

MRes in Railway Systems Engineering and Integration

College of Engineering, School of Civil Engineering

University of Birmingham



**Integration of a Mechanical Interlocking
Lever Frame into a Signalling
Demonstrator**

By: Mersedeh Maksabi

Supervisor: Prof. Felix Schmid

DATE SUBMITTED: 2013-10-30

UNIVERSITY OF
BIRMINGHAM

University of Birmingham Research Archive

e-theses repository

This unpublished thesis/dissertation is copyright of the author and/or third parties. The intellectual property rights of the author or third parties in respect of this work are as defined by The Copyright Designs and Patents Act 1988 or as modified by any successor legislation.

Any use made of information contained in this thesis/dissertation must be in accordance with that legislation and must be properly acknowledged. Further distribution or reproduction in any format is prohibited without the permission of the copyright holder.

Executive Summary

Railway signalling has experienced numerous changes and developments, most of which were associated with its long evolutionary history. These changes have occurred gradually from the earliest days of the railway industry when fairly safe distances between the trains were controlled by signalmen with their rudimentary tools to multiple aspects colour light signalling systems and complicated operating systems as well as computerised traffic information systems. Nowadays signalling technology is largely affected by the presence of high performance electromechanical relays which provide the required logic on one hand and securely control the train movement on the other. However, this kind of control system is bulky and requires large space to accommodate. Therefore, such a technology will be expensive as it requires intensive efforts for manufacturing, installation and maintenance.

The mechanical interlocking is an alternative substitution for the electromechanical relay-based systems. All interlocking systems have been developed by considering all the safety issues and existing standards within the railway signalling. The current project introduces a new electromechanical approach which utilises the real time data from a mechanical unit and produces the outputs in the form of electrical signals. To achieve this, an electronic circuit has been designed to use the extracted data to represent the status of each lever within the mechanical interlocking. Ultimately, this electronic circuit determines two aspects of signalling concerning straight and reverse route.

This procedure is performed with the aid of magnetic contacts which are formed by two parts, magnet and reed switch. The magnets are fixed underneath each lever. As a lever is reversed, the reed switch will move to close proximity to the magnet and a propagated magnetic field by the magnet will trigger the internal switch to close based on their polarity. The reed switches are connected to a microcontroller. Each lever status (reverse/normal) will be sent to the microcontroller as an input. Consequently, the corresponding output due to activation of each lever will be generated from PIC microcontroller as an electrical signal with enough amount of current to activate the related LED. A track layout with two switch points has also been designed and fixed on a wooden board, which has two holes, at the location of the switches. The directions of the switches are controlled by

two AC motors, which are connected to the switches and energised with the help of a microcontroller and reed relays for a short period of time.

Acknowledgments

It is a pleasure to thank those who made this work possible; this project would not have been possible without their help and support.

Firstly, I would like to show my gratitude to Prof. Felix Schmid for his superior supervision throughout the whole period of this project. He was always willing to guide me from the initial to the final stage of the project. He encouraged me to be an independent thinker and have an engineering point of view. I owe sincere and earnest thankfulness to him for his support, careful guidance and understanding. I am sure it would have not been possible to complete my project without his great efforts.

I am truly indebted and thankful to Dr.Charles Watson for his efforts and great company. He has always been accessible and provided me with great information resources, superior teaching and good advice. He showed me the right way and I would have been lost without him.

I would like to thank all the members of the Electronic, Electrical and Computer Engineering Department, in particular Mrs Joy Grey and Mr Andy Dunn.

Ultimately, I dedicate my dissertation to my father who always supported me and encouraged me throughout my way, my mother who has always been a source of love to me and my beloved sister Mahsa. None of this would have been possible without their love, support and patience.

Table of Contents

Executive Summary	i
List of Abbreviations	x
CHAPTER ONE	1
1 Introduction	1
1.1 Motivation for Building Interlocking Machines	1
1.2 Objectives of an Interlocking	1
1.3 Project Scope	2
1.4 Methodology	2
1.5 Thesis Layout	3
CHAPTER TWO	5
2 Literature Review	5
2.1 Configuration and application of Interlocking	5
2.2 History of Interlocking	6
2.3 Past Interlocking Machines	8
2.3.1 Tappet Interlocking	8
2.3.2 Mutual Locking	9
2.3.3 Exclusion	9
2.3.4 Conditional Locking	10
2.3.5 Crossing	10
2.3.6 Junction	11
2.3.7 Locking Sheet	12
2.3.8 First Motion	12
2.3.9 Cam Plate	13
2.3.10 Saxby and Farmer Rocker	13
2.3.11 Saxby Gridiron Locking	14
2.4 Present Day Interlocking Machines	15
2.4.1 Safe Data Transmission	15
2.4.2 SSI System	15
2.4.3 The Interlocking Cubicle	16
2.4.4 Multi-Processor Module	18
2.5 Chapter Summary	18

CHAPTER THREE.....	19
3 Visualisation of Interlocking Systems through LabVIEW.....	19
3.1 Testing over Straight Route (set from A to B).....	20
3.2 Testing over Reverse Route (set from C to D).....	20
3.3 Logical Simulation of the Normal Route.....	21
3.4 Logical Simulation of the Reverse Route	23
CHAPTER FOUR	25
4 Simulation results using Proteus ISIS	25
4.1 System Approach	25
4.2 Design Description.....	27
4.2.1 Design Optimisation	30
CHAPTER FIVE	32
5 Introduction to Linear Machine and PECO Turnout Motor.....	32
5.1 Linear Machine	32
5.1.1 Working Mechanism	33
5.1.2 Linear Motor.....	35
5.2 PECO Turnout Motor.....	36
5.3 Software Implementation	38
5.4 Hardware Implementation.....	39
5.5 Introduction to Circuit Components	41
5.5.1 Reed Relay.....	41
5.5.2 Operational Amplifier.....	41
5.5.3 Diode	43
CHAPTER SIX.....	44
6 Experimental Setup and Practical Results.....	44
6.1.1 Case Study	46
6.1.2 Experimental Setup (second phase).....	48
6.2 Overview of Components in Final Circuit.....	48
6.2.1 Introduction to magnetic actuators and magnetic sensor:	48
6.2.2 Magnetic Contact.....	49
6.2.3 Sensitivity Evaluation of Magnetic Contact	49
6.2.4 Switch Bounce and Filter Circuit	51
6.2.5 Real Time Data Acquisition	53

6.2.6	Circuit Enhancement	56
CHAPTER SEVEN		58
7	Conclusion.....	58
7.1	Findings.....	58
7.2	Recommendation	58
7.3	Review of Approach	59
7.4	Areas for Further Work.....	59
7.5	Word count.....	60
8	References	61
8.1	Documents	61
8.2	Web Site Links.....	62
8.3	Bibliography.....	63
Appendix A.....		65
Appendix B.....		104

List of Figures

Figure 1: Project block diagram	3
Figure 2: General objectives of Interlocking Systems	5
Figure 3: Railway interlocking and main communication line	5
Figure 4: Old Interlocking (Lawrence, 2011)	6
Figure 5: Tappet Interlocking (Calvert, 2008)	9
Figure 6: Mutual Locking (Calvert, 2008)	9
Figure 7: Exclusion (Calvert, 2008)	10
Figure 8: Conditional Locking (Calvert, 2008)	10
Figure 9: Crossing (Calvert, 2008)	11
Figure 10: Junction (Calvert, 2008)	11
Figure 11: Locking Sheet (Calvert, 2008)	12
Figure 12: First Motion (Calvert, 2008)	12
Figure 13: Illustration of Cam Plate (Calvert, 2008)	13
Figure 14: Saxby and Farmer Rocker (Calvert, 2008)	14
Figure 15: Saxby Gridiron locking (Calvert, 2008)	14
Figure 16: SSI System (Waller, 1991)	16
Figure 17: Entire Installation of SSI System (Waller, 1991)	17
Figure 18: Track layout with signals and points	19
Figure 19: Straight route set from A to B	20
Figure 20: Reverse route set from C to D	20
Figure 21: Truth table for lever frame working mechanism	21
Figure 22: Simulation model for the straight route	21
Figure 23: Control unit for the straight route	22
Figure 24: Simulation model for the reverse route	23
Figure 25: Control unit for the reverse route	24
Figure 26: Simulation model concerning the straight route	25
Figure 27: Demonstration of preliminary setup	26

Figure 28: Levers are reversed	26
Figure 29: First shot from simulation	27
Figure 30: Activation of levers for normal route.....	28
Figure 31: Activation of levers for reverse route	28
Figure 32: Activation of lever 4	29
Figure 33: Design Evaluation	30
Figure 34: Lever's mechanism.....	31
Figure 35: Linear machine (Chapman, 1999)	32
Figure 36: Starting a linear machine (Chapman, 1999)	33
Figure 37: Linear motor (Chapman, 1999)	35
Figure 38: PL-10e PECO motor	37
Figure 39: PECO motor with bended central legs.....	37
Figure 40: Illustration of switches and part of the track layout on the baseboard.....	37
Figure 41: Pulse generation	38
Figure 42: Pulse generating flowchart.....	39
Figure 43: Circuit block diagram	40
Figure 44: NC and NO relays	41
Figure 45: Op-amp Symbol	41
Figure 46: Negative feedback network.....	42
Figure 47:LM741 internal architecture	42
Figure 48: Position of op-amp, reed relay and diodes.....	43
Figure 49: Demonstration of preliminary circuit.....	44
Figure 50: Negative voltage square pulse from the microcontroller	44
Figure 51: Detected pulse from output of op-amp	45
Figure 52: Experimental setup.....	45
Figure 53: Illustration of PNP transistor	46
Figure 54: Collected result from PNP transistor	47
Figure 55: Collected result from microcontroller.....	47
Figure 56: Demonstration of finial circuit.....	48

Figure 57: Magnetic actuator block diagram.....	48
Figure 58: Magnetic sensor block diagram	49
Figure 59: Reed switch sensitivity evaluation	49
Figure 60: The layout of the circuit	50
Figure 61: Pull-up connection	51
Figure 62: Switch bounce	51
Figure 63: Delay routine.....	52
Figure 64: Filter circuit.....	52
Figure 65: Two aspects of signalling (first shot).....	53
Figure 66: Interlocking (top-view)	53
Figure 67: Position of magnetic contact	54
Figure 68: Home signal	54
Figure 69: Station started signal	54
Figure 70: Distant signal	55
Figure 71: Station-started signal for the reverse route	55
Figure 72: Shunting signal and point.....	55
Figure 73: Shunting signal and point #2.....	56
Figure 74: LCD simulation result.....	57

List of Abbreviations

Term	Explanation / Meaning / Definition
AC, ac	Alternating Current
Back EMF	Back Electromotive force
BR	British Railway
DC, dc	Direct Current
GWR	Great Western Railway
KVL	Kirchhoff Voltage Law
LAN	Local Area Network
LCD	Liquid Crystal Display
LED	Light Emitting Diode
Op-Amp	Operational Amplifier
PCB	Printed Circuit board
PIC	Peripheral Interface Controller
SSI	Solid State Interlocking
TFMs	Trackside Functional Modules
MPM	Multi Processor Module

CHAPTER ONE

1 Introduction

1.1 Motivation for Building Interlocking Machines

In the first years of the railway industry, signals and switches were distributed over a predefined area in the vicinity of the stations. Switch tender or signalmen had to walk a long distance when signals used to be operated at their locations. An obvious problem was that each signalman was able to cover only a limited number of signals. Therefore, the switch levers were connected to the switches by rods and placed in an elevated platform. Signals on top of the elevated platform were operated by stirrups to keep them clear. A constant effort and supervision was required. Although the suggested system was affordable at first sight, it was impregnated with dangerous hazards (Calvert, 2008) .

The hazardous conditions occurred due to errors in operation of both signals and switches, because it was not possible to identify which lever relates to which switch and signal. Besides, signalmen were unable to observe the switches in operation. Consequently, they could not locate the operated signals and switches fitted for the certain route. In addition, if the driver could identify the applied signal, he was still in doubt for the proper place to stop. Hence, to mitigate hazardous situations in railway signalling systems and in order to have a safe and more logical operation, it is necessary to connect signals and switches together.

1.2 Objectives of an Interlocking

Signalling is known as the most vital section of the railway system because controlling and management and safe movements of trains are dependent on it. Signalling and train control systems have been improved over the years and recently they have developed as an extremely technical and complex industry. In the mid-19th Century, mechanical interlocking was introduced, in which signals and points were connected together to perform more secure and logical operation. However, there was no connection between signals and switches. As a result, certain routes were conflicted by different contradictory signals and train collision would ultimately occur. The purpose of such a safety feature in the railway industry was to provide a clear and safe operation by preventing the route for a train being set up and its

protecting signal cleared if there was already another signal in operation (Signalling, 2013) .

1.3 Project Scope

The aim of this project is to introduce a new electromechanical approach that uses the real-time data from a mechanical unit and demonstrates system outputs as electrical signals. An electronic circuit is designed to extract the information about the status of each lever within interlocking (made by Atkins) and effectively indicate the two aspects of signalling concerning both the straight and reverse route. This task has been performed with the aid of magnetic contacts, which are formed by two parts, magnet and reed switch. The former is fixed underneath each lever and as the lever gets reverse, the latter part will move to a close proximity to the magnet. The propagated magnetic field will trigger the internal switch to close based on their polarity. The reed switches are connected to a microcontroller. Each individual lever status (reverse/ normal) will be sent to that microcontroller (PIC) as a system input. Consequently, due to the activation of the levers, the corresponding electrical signal outputs will be generated from the PIC which have sufficient amount of current to activate the related LED. A track layout with two switch points is designed and fixed on a wooden board, which has two holes at the switches' location. The directions of the switches are controlled by two AC motors, which are connected to the switches and are energised by the PIC and reed relays for a short period of time.

1.4 Methodology

According to the scope of the project, several phases are covered. The objective of this project is to obtain information from the interlocking lever frame in order to illustrate two aspects of signalling. To determine the scope of the research methodology, the main issues are as follows:

- Comprehensive literature review about railway signalling
- Visualisation of the interlocking system through Lab VIEW
- Signalling-based simulation on the control logic of interlocking and behaviour of every single lever
- Investigating appropriate solutions for real-time data acquisition
- Demonstrating two aspects of signalling due to the obtained data through an electronic circuit

The following block diagram illustrates the procedure in its entirety.

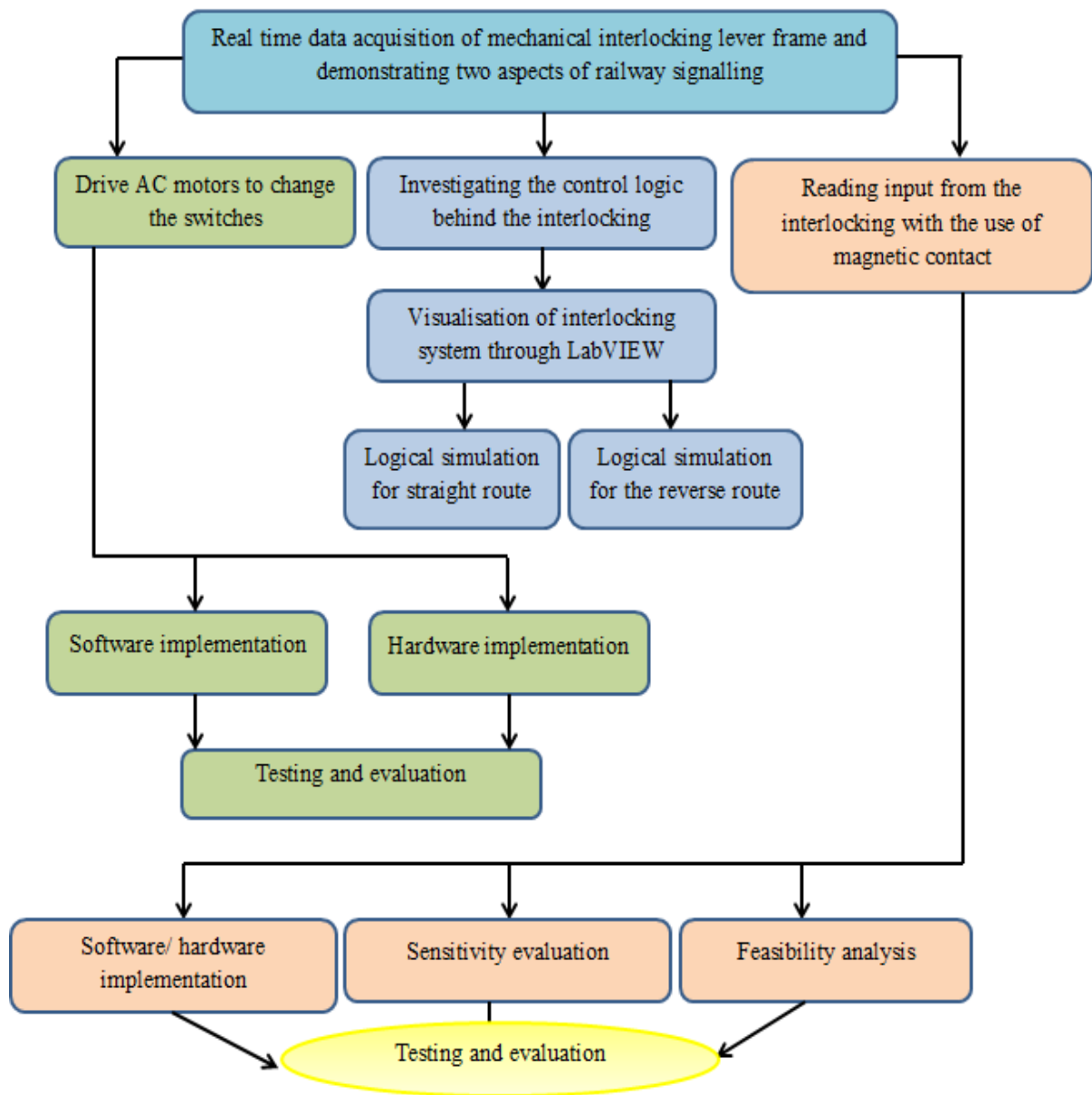


Figure 1: Project block diagram

1.5 Thesis Layout

This thesis is organised into seven chapters. Chapter one briefly explains the aim and objectives of the project, as well as the project's scope and methodology. Chapter two contains a literature review and history of the interlocking machine. Chapter three concentrates on system visualisation and demonstrates functionality of the lever frame due to both straight and reverse route which is accomplished by LabVIEW software. Chapter four presents the operation mechanism of the mechanical interlocking lever frame through three

different simulation models. In chapter five basic principles, characteristics and the working mechanism of the linear machine are discussed. The analysis of the experimental setup, practical results, the schematic of the entire system and the finalised circuit are discussed in chapter six. Chapter seven includes the conclusion, problems and future work and suggestions.

