is also a graphical environment for visualizing real time simulation. Simulink simulates analogue systems and discrete digital systems. It also gives the opportunity to transform other programming languages.

4.1 Lift design and control analysis through Simulink

Real world design and control needs clear knowledge of parameter specification of permanent magnet motor and other related components to which the design is concerned.

After selecting the specification of permanent magnet dc motor (torque output, power specification and current rating) in relation with desired load capacity, it is essential to determine the dynamic and static parameters of the motor. According to [29] of chapter 3, the steady state parameters of motors are armature resistance, static frictional torque, motor constant (k) and motor friction coefficient (B) while the dynamic parameters are armature inductance(La) and motor rotor moment of inertia(Jm). These parameters of the motor plays a vital role in the control and design of it. Those all PMDC motor parameters can be found by using laboratory experiment as stated in [29].

Selecting reasonable value for PMDC motor with approaching values as that of [29], the simulation has been done using the control analysis of MATLAB Simulink. For real time implementation, there may be some changes on the controller parameters due to deviation of parameters of the real motor with this modeled motor parameters.

The below Simulink model has the objective of illustrating the speed response of the lift with respect to the reference speed graphically. As a means of observing model performance, lift speed versus reference speed is plotted. This plot gives visual evaluation of how the response is tracking the reference.

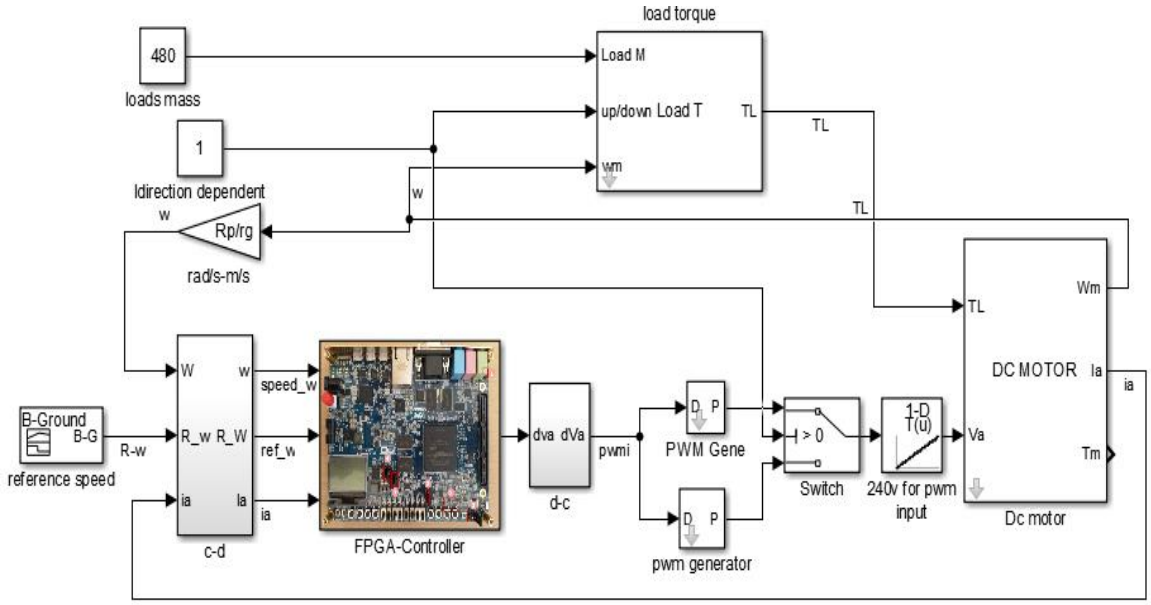


Fig 4 2 Simulink model of lift electric drive with speed & position control

4.1.1 Simulink Subsystem construction

Dc motor (PMDC motor)

Before looking the simulation result, the constructional insider subsystem features of the above figure will be given below.

By using equations of (3.22), (3.23), (3.24) and (3.28), the PMDC motor can be represented by Simulink diagram of fig 4.2 below.

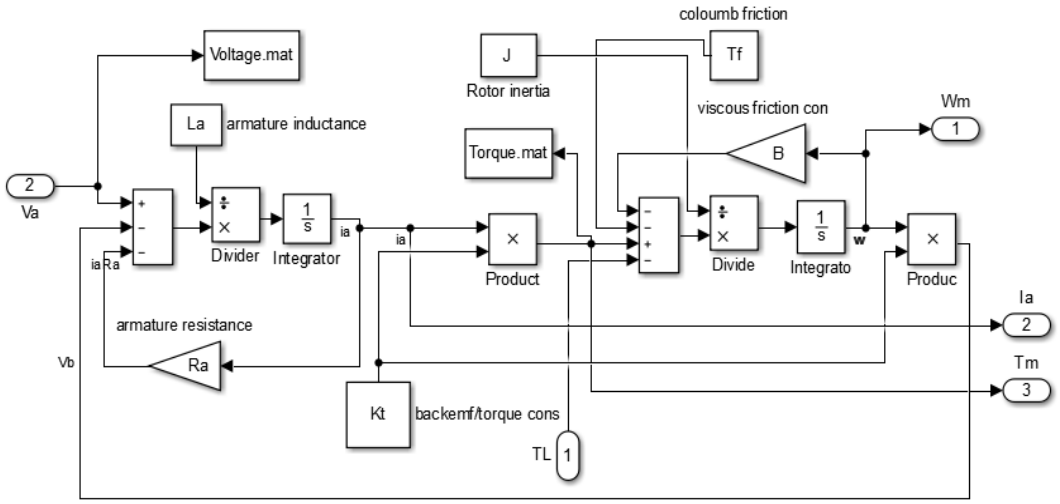


Fig 4 3 PMDC motor Simulink model

Load torque (TL)

The load torque placed on the drive is a function of the drive's speed and the masses of the counterweight load, car load and the load placed in the lift car. According to **chapter 3** of **Eqn. 3.34a** and **Eqn. 3.34b**, the equation of load torque for the upwards motion is slightly different from that of the downwards motion.

For the upward motion, the car-side mass is added to the total load torque as a function of increased car mass and increased load inertia. For the downward motion, the car-side mass detracts from the total load torque and the load's inertia is not considered as only the counter-weight's inertia. These similar yet different subsystems are illustrated in **Fig 4 4**.

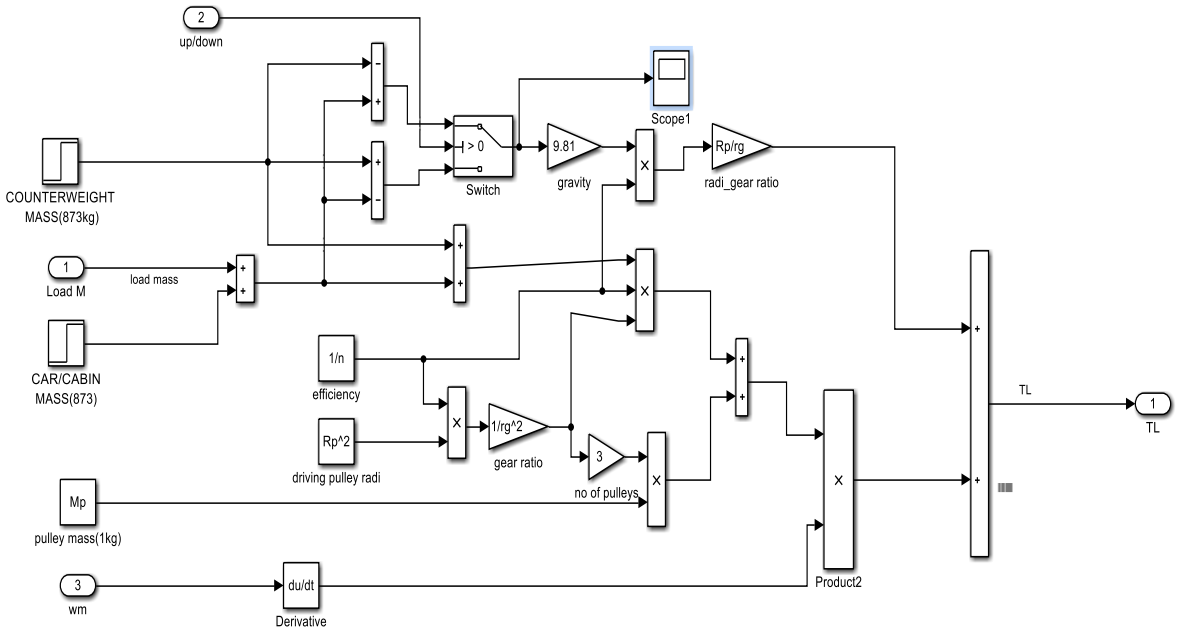


Fig 4 4 Load torque subsystem for ascending (up+) &descending (down-) lift car

FPGA controller

The current and speed control loops each are contained within their own respective subsystem. Each control subsystem calculates the error between the reference value and the actual value. Each control subsystem also contains the appropriate proportional and integral gains in addition to a saturation block which accounts for physical limitations such as the maximum speed at which the lift can safely operate. Since the controller is needed to have fixed point data precision for HDL code generation, its parameter configuration and solver is totally different from the whole (overall) simulation. It's all blocks are HDL code generation supported blocks with fixed step and discrete solver.

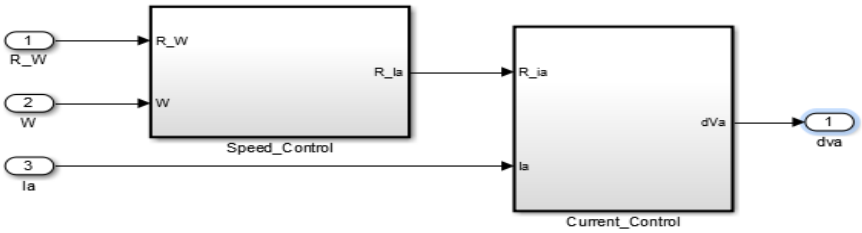


*Fig 4 5 FPGA PI speed and current control(cascade version of **Fig 3 7** & **Fig 3 8**)*

ADC and DAC

On **Fig 4 2**, there is a subsystem at the entrance and leaving of the model of the controller. The subsystem at the input side of the controller namely "c-d" is used to convert continuous time to discrete time fixed data type. The output of the controller is also converted from discrete to continuous by "d-c" subsystem block. They have an internal structure as given below.

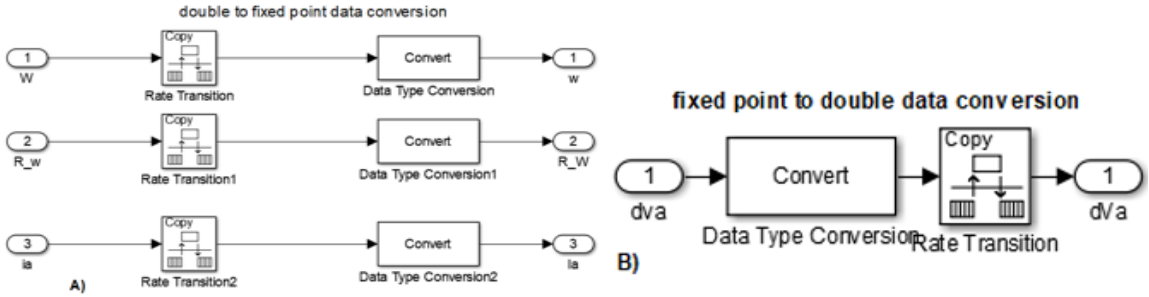


Fig 4 6 data type converter a)continous(double) to disccrete(fixed point) data type b)discrete-to-continous data type.

Direction wise PMW generator

The direction controller block is connected to load and PWM generator selecting switch. The block is positive for upward movement and negative for downward movement of lift car. The negative block multiplication is added to the PWM for downward travel of the car as shown below.

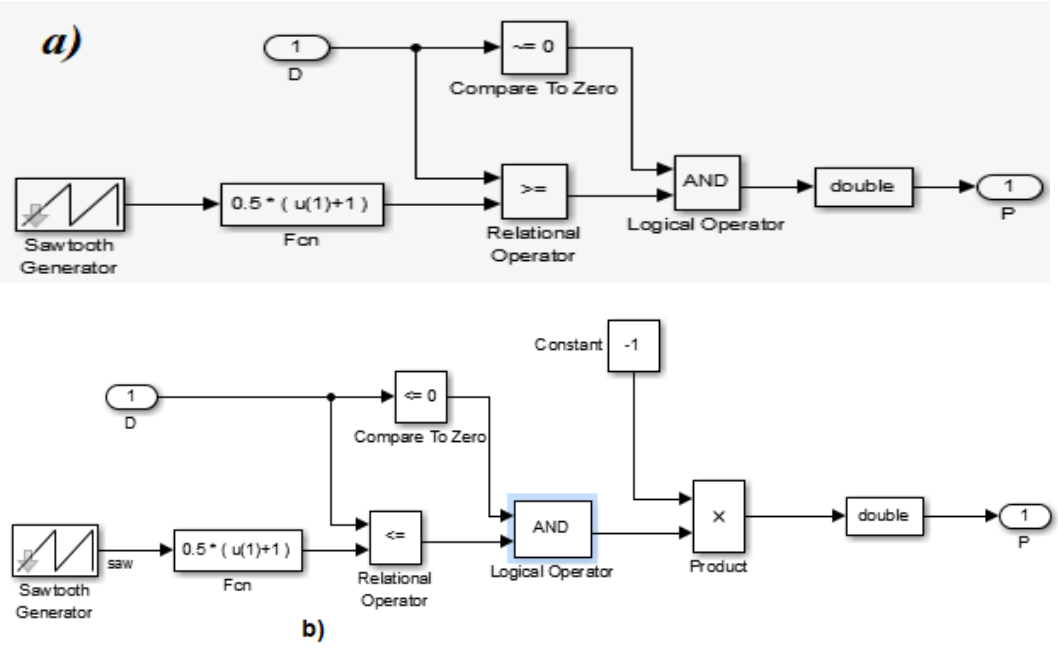


*Fig 4 7 PWM generator for **a) upward** & **b)downward** direction*

Fig 4 2 has also a lookup table which switches on/off based on the controlled output of the PWM. The output of PWM can be shown by the following figure.