CHAPTER TWO

2 Literature Review

2.1 Configuration and application of Interlocking

Interlocking has been introduced as the most vital section for train control movements as it determines fairly safe operation by considering and managing of signals, point machines, crossings at grade and movable bridges as well as other field elements. Fundamental principles of an interlocking are listed as below:

- Pre-defined restrictions for signal operation, in which signals may not operate at the same time to allow conflicting train movements to take a place.
- Points and other field elements in the route need to be in position or set properly prior any permission for train operation.
- As route get sets and train receives the corresponding signal. Train can proceed over that route safely. Whilst all other movable appliances are locked on their location.
- All the locked components in the route need to hold locked until the train pass that portion of route or either the signal to proceed is withdrawn. Therefore, enough time has been given, to guarantee that the approaching train from the route has got enough time to proceed over the route and stop before passing signal.

Figure 2 determines general objectives of interlocking systems.

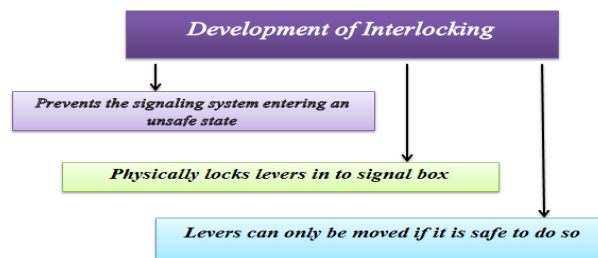


Figure 2: General objectives of Interlocking Systems

The status information of each field elements will be transferred to train control systems, which is connected to interlocking components.

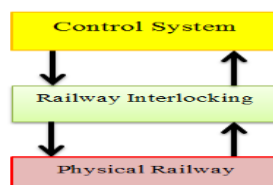


Figure 3: Railway interlocking and main communication line

2.2 History of Interlocking

At the beginning of the railway industry, signalmen were in charge of operating signals by pressing down the stirrups with their foot. The stirrups were arranged at Bricklayer's Arms junction by Charles Hutton Gregory in 1843. As one stirrup was pressed, it interfered with other stirrups, hence there was not any contradictory signal and the installed points on the ground were operated separately from the signals. In 1856, for the first time both points and signals levers were arranged to install in a row by means of spring catch. The invented frame by Saxby was installed at Keyham Junction in Brighton. Although it was the first interlocking which connected both signals and points, in terms of the operation principles it was completely different from a modern interlocking. Accordingly, as a point lever was pulled for a specific route, at the same time, the corresponding signals were operated. On the other hand, in modern interlocking, signals will operate once the operation of points is completed. A great improvement of Saxby's machine, which was installed at Keyham and moved both signals and points at the same time, appeared in 1860. According to this new technology, the locking bar at the top of the frame is driven with the movement of each lever. The essential levers are locked in the upper jog, so the middle part the locking bar is static. Whereas, in the lower jog the essential levers are free to move, which introduces early and late motion within the levers' frame. The Saxby and Farmer Hook Interlocking were installed at the new Victoria Station, London. This machine was the first interlocking machine imported into United State and installed at Trenton in 1870. In 1874, a rocker type with gridiron locking machine was used for East Newark for the first time.

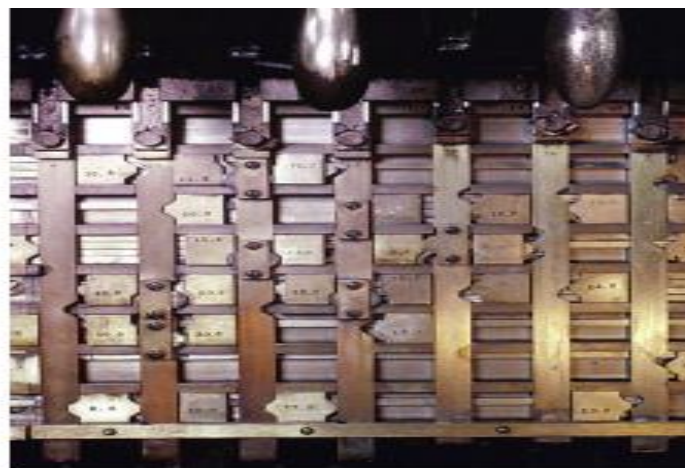


Figure 4: Old Interlocking (Lawrence, 2011)

In 1859, the points and signals operated independently one after the other. It was the first modern sense of interlocking introduced by Steven and Sons in collaboration with Austin Chambers. Based on the provided interlocking machine, signal stirrups get into the holes with the usage of rods and move by the means of point levers. Therefore, signals were only cleared as the rod gets into all the plates. In 1880, the invented machine by Steven and Sons earned good reputation as it was simple and easy to maintain (Calvert, 2008) .

Chambers' idea was improved by Pierre Auguste Vignier, who was the inventor of the interlocking between points and signals for four junctions. The devised machine by Vignier was similar to Chambers', but both rods and holes were redesigned and moved underneath the floor where the cabin was positioned.

In 1874, the levers were combined with a large rotating motion around the axis which is known as Rapiers' locking. The locking was located in the middle of notches and locking bars. The related locking bar to each lever is driven with the use of a cam. All the interlocked levers with certain levers are locked at the occurrence of the first motion of the levers and at the last motion of the levers, the required levers are unlocked. An 80-levers frame of Rapier was installed at Lincoln (In on the fixed signals of railways, 1874) .

The first American interlocking machine was designed in 1870 by Toucey and Buchanan, and it was installed at "Duyvil at the north end of Manhattan" in 1875. In 1888, a locking machine similar to that of Saxby and Farmer's machine was designed by Arthur Henry Johnson with the cooperation of his nephew C.R. Johnson. However, in this machine the links were connected in reversed manner and under the ground. Moreover, "the Webb and Thompson electric train staff machine" was introduced in the United States by Johnson in 1892 (Calvert, 2008) .

The devised machine by Johnson was applied for the first time between Savanna, Illinois and Sabula, Iowa. The American Interlocking Machine was either like the Saxby and Farmer machine or the Johnson machine. The produced machine by National Signal Company was almost the same as the Saxby machine, but the tappet locking was organized to locate vertically and the links were placed under the lever. The inventor of this idea was J.A.Bonnell. In 1865, Michael Lane who was a railway engineer and worked on the Great Western Railway devised a new frame. In his frame a rack was attached on the lever and a

pinion was driven by the means of that rack and the rotation of the shaft was from left to right.

Although the provided frame by Lane was a fairly safe locking, it was unable to produce early and last motion within the lever. Consequently, if two unlocked and contradictory levers were applied, the result was a jammed frame. In 1860, the Great Western Railway's locking machine was installed at the junction of Paddington (GWR 2013). In 1863, T.Blackall devised a frame which was made in Reading and was applied to the departure part of Paddington station. In 1885, all the Great Western Railway's signal machinery was made in Reading with the efforts of T.Blackall. In 1872, Tom Gooderson invented a single twist frame which supplied a quick first motion and improved Lane's machine. Both first and last motions were introduced by double twist machines, which were improved by Great Western Railway. The improved twist frame, a twisted bar was rotated corresponding to the lever's motion and locking was driven by the rotation of a twisted bar.

HT3 was the last form of the mechanical interlocking machine, which was used by the Great Western Railway (1906-1926). The HT3 locking machine included locking trays which were located in both a vertical and a horizontal manner and driven by the means of bell cranks. Moreover, each tappet blade of the HT3 machine had three bridles. VT5 was another version of the HT3 locking machine with 5 bridles on each tappet blade appeared in 1926-1966. In the VT5 machine, the lever was driven by the motion of both tappet blade and cam plate. In 1943, the largest VT5 frame with 222 levers was "At Reading Main Line West" (GWR, 2013) .

2.3 Past Interlocking Machines

2.3.1 Tappet Interlocking

In 1870 James Deakin created an effective and straightforward method for interlocking the levers, which is tappet interlocking. Based on this method, the levers force the notched tappet blades, sword irons or plungers to move directly or with the help of any other intervening mechanisms. The bridles or locking bars are perpendicularly located on the tappet blades, which are capable of moving sideways, see figure 5. The tappets were riveted to the bridles and fixed in the locking bars through the notches. The tappet is engaged with blades and as the blades move, the tappet will move to the right or left accordingly. Sometimes, the tappets

are occupied with the blades and the bridle cannot be moved. In such a case the blades will not move as movement of blades is dependent on the movement of the bridle (Calvert, 2008) .

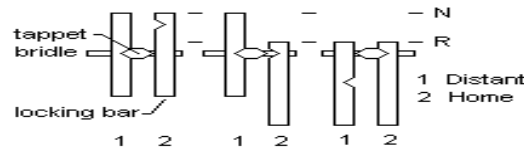


Figure 5: Tappet Interlocking (Calvert, 2008)

2.3.2 Mutual Locking

Figure 6 illustrates the interlocking mechanism of levers based on both home signal and distant signal. As it can be seen, as far as the home signal (2) is normal at stop, the distant signal (1) does not have the ability to be cleared (2N/1N). As soon as the home signal becomes reversed the distant signal lever can be operated, if required. The distant signal (lever 1) is locked by the home signal (lever 2). Therefore, as long as lever 1 (distant signal) is reversed, lever 2 is locked (1R/2R). The interlocked levers are always considered as pairs and a reciprocal relation always exists between them. Once the distant signal is reversed, the home signal (2) is not able to return back to normal state, which is known as back-locking. In back-locking a clear distant signal cannot be followed by a stop signal (Calvert, 2008) .

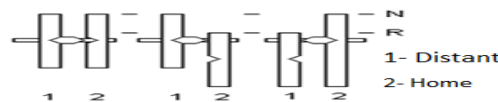


Figure 6: Mutual Locking (Calvert, 2008)

2.3.3 Exclusion

Figure 7 clearly shows that the levers can be reversed at the same time, as they are both able to operate signals and, consequently, conflicting routes. However, they can be in the normal state at the same time and they are allowed to be reversed and are permitted to have corresponding movement. Based on the above, as lever 2 is reversed, lever 1 is locked to the normal state (2R/1N) and vice versa (1R/2N), this indicates the reciprocal relation between each pair. In addition, the locking might be termed exclusion. In such a case lever 2 reversing is locked by lever 1 normal and vice versa (1N/2R and 2N/1R). Usually, all the levers can

hold the normal state at the same time in such a case exclusion does not exist (Calvert, 2008) .

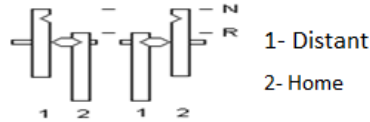


Figure 7: Exclusion (Calvert, 2008)

2.3.4 Conditional Locking

Figure 8 illustrates conditional locking, which became possible through the tappet locking.

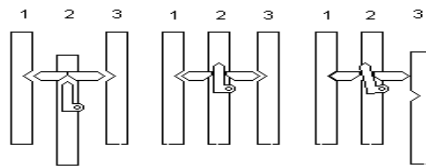


Figure 8: Conditional Locking (Calvert, 2008)

As the above diagram shows (at the left), the lever 2 gets reversed both the lever 1 and lever 3 are free to move in any way. The movable portion will be located between two tappets (1 and 3) once lever 2 get locked into the normal state. This will allow lever 1 to locks lever 3 to normal state and vice versa. Indeed, whenever lever 2 get normal, lever 2 will get lock to normal by lever 1. Hence, the position of lever 2 is governing the relation and state of lever 1 and 3. Indeed, once lever

2.3.5 Crossing

Having home signal for each direction and a unique distant signal for each home signal is required to protect level crossing properly as shown in figure 9. In this case we consider home/distant signal with two aspects of signalling. Although eight levers will be required in reality, to make it simpler, let us assume there are only four levers which can combine home and distant signals. Accordingly, the two home signals cannot be reversed simultaneously and a distant signal should be reversed whenever the related home signal is reversed. Assuming levers 1, 2, 3 and 4 as home signals, lever 1 is reversed and locks lever 2, 3 and 4 normal. This indicates that lever 1 will be locked to normal by lever 2 reversed. Moreover, lever 2 reversed locks levers 3 and 4 to normal. Ultimately, lever 4 should be locked to normal state by lever

3. As is illustrated in figure 9, as the signal lever is reversed, all the other levers will be locked to the normal state.

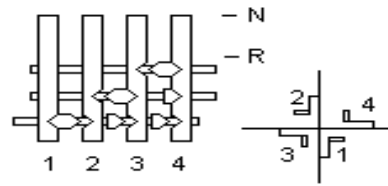


Figure 9: Crossing (Calvert, 2008)

In fact, we need to consider locking with higher-numbered signals. Correspondingly, when considering the switches, there is a need to consider locking with higher-numbered switches. Consequently, switch levers are locked by signal levers. Based on the above, each pair of levers will be considered once to "avoid the provision of superfluous interlocking" (Calvert, 2008).

2.3.6 Junction

Figure 10 shows junction. Although both the distant signals and the facing point lock are not presented here, but they will normally nominated.

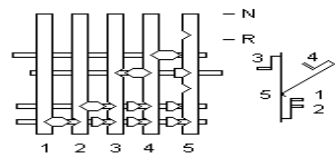


Figure 10: Junction (Calvert, 2008)

The distant signal have only one arm and it will be cleared once the straight route with all the corresponding field elements were completely set. Accordingly, lever 1 and 2 will be pulled for straight-through movement and also movement to the branch respectively. The mentioned movements cannot happen at the same time as the corresponding arms for them (arms 1 and 2) cannot get lowered at the same time. The situation will fully different for the arms 1 and 3 as they both can get lowered (reversed) once the lever 5 gets lock to the normal state. Therefore, as the lever 5 gets reversed, arms 2 and 4 will get lowered.

2.3.7 Locking Sheet

Figure 11 presents a locking sheet for the mentioned interlocking. In this locking sheet, the reversed lever is presented by a circle around the lever's number.

lever	when	locks
1		5 2 3 4
2		⑤ 3 4
3		5
4		⑤

Figure 11: Locking Sheet (Calvert, 2008)

The above locking sheet can be read as follow:

- Levers 2, 3, 4 and 5 will be locked to normal state, once lever 1 reversed.
- Levers 3 and 4 will be reversed and the fifth lever will locked to normal, once lever 2 reversed.
- Levers 5 and 4 will be locked to normal and reverse state respectively, as the lever 3 get reversed.
- The "when" column is empty as there is not any conditional locking.

2.3.8 First Motion

In order to move the tappet from the notch and lock related levers, motion of locking bars within one-half of the tappet is required. In last moving part of the locking bars, all the associated levers are completely unlocked. It is important to bear in mind that both first and last motion of the locking bars is necessary for having a safe operation of the frame.

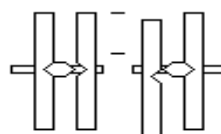


Figure 12: First Motion (Calvert, 2008)

Assume a short distance in one lever's movement, an effective attempt to move the second lever (if the second lever is not locked) could conflict the routes. The frame can get damaged in the case of considerable leverage as the locking bars are moved by levers directly. Thereby, it is necessary for all the levers that are going to be locked to be locked in the first motion of the lever, and all the levers that need to be unlocked, to get unlocked on the last motion.

In order to provide an early and last movement, the easiest solution is to create the stroke of the locking bar larger than the width of the tappet which was done by Steven and sons through a small frame which was often used in ground frames.

2.3.9 Cam Plate

The provided lever locking model by Steven was inappropriate for small frames as it was huge and required significant space. Accordingly, a Railway signal company in Liverpool found an alternative, where the lever moves a cam plate that then moved the locking bars, the inventor of this idea is George Edwards. The early and late motions are provided by the S-shaped cam slot. (Calvert, 2008) In 1904, this method was applied on the Great Western Railway's HT and VT tappet interlocking frames (GWR, 2013) .

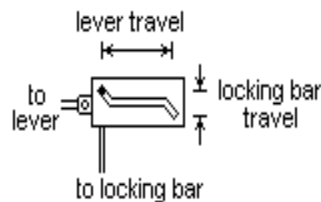


Figure 13: Illustration of Cam Plate (Calvert, 2008)

The general logic behind the provided method is based on locked levers; as a lever gets locked, small trail movement can identify that. Despite all, the locking has to be made very large and bulky and has to be installed underneath the operating level in a vertical manner. Further, electric locks are located at the end of each tappet blade.

2.3.10 Saxby and Farmer Rocker

It is not required to mechanically interfere with movement of lever as it is enough to avoid the catch dog to run off from the quadrant. Moreover, as the catch handle is released, the movement of catch rod is sufficient enough to force the interlocking to work. In addition, the motion of catch dog which happened at the start and end of each lever movement can perfectly assure both early and late motion. Earlier than any motion of the levers locking is actuated. Accordingly, a skilful arrangement was provided by Saxby, which was based on that the catch rod motivates the rocker or link to move. As the lever is in the normal state, the depressed catch handle drives rear end of the link up and eventually supplies half of the first motion. Both lever and link slot have the same centre of curvature. Therefore, as lever moves from normal to reverse state the link remains in static mode. The second and final half of the

first motion occurs as the front part of the link becomes more depressed through the downward motion of the catch rod and the released catch handle.