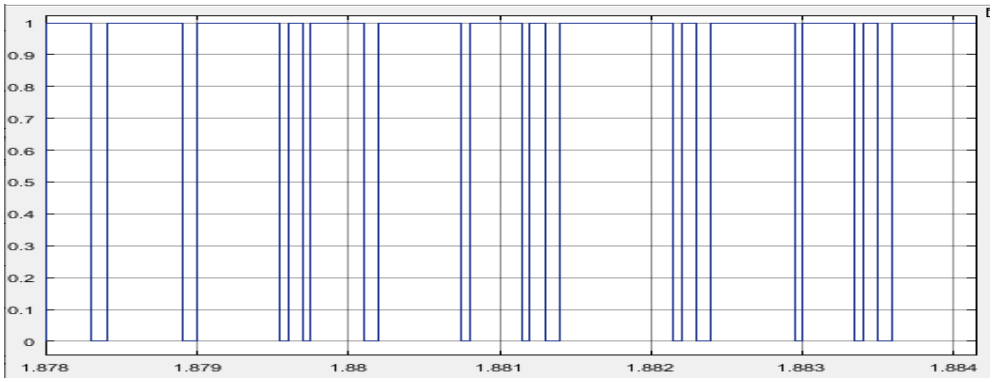


Fig 4 8 PWM output (snipped from Simulink scope) for moving up

Dc voltage for motor

PWM output result can be connected with the 240v dc source using **lookup table** or "product" Simulink block. In real world application, the converted (ac to dc) voltage is supplied to the motor using H-bridge circuit. Below is the figure representing PWM output relation with dc voltage source.

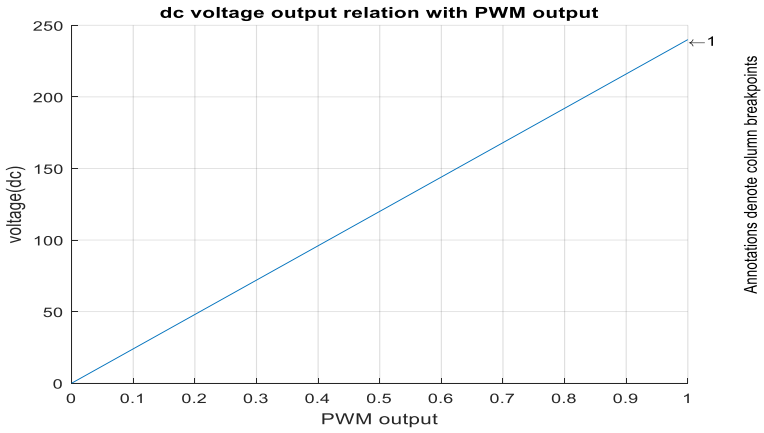


Fig 4 9 PWM output relation with 240v source for the motor operation

4.1.2 Reference speed/position and simulation results

Using floor length among floors listed in **Table 3 2**, the reference speed and their response will be given below.

Upward movement of car with no-load and full-load

The car must move 4m height from basement floor to reach on ground floor. As already mentioned above, the reference speed being integrated must result the reference position i.e. 4m. In either no-load or full-load case, the output height of the design should be the same. The reference speed, its speed, position acceleration output is given below.

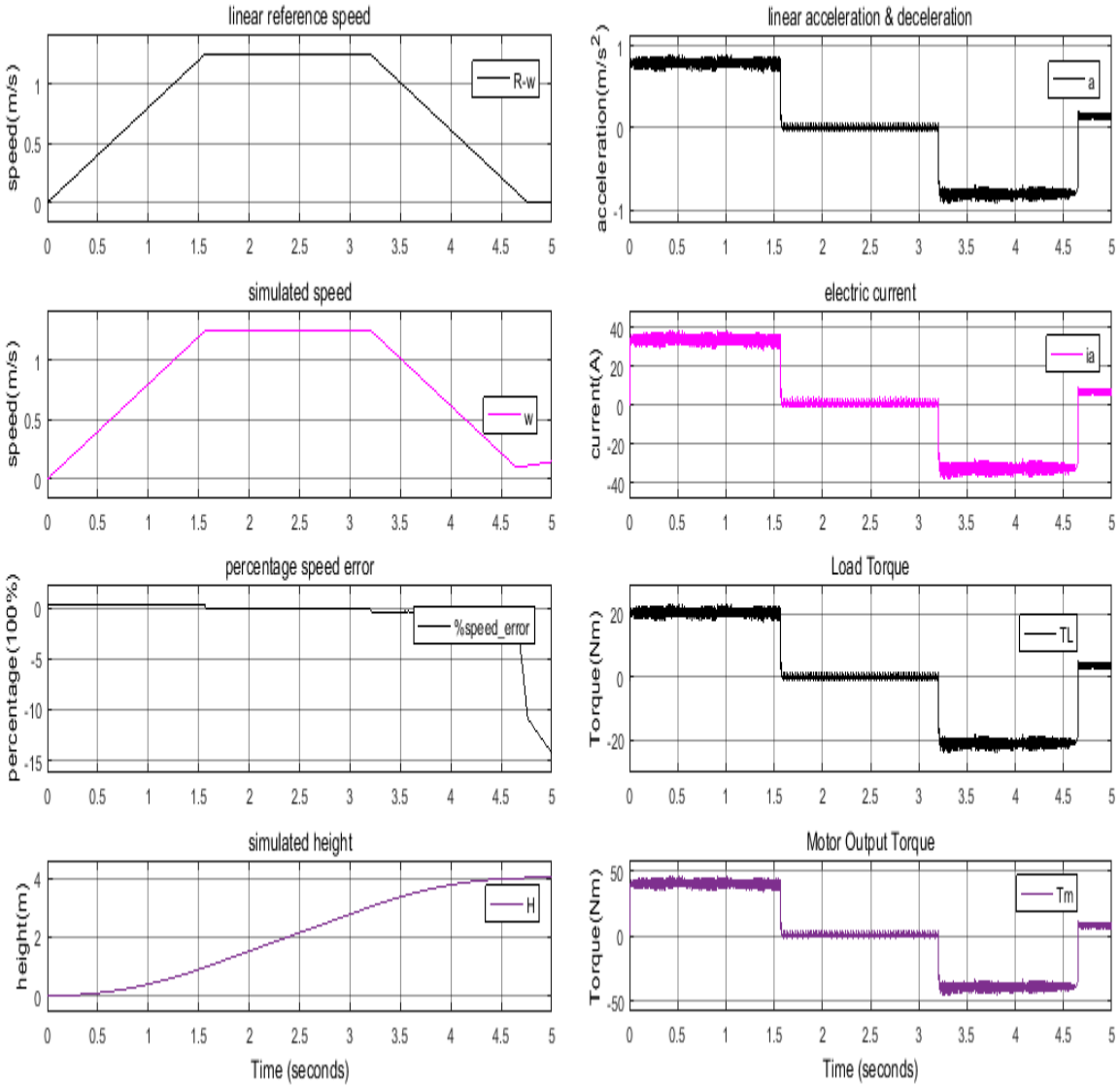


Fig 4 10 No-load speed characteristics for 4m upward movement

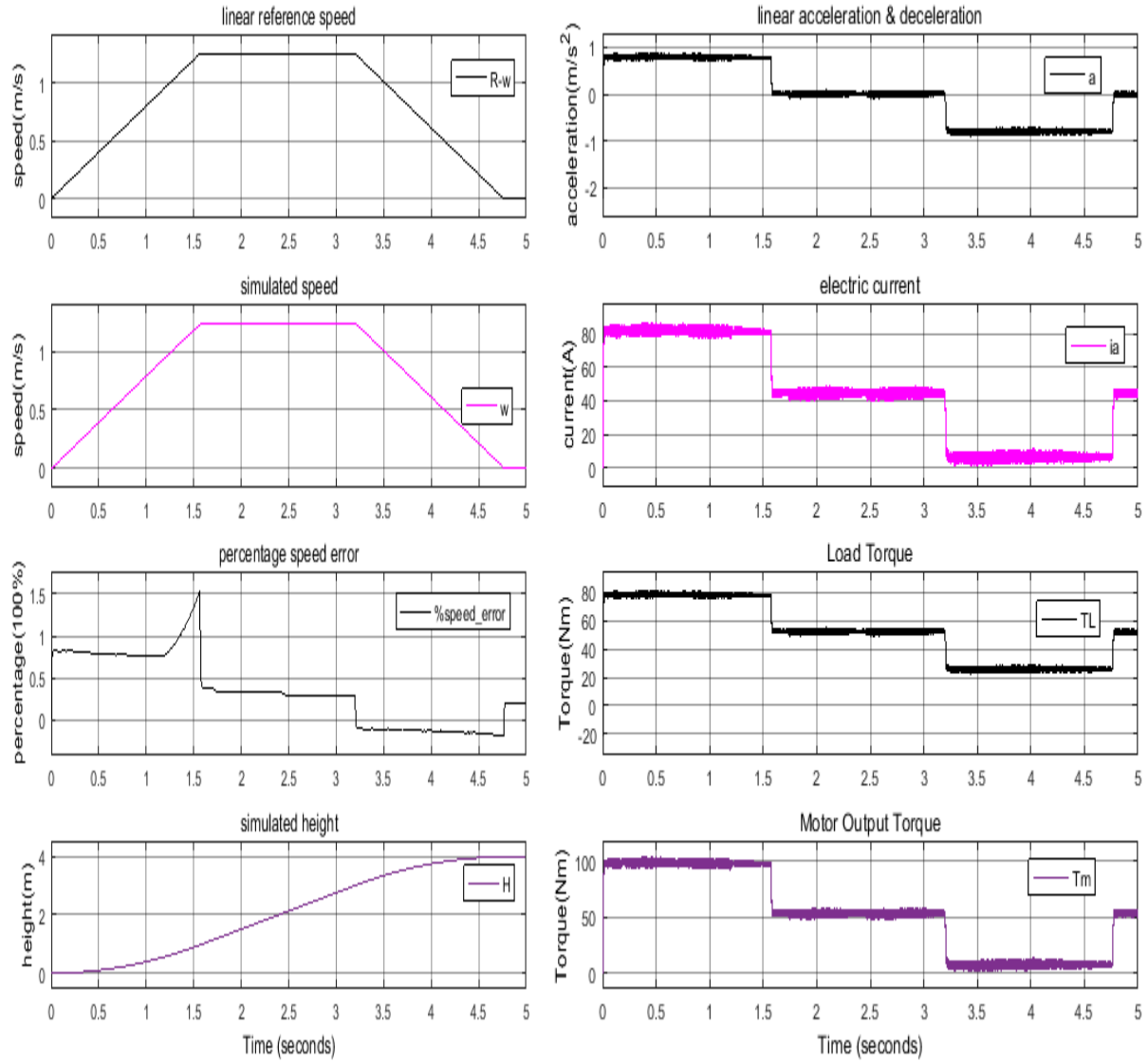


Fig 4 11 Full-load(480kg) motion characteristics for 4m height

The above **Fig 4 11 Full-load(480kg) motion characteristics for 4m height** has the objective of bringing the car from basement floor to the ground. What will be the reference speed for other floor levels? The answer has been clearly defined by **Table 3 2** on the previous chapter.

Consider another example which show the simulation result for the lift car to move from **basement** to **floor 1** by drawing the reference speed as shown below i.e. (4m+12m=16m).