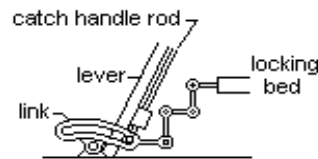


Figure 14: Saxby and Farmer Rocker (Calvert, 2008)

Based on the above, two general ideas are identified:

- Locking levers through locking their catches
- Motion of catch rod which drives interlocking

The inventor of the first idea was Easterbrook, whilst the locking operation by the catch rod was suggested by Saxby.

2.3.11 Saxby Gridiron Locking

The gridiron interlocking was an initial catch-locking machine without tappet interlocking. With the help of cranks, the provided locking machine moves the links by the catch handle rods known as gridirons (flops). As all levers enter in a normal state, the gridirons will be horizontal. In figure 15 upward view of the locking machine (only the upper locking bar) is illustrated which shows five slots gridiron with five locking bars at below and five locking bars at above.

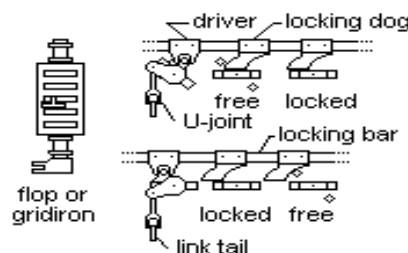


Figure 15: Saxby Gridiron locking (Calvert, 2008)

In this method of locking, the locking bar was motivated to move by means of a pin and one gridiron's driver. Moreover, a gridiron was either driven freely to rotate or to be prohibited from any movement by locking dogs which were located on the locking bars. Figure 15 clearly indicates both locked and free gridirons. The catch of a locked gridiron cannot be released. This locking mechanism was fully approved and accepted in France. However, in Britain and America the provided model was changed by inserting tappet locking around. The

gridiron interlocking has been known as an adequate interlocking. Besides, there is no need for having a lever to exert force. This is made in a smaller size and lighter weight compared to lever locking. Moreover, this locking was located on the operating level behind the levers, whilst the lever locking interlocking which is heavier has to be located under the operating floor. The general logic behind the gridiron locking is almost the same as the suggested locking frame by John Imray. In Imray's Locking frame, rotating cylinders get blocked by located locking dogs on locking bars.

Ultimately, the locking mechanism before lever movement is known as preliminary locking. According to the preliminary locking, all locking procedures have to be finished prior to the movement of the lever and all unlocking procedures have to be done afterwards. The preliminary locking in lever locking cannot be achieved precisely, but in catch locking it can.

2.4 Present Day Interlocking Machines

2.4.1 Safe Data Transmission

Railway signalling is a main example of data transmission between different places. Hence system reliability is dependent upon the information received being similar to the one sent, in the context of fail-safe system. Over the years, since the advent of computer-based safety systems, well-developed techniques have been used to protect data which is sent over serial telecommunication channels (Waller, 1991) .

2.4.2 SSI System

Up until the 1980s, the functionality of the relays was well defined. Whilst the behaviour of electronic was not. The mentioned situation has been reviewed by advent of microprocessors. Hence, a new generation of SSI (Solid State Interlocking) was designed. The designed system was very much safer in compare with corresponding relay counterparts.

SSI system mainly consists of two parts: a control centre and line side subsystems. An important part of the control centre is interlocking, in which an essential logic is implemented to provide safety commands for the other parts of the SSI system (line side subsystem). Moreover, a secure communication line between interlocking has been provided by the means of an internal network on the control centre part, which is based on information corresponding to each signalling area that will be exchanged in a secure manner.

In line side subsystem, a fail-safe LAN has been formed by the lineside data cable and TFMs. This effectively drives "signals, points and other signalling equipment according to command

telegrams generated by interlocking, and transfers back to the interlocking information from track circuits, points and other equipment" (Waller, 1991) .

Figure 16 illustrates the schematic form of SSI system as well as communication line between subsystems.

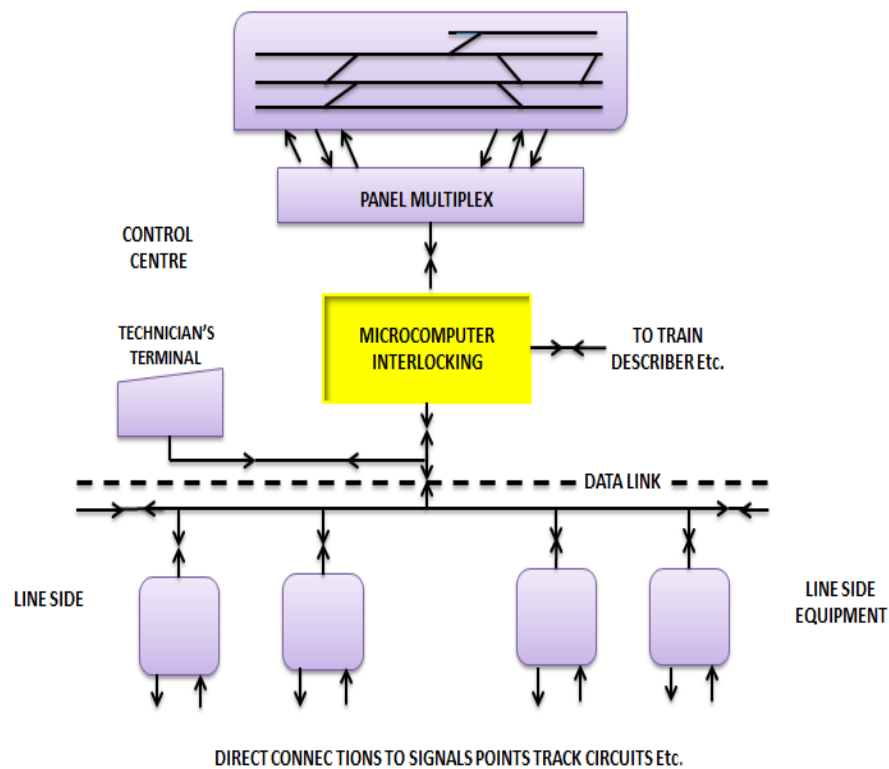


Figure 16: SSI System (Waller, 1991)

2.4.3 The Interlocking Cubicle

Figure 17 shows a common SSI installation. The important equipment of interlocking is accommodated in a cubicle with a 19-inch size.

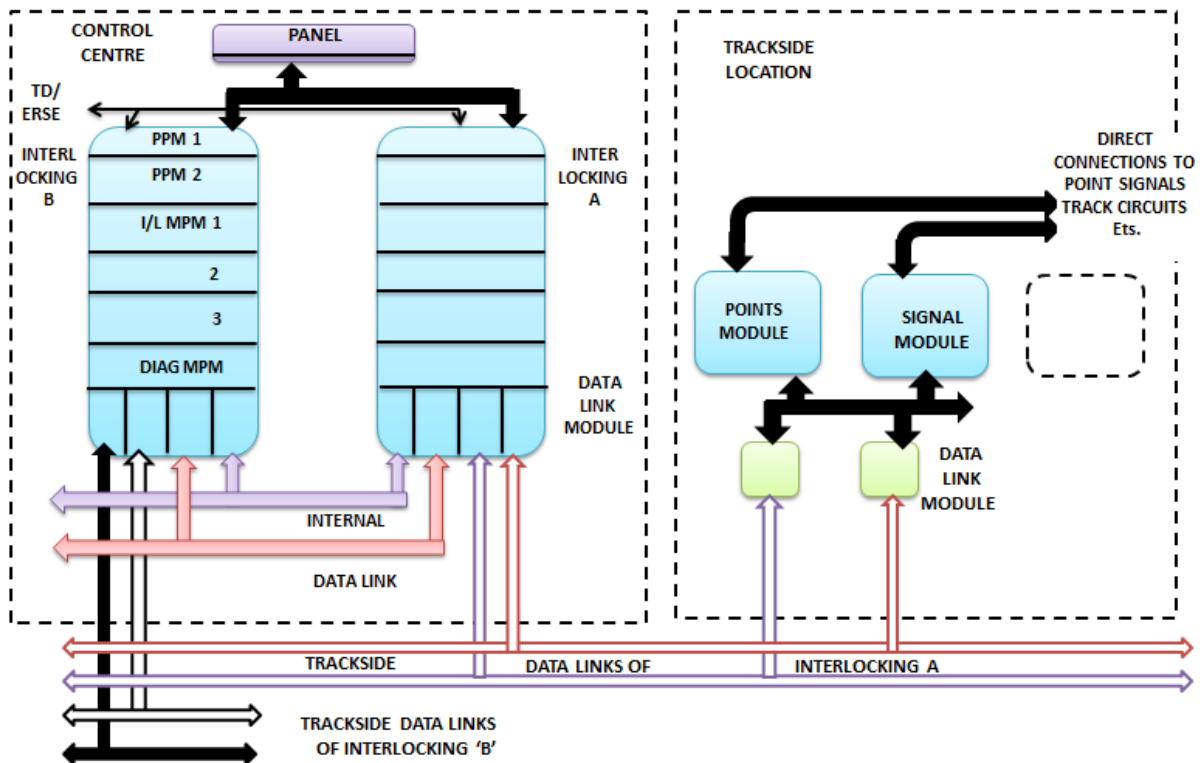


Figure 17: Entire Installation of SSI System (Waller, 1991)

The size of the controlled area by each interlocking is dependent on the number of trackside equipment. To extend the areas more than one interlocking is required since each interlocking is only capable of controlling 20 to 40 sets of points and managing 40 signals in total. The following are the equipment which is located in the interlocking cubicle: Three interlocking multi-processor modules

- Two panel processor modules
- The diagnostic multi-processor module
- Data link modules and power supplies

The interlocking cubicle wiring to multi-processor module includes an 8-bit binary address. Each wire link is specified for one bit address and is made to either high or low voltage, 1 or 0. Five bits represent the system identity, and identify the individual interlocking within the control centre. The remaining three bits are available to distinguish successive versions of geographical data. The designed interlocking requires power at 110 VAC and the amount of power consumption for each interlocking cubicle is 375 watts approximately. In addition, the

hardware of the SSI system is managed by the software named by MPM, which provides a safe operation and secure data transmission. The MPM software is divided into programs providing initialisation, redundancy management, interfacing with panel processors and communications links.

2.4.4 Multi-Processor Module

Multi-processor module is the name of software, which is applied in SSI systems for a railway network. The MPM software may contains one processor for receiving commands by considering the operating parts of trackside equipment and also transporting other corresponding commands relates to present state of other trackside equipment. The second processor will be applied to achieve the final commands from the first one, which ultimately would have an effect on operation within trackside equipments through a data link.

The first processor runs a higher speed as its clock speed is more than that of in the second processor. However, the speed of the first processor for some of the program, which interface with the second processor has been arranged to be slower.

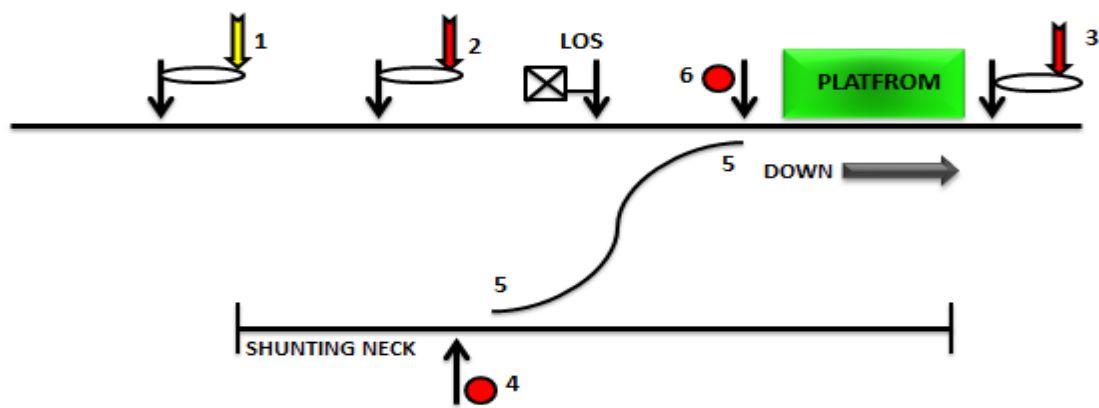
2.5 Chapter Summary

The history of interlocking control systems from the earliest days of the railway industry when fairly safe distances between trains in operation were governed by signalmen with their rudimentary tools to complicated signalling operating systems has been covered in this chapter. In addition, various types of locking machines invented from the earliest days of railway industry to the present interlocking and computer-based signalling systems, which are applied to the modern railway industry, have been introduced. The functionalities and weaknesses of each invented locking machine as well as their working mechanisms are described.

CHAPTER THREE

3 Visualisation of Interlocking Systems through LabVIEW

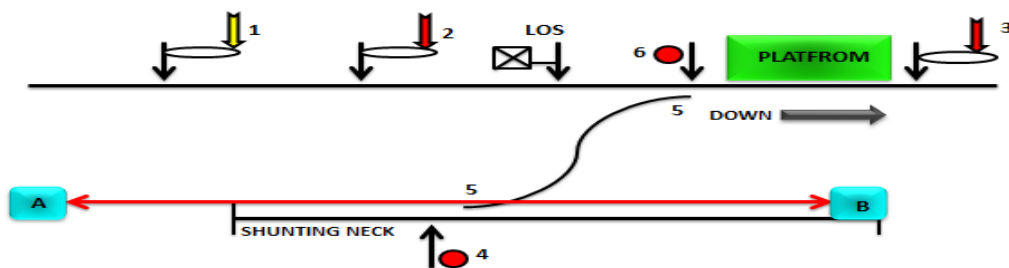
This chapter explains the working mechanism of the mechanical interlocking lever frames. The idea is to identify the general logic behind each lever's movement and then to simulate the behaviour of the levers with respect to both straight and reverse route through LabVIEW software. Testing procedure of the mechanical interlocking lever frame is performed twice with regards to straight and reverse route using a shunt neck. Figure 18 below shows a track layout with an interlock, which is divided into two separate routes.



Lever Number	Control Signal
Lever 1	Distant Signal
Lever 2	Home Signal
Lever 3	Station-started-Signal
Lever 4	Shunting Signal
Lever 5	Shunt Point
Lever 6	Shunting Signal

Figure 18: Track layout with signals and points

The first test requires levers 1, 2 and 3 to be unlocked in order to clear the route by making distant signal, home signal and station-started-signal to green respectively. However, based on the working mechanism of the interlocking frame, lever 2 has to get reversed first to clear the home signal to green. Due to safety reasons, preventing accidents and providing collision-free operation, the platform needs to be clear. Hence, lever 3 has to get unlocked to make the station-started-signal to green. Ultimately, lever 1 has to get reversed, which makes the distant signal turn double yellow. This is the normal and safe approach of setting the route for trains to pass from departure A to destination B.



3.2 Testing over Reverse Route (set from C to D)

The diagram illustrates a railway track layout. At the top, a horizontal track has several segments: a green circle labeled '1' with a red arrow pointing down, a green circle labeled '2' with a red arrow pointing down, a square labeled 'LOS' with a red arrow pointing down, a green circle labeled '6' with a red arrow pointing down, a green rectangle labeled 'PLATFORM', and a green circle labeled '3' with a red arrow pointing down. Below this track, a curved track segment labeled '5' leads to a green rectangle labeled 'C'. A dashed red line connects 'C' to a green rectangle labeled 'D' on the left. Below 'D' is a horizontal track segment labeled 'SHUNTING NECK'. A green circle labeled '4' with a red arrow pointing up is located below the 'SHUNTING NECK' segment. A dashed red line also connects '4' to the curved track segment labeled '5'.

MRes Thesis

Figure 21 explains the working mechanism of mechanical interlocking lever frame.

No	Description	No	Released by	Locks		Releases
				Normal, Both Ways		
1	Down Distant	1	2 & 3			
2	Down Home	2		5, 6		1
3	Down Starting	3			5	1
4	From Sliding Shunt	4		6w 5 R	5	
5	Siding Points	5		3		
6	From Down Main shunt	6		2(4w 5R)	5	

Figure 21: Truth table for lever frame working mechanism

3.3 Logical Simulation of the Normal Route

For further understanding and to show how a 2-aspects signalling with mechanical interlocking operates, the behaviour of each lever due to the state (Lock/Unlock) of other levers has been examined. Figure 22 shows the first simulation model for the straight route using three levers.

In 1	In 2	In 3	Behav1	Behav2	Behav3
0	0	0	dis	en	en
0	0	1	dis	en	en
0	1	0	dis	en	en
0	1	1	en	en	en
1	0	0	en	dis	dis
1	0	1
1	1	0
1	1	1	en	dis	dis

Figure 22: Simulation model for the straight route

The model has 3 inputs (due to associated levers) and eight possible outputs. In the design levers are designed as push button so two states of each push button (0 and 1) model the functionality of each lever ON and OFF (0 and 1). The outputs of the system show each lever's moving ability. Hence, based on three inputs and two states , eight conditions have been tested on the first three levers, in which 6 states out of 8 are experimentally achievable, however, not all of them are logically safe.

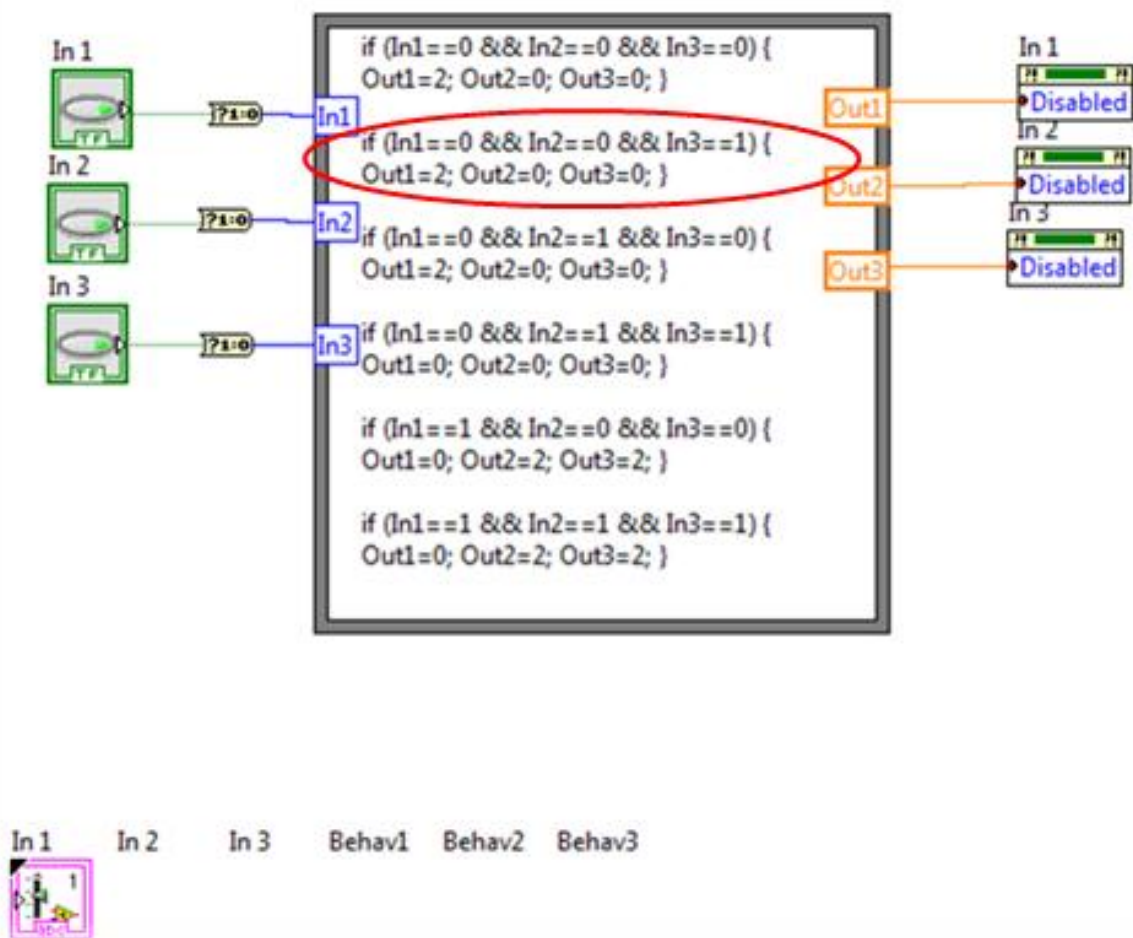
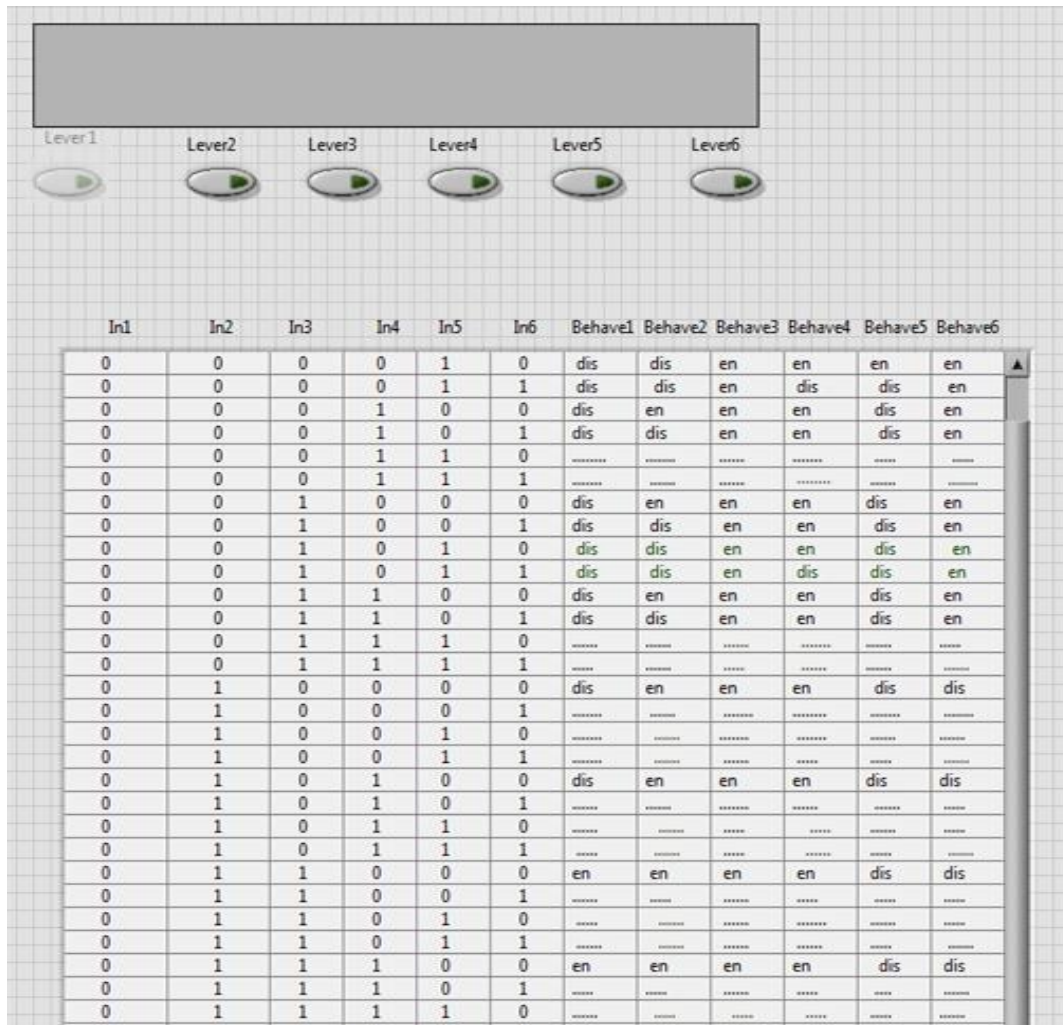


Figure 23: Control unit for the straight route

As is obvious from figure 23, the states of the push buttons are governed by six if statements (due to 6 possible outputs) inside a formula node . Let us consider the first and fourth if statement inside a box. The first if statement disabled In 1 by giving its corresponding output value 2 and it also enabled both In 2 and In 3 when all the push buttons are disabled. In the fourth if statement both In 2 and In 3 are enabled. Therefore In 1 will be enabled, which shows that the state of the first pushbutton is governed by state of the second and third one.

3.4 Logical Simulation of the Reverse Route

The same strategy has been adopted to simulate the functionality of the levers due to reverse route. Figure 24 shows the second model in which 64 states need to be examined. Only 18 states out of 64 states could physically happen. However, not all of them are certain to happen.



In1	In2	In3	In4	In5	In6	Behave1	Behave2	Behave3	Behave4	Behave5	Behave6
0	0	0	0	1	0	dis	dis	en	en	en	en
0	0	0	0	1	1	dis	dis	en	dis	dis	en
0	0	0	1	0	0	dis	en	en	en	dis	en
0	0	0	1	0	1	dis	dis	en	en	dis	en
0	0	0	1	1	0	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXX	XXXXX
0	0	0	1	1	1	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	0	1	0	0	0	dis	en	en	en	dis	en
0	0	1	0	0	1	dis	dis	en	en	dis	en
0	0	1	0	1	0	dis	dis	en	en	dis	en
0	0	1	0	1	1	dis	dis	en	en	dis	en
0	0	1	1	0	0	dis	en	en	en	dis	en
0	0	1	1	0	1	dis	dis	en	en	dis	en
0	0	1	1	1	0	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	0	1	1	1	1	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	0	0	0	0	dis	en	en	en	dis	dis
0	1	0	0	0	1	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	0	0	1	0	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	0	0	1	1	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	0	1	0	0	dis	en	en	en	dis	dis
0	1	0	1	0	1	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	0	1	1	0	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	0	1	1	1	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	1	0	0	0	en	en	en	en	dis	dis
0	1	1	0	0	1	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	1	0	1	0	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	1	0	1	1	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	1	1	0	0	en	en	en	en	dis	dis
0	1	1	1	0	1	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	1	1	1	0	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX
0	1	1	1	1	1	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX

Figure 24: Simulation model for the reverse route

Figure 25 is presenting the formula node for the second simulation. According to figure 24, there are 18 if statements inside the control unit, which will disable/ enable each push button by giving them value 2 and value zero to their corresponding outputs.

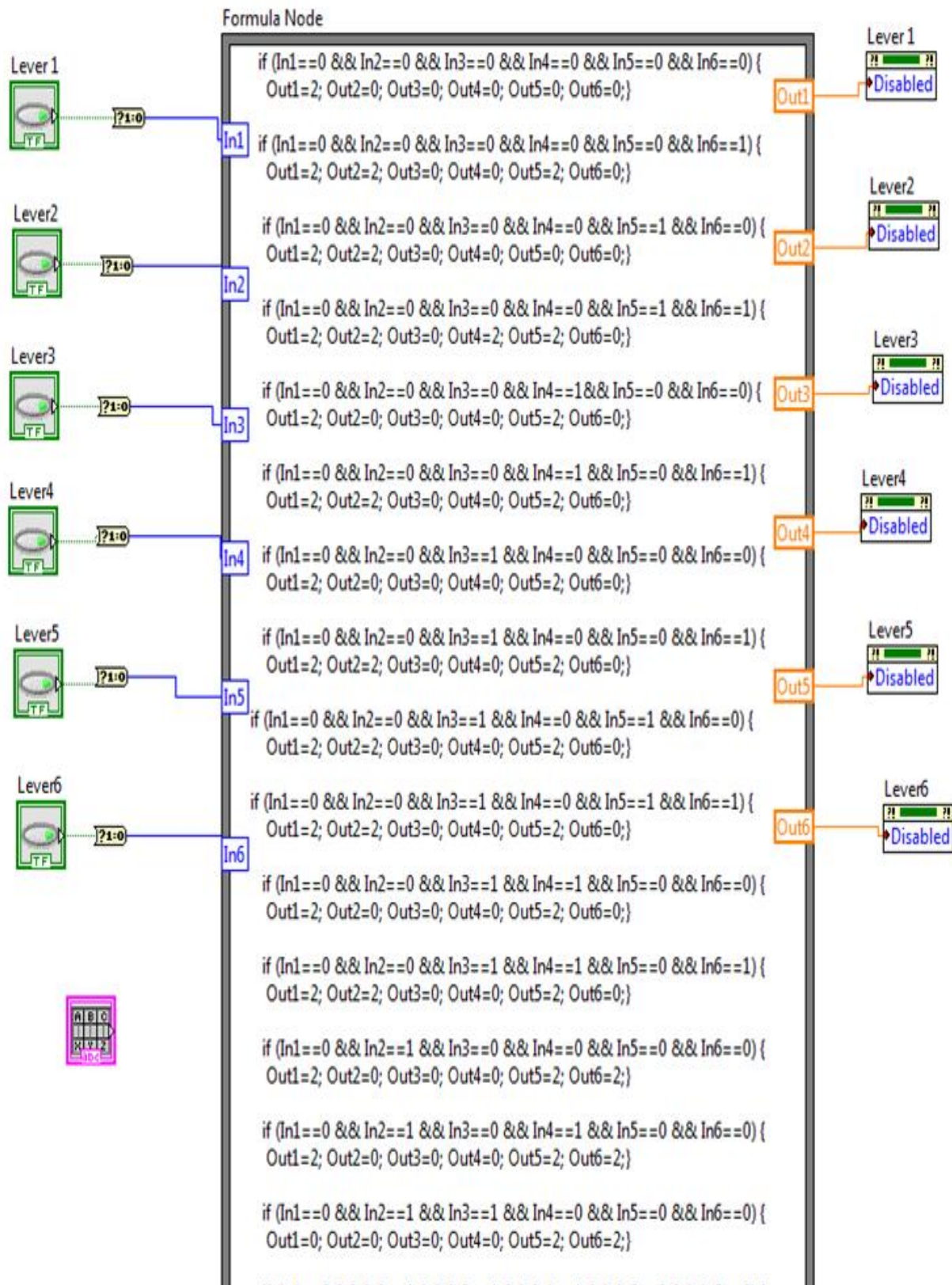


Figure 25: Control unit for the reverse route

CHAPTER FOUR

4 Simulation results using Proteus ISIS

4.1 System Approach

In order to model the working mechanism of the interlocking and to show how two- aspect signalling (red/ green) is created based on the state (lock/unlock) of each lever, a set of codes have been written through MPLAB software. The selected PIC microcontroller is PIC18F452, which has 40 pins and 5 ports. The idea is to consider virtual locks between the levers and control the signalling based on the actual logic behind each lever's movement. The code has been compiled by using C18 compiler and the provided hex file has been imported to corresponding library of the simulator to know whether the code makes sense experimentally or not. Figure 26 presents the first simulation model due to straight route (The codes are provided in Appendix A)

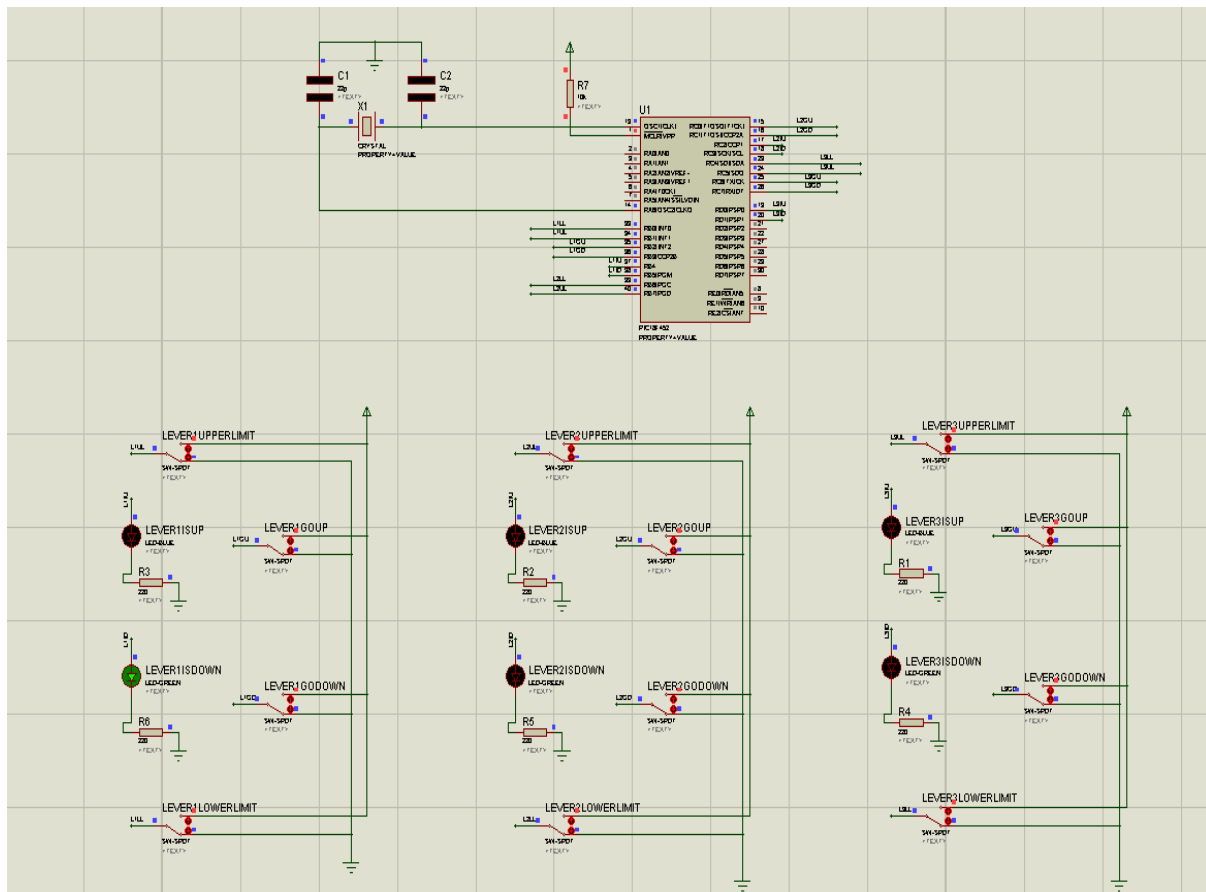


Figure 26: Simulation model concerning the straight route

The system has 12 inputs and 6 outputs, for every lever 4 inputs and 2 outputs have been assigned. For example, lever 1 has 4 inputs which are Lever1UpperLimit, Lever1LowerLimit, Lever1GoUp and Lever1GoDown. The outputs are Lever1IsUp and Lever1IsDown which shows the reversed and normal states through two-aspects signalling.