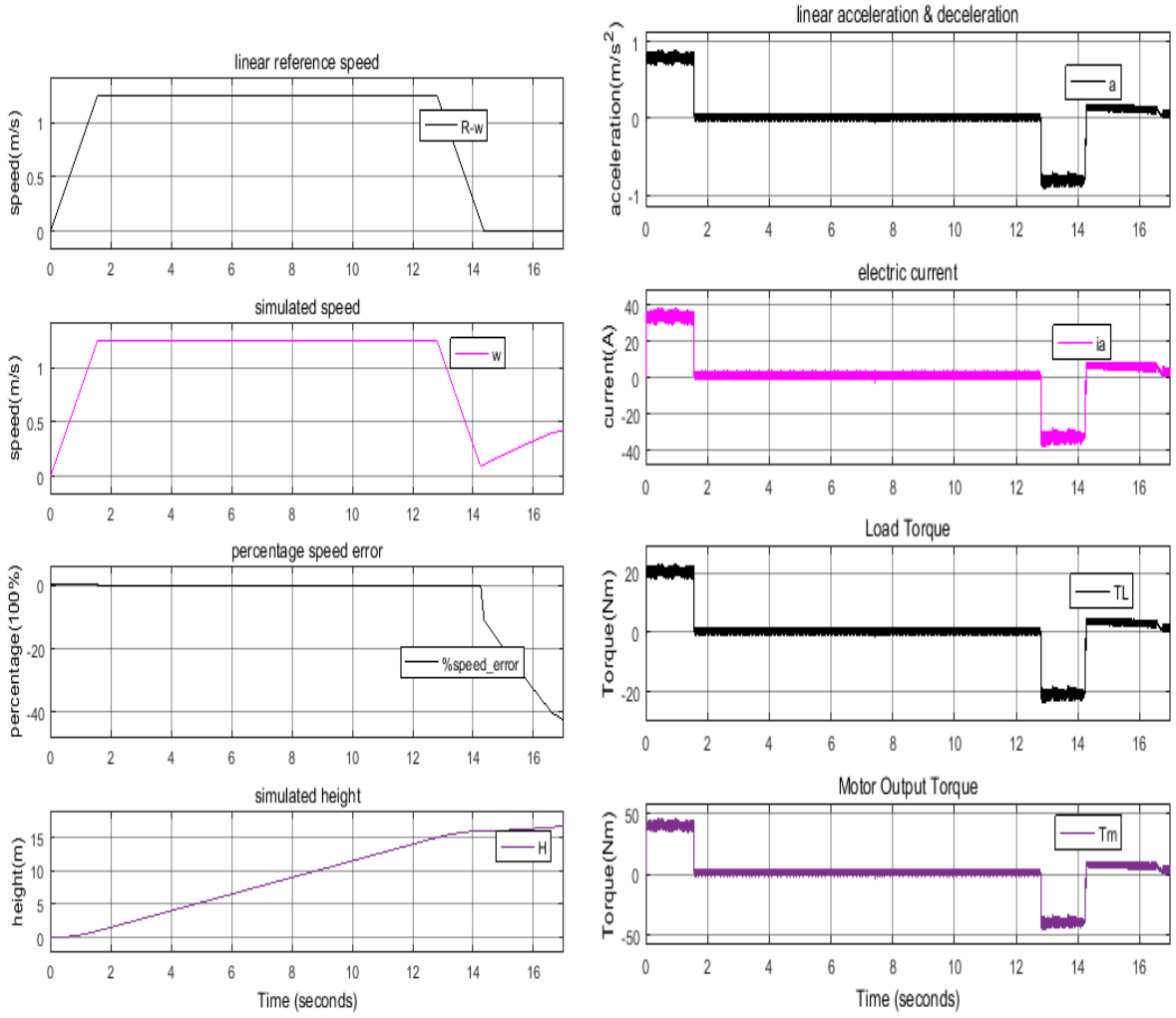


Fig 4 12 no-load 16m upward motion characteristics

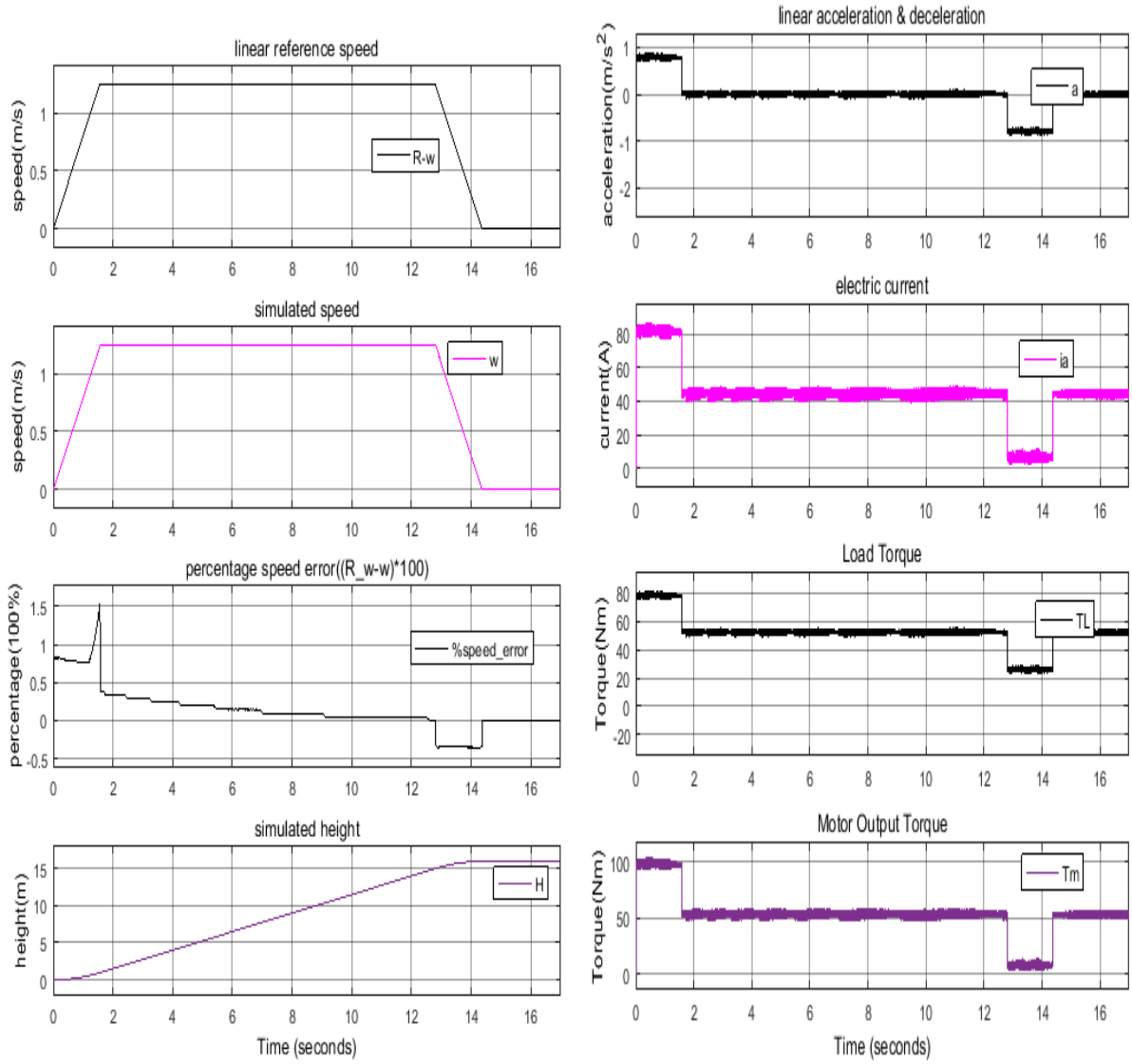


Fig 4 13 full-load 16m motion characteristics

Downward movement of car with no-load and full-load

The car must move 4m down from ground floor to reach on basement floor. As already mentioned, the reference speed being integrated must result the reference position i.e. -4m. In either no-load or full-load case, the output position measurement of the design should be the same. The reference speed, its speed, its speed error percentage and position output is given below.

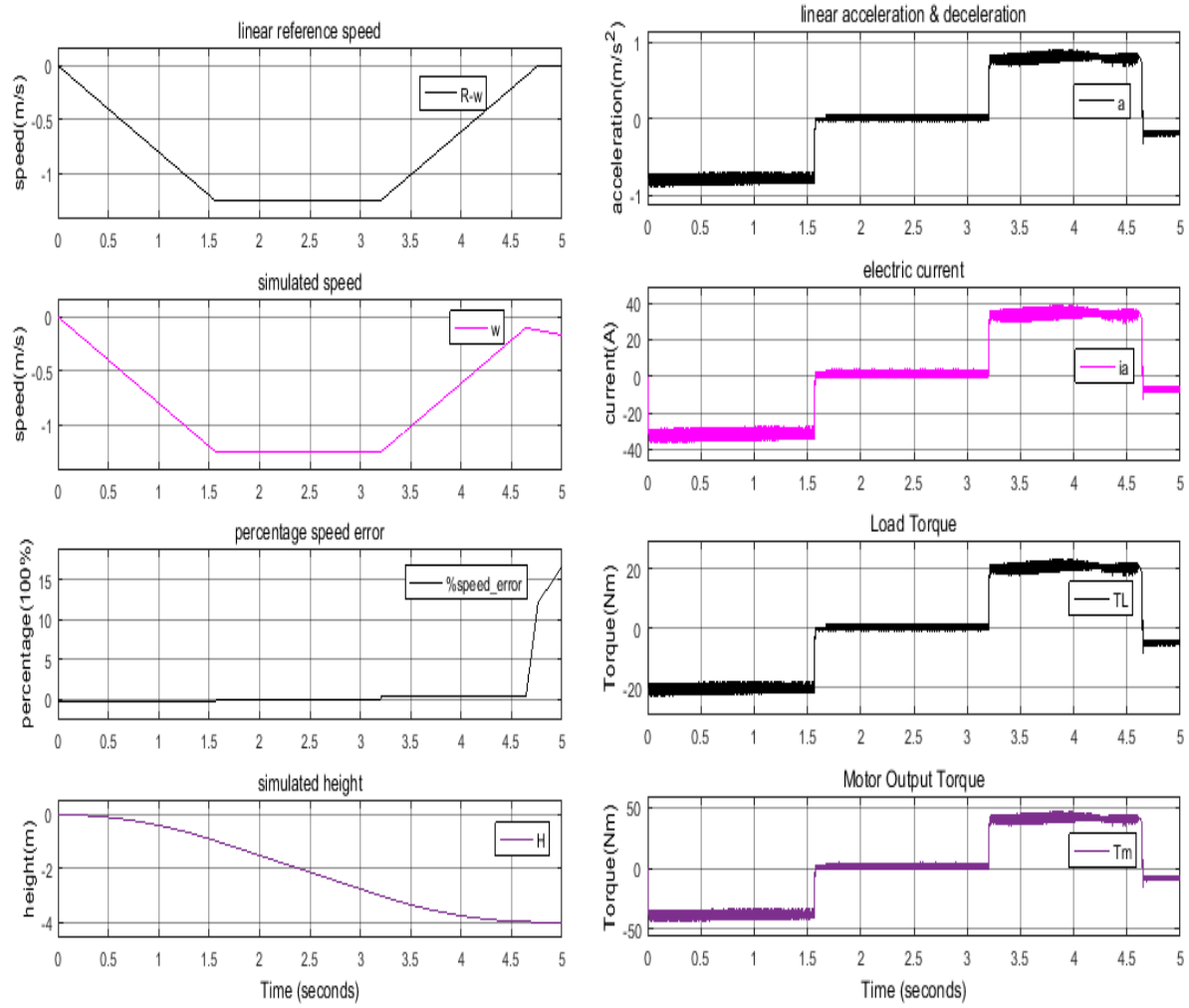


Fig 4 14 no-load motion characteristics for 4m downward movement

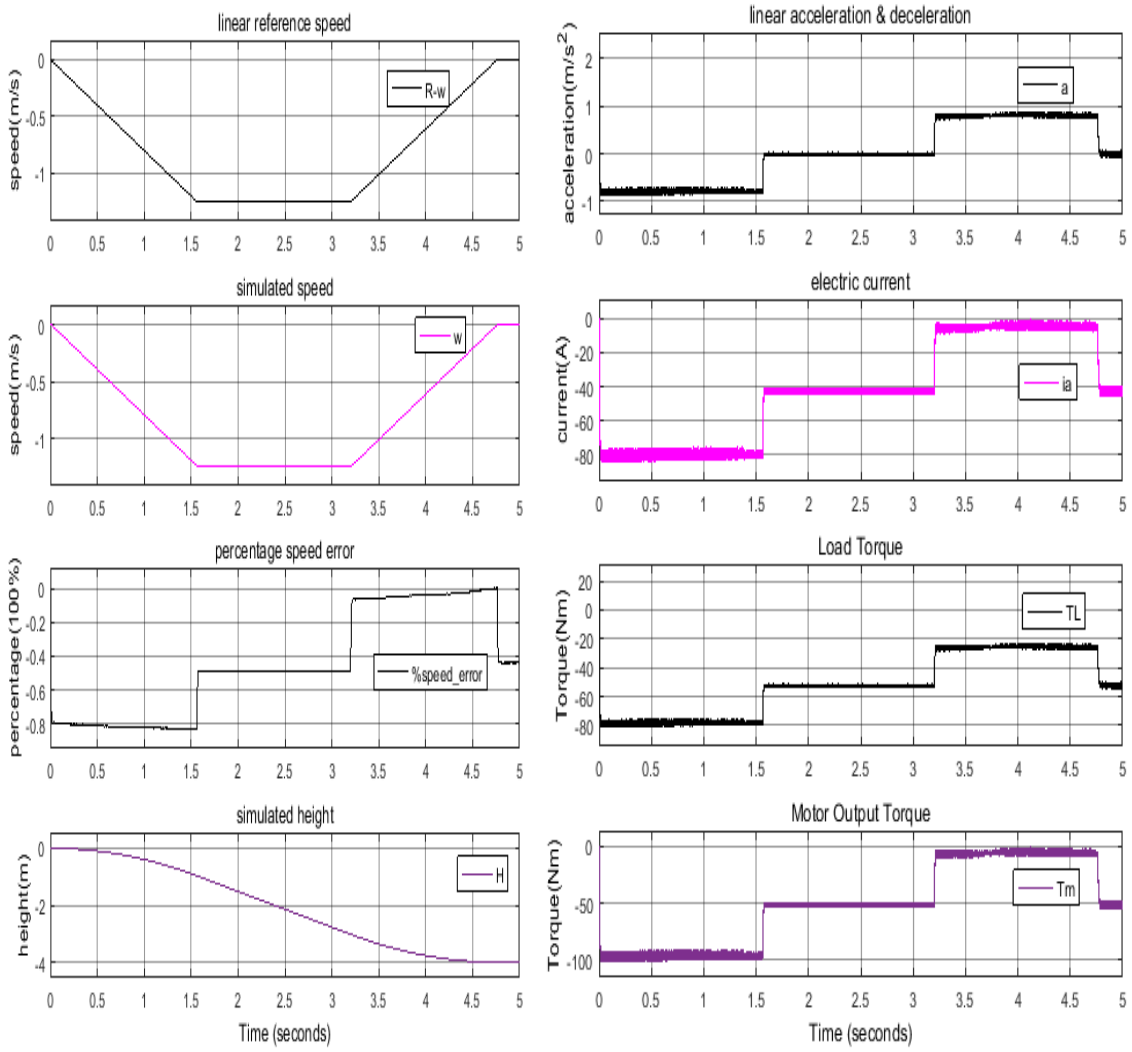


Fig 4 15 full-load -4m movement motion characteristics

Note that the period of time the lift car takes for acceleration and deceleration is similar/same for all speed reference input. But, the duration of the constant speed is different based on the height between the floors. Consider and compare the above figure with 4m height at **Fig 4 11** and lift car moving from **basement** to **floor 1 (Fig 4 13)** (4m+12m=16m shown above).

It is clear that the acceleration time (0-1.56sec) for both figure is the same. The time difference between constant speed and zero speed or the deceleration time is also similar (4.76-3.2=1.56sec for 4m height and 14.36-12.8=1.56sec for 16m height).

The simulation result of acceleration, current and torque graph is not smooth Due to the pulse width modulation (PWM) and controller discrete time feature as shown above on all figures.

From the above figures it is observed that the lift is able to transport six persons load (480kg) and empty car within the same duration of time to a height of 4 meter, 16m and -4m. One can understand the differences in current, torque and supplied voltages for these load variation. The simulation results show that as the load increases the voltage supplied to the motor should increase.

4.2 State CAD results of Xilinx

The second simulation is the Xilinx ISE state CAD result of the test bench, which is one method of expressing FSM. The lift control system is basically finite state machine (FSM). FSM is a digital sequential circuit that consists of different defined states that are controlled by inputs. The given lift control system is based on **Mealy machine** because outputs is generated by using current state and the input variables.

A state diagram is simply used to explain the state machine automation graphically. State CAD is used to implement state diagram of lift control system, which is a graphical tool that express ideas of state machines, as state diagrams. After making state diagram, State CAD simplify the process of converting the state diagram into Hardware Description Language (HDL) that can be used as source file in Xilinx Project and changed into schematic symbol. Xilinx State CAD also include State Bench which is used to show results as waveform simulation.

Here, the state CAD analysis, which has a relation with the human interaction for real world lift application of the five floor lift is presented. The test bench can show useful output response of the lift for different push button from different floors as the desire of human.

The Xilinx software state CAD for the five floor building is drawn as shown below.