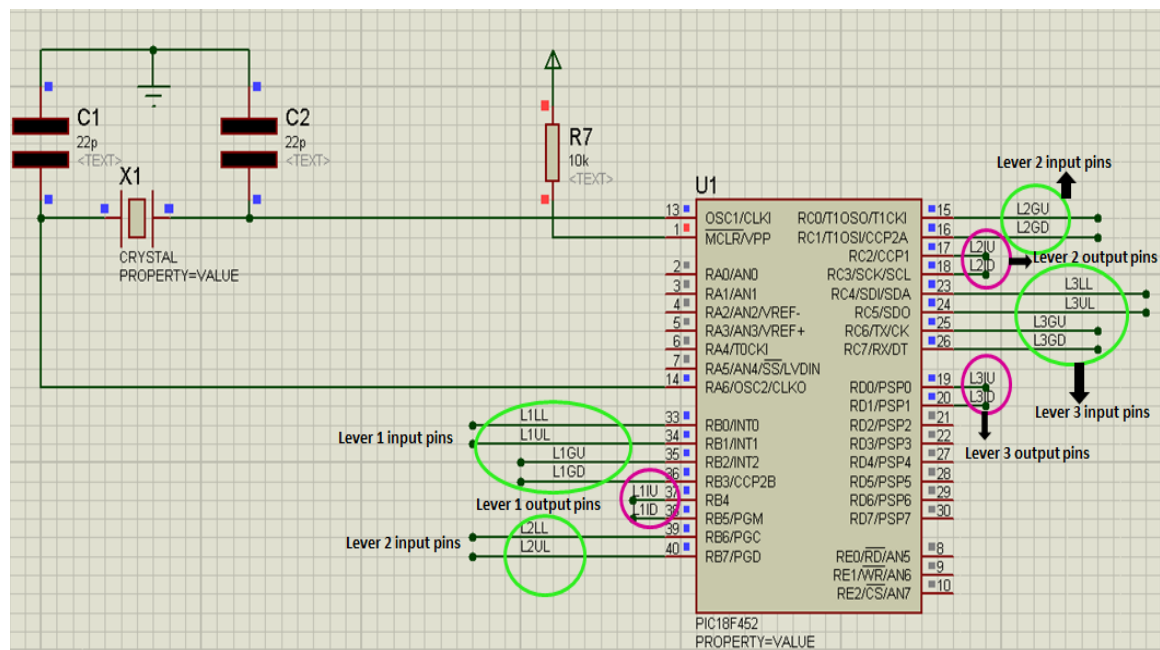


Figure 27: Demonstration of preliminary setup

The outputs are controlled by GoUp and GoDown SPDT switches. To simulate each lever, 4 switches (SW-SPDT) have been intended through Proteus software and they create 3 input chains corresponding to 3 levers. The outputs are the LEDs. The blue LED will come up as each lever GoUp switch is activated and the green one will be blinking as the GoDown switches are activated.

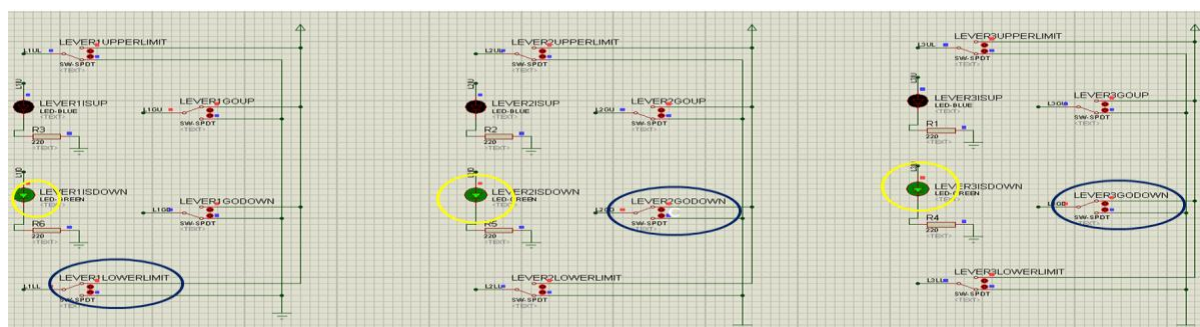


Figure 28: Levers are reversed

4.2 Design Description

In this model lever1 GoDown switch is activated from the beginning, which means the corresponding green LED is blinking, this indicates the first lever is locked by default. Hence, to reverse lever 1, the GoUp switch has to be activated which means Lever2UpperLimit and Lever3UpperLimit have to be achieved first. Figure 29 shows independent logical operation within both lever 2 and lever 3. As is obvious from the figure, both blue LEDs corresponding to Lever 2 and Lever 3 is ON through activation of Lever 2 GoUp and Lever 3 GoUp. This means that the levers can get reversed independently. Accordingly, the Lever 1 GoUp switch is ON, but the corresponding blue LED is OFF, which means that lever 1 is still locked and the distant signal (blue LED) is not clear yet.

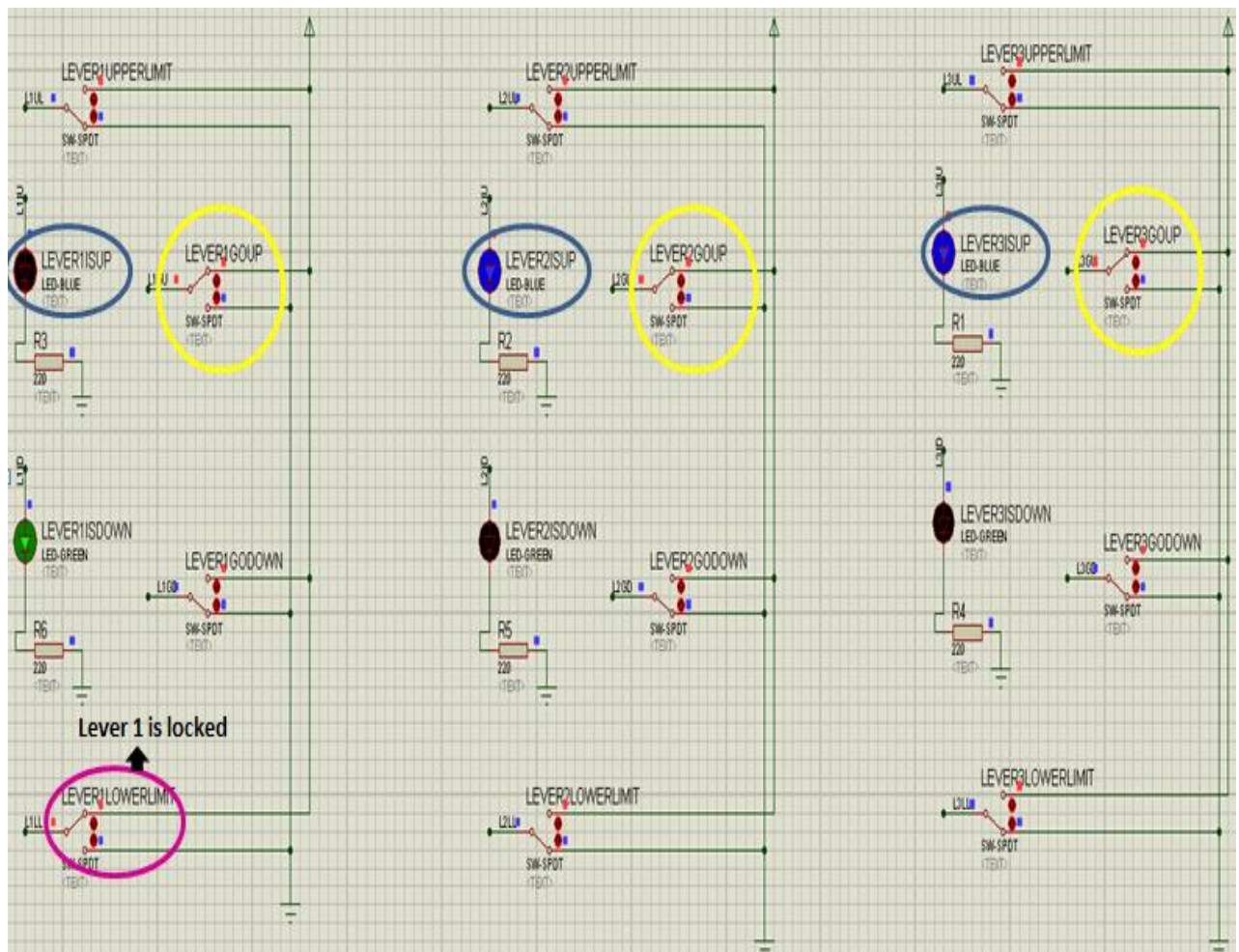


Figure 29: First shot from simulation

Figure 30 presents activation of Lever1GoUp switch based on activation of Lever2UpperLimit and Lever3UpperLimit SPDT switches.

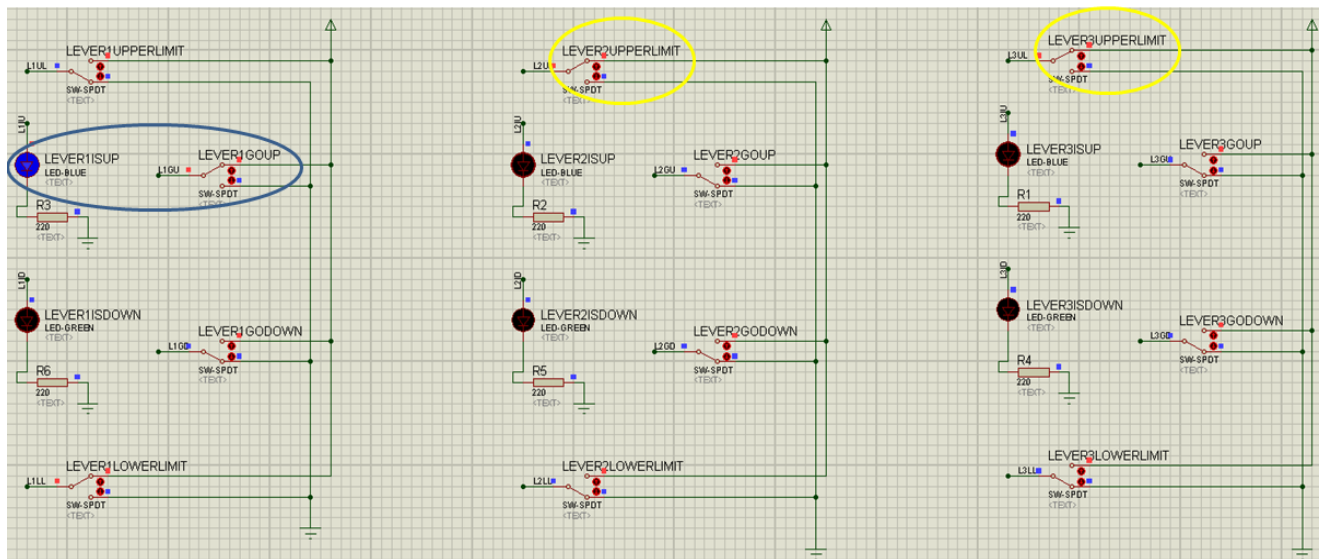


Figure 30: Activation of levers for normal route

The same approach has been adopted to simulate lever's working mechanism due to the reverse route. Figure 31 shows the system approach for the reverse route. In this model states of lever 4, 5 and 6 have been examined. Levers 4 and 6 create a shunting signal and they cannot get reversed at the same time.

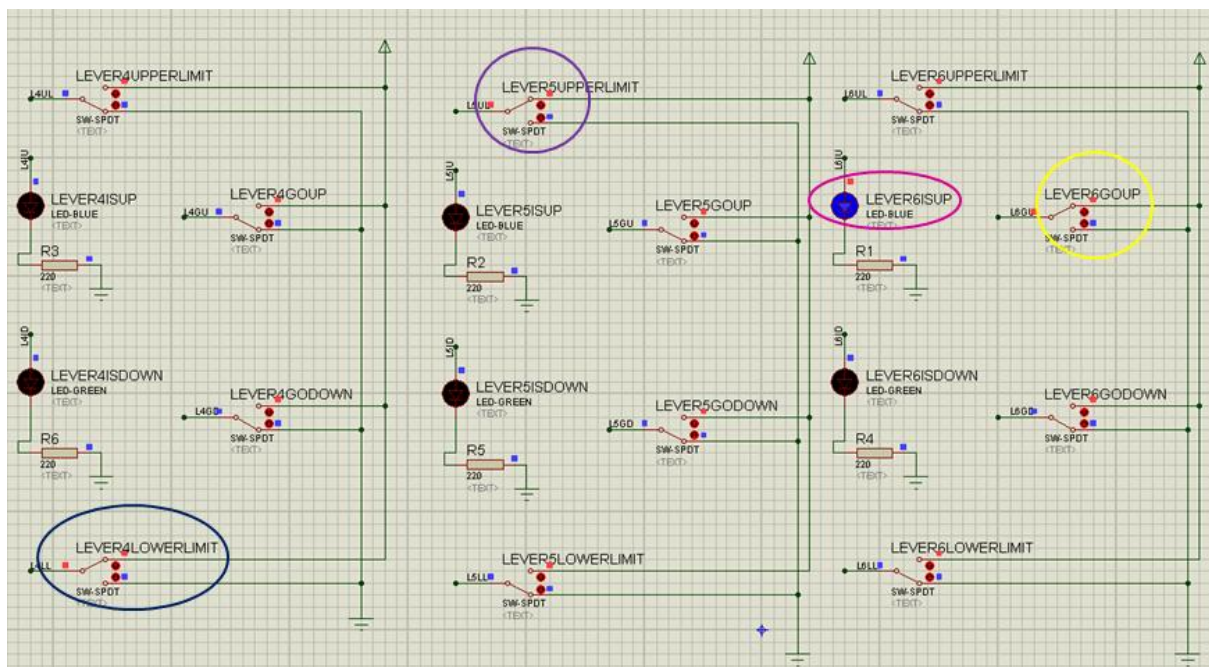


Figure 31: Activation of levers for reverse route

To clear the reverse route, the station-started-signal has to be green first. Subsequently, the shunting signal will be clear as Levers 4 and 6 get reversed. However, they cannot get reversed at the same time. Hence, the shunting signal due to lever 6 will be clear and according to that the shunt point, which is controlled by lever 5 will change track direction through activating the switches (Refer to chapter 5). Ultimately, lever 6 will get normal (locked) and a corresponding shunting signal to lever 4 will come up. As is clear from figure 31, Lever 5's Upper Limit switch is active, this means that lever 5 is reversed and Lever 4's Lower Limit switch is active, which means lever 4 is locked. Thus, as Lever 6 GoUp operates, the blue signal corresponding to lever 6 will come up. Figure 32 presents the activation of lever 4 through opposite operation.

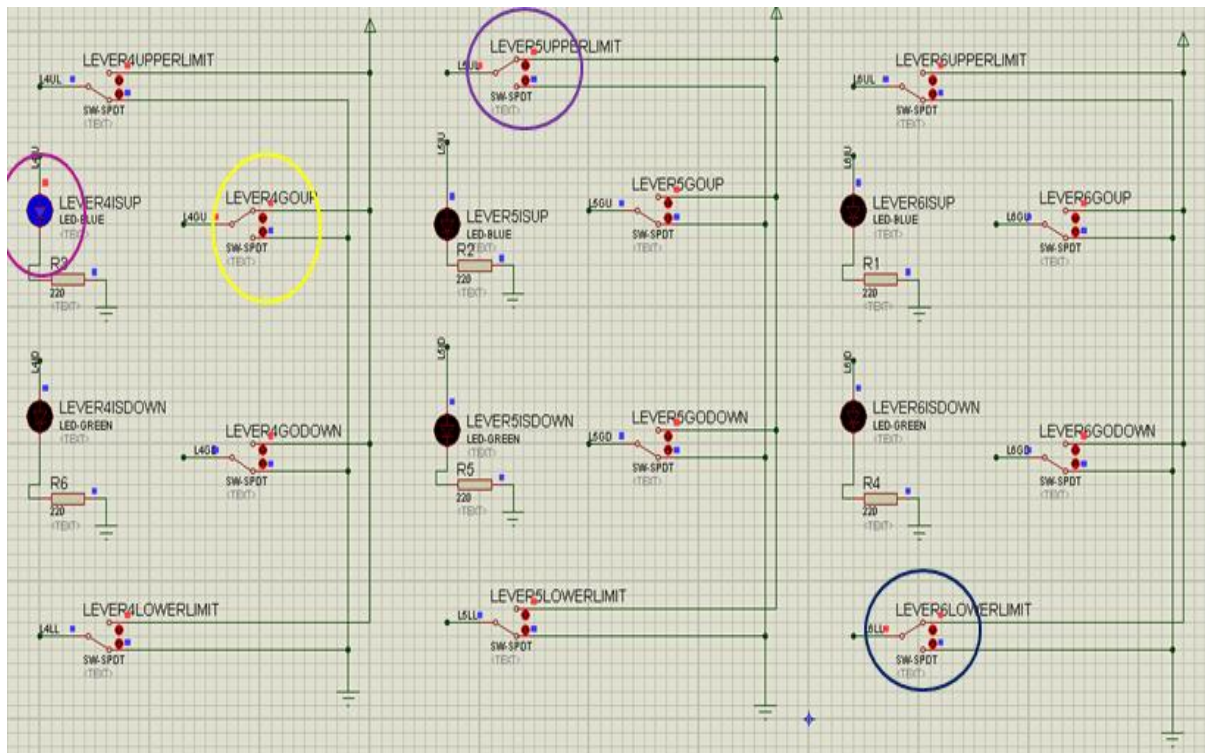


Figure 32: Activation of lever 4

As is obvious from figure 32, the corresponding shunting signal to lever 4 has been turned on. This is governed by two pre-defined variables, namely: Lever 5 Upper Limit and Lever 6 Lower Limit, which means lever 4 can be reversed if lever 6 gets locked, and accordingly lever 5 get reversed.

To evaluate the presented design, the simulation has been run under activation of Lever 5 Upper Limit switch; this is to show the effect of Lever 6 Lower Limits on the LED response.

Figure 33 shows the LED response due to activation of Lever 5 Upper Limit switch and deactivation of Lever 6 Lower Limit.

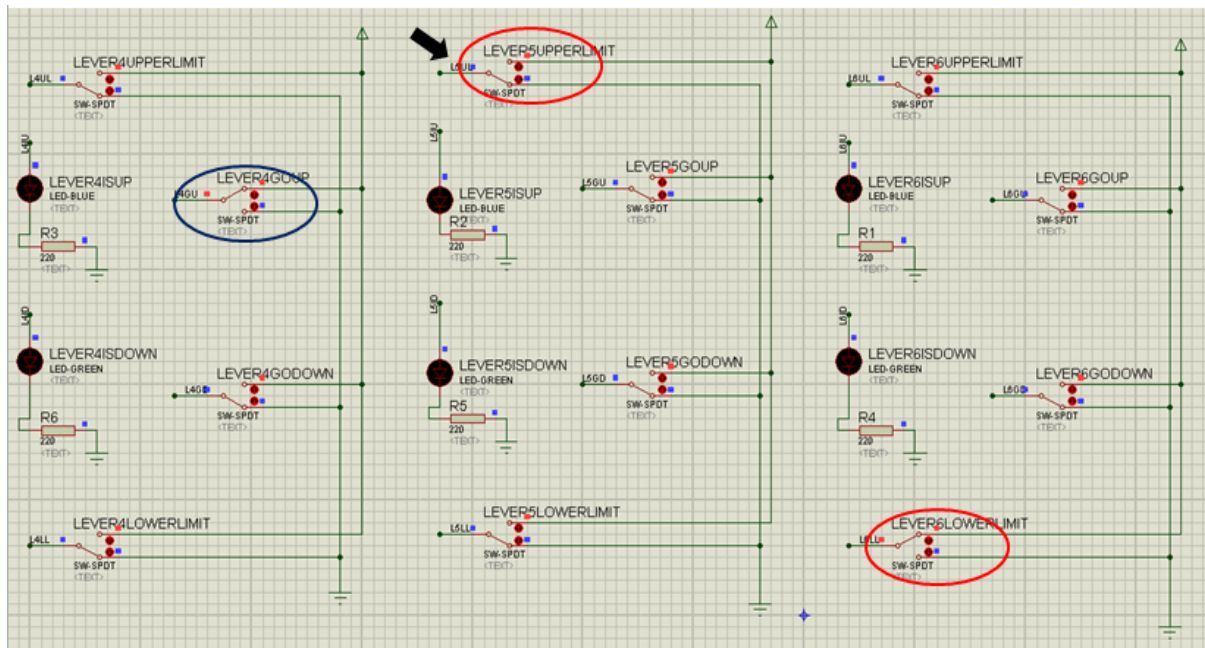


Figure 33: Design Evaluation

As is obvious, lever 4 cannot get reversed unless lever 5 is reversed and lever 6 is locked. In figure 33, the red circle shows that Lever5UpperLimit switch is not active, which means lever 4 is in normal state and hence there is no signal.

4.2.1 Design Optimisation

The functionality of the mechanical lever frame has been simulated through two sets of design concerning both straight and reverse routes. The adopted logic has been extended both through MPLAB and Proteus in order to simulate the behaviour of all the levers in one workspace. Figure 34 determines the final simulation model concerning both the straight and reverse route through the proteus software

As it is understood so far, lever 5 pursues independent operation, which means it can move freely without any restriction. However, to clear the reverse route, lever 5 needs to be unlocked to allow proceeding in the shunt neck. Hence, the switches which simulate the normal state of lever 5 have been omitted from the design.

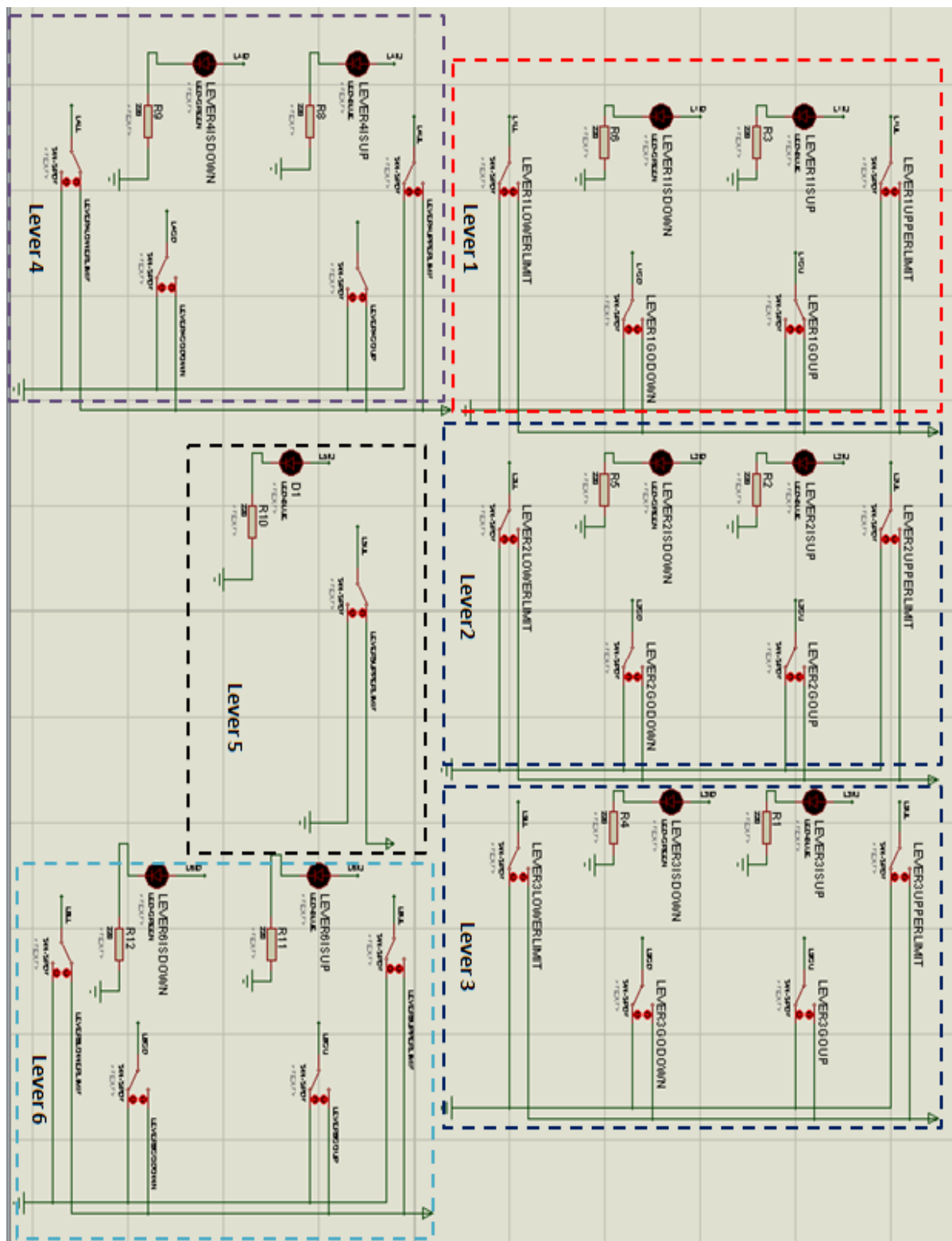


Figure 34: Lever's mechanism

CHAPTER FIVE

5 Introduction to Linear Machine and PECO Turnout Motor

5.1 Linear Machine

A linear machine is about the simplest version of the machines, which obey the same principles and exhibit the same behaviour as real generators and motors. Figure 35 illustrates a linear machine. As is obvious, a battery and resistance are connected with a switch to a pair of frictionless rails.

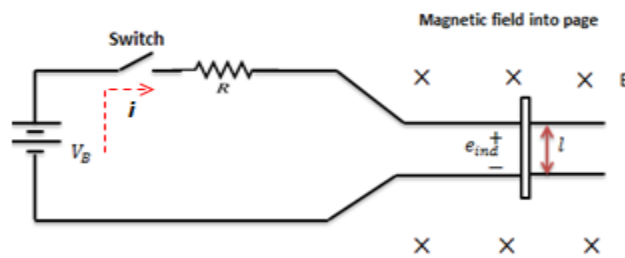


Figure 35: Linear machine (*Chapman, 1999*)

The railroad track along the bed is constant, as the magnetic field goes into the page of the bar which is made by a conducting metal laid across the tracks. The working mechanism of this device can be described through the use of the four following fundamental equations: (Chapman, 1999)

- Calculation of force on the wire according to produced magnetic field:

$$F = i(I \times B) \quad eq(1)$$

Where F = force on wire

i = magnitude of current in wire

I = length of wire, direction of I defined to be in the direction of current flow

B = magnetic flux density vector

- The induced voltage on a wire in the presence of magnetic field:

$$e_{ind} = (v \times B) \cdot I \quad eq (2)$$

Where v = velocity of the wire

B = magnetic flux density vector

l = length

- Calculation of Kirchhoff's voltage law for figure 35:

$$V_B - iR - e_{ind} = 0 \quad eq (3)$$

$$V_B = e_{ind} + iR = 0$$

- Newton's law for the bar across the tracks:

$$F_{net} = ma \quad eq (4)$$

Where

F_{net} = vector sum of forces F

m = object mass

a = acceleration vector a of subject

5.1.1 Working Mechanism

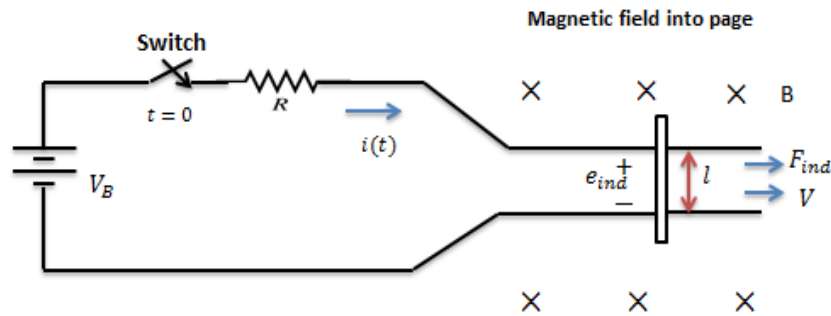


Figure 36: Starting a linear machine (Chapman, 1999)

Figure 36 shows the linear machine under starting conditions. As the switch closes the current flows in the bar, this triggers the machine to work. The current is given by Kirchhoff's voltage law:

$$i = \frac{V_B - e_{ind}}{R} \quad eq (5)$$

$e_{ind} = 0$ When the bar is at rest, hence $i = \frac{V_B}{R}$, this moves down through the bar.

According to equation (1), the current flows through the wire due to the presence of a magnetic field, which ultimately induces a force on the wire and based on the geometry of the machine, this force can be calculated by the formula:

$$F_{ind} = ilB \quad eq (6)$$

Based on above equation and Newton's law, the bar will accelerate to the right. Voltage will appear across the bar as the velocity of the bar starts to increase. The voltage can be determined by equation (2), which will reduce due to the geometry of the device to

$$e_{ind} = vBl \quad eq (7)$$

According to Kirchhoff's voltage law, the voltage will reduce as the current flow in the bar

$$i \downarrow = \frac{V_B - e_{ind} \uparrow}{R} \quad eq (8)$$

The current will decrease as e_{ind} increases.

e_{ind} Will increase to be equal to V_B , accordingly, the net force on bar will reduce to zero and the bar will reach a constant steady-state speed. The speed of the moving bar can be determined by:

$$V_B = e_{ind} = V_{ss}Bl \quad eq (9)$$

$$V_{ss} = \frac{V_B}{Bl} \quad eq (10)$$

The bar will continue to move at this no-load speed until an external force interrupts it.

5.1.2 Linear Motor

A force (F_{Load}) is applied in an opposite direction to the bar. The direction of net force on the bar will be in the opposite direction of motion as the bar was running under no-load steady-state:

$$F_{net} = F_{Load} - F_{ind} \quad eq (11)$$

The bar will slow down due to the presence of external force and the induced voltage across the bar will drop due to $e_{ind} = v \downarrow Bl$. The current in the bar will increase according to the reduction of induced voltage:

$$i \uparrow = \frac{V_B - e_{ind} \downarrow}{R} \quad eq (12)$$

Hence, the induced forces will increase by $F_{ind} = i \uparrow lB$. The induced force will increase as far as it is equal and opposite to the load force, therefore the bar slowly goes to steady state.

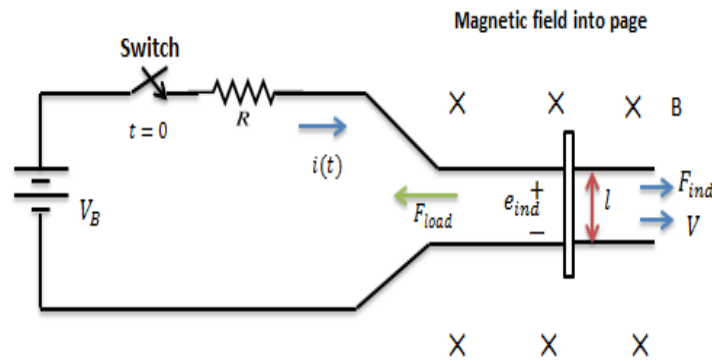


Figure 37: Linear motor (Chapman, 1999)

The induced force is in the same direction as the motion of the bar and the bar will continue to move as the power converts from electrical form to mechanical form by:

$$P_{conv} = e_{ind}i = F_{ind}v \quad eq (13)$$

The electrical power is equal to $e_{ind}i$ and it all consumes in the bar and eventually will be replaced by mechanical power $F_{ind}v$. The bar will operate as a motor as its power changes from electrical form to mechanical form.

A dc motor operates in an analogous fashion as it is loaded, which means as the load enters the shaft the internal voltage will decrease and the current flow will increase and the motor starts to slow down. As a result of increased current, the induced torque will increase. The induced torque will equal the load torque of the motor at a lower speed. The converted power from electrical form to mechanical form for a real rotating motor can be determined by:

$$P_{conv} = \tau_{ind}\omega \quad eq (14)$$

From equation 14, τ_{ind} (induced torque) is the rotational analogue form of the F_{ind} (induced force) and the ω (angular velocity) is the rotational analogue of the linear velocity v .

To summarize the behaviour of a linear machine:

- F_{load} is attached in opposite direction of motion, which triggers the F_{net} to be in opposite direction of motion.
- The shaft will slow down due to a negative acceleration by:

$$a = \frac{F_{net}}{m} \quad eq (15)$$

- The induced voltage will decrease through $e_{ind} = v \downarrow Bl$ and according to that current will increase by: $i = (V_B - e_{ind} \downarrow)/R$.
- The induced force will increase by $F_{ind} = i \uparrow lB$ since at a lower speed $|F_{ind}| = |F_{load}|$.
- Ultimately, $e_{ind}i$ which is the electrical power, will convert to mechanical power $F_{ind}v$, and through that the machine starts to operate as a motor.

5.2 PECO Turnout Motor

The intended motor for this project is turnout peco motor (PL-10e), which is using for low voltage applications. The motor should be energized momentarily and the maximum operating voltage is 16v AC. The motor is formed by two windings and a shaft in the middle part.



Figure 38: PL-10e PECO motor

The recommended track type for this motor is N/009/H0e, which has two holes on each side and one hole in the middle to fit the tie bar. To fix the motor to the switch the centre legs of the motor have to be bent over, so the tie bar in the middle moves and accordingly changes direction of switch to left and right.

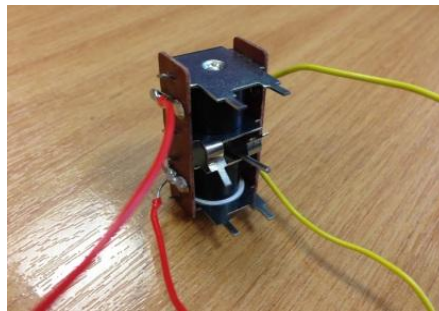


Figure 39: PECO motor with bended central legs

A track layout with two points has been designed and fixed on a wooden board that is 1.5 metres in length and 0.35 metres in width. The baseboard has two holes with size of 40mm × 24mm, where points are located and motors will be passed through the holes and connected to the switches.



Figure 40: Illustration of switches and part of the track layout on the baseboard

5.3 Software Implementation

As mentioned above, the motor has to energise for a short period of time with maximum voltage of 16 VAC. Therefore, to energise the motor momentarily, the idea is to generate a square wave with 50 ms duty cycle. In order to avoid winding and prevent getting warm, 2 seconds delay will come up after each clock rising edge.