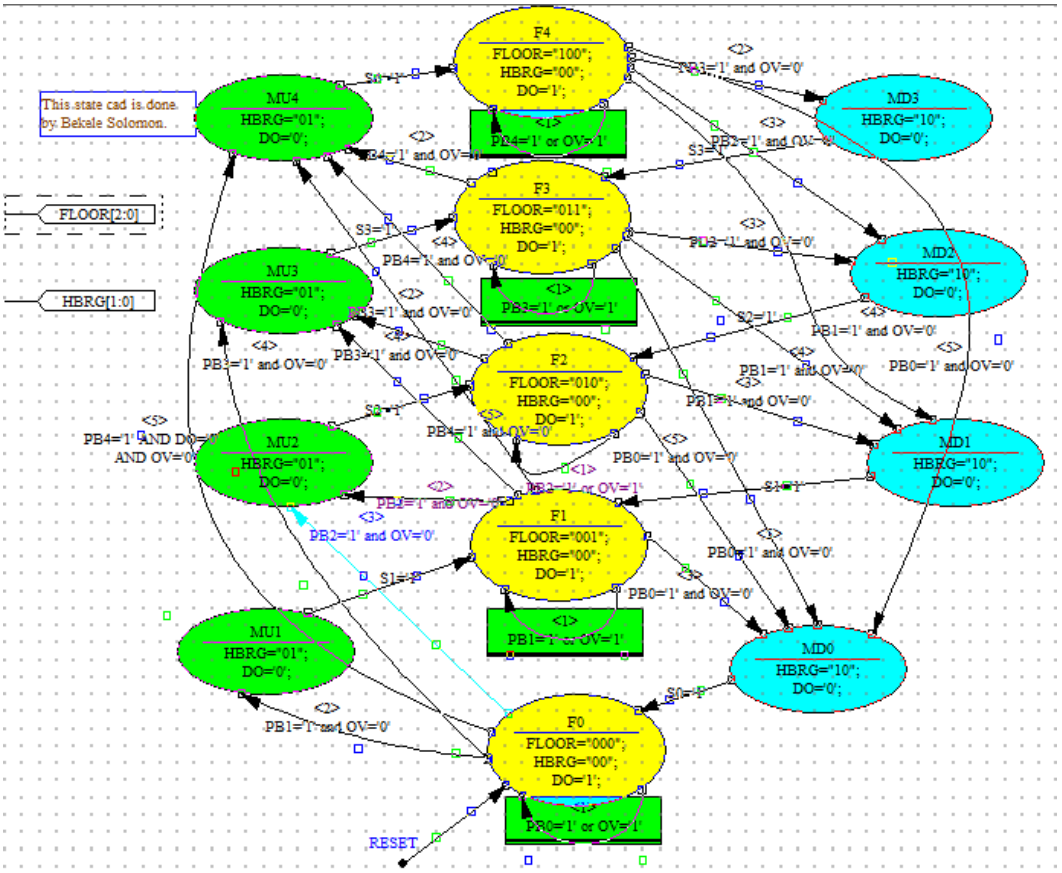


Fig 4 16 design and implementation of efficient lift control system using FPGA [5]

For this simulation analysis, Spartan 3E has been used for this drawing.

There are THIRTEEN states for FIVE level lift control systems. To control the lift system on each floor, input request are provided and push buttons are used for these requests, as shown in **Fig 4 16**. Description of lift control system is specified in **Table 4 1.**

- Three states F0, F1, F2, F3 and F4 represent the floors.
- Four states MU1, MU2, MU3 and MU4 represents the state for which motor rotates in upward direction.
- At MU1, MU2, MU3 and MU4 states output HBRG=”01” is generated which is attached to H-bridge to control the direction of motor.

- Four states MD0, MD1, MD2 &MD3 represent the state for which motor rotates in downward direction.
- At MD0, MD1, MD2 &MD3 states output HBRG=”10” is generated which is attached to H-bridge to control the direction of motor.
- The transitions PB0, PB1, PB2, PB3 and PB4 are inputs connected to Push buttons.
- The transitions S0, S1, S2, S3 and S4 are inputs which are connected to sensors.

The simulation blink can show some information of the state CAD at its compilation. It show the 394 lines of VHDL generated code, 13 states and 47 transitions as shown on figure given below. The generated VHDL code is shown in **APPENDIX D**.

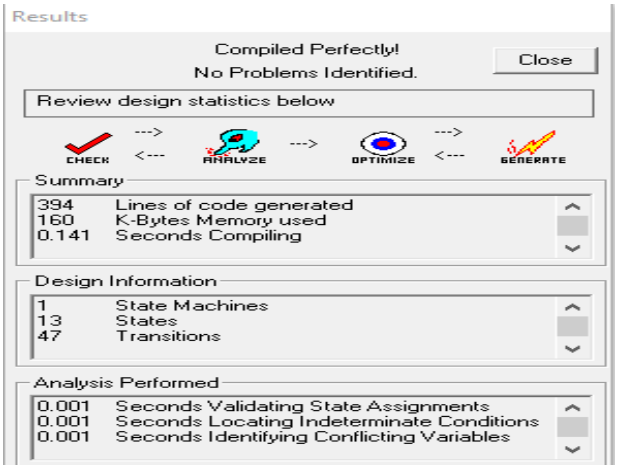


Fig 4 17 resource usage Feature of the stateCAD program

The inputs of the lift control system can be changed any time in State Bench simulator to see the simulation for these inputs. A state can be changed in simulator and the simulator will adjust the inputs automatically.

Table 4 1 lift sensor and states

State Name	Description
Floor States	F0, F1, F2,F3 and F4
Moving States (Moving Motor Up-ward) (Moving Motor Down-ward)	MU1, MU2, MU3 & MU4 MD0, MD1,MD2 & MD3

Input Transition (Push Button) (Sensor) Overload sensor	PB0, PB1, PB2, PB3 & PB4 S0, S1,S2,S3 & S4 Ov
Outputs	DOOR OPEN (DO) HBRG[0:1] FLOOR[0:2]

Simulation for different positions

At state F0, when PB0 is pressed, the state remains same using PB0 transition as shown in **Fig 4 18.**

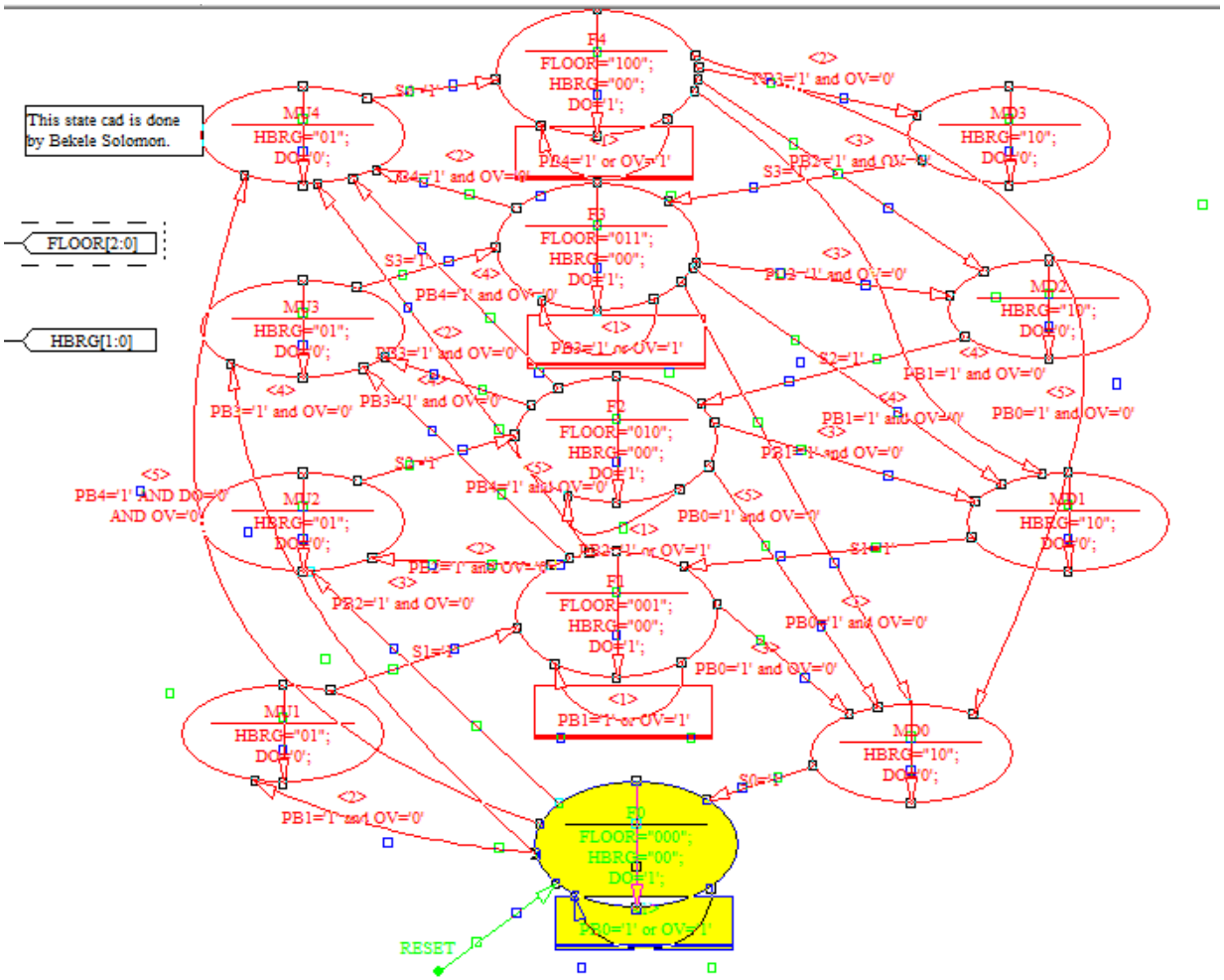


Fig 4 18 floor0 activation during po pressing

When PB1 is pressed the state changes into MU1 state as shown in **Fig 4 20** and output HBRG=”01” is generated and motor starts to rotate upward. When sensor S1 is detected due to transition S1 at that state, state will change to F1 state as shown in **Fig 4 21**. Output HBRG=”00” is generated which stops the motor and one instruction is completed. At F1, When PB3 is pressed the state changes into MU2, output HBRG=”01” is generated and motor starts to rotate upward. When sensor S2 is detected due to transition S2 at that state, state will change to F2. Output HBRG=”00” is generated which stops the motor and one instruction is completed. At state F3, F4 the instructions for inputs PB3, PB4 will executed in the same manner.

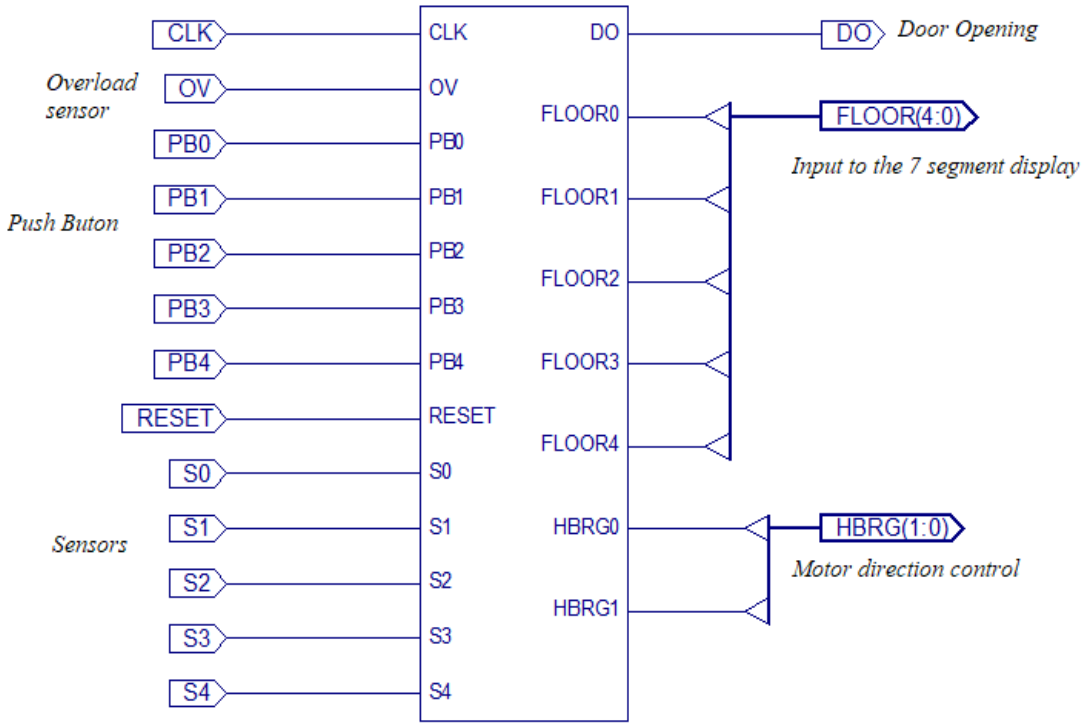


Fig 4 19 the RTL Schematic diagram

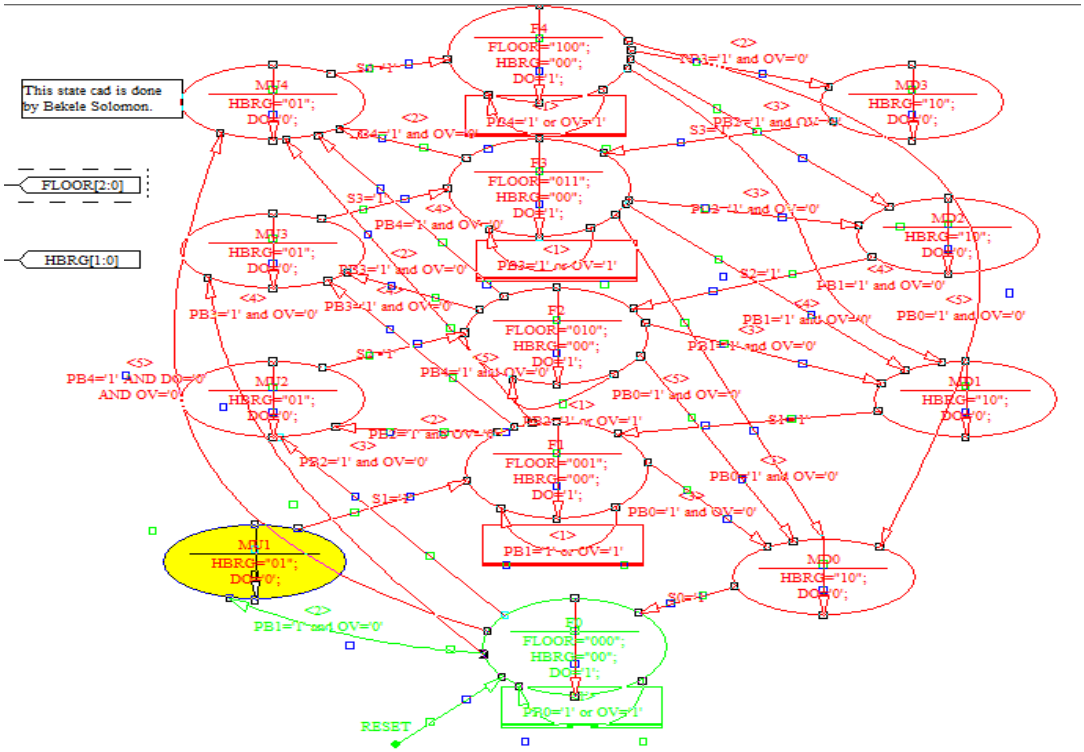


Fig 4 20 lift status when push button 1(p1) and MU1 is active

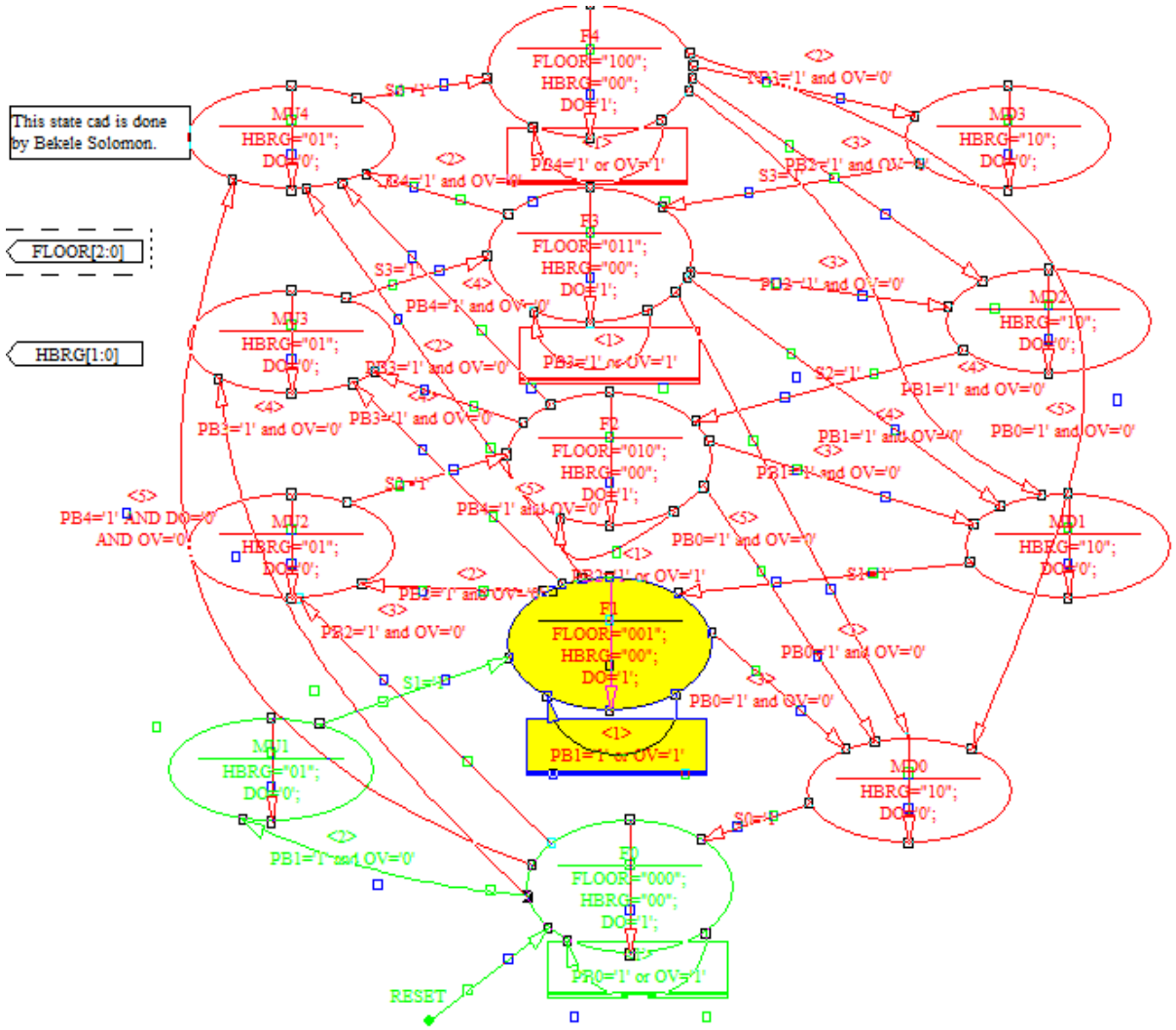


Fig 4 21 when lift is at floor1 and overload is active

The state bench results is shown below being simulated using Xilinx stateCAD (State Bench Simulator) for different floors. When the reset is active, the lift car is on floor 0. By the time push button PB0 is pressed, door open and over load sensor become active, staying at floor 0.

In 2nd lock, PB2 is active state change to MU2 state and output HBRG "01"is generated which moves the motor upward direction. When lift car reached at floor 2, in the 3rd clock, S2 will be active (high), sate will change to F2 state and at this state output is HBRG"00" means motor is at stop condition. The FLOOR (00010) reads 02, which is the result obtained from seven segment

display being converted from binary digit to decimal digit/bit. The car door will be opened immediately as the reaches the floor 2. It stay opened until other call is presented. In the same manner the lift moves upward and downward among the five floors.

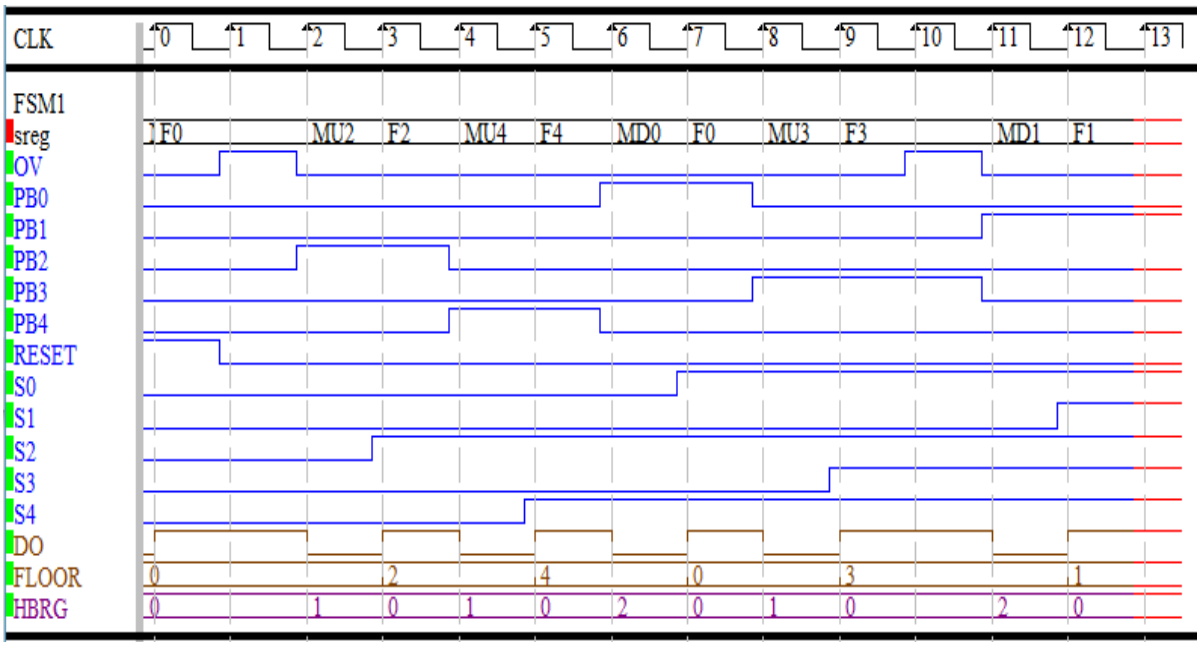


Fig 4 22 the state Bench result for the lift movement

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	58	960	6%
Number of Slice Flip Flops	19	1920	0%
Number of 4 input LUTs	107	1920	5%
Number of bonded IOBs	21	66	31%
Number of GCLKs	1	24	4%

Fig 4 23 resource usage on the Spartan 3E FPGA device

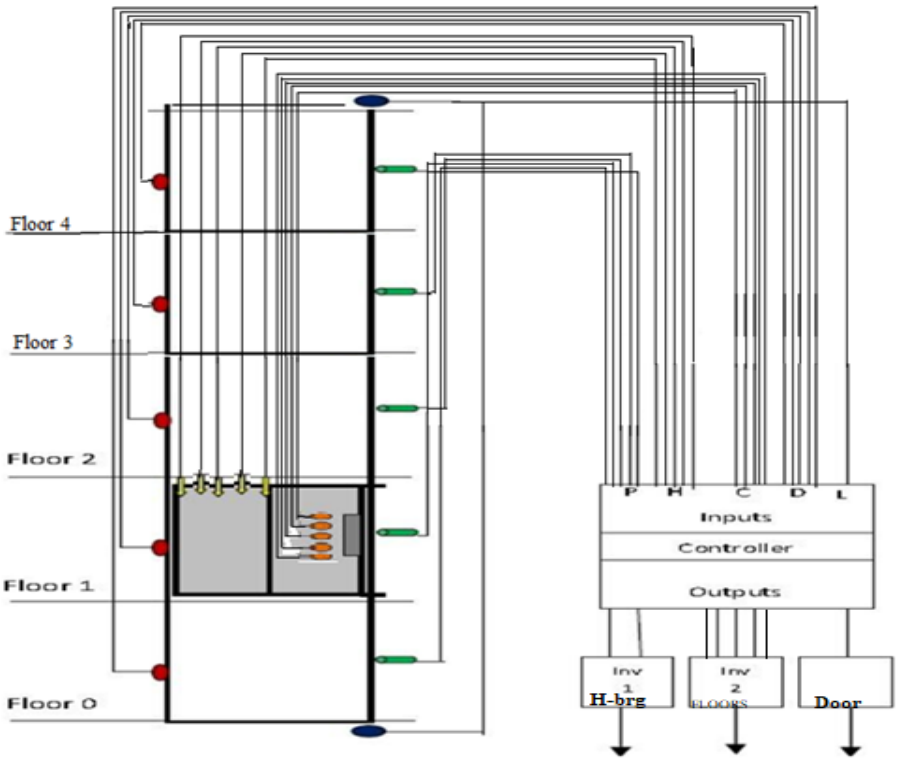


Fig 4 24 overview of lift circuit connection by some modification on [31]

4.3 The Simulink state flow results

The third result show the lift physical world functioning, designed by the Simulink state flow. This result shows a real world lift features. This was done by using Matlab code and state flow concept. This result also has not considered the mathematical modeling of the lift. That is why this thesis

has not more focus on it in the implementation of lift control. Even if the simulation can be integrated with the mathematical modeling, it has been omitted due to its difficulty to make a smooth relation between the state flow and mathematical model. In short, it needs advanced knowledge.

The Simulink state flow diagrams can generate VHDL code of FPGA by using Matlab tool known as "HDL work advisor". The complexity of the state flow determines the code generation of HDL from it. If the state flow has used more complex c code or MATLAB program, it will become complex to generate HDL codes. The Simulink state flow and the Xilinx ISE state CAD (earlier version below Xilinx ISE 9.1i) has close relation. But, the Simulink state flow has greater advantages than the state CAD of the Xilinx due to its ability to be used together with the mathematical modeling on the Simulink analysis. From Simulink state flow and mathematical modeling, it is possible to generate HDL code (for FPGA devices) as well as c code (for microcontroller) though it involves complex procedures and steps. State flow has also great advantage in robotics design.

The state flow simulation has a benefit for visualization purpose of lift operation. It was obtained from Simulink examples by some arrangement of the states, transitions and MATLAB codes. For the figure given below, the first figure show that the car has reached to the third floor due to the hall call from the 3rd floor. The second figure indicates that the lift has reached to the fifth floor, and the fire/overload sensor has resulted active. As a result, the elevator/lift car opens the door to wait until the load/fire condition is changed. The 3rd figure, on the other hand, shows the car call from the ground floor, so it travels to the destination and activate the door opening.

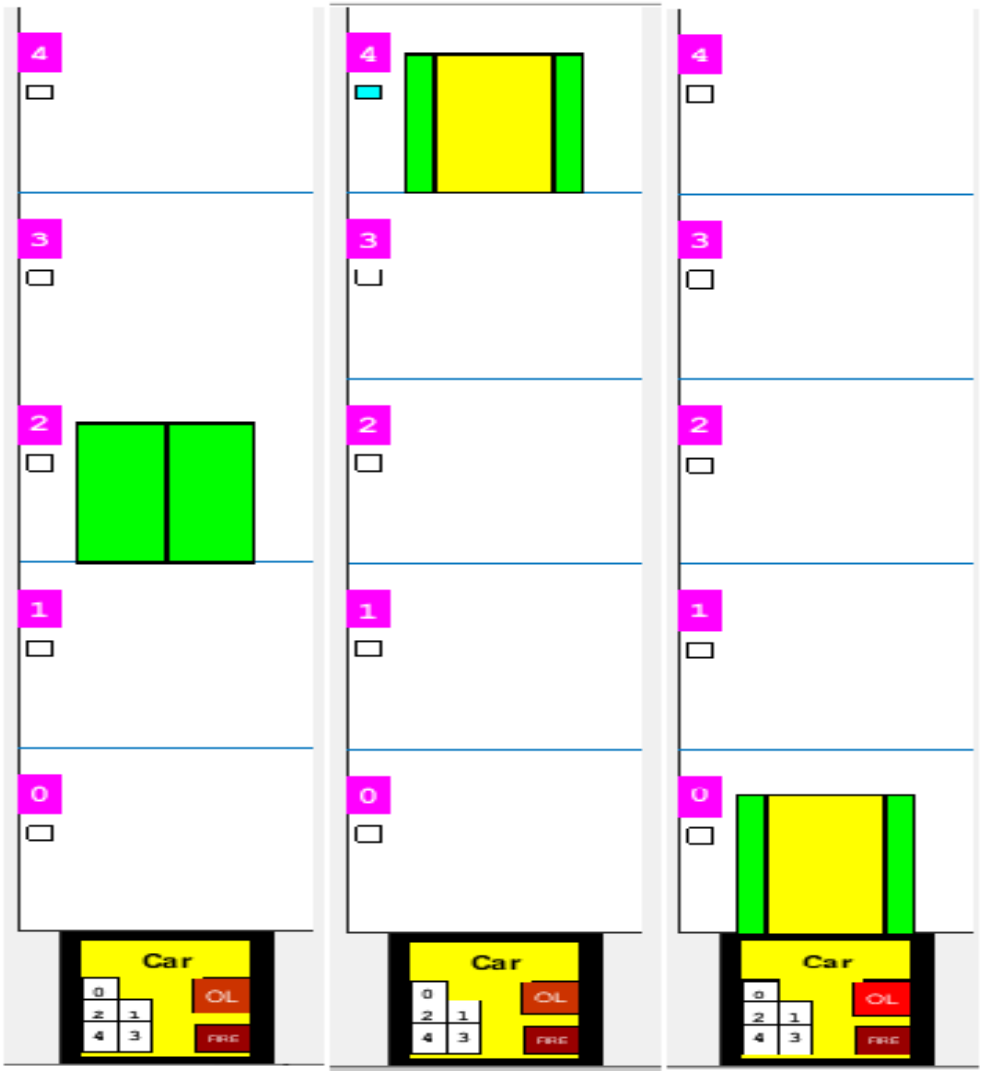


Fig 4 25 THE STATE FLOW ELEVATOR/LIFT DIAGRAM

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

In this thesis, speed and position control of PMDC motor using HDL blocks has been discussed for lift control of Assela malt factory using MATLAB Simulink platform and its feature on Xilinx software. The FPGA controller block was developed after the continuous/discrete time PI controller of speed and current is designed in Simulink main model. Procedurally converted pi speed and current controller blocks from discrete time double data type to fixed point data type does not behave as its original simulation result(continuous or discrete). But, the original simulation result simulated by discrete/continuous time controller (double data type) is used as reference frame for HDL compatible PI controller (fixed point data type) simulation on the main Simulink model.