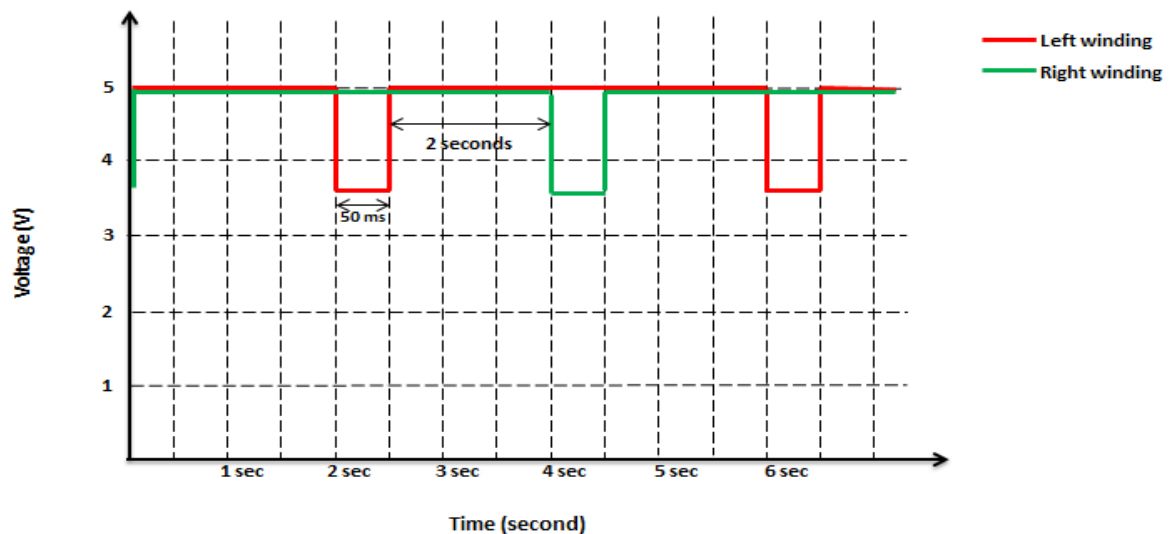


Figure 41: Pulse generation

To perform this task, a PIC microcontroller (PIC18F2550) has been programmed through the MPLAB IDE v8.84 (C18 compiler) software. The idea is to define two variables; output left and output right. Each output variable has two states of ON and OFF. Accordingly, the state of each output variable was examined and based on that they will get a delay of 2 seconds. (The code for square pulse generation is presented in Appendix A)

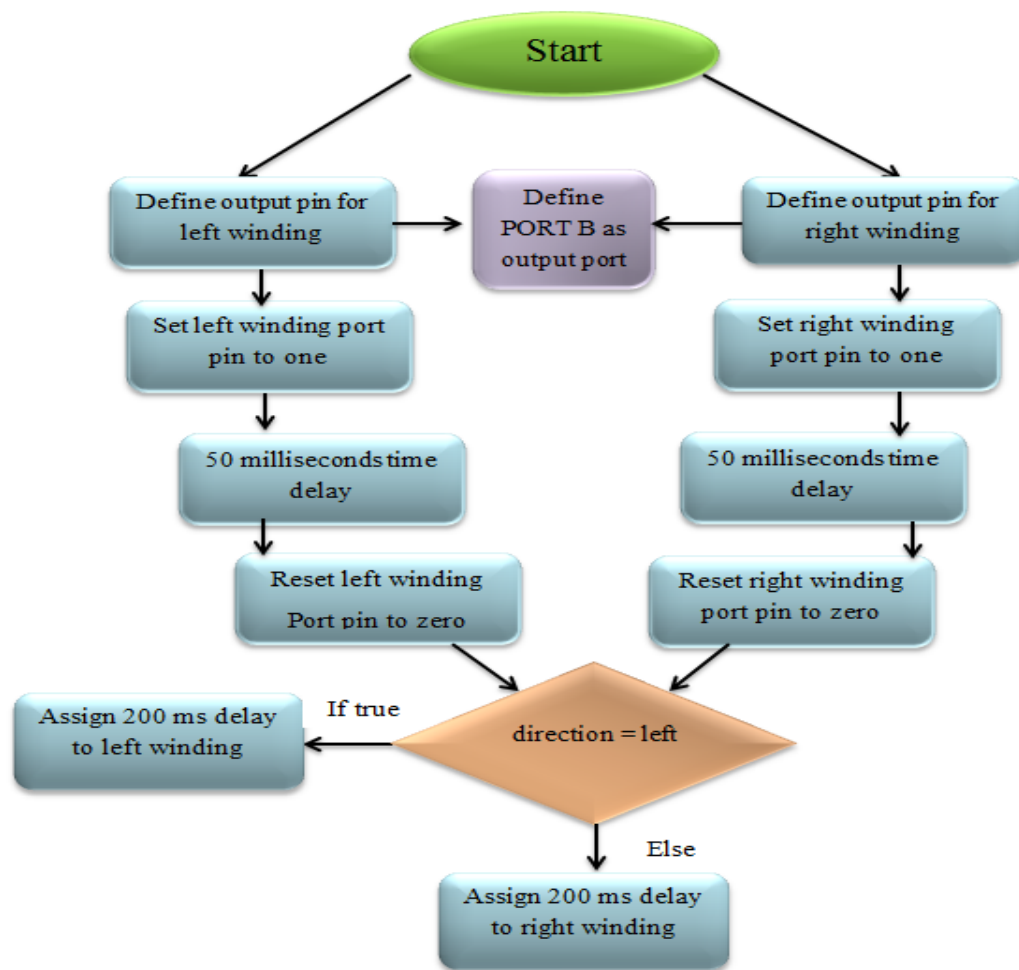


Figure 42: Pulse generating flowchart

5.4 Hardware Implementation

In order to energise the motor and to trigger the switch to move the track to left and right directions, two output pins of the PIC microcontroller, have to be connected to two operational amplifiers. The amplifiers have to be connected in such a manner as to provide negative feedback as oncoming pulses from the outputs of the PIC held negative value. The output of negative feedback op-amp will go to the relay. Ultimately, as each +5 VDC pulse arrives at the internal switch of the relay, the switch will be closed due to different polarity and eventually the motor will be energised for a short period of time, changing the direction of the switch.

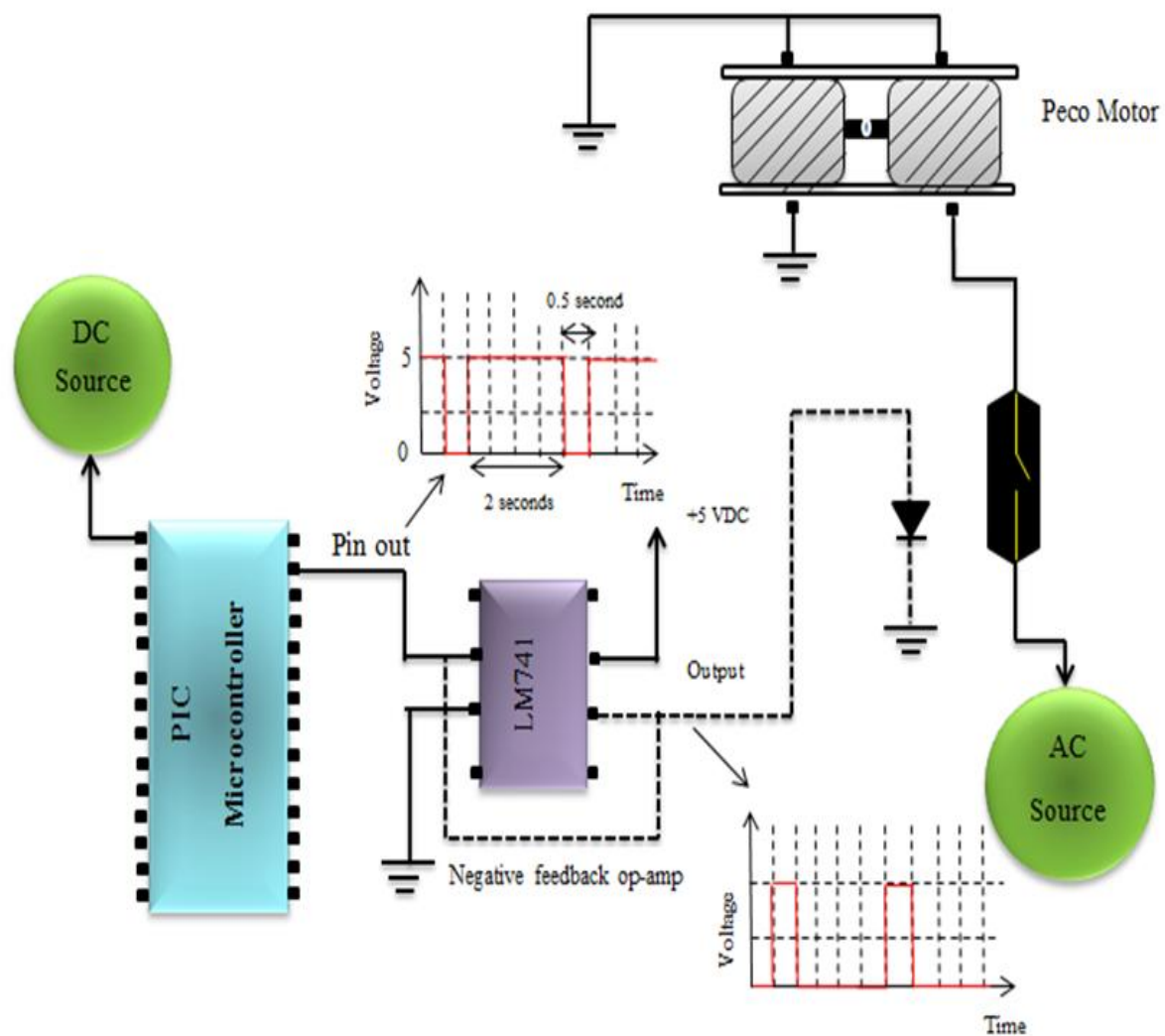


Figure 43: Circuit block diagram

5.5 Introduction to Circuit Components

5.5.1 Reed Relay

Reed relay is a switch, formed by two internal iron stripes, which are covered by coils of wire. When the coil is supplied with power, the wire will generate a magnetic field and the propagated magnetic field will be sensed by two iron stripes inside the relay's body and ultimately, the relay will be operated as a mechanical switch.

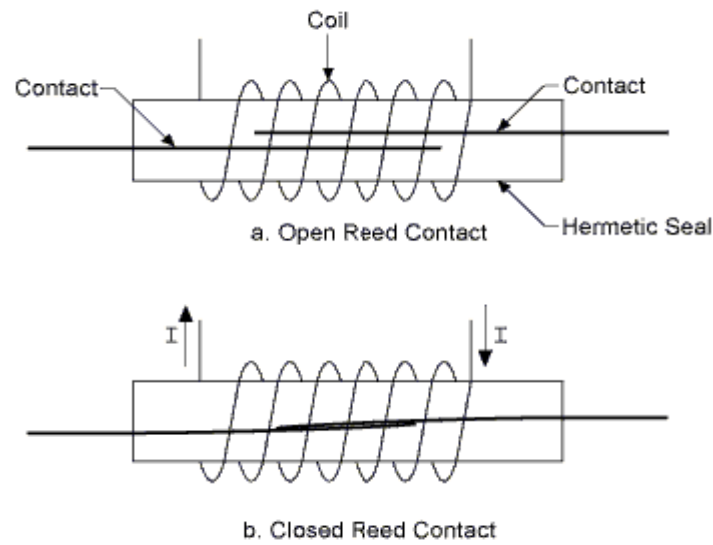


Figure 44: NC and NO relays

The internal switch in NO, or normally-open relays, is open by default. Hence, the internal switch will be closed and the circuit will be connected due to activation of the relay, according to the presence of the magnetic field. The NC, or normally-closed relays, obeys the same principles for activation, however, they exhibit an opposite operation due to their different switch architecture.

5.5.2 Operational Amplifier

Operational amplifiers are essential components in analogue electronic circuits with the ability of amplifying analogue signals. An op-amp is usually formed by three terminals, which are two high impedance inputs (inverting and non-inverting inputs) and an output which is able to sink voltage or current. However, the majority of the circuits require op-amp for voltage amplification.

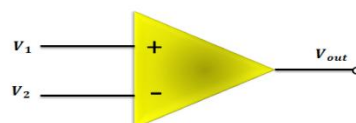


Figure 45: Op-amp Symbol

An ideal op-amp has particular characteristics as follows:

- Infinite open loop gain
- Infinite input resistance
- Zero output resistance
- Bandwidth from 0 to ∞
- Zero offset

As is obvious from figure 46, the operational amplifier is connected in such a manner to provide negative feedback. Negative feedback is defined as a process in which part of the output voltage will be return back to the input with a phase angle that opposes the input signal. Accordingly, the output is not driven into saturation and more realistic gain is achievable.

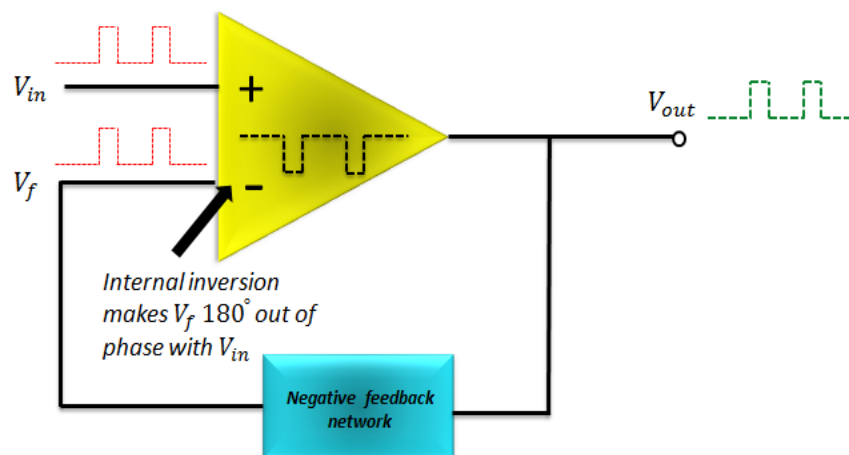


Figure 46: Negative feedback network

The applied IC package for this project is LM741, which is a single device with a single op-amp inside.

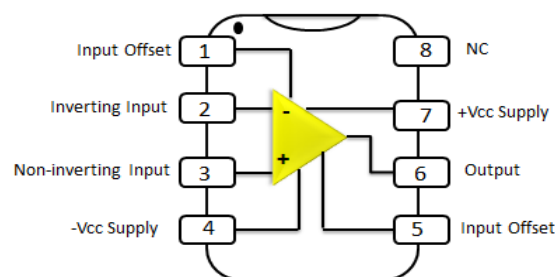


Figure 47:LM741 internal architecture

5.5.3 Diode

Indeed, when the relay is released from its energised state, there is a risk of damage for the op-amp or microcontroller which drives the relay. Therefore, to protect the circuit from back emf, it is vital to connect a rectifier in reverse bias manner across the relay coil. Accordingly, the diode will short out the Back emf created by the coil when it turned off from on state.

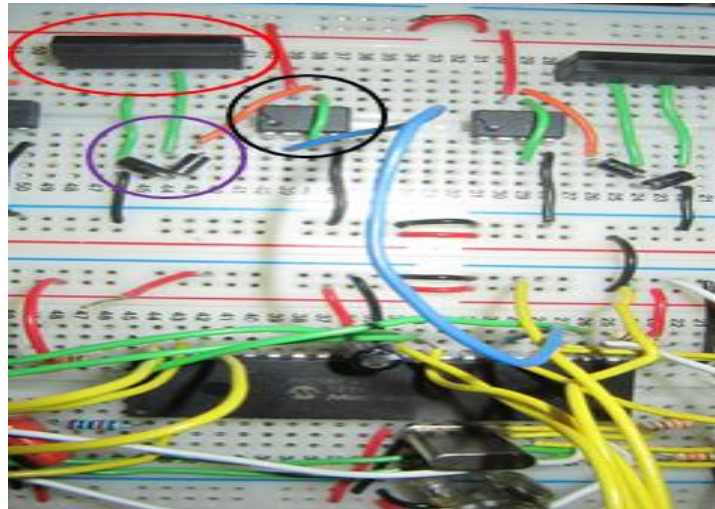


Figure 48: Position of op-amp, reed relay and diodes

Figure 48 demonstrates the connection between circuit components from output of the microcontroller (blue wire) to inverting input of negative feedback op-amp and from output of op-amp to applied diodes across relay coil.

CHAPTER SIX

6 Experimental Setup and Practical Results

Following to the previous chapter, an electronic circuit has been designed and examined in a real experimental situation. The aim of this experiment was to energise the motor based on a correct communication between the control unit (microcontroller) and other adopted electrical components.

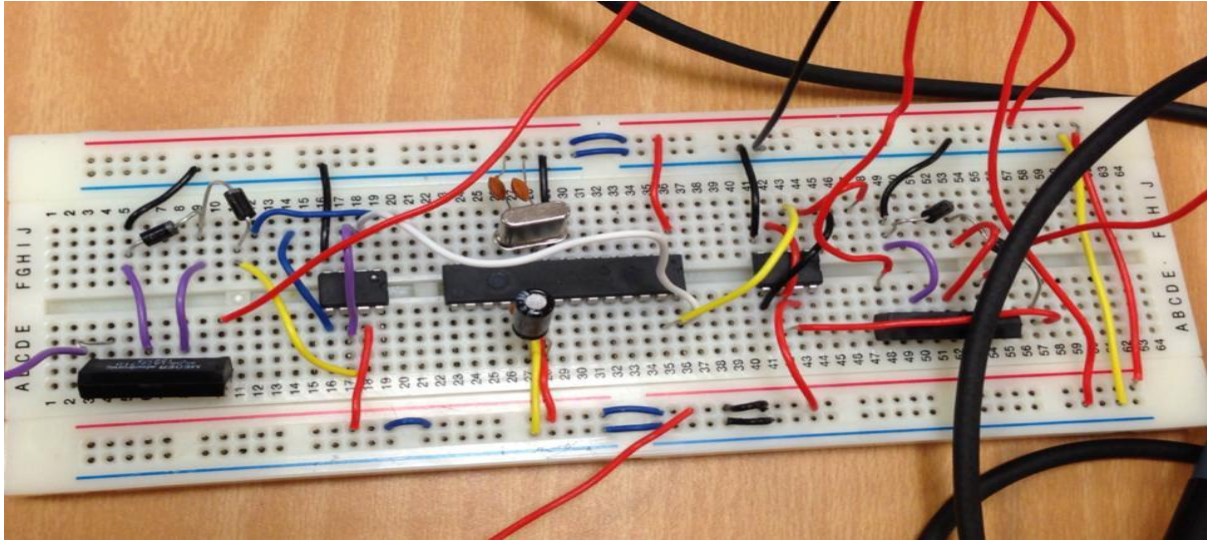


Figure 49: Demonstration of preliminary circuit

The intended motor was energising due to the activation of a reed relay. The intended reed relay required 3.5 VDC to operate. However, the generated square wave from the microcontroller was not able to provide the required current for the relay for switching.

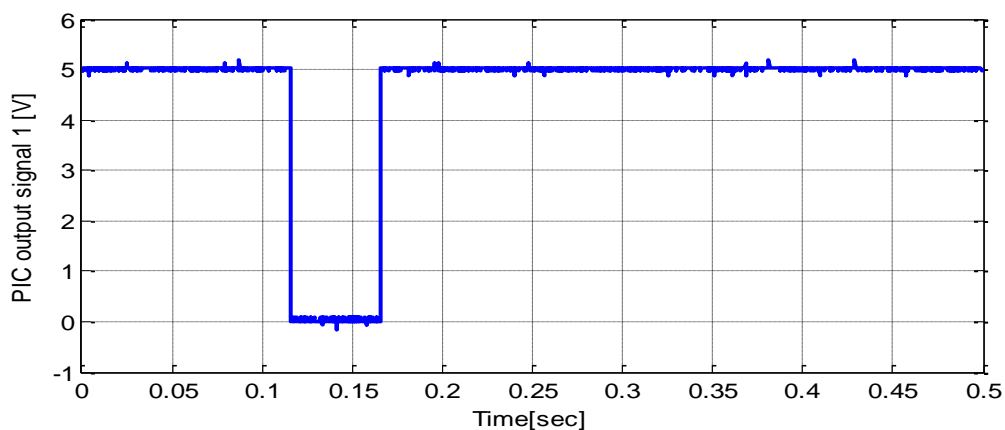


Figure 50: Negative voltage square pulse from the microcontroller

To energise the motor, the relay has to be activated first, which means the internal switch has to be closed. Hence, an amplifier with negative feedback has been applied. Accordingly, the output pin of the microcontroller was connected to inverting input of the op-amp. At the same time, the output of the operational amplifier was connected to the non-inverting input, which caused internal phase inversion by 180 degrees. Therefore, the generated negative voltage square pulse from the microcontroller was changed to positive voltage square pulse.