For No-load case analysis shown above, the desired output of the motion has some deviation when compared with reference speed. In reality, lift has a braking system after some duration of deceleration. Not only for no-load but also for different load changing, the motor brake initiates its stopping, hence removes the error occurred on the speed response.

Even if the two simulations has not been done on the same software platform, the two simulation can be interrelated each other. HBRG output result of the states MU1, MU2… MU4 & MD3, MD2…, MD0 of the stateCAD could be replaced by the controller output H-bridge of the first Simulink simulation. The stateCAD state MU2 has an access for reference speeds of floor1-floor2 and floor0-floor2. MU3 can choose reference speeds of floor2-floor3, floor1-floor3 and floor0-floor3 while MU4 can choose reference speeds of floor (3-4, 2-4, 1-4 & 0-4) respectively. The same is true for the downward movement states of MD3, MD2, MD1 & MD0 with according to **Fig 4 22**.

The stateCAD simulation result shown on ***Fig 4 22*** by state bench diagrams for different floors hasn't considered the reference speed timing. It rather is used to show the simulation feature of a lift. In reality, the duration of time taken by MU1 state should be equal with the time duration elapsed by reference speed for the four meter (4m) height floor(from basement to ground). So, this problem can be solved by the use of appropriate sensors since Sensors are common lift control

signals. The Simulink result and test bench responses can be synchronized using appropriate FPGA device to function as Simulink result with the help of sensor effect on the device responses.

5.2 Recommendation

All things have their own endings. All activity in this world could have their final stage. A man will born and live some duration within this world. Technologies are the same. They usually replaced by newer one when new innovation come to appear.

In control engineering similar replacement of old controlling mechanisms have been revealed. The relay based control has been replaced by ladder logic control. The ladder logic control has been replaced by classical control methods like PI, PD or PID control. Due to their high advantages, they still have been in use. The new advanced control methods like neural network control, adaptive control and fuzzy logic control are replacing the older ones. All these controlling methods use microcontroller or microprocessors for real time implementation. These methods linkage to microcontroller or processors is due to the cheap cost of microcontrollers or microprocessors when compared with FPGA devices. But, they have re-cycling reading of programs which is problem for high speed output desires. Even if their cost is high, FPGA devices have a solution for the speed demand of any human desires. The programming technique used to program is similar. But, their software's are different. Microcontroller/microprocessors use c programing software's while FPGA devices use HDL programming software.

At my starting of thesis title searching I read these advantages of FPGA over microcontroller c programming. Then I wanted to relate my thesis problem to be solved by FPGA. After presenting my proposal I entered to see more software which have linkage with FPGA. I get four software namely: Xilinx ISE, Altera quartus, LabVIEW and Modelsim. After downloading and installing them to my computer I see their relation with my objective. LabVIEW is a more graphical software which has some similarity with Simulink software in some extent. But, it has a drawbacks of inability to generate HDL code from it (I haven't see the more details of it). The other three software could be programmed by writing their respective codes. In short, they couldn't enable me to conduct the simulation of mathematical modeling. They are non-beneficial for load increment or decrement analysis on the mathematical model simulation.

When the above problem faced me, I started to study MATLAB Simulink as a solution. Of course, the Matlab Simulink software was the right software for mathematical modeling and FPGA device compatible HDL code generation from the model to load the program into FPGA devices. Now, my focus has drifted to Simulink HDL code generation from Simulink.

In reality, the study of HDL programming by using Simulink library block is difficult. It involves complex analysis. Those complexities are

a) Installation of newer "b" version of matlab and newer version of Xilinx/Altera software. The most online easily accessible and downloadable matlab version is matlab "a" version. Those oldest matlab version like that of "matlab R2012a" and its earlier do not have "system generator" that enables for integration with Xilinx software. But, the new version has some chance of integration in their configuration. The "matlab R2012b" or others with "b" letters have the capability to integrate with Xilinx system generation configuration. Once the configuration is established, the DSP HDL blocks will be added and seen in the Simulink library. One can use those blocks for Simulink modeling analysis usage.
b) Using of Simulink HDL blocks for control analysis in Simulink modeling. The Simulink library has different blocks like that of continuous time block, the discrete time blocks and HDL blocks. The HDL blocks can be used for the model analysis.
c) Analyzing in continuous time domain and converting the control subsystem into discrete time, then to single floating point for HDL code generation is another option. In order to use efficiently the limited HDL device hardware, it is obligatory to convert double floating point to single floating point and changing some library blocks to other HDL compatible form to use hardware device.

This paper has led me to further explore different software and see their pursuits. The title I tried to solve has been vast to easily analyze. It could be two thesis titles as my imagination. The first will be the *speed control analysis of dc/AC machines using FPGA device on matlab software* and the second will be *state flow and matlab code integration with HDL programming*. In the future, my study shall focus on the speed control of synchronous motor using FPGA device for lift car driving of upward/downward. Synchronous motor is chosen due to its good behavior of precise timing and exact RPMs behavior when compared with other motors.

The major problem that I faced in this paper is the inability of simulating the two simulations in one software environment .i.e. Simulink or Xilinx ISE. In the future, I need to draw the two simulation result on one Simulink environment with all features included in it.

References

[1] I. O. A. N. B. G. O. C. L. O. M. C. M. C. O. Abdulmalik, "The Design of A One-Man Passenger Electric Elevator," *Scholars Journal of Engineering and Technology (SJET),* vol. 2, no. ©Scholars Academic and Scientific publisher, pp. 1-6, 2014.

[2] *About Elevator by Otis,* 1970.

[3] D. Loker, "ELEVATOR CONTROL SYSTEM PROJECT," in *American Society for Engineering Education*, Pennsylvania State University, Erie, 2010.

[4] Prof. Dr. A. Bode, "Specification of an Elevator Control System," *Frank Strobl and Alexander Wisspeintner,* no. Institut fur Informatik, p. 63, 1999.

[5] M. A. U. a. M. A. Saeed, "Design and Implementation of Efficient Elevator Control System using FPGA," *Innovative Space of Scientific Research,* vol. 9, no. Innovation and Applied Studies, pp. 1541-1546, 2014.

[6] R. E. S. S. S. G. A. S. D. Sithumini Ekanayake, "FPGA Based Elevator Controller with Improved Reliability," *International Conference on Computer Modelling and Simulation,* vol. 15, no. University of Peradeniya, Sri Lanka, p. 6, 2013.

[7] K. Al-Kodmany, "Tall Buildings and Elevators: A Review of Recent Technological Advances," *Buildings,* no. College of Urban Planning and Public Affairs, University of Illinois at Chicago, Chicago, IL 60607, USA, pp. 1-35, 2015.

[8] heavy-duty-elevator-machine-room-less-mrl, "http://www.railsystem.net," [Online]. Available: http%3A%2F%2Fwww.railsystem.net. [Accessed 18 05 2020].

[9] G. G. G. H. Menbere Alemu, "ELEVATOR CONTROL SYSTEM BASED ON PLC SIMULATION," in *ETHIOPIAN INSTITUTE OF TECHNOLOGY*, MEKELLE, 2013.

[10] A. L. S. Dr. Jamal A. Mohammed, "Modeling of DC Elevator Motor Drive for Mid-rise Building," *Eng. &Tech. Journal,* vol. 31, no. 12, pp. 1-23, 2013.

[11] D. L. Al-Sharif, "LIFT SAFTY GEAR TESTING WITHOUT WEIGHTS: A CRITIQUE AND OVERVIEW," Buchanan House, Holborn,London EC1N 2HS.

[12] "Basic Elevator components," www..slideshare.net, 12 4 2018. [Online]. Available: http://www.basic-elevator-components. [Accessed 10 6 2020].

[13] m. e. q. a. answers, "Question: A Geared Elevator System Used To Raise A Car Is Represented...," Chegg Study textbooks and solutions, [Online]. Available: www.chegg.com/homework-help/questions-and-answers/geared-elevator-system. [Accessed 18 05 2020].

[14] alibaba, "www.alibaba.com/product-detail," alibaba, [Online]. Available: www.alibaba.com/product-detail/1150-1600KG-Gearless-Traction-Elevator-Machine. [Accessed 18 05 2020].

[15] S. H. a. S. S. Y. Mon, "Implementation of PLC Based Elevator Control System," *International Journal of Electronics and Computer Science Engineering,* vol. 3, no. 2277, pp. 95-96, 1956.

[16] K. P. P. N. Kapil, "SIMULATION OF PWM CONTROLLER BASED DC MOTOR," *Technical Research Organization India(TROI),* vol. 2, no. 5, pp. 65-68, 2015.

[17] u. martin, "ELEVATOR ROPES," Usha martin limited.

[18] R. Đ. i. M. K. M. K. J. Vladić, "MODELLING AND SIMULATIONS OF ELEVATOR DYNAMIC BEHAVIOUR," *Modeliranje i simulacije dinamičkog ponašanja dizala,* vol. 3, no. Technical Gazette , 18, p. 427, 2011.

[19] P. D. D. G. &. CO.KG, "Steel Wire Ropes in Elevators," PFEIFER DRAKO, 2015.

[20] G. W. Gibson, "ELEVATOR HOISTWAY EQUIPMENT:Mechanical and Structural Design, Part II," *ELEVATOR WORLD,* p. 102, 01 February 2009.

[21] electrical-knowhow, "electrical-knowhow," [Online]. Available: http://www.electrical-knowhow.com/2012/04/elevator-control-system.html. [Accessed 24 4 2015].

[22] R. T. a. J. R. I. Kuon, "FPGA Architecture: Survey and Challenges," in *Foundations and Trends in Electronic Design Automation*, Toronto, Canada, eecg.utoronto.ca, 2007, pp. 135-253.

[23] A. Moore, FPGAs For Dummies®, Altera Special Edition, Hoboken, New Jersey: John Wiley & Sons, Inc., 2014.

[24] K. K. a. Y. Gu, "Model-Based Design with Simulink, HDL Coder, and Xilinx System Generator for DSP," The MathWorks, Inc., washington DC, 2012.

[25] P. P.Chu, "FSM," in *FPGA PROTOTYPING BY VERILOG EXAMPLES*, Hoboken,New Jersey, JOHN WILEY & SONS Inc, 2008, pp. 119-137.

[26] B. T. Boulter, "Elevator Modeling and DC Drive Speed Controller Design," © ApICS ® LLC, 2000.

[27] D. S. C. M. Hui, "Lift and Escalators," Department of Mechanical Engineering, Hong Kong, sep,2010.

[28] D. L. R. Al-Sharif, "Motor Sizing and Selection for Geared Hoisting Systems," [Dr. Lutfi R. Al-Sharif, 21 12 2007. [Online]. Available: http://www.ju.edu.jo/sites/Academic/l.sharif/Material/Forms/AllItems.aspx. [Accessed 28 05 2019].

[29] P. J. Ford, "ELECTRIC ELEVATOR DRIVE WITH POSITION CONTROL," 5 05 2012. [Online]. Available: http://www.elevator position control of.com . [Accessed 15 03 2015].

[30] H. Semat, "Physics, Chapter 11: Rotational Motion," in *The Dynamics of a Rigid Body*, City College of New York, Robert Katz, 1958, p. 141.

[31] S. M. I. H. A. S. M. I. P. P. W. J. S. A. M. I. a. S. D. H.P.A.P. Jayawardana, "Design and Implementation of a Statechart Based Reconfigurable Elevator Controller," in *International Conference on Industrial and Information Systems*, Sri Lanka, 2011.

[32] R. E. S. S. s. G. D. Suthumini Ekanayake, "FPGA Based Elevator Controller with Improved Reliability," *International Conference on Computer Modelling and Simulation,* vol. 15, no. University of Peradeniya, Sri Lanka, p. 6, 2013.

[33] "Apparent Weight: Person on Scale in Elevator".

[34] "Modeling of DC Elevator Motor Drive for Mid-rise Building".

APPENDIX A

Simulink parameters

PMDC motor

Ra =0.5
La = 4.3e-3
Jm=0.2
Kt=1.5
B=6.4e-5 %Friction coefficient

Tf = 1 Nm % frictional torque loss

Load of lift

Mp = 1kg %drive pulley mass
Rp = 0.1 m %drive pulley radius
g = 9.81m/s^2 %gravitational constant
Mc = (0-480 + 873) kg %mass of car
Mcw = 873 kg %mass of counter-weight
rg = 12 % gear ratio
n= 0.75 % gear and motor efficiency

w = 150 rad/s

Appendix B

Prepare Floating-Point Model for Conversion to Fixed Point

The Fixed-Point Advisor provides a set of tasks that help you prepare a floating-point model or subsystem for conversion to an equivalent fixed-point representation. After preparing your model, you use the Fixed-Point Tool to perform the fixed-point conversion.

In this part of the example, you use the Fixed-Point Advisor to prepare the Controller Subsystem in the ex_fixed_point_workflow model for conversion.

Open the Fixed-Point Advisor

1. In the ex_fixed_point_workflow model menu, select **Analysis** > **Fixed-Point Tool**.
2. In the Fixed-Point Tool:
   a. Under **System Under Design**, click **Change**.
   b. In the System Selector dialog box, select Controller Subsystem and click **OK**.
   c. Under **Fixed-point preparation**, click the **Fixed-Point Advisor** button.

   You run the Fixed-Point Advisor on the ex_fixed_point_workflow Controller Subsystem because this is the system of interest. There is no need

to convert the system inputs or the display to fixed point.

Prepare Model for Conversion

1. In the Fixed-Point Advisor left pane, expand the **Prepare Model for Conversion** folder to view the tasks. For the purpose of this example, run the tasks in the this folder one at a time. Select**Verify model simulation settings** and, in the right pane, select **Run this task**.

   This task validates that model simulation settings allow signal logging and disables data type override in the model and for fi objects or embedded numeric data types in your model or workspace. These settings facilitate conversion to fixed point in later tasks.

   A waitbar appears while the task runs. When the run is complete, the result shows that the task passed.

2. Select and run **Verify update diagram status**.

   Verify update diagram status runs. Your model must be able to successfully update diagram to run the checks in the Fixed-Point Advisor.

   The task passes.

3. Select and run **Address unsupported blocks**. This task identifies blocks that do not support fixed-point data types.

   The task passes because the subsystem contains no blocks that do not support fixed-point data.

4. Select and run **Set up signal logging**. Prior to simulation, you must specify at least one signal for the Fixed-Point Advisor to use for analysis and comparison in downstream checks. You should log, at minimum, the unique input and output signals.

   The task generates a warning because signal logging is not specified for any signals.

5. Fix the warning using the Model Advisor Result Explorer:

   a. Click the **Explore Result** button.

      The Model Advisor Result Explorer opens.

   b. In the middle pane, select each signal you want to log and, next to the signal, select the corresponding **EnableLogging** check box.

      For this example, log these signals:

   - Lookup Table for Gain
   - Lookup Table for Chart
   - Chart
   - Discrete Filter

   c. Close the Model Advisor Result Explorer.

   d. In the Fixed-Point Advisor window, click **Run This Task**.

      The task passes because signal logging is now enabled for at least one signal.

6. Select and run **Create simulation reference data**.

   The Fixed-Point Advisor simulates the model using the current solver settings, and creates and archives reference signal data in a run named FPA_Reference to use for analysis and comparison in later conversion tasks. This task also validates that model simulation settings allow signal logging and that the **Fixed-point instrumentation mode** is set to Minimums, maximums and overflows.

   The Fixed-Point Advisor issues a warning and provides information in the Analysis Result box that logging simulation minimum and maximum values failed.

   Logging failed because the **Fixed-point instrumentation mode** is Use local settings, but the recommended setting is Minimums, maximums and overflows.

7. To fix the failure, in the **Action** pane, click **Modify All**.

   The Fixed-Point Advisor configures the model to the settings recommended in the Analysis **Result** pane. The **Action** pane displays a table of changes showing that the **Fixed-point instrumentation mode** is now Minimums, maximums and overflows

8. Click **Run This Task**.

   Running the task after using the Modify All action verifies that you made the necessary changes. The Analysis **Result** pane updates to display a passed result and information about why the task passed.

   Tip You can view the reference run data in the Fixed-Point Tool **Contents** pane i for both runs using the same name (FPA_Reference).

9. In the **Verify Fixed-Point Conversion Guidelines** folder, select and run **Check model configuration data validity diagnostic parameters settings**. This task verifies that the **Model Configuration Parameters** > **Diagnostics** > **Data Validity** > **Parameters** options are all set to warning. If these options are set to error, the model update diagram action fails in downstream checks.

   The task passes because none of these options are set to error.

10. Select and run **Implement logic signals as Boolean data**. This task verifies that **Model Configuration Parameters** > **Optimization** > **Implement logic signals as Boolean data** is selected. If it is cleared, the code generated in downstream checks is not optimized.

    The task passes.

11. Select and run **Check bus usage**. This task identifies:

- Mux blocks that are bus creators
- Bus signals that the top-level model treats as vectors

Note: This is a Simulink® check. For more information, see Check bus usag

12. The task runs and generates a warning because this check works only from top-level models and you are running from the subsystem. Because this model uses no buses, ignore this warning. For models containing buses, you must run the Fixed-Point Advisor from the top-level model to perform this check.
13. Select and run **Simulation range checking**. This tasks verifies that the **Model Configuration Parameters** > **Diagnostics** > **Data Validity**> **Simulation range checking** option is not set tonone.

    The task generates a warning because the Simulation range checking option is none.
14. To fix the warning, in the **Action** box, click **Modify All**.

    The Fixed-Point Advisor sets the Simulation range checking option to warning.
15. Rerun the task.

    The task now passes because the **Simulation range checking** option is correct.
16. Select and run **Check for implicit signal resolution**. This task checks for models that use implicit signal resolution.

    The task fails because implicit signal resolution is enabled.
17. To fix the failure, in the **Action** box, click **Modify All**.

    The Fixed-Point Advisor sets the **Signal resolution** option to Explicit only.
18. Rerun the task.

    The task now passes.

You have completed all the tasks for the **Prepare Model for Conversion** folder. At this point, you can review the results report found at the folder level, or continue to the next folder.

Prepare for Data Typing and Scaling

This folder contains tasks that set the block configuration options and set output minimum and maximum values for blocks. The block settings from this task simplify the initial data typing and scaling. Later tasks set optimal block configuration. The tasks in this folder prepare the model for automatic data typing in the Fixed-Point Tool.

1. For the purpose of this example, run the tasks in the **Prepare for Data Typing and Scaling** folder one at a time.

   Open the **Prepare for Data Typing and Scaling** folder then select and run **Review locked data type settings**. This task identifies blocks that have their data type settings locked down which excludes them for automatic data typing.

   This task passes because the model contains no blocks with locked data types.
2. Select and run **Verify Stateflow charts have strong data typing with Simulink**. This task verifies that the configuration of all Stateflow® charts ensures strong data typing with Simulink I/O.

   The task passes because the configuration of the Stateflow chart in the subsystem is correct.
3. Select and run **Specify block minimum and maximum values**. Ideally, you should specify block output and parameter minimum and maximum values for, at minimum, the Inport blocks in the system. You can specify the minimum and maximum values for any block in this step. Typically, you determine these values during the design process based on the system you are creating.
4. The tool advises you to specify minimum and maximum values for all blocks if possible. For the purpose of this example, specify the minimum and maximum values for the Inport blocks in the system.
   a. Click the **Explore Result** button.

      The Model Advisor Result Explorer opens, showing that the Inport blocks, In1 and In2, do not have output minimum and maximum values specified.
   b. In the center pane, select In1. This block receives the output from Repeating table Source, which has a minimum value of 10 and a maximum value of 20. Therefore, set **OutMin** to 10 and set **OutMax** to 20 as follows:
      i. In the **OutMin** column for In1, select [] and replace with 10.
      ii. In the **OutMax** column for In1, select [] and replace with 20.
   c. Select In2. This block receives the output from Sine Wave block, which has a minimum value of -1 and a maximum value of 1. Therefore, set **OutMin** to -1 and set **OutMax** to 1.
   d. Close the Model Advisor Result Explorer.
   e. In the Fixed-Point Advisor, rerun the task.

      The task passes because you specified minimum and maximum values for all Inport blocks.

You have completed all tasks in the **Prepare for Data Typing and Scaling** folder. At this point, you can review the results report found at the folder level, or continue to the next folder.

Return to Fixed-Point Tool to Perform Data Typing and Scaling

Select and run this task to close the Fixed-Point Advisor and return to the Fixed-Point Tool.

Propose Data Types

Use the Fixed-Point Tool to propose fixed-point data types for appropriately configured blocks based on the double-precision simulation results stored in the simulation reference run that the Fixed-Point Advisor created. These results are stored in the run named **FPA_Reference**. You can view the results in the Fixed-Point Tool **Contents** pane.

The tool proposes fixed-point data types and scaling based on the ranges of the Repeating table Source and Sine Wave inputs. You can then use the tool to accept and apply the proposed data types selectively. In this example, you propose fraction lengths for the specified word lengths.

1. In the Fixed-Point Tool, Under **Automatic data typing**, click the **Propose data types** button DT.

   The Fixed-Point Tool analyzes the scaling of all fixed-point blocks whose **Lock output data type setting against changes by the fixed-point tools** parameter is not selected.

   The Fixed-Point Tool uses the default proposal settings to propose data types with 16-bit word length and best-precision fraction length and updates the results in the **Contents** pane.
2. The tool displays the proposed data types in the **ProposedDT** column in the **Contents** pane.

   By default, it selects the **Accept** check box for these signals because the proposed data type differs from the object's current data type. If you apply data types, the tool will apply the proposed data types to these signals. For more information, see Apply Proposed Data Types.
3. Examine the results to resolve any conflicts and to ensure that you want to accept the proposed data type for each result.

   In the Fixed-Point Tool toolbar, select **Show** > **Conflicts with proposed data types**.

   The Fixed-Point Tool detected no conflicts.

   Tip If the tool does detect conflicts, you must resolve these before applying data t

Now that you have reviewed the results and ensured that there are no issues, you are ready to apply the proposed data types to the model, as described in Apply Fixed-Point Data Types.

Apply Fixed-Point Data Types

1. Click the **Apply accepted data types** button to write the proposed data types to the model.

   The Fixed-Point Tool applies the data type proposals to the subsystem blocks.
2. In the Fixed-Point Tool toolbar, select **Show** > **All results**.

   The tool has set all the specified data types to the proposed types.

You are now ready to check that the new data types are acceptable, as described in Verify Fixed-Point Settings.

Verify Fixed-Point Settings

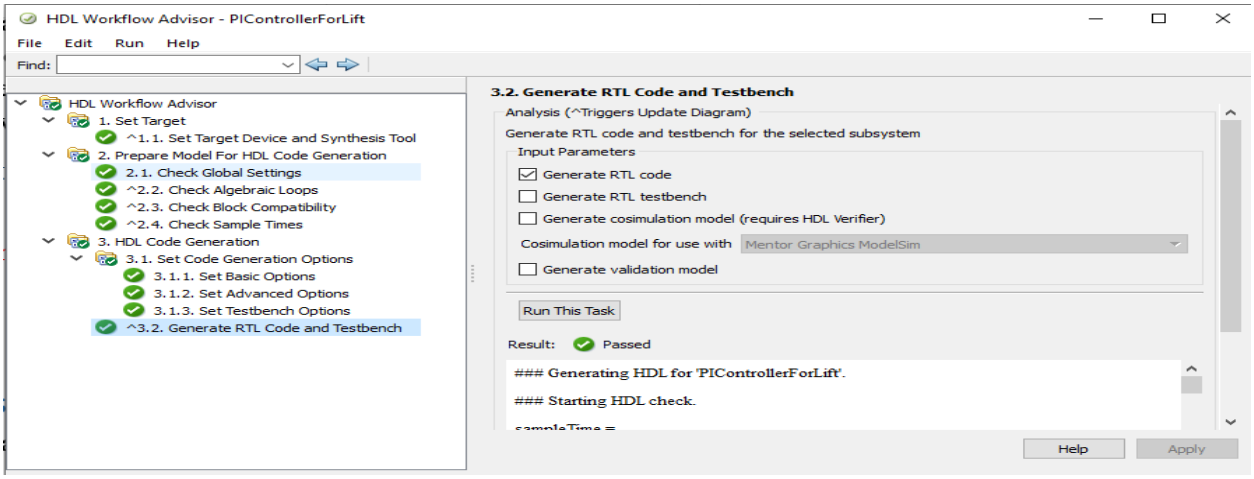
Next, you simulate again using the new fixed-point settings. You then use the Fixed-Point Tool plotting capabilities to compare the results from the floating-point **FPA_Reference** run with the fixed-point results.

1. Under **Range collection**, set **Run name** to Initial_fixed_point. You specify a new run name to prevent the tool from overwriting the results that you want to retain in the **FPA_Reference**run.
2. Click the Fixed-Point Tool **Simulate model** button to run the simulation.

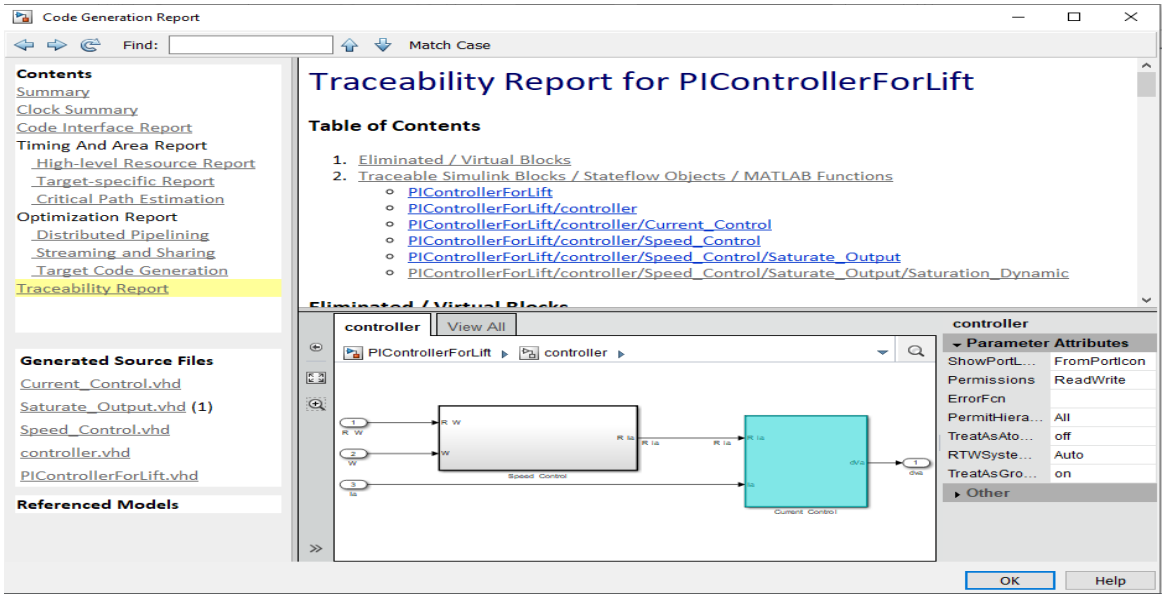
   The Simulink software simulates using the new data types that you applied in the previous step. Afterward, the Fixed-Point Tool displays in its **Contents** pane information about blocks that logged fixed-point data.
   The **CompiledDT** (compiled data type) column for the run shows that the Controller Subsystem blocks use fixed-point data types with the new data types.

   Tip In the **Contents** pane, click the **Run** column heading to sort the runs.
3. Examine the results to verify that there are no overflows or saturations.

APPENDIX C



Its Generated VHDL Code Summary is given below.



APPENDIX D

(The VHDL generated code from the state CAD)

```
-- C:\XILINX\ELEVATOR\ELL.vhd

-- VHDL code created by Xilinx's StateCAD 8.1i

-- Thu May 23 15:57:01 2019

-- This VHDL code (for use with Xilinx XST) was
generated using:

-- enumerated state assignment with structured code
format.

-- Minimization is enabled, implied else is enabled,
```

```
-- and outputs are speed optimized.

LIBRARY ieee;

USE ieee.std_logic_1164.all;

ENTITY SHELL_ELL IS

PORT
(CLK,OV,PB0,PB1,PB2,PB3,PB4,RESET,S0,S1,S2,S3,S4
: IN std_logic;

DO, FLOOR0, FLOOR1, FLOOR2, FLOOR3, FLOOR4,
HBRG0, HBRG1 : OUT std_logic);

END;

ARCHITECTURE BEHAVIOR OF SHELL_ELL IS

TYPE type_sreg IS
(F0,F1,F2,F3,F4,MD0,MD1,MD2,MD3,MU1,MU2,MU3,
MU4);

SIGNAL sreg, next_sreg : type_sreg;

SIGNAL
next_BP_DO,next_BP_FLOOR0,next_BP_FLOOR1,next_
BP_FLOOR2,next_BP_FLOOR3,next_BP_FLOOR4,next_
HBRG0,

next_HBRG1 : std_logic;

SIGNAL BP_FLOOR : std_logic_vector (4 DOWNTO 0);

SIGNAL FLOOR : std_logic_vector (4 DOWNTO 0);

SIGNAL HBRG : std_logic_vector (1 DOWNTO 0);

SIGNAL
BP_DO,BP_FLOOR0,BP_FLOOR1,BP_FLOOR2,BP_FL
OOR3,BP_FLOOR4: std_logic;

BEGIN

PROCESS (CLK, next_sreg, next_BP_DO,
next_BP_FLOOR4, next_BP_FLOOR3,
next_BP_FLOOR2, next_BP_FLOOR1,

next_BP_FLOOR0, next_HBRG1, next_HBRG0)

BEGIN

IF CLK='1' AND CLK'event THEN

sreg <= next_sreg;

BP_DO <= next_BP_DO;

BP_FLOOR4 <= next_BP_FLOOR4;

BP_FLOOR3 <= next_BP_FLOOR3;

BP_FLOOR2 <= next_BP_FLOOR2;

BP_FLOOR1 <= next_BP_FLOOR1;

BP_FLOOR0 <= next_BP_FLOOR0;

HBRG1 <= next_HBRG1;

HBRG0 <= next_HBRG0;

END IF;

END PROCESS;

PROCESS
(sreg,BP_DO,BP_FLOOR0,BP_FLOOR1,BP_FLOOR2,B
P_FLOOR3,BP_FLOOR4,OV,PB0,PB1,PB2,PB3,PB4,

RESET,S0,S1,S2,S3,S4,BP_FLOOR,HBRG)

BEGIN

next_BP_DO <= BP_DO;next_BP_FLOOR0 <=
BP_FLOOR0;next_BP_FLOOR1 <=

BP_FLOOR1;next_BP_FLOOR2 <=
BP_FLOOR2;next_BP_FLOOR3 <=
BP_FLOOR3;next_BP_FLOOR4 <= BP_FLOOR4;

BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));

next_HBRG0 <= '0'; next_HBRG1 <= '0';

HBRG<=std_logic_vector'("00");

next_sreg<=F0;

IF ( RESET='1' ) THEN

next_sreg<=F0;
```

```
next_BP_DO<='1';
HBRG <= (std_logic_vector("00"));
BP_FLOOR <= (std_logic_vector("00000"));
ELSE
CASE sreg IS
WHEN F0 =>
IF ( PB0='1' ) OR ( OV='1' ) THEN
next_sreg<=F0;next_BP_DO<='1';
HBRG <= (std_logic_vector("00"));
BP_FLOOR <= (std_logic_vector("00000"));
ELSIF ( PB1='1' AND OV='0' ) THEN
next_sreg<=MU1;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector("01"));
ELSIF ( PB2='1' AND OV='0' ) THEN
next_sreg<=MU2;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector("01"));
ELSIF ( PB3='1' AND OV='0' ) THEN
next_sreg<=MU3;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector("01"));
ELSIF ( PB4='1' AND BP_DO='0' AND OV='0' ) THEN
next_sreg<=MU4;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector("01"));
ELSE
next_sreg<=F0;
next_BP_DO<='0';
HBRG <= (std_logic_vector("00"));
BP_FLOOR <= (std_logic_vector("00000"));
END IF;

WHEN F1 =>
IF ( PB1='1' ) OR ( OV='1' ) THEN
next_sreg<=F1;
next_BP_DO<='1';
HBRG <= (std_logic_vector("00"));
BP_FLOOR <= (std_logic_vector("00001"));
ELSIF ( PB2='1' AND OV='0' ) THEN
next_sreg<=MU2;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector("01"));
ELSIF ( PB0='1' AND OV='0' ) THEN
```

```
next_sreg<=MD0;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector'("10"));
ELSIF ( PB3='1' AND OV='0' ) THEN
next_sreg<=MU3;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector'("01"));
ELSIF ( PB4='1' AND OV='0' ) THEN
next_sreg<=MU4;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector'("01"));
ELSE
next_sreg<=F1;
next_BP_DO<='0';
HBRG <= (std_logic_vector'("00"));
BP_FLOOR <= (std_logic_vector'("00001"));
END IF;

WHEN F2 =>
IF ( PB2='1' ) OR ( OV='1' ) THEN
next_sreg<=F2;
next_BP_DO<='1';
HBRG <= (std_logic_vector'("00"));
BP_FLOOR <= (std_logic_vector'("00010"));
ELSIF ( PB3='1' AND OV='0' ) THEN
next_sreg<=MU3;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector'("01"));
ELSIF ( PB1='1' AND OV='0' ) THEN
next_sreg<=MD1;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector'("10"));
ELSIF ( PB4='1' AND OV='0' ) THEN
next_sreg<=MU4;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector'("01"));
ELSIF ( PB0='1' AND OV='0' ) THEN
next_sreg<=MD0;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
```

```
HBRG <= (std_logic_vector("10"));
ELSE
next_sreg<=F2;
next_BP_DO<='0';
HBRG <= (std_logic_vector("00"));
BP_FLOOR <= (std_logic_vector("00010"));
END IF;

WHEN F3 =>
IF ( PB3='1' ) OR ( OV='1' ) THEN
next_sreg<=F3;
next_BP_DO<='1';
HBRG <= (std_logic_vector("00"));
BP_FLOOR <= (std_logic_vector("00011"));
ELSIF ( PB4='1' AND OV='0' ) THEN
next_sreg<=MU4;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector("01"));
ELSIF ( PB2='1' AND OV='0' ) THEN
next_sreg<=MD2;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector("10"));
ELSIF ( PB1='1' AND OV='0' ) THEN
next_sreg<=MD1;next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector("10"));
ELSIF ( PB0='1' AND OV='0' ) THEN
next_sreg<=MD0;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector("10"));
ELSE
next_sreg<=F3;
next_BP_DO<='0';
HBRG <= (std_logic_vector("00"));
BP_FLOOR <= (std_logic_vector("00011"));
END IF;

WHEN F4 =>
IF ( PB4='1' ) OR ( OV='1' ) THEN
next_sreg<=F4;
next_BP_DO<='1';
HBRG <= (std_logic_vector("00"));
BP_FLOOR <= (std_logic_vector("00100"));
ELSIF ( PB3='1' AND OV='0' ) THEN
next_sreg<=MD3;
next_BP_DO<='0';
```

```
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));

HBRG <= (std_logic_vector'("10"));

ELSIF ( PB2='1' AND OV='0' ) THEN

next_sreg<=MD2;

next_BP_DO<='0';

BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));

HBRG <= (std_logic_vector'("10"));

ELSIF ( PB1='1' AND OV='0' ) THEN

next_sreg<=MD1;

next_BP_DO<='0';

BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));

HBRG <= (std_logic_vector'("10"));

ELSIF ( PB0='1' AND OV='0' ) THEN

next_sreg<=MD0;

next_BP_DO<='0';

BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));

HBRG <= (std_logic_vector'("10"));

ELSE

next_sreg<=F4;

next_BP_DO<='0';

HBRG <= (std_logic_vector'("00"));

BP_FLOOR <= (std_logic_vector'("00100"));

END IF;


WHEN MD0 =>

IF ( S0='1' ) THEN

next_sreg<=F0;

next_BP_DO<='1';

HBRG <= (std_logic_vector'("00"));

BP_FLOOR <= (std_logic_vector'("00000"));

ELSE

next_sreg<=MD0;

next_BP_DO<='0';BP_FLOOR <=
(( std_logic_vector'(BP_FLOOR4, BP_FLOOR3,
BP_FLOOR2, BP_FLOOR1, BP_FLOOR0)));

HBRG <= (std_logic_vector'("10"));

END IF;


WHEN MD1 =>

IF ( S1='1' ) THEN

next_sreg<=F1;

next_BP_DO<='1';

HBRG <= (std_logic_vector'("00"));

BP_FLOOR <= (std_logic_vector'("00001"));

ELSE

next_sreg<=MD1;

next_BP_DO<='0';

BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));

HBRG <= (std_logic_vector'("10"));
```

```
END IF;

WHEN MD2 =>
IF ( S2='1' ) THEN
next_sreg<=F2;
next_BP_DO<='1';
HBRG <= (std_logic_vector'("00"));
BP_FLOOR <= (std_logic_vector'("00010"));
ELSE
next_sreg<=MD2;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector'("10"));
END IF;

WHEN MD3 =>
IF ( S3='1' ) THEN
next_sreg<=F3;
next_BP_DO<='1';
HBRG <= (std_logic_vector'("00"));
BP_FLOOR <= (std_logic_vector'("00011"));
ELSE
next_sreg<=MD3;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector'("10"));
END IF;

WHEN MU1 =>
IF ( S1='1' ) THEN
next_sreg<=F1;
next_BP_DO<='1';
HBRG <= (std_logic_vector'("00"));
BP_FLOOR <= (std_logic_vector'("00001"));
ELSE
next_sreg<=MU1;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
HBRG <= (std_logic_vector'("01"));
END IF;

WHEN MU2 =>
IF ( S2='1' ) THEN
next_sreg<=F2;
next_BP_DO<='1';
HBRG <= (std_logic_vector'("00"));
BP_FLOOR <= (std_logic_vector'("00010"));ELSE
next_sreg<=MU2;
next_BP_DO<='0';
BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
```

```
HBRG <= (std_logic_vector'("01"));

END IF;


WHEN MU3 =>

IF ( S3='1' ) THEN

next_sreg<=F3;

next_BP_DO<='1';

HBRG <= (std_logic_vector'("00"));

BP_FLOOR <= (std_logic_vector'("00011"));

ELSE

next_sreg<=MU3;

next_BP_DO<='0';

BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));

HBRG <= (std_logic_vector'("01"));

END IF;


WHEN MU4 =>

IF ( S4='1' ) THEN

next_sreg<=F4;

next_BP_DO<='1';

HBRG <= (std_logic_vector'("00"));

BP_FLOOR <= (std_logic_vector'("00100"));

ELSE

next_sreg<=MU4;

next_BP_DO<='0';

BP_FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));

HBRG <= (std_logic_vector'("01"));

END IF;


WHEN OTHERS =>

END CASE;

END IF;

next_BP_FLOOR4 <= BP_FLOOR(4);

next_BP_FLOOR3 <= BP_FLOOR(3);

next_BP_FLOOR2 <= BP_FLOOR(2);

next_BP_FLOOR1 <= BP_FLOOR(1);

next_BP_FLOOR0 <= BP_FLOOR(0);

next_HBRG1 <= HBRG(1);

next_HBRG0 <= HBRG(0);

END PROCESS;

PROCESS (BP_DO)

BEGIN

IF (( BP_DO='1' )) THEN DO<='1';

ELSE DO<='0';

END IF;

END PROCESS;

PROCESS
(BP_FLOOR0,BP_FLOOR1,BP_FLOOR2,BP_FLOOR3,
BP_FLOOR4,FLOOR)

BEGIN

FLOOR <= (( std_logic_vector'(BP_FLOOR4,
BP_FLOOR3, BP_FLOOR2, BP_FLOOR1,
BP_FLOOR0)));
```

```
FLOOR0 <= FLOOR(0);

FLOOR1 <= FLOOR(1);

FLOOR2 <= FLOOR(2);

FLOOR3 <= FLOOR(3);

FLOOR4 <= FLOOR(4);

END PROCESS;

END BEHAVIOR;

LIBRARY ieee;

USE ieee.std_logic_1164.all;

ENTITY ELL IS PORT (FLOOR : OUT std_logic_vector
(4 DOWNTO 0);

HBRG : OUT std_logic_vector (1 DOWNTO 0);

CLK,OV,PB0,PB1,PB2,PB3,PB4,RESET,S0,S1,S2,S3,S4:
IN std_logic;

DO : OUT std_logic);

END;

ARCHITECTURE BEHAVIOR OF ELL IS

COMPONENT SHELL_ELL

PORT
(CLK,OV,PB0,PB1,PB2,PB3,PB4,RESET,S0,S1,S2,S3,S4
: IN std_logic;

DO,FLOOR0,FLOOR1,FLOOR2,FLOOR3,FLOOR4,HBR
G0,HBRG1 : OUT std_logic);

END COMPONENT;

BEGIN

SHELL1_ELL : SHELL_ELL PORT MAP
(CLK=>CLK,OV=>OV,PB0=>PB0,PB1=>PB1,PB2=>PB
2,

PB3=>PB3,PB4=>PB4,RESET=>RESET,S0=>S0,S1=>S1
,S2=>S2,S3=>S3,S4=>S4,DO=>DO,

FLOOR0=>FLOOR(0),FLOOR1=>FLOOR(1),FLOOR2=
>FLOOR(2),FLOOR3=>FLOOR(3),FLOOR4=>FLOOR(4
),HBRG0=>HBRG(0),HBRG1=>HBRG(1));

END BEHAVIOR;
```