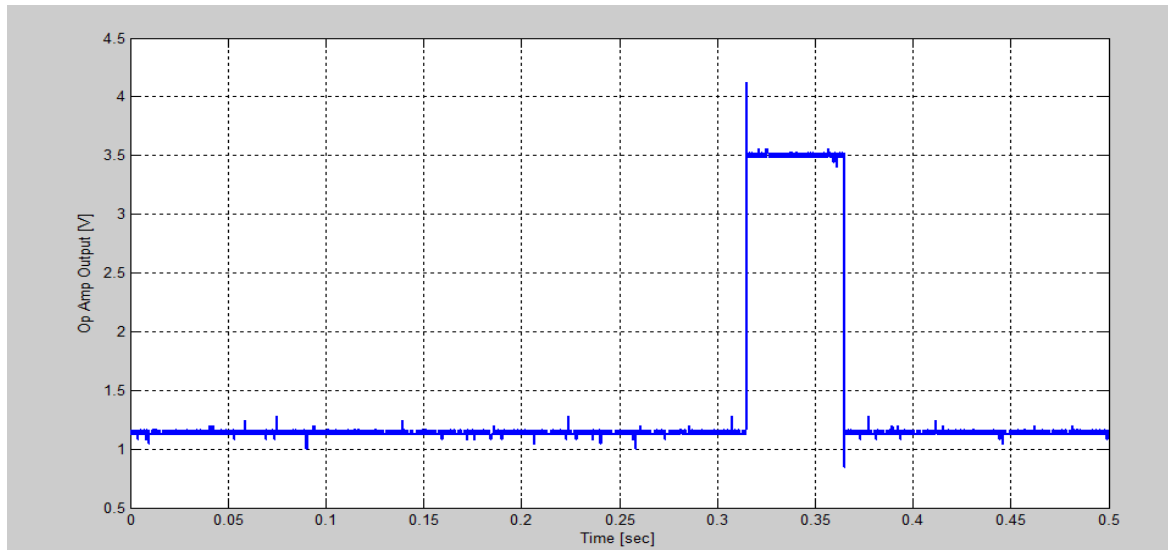


Figure 51: Detected pulse from output of op-amp

The +5 VDC square pulses were passed from output of the op-amp to two rectifiers which were connected across the relay coil. The intended diodes protected the microcontroller and op-amp from the harmful effects of back emf. Besides, both current and voltage will be reduced due to the internal resistance of the rectifiers. Therefore, a secure amount of dc voltage was provided for the reed relay to operate.

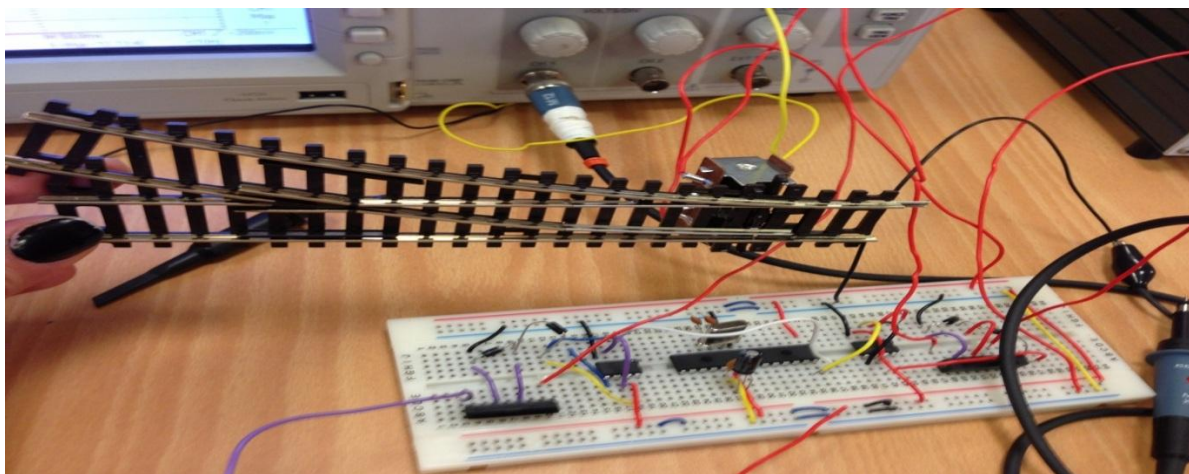


Figure 52: Experimental setup

Figure 52 demonstrates the first experimental setup within the project. As is obvious from the figure, the motor was fixed to the switch point of the track. Hence, by arriving at each oncoming square pulse, the internal switch of the reed relay was activated and the direction of the switch was changed due to a short linear movement created by the motor. Each winding of the motor has a ground and a source pin, which indicates yellow and red wires. The yellow wires were connected to the ground of voltage supplier and the red wires were connected to the first pin of the relay. Besides, the fourth pin of the relay was connected directly to a voltage supplier in order to provide the required voltage for the motor to operate.

6.1.1 Case Study

As mentioned, the relay coil requires more than 100mA current. On the other hand, the maximum output current sunk by each output of the microcontroller is 25mA. Hence, an alternative was to use PNP transistor, this is to drive the relay and to make sure that the relay will stay on when the microcontroller sends -5 VDC to the base pin of the transistor. In the practical test, a PNP transistor was connected in series with PIC microcontroller, the output pin of the microcontroller was connected to transistor base pin and the emitter pin of the transistor was connected to the source through a 1K Ω resistor, which was used to reduce the amount of sunk current from voltage supplier. The black circle in figure 53 shows the transistor unit.

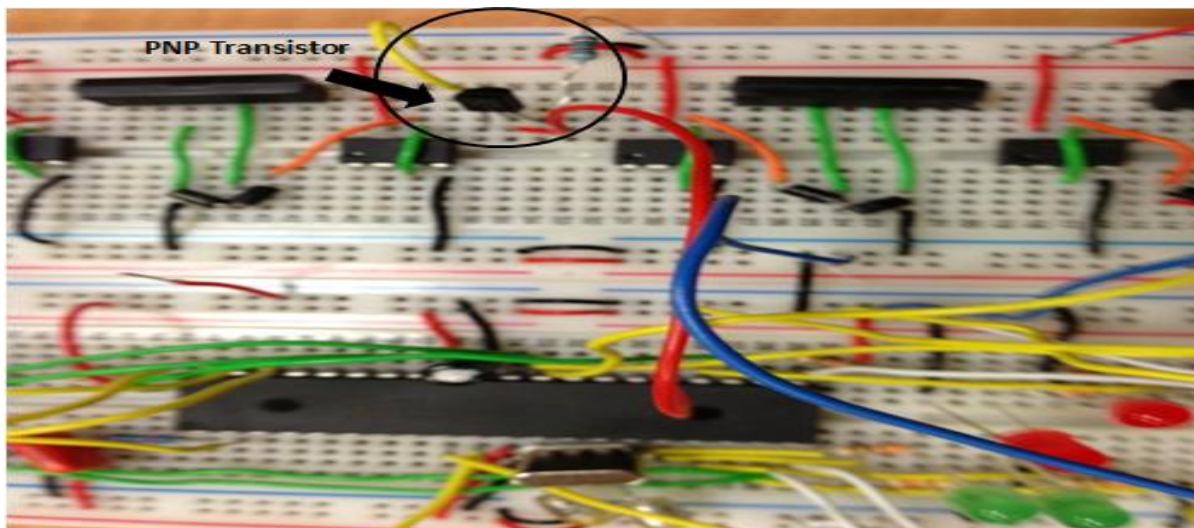


Figure 53: Illustration of PNP transistor

The PNP transistor has been intended to make sure that the reed relay will be activated only due to presence of logic 1 at the microcontroller output. However, interface between PIC microcontroller and PNP transistor in the real experimental situation confirmed that the switching frequency of the transistor is lower than that of in PIC microcontroller.

Figure 54 shows the generated square pulse from the transistors.

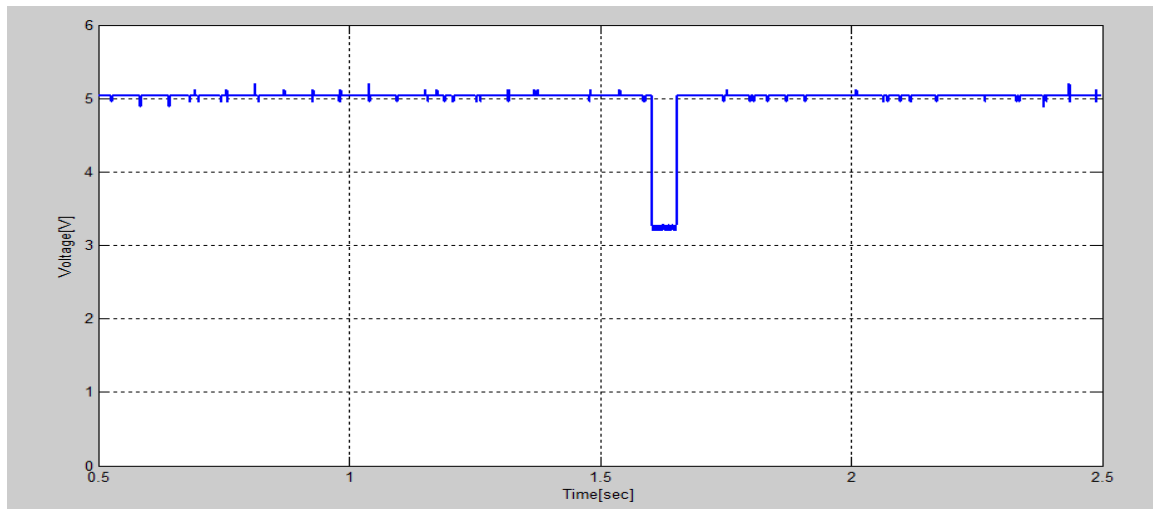


Figure 54: Collected result from PNP transistor

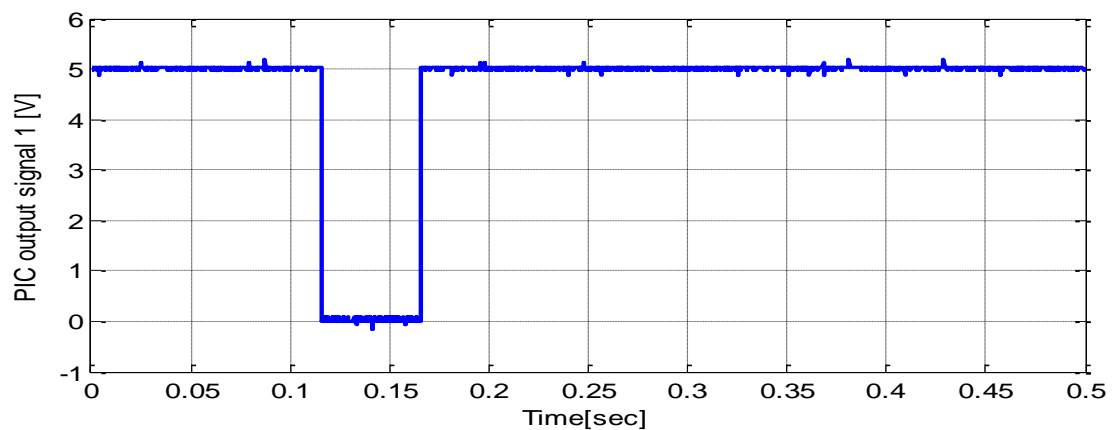


Figure 55: Collected result from microcontroller

As it shown in figure 54, the switching frequency of the transistor is lower than that of the PIC microcontroller. Accordingly, the transistor was not able to operate properly for the required application due to its low switching rate.

6.1.2 Experimental Setup (second phase)

Following the first chapter, the last objective of the defined project is to identify the state of each lever (normal/reverse) and effectively demonstrate 2-aspects of signalling based on the operational logic of each lever through an electrical circuit. Accordingly, the adopted strategy to achieve this task is to use a magnet and reed switch. The magnet can easily attract the levers as they are constructed of metal. The propagated magnetic field by magnets will be sensed by an internal switch of the reed switch (magnetic contact). Subsequently, activation of the internal switch indicates that the corresponding lever has been reversed or unlocked and ultimately the related signal to each particular lever will come up as a blinking LED.

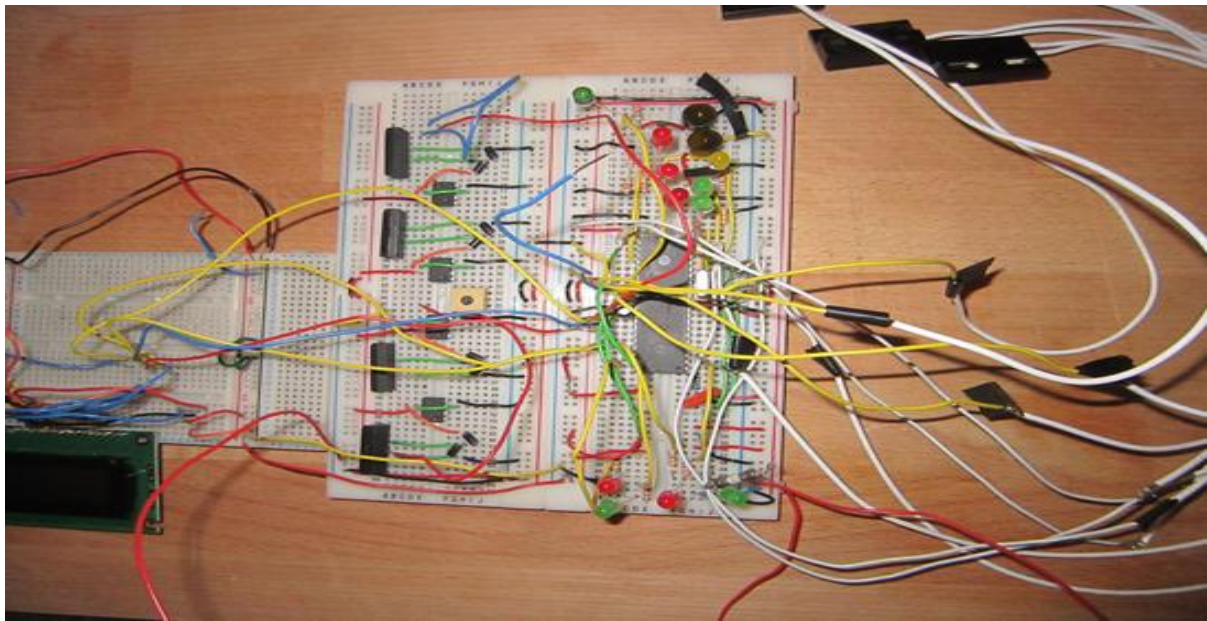


Figure 56: Demonstration of final circuit

6.2 Overview of Components in Final Circuit

6.2.1 Introduction to magnetic actuators and magnetic sensor:

Magnetic actuators are able to convert electrical energy input (voltage/ current) which converts to magnetic energy.

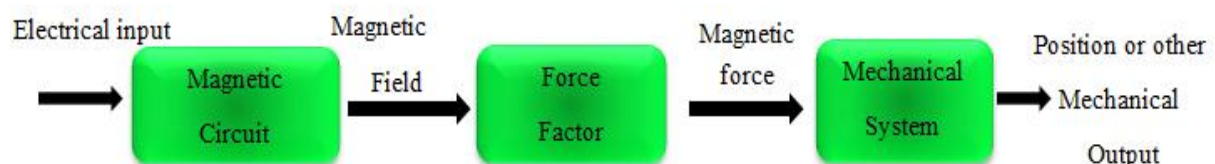


Figure 57: Magnetic actuator block diagram

The energy flow in the magnetic sensor is different from the magnetic actuator. The input in the magnetic sensor is mechanical energy (velocity/ energy). Mechanical energy will be converted to magnetic field and the output will be an electrical signal, which is very small due to the signal's small current. Hence, magnetic devices with large amounts of electrical energy at the output are not classified as sensors.

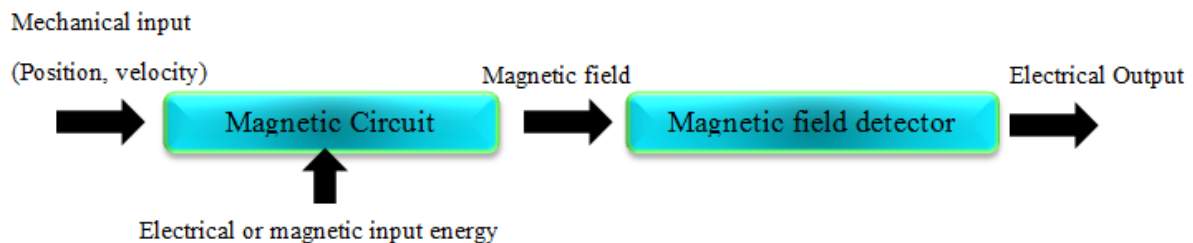


Figure 58: Magnetic sensor block diagram

6.2.2 Magnetic Contact

The majority of magnetic contacts are NC (Normally Close). They are formed of two parts: magnet and reed switch. As the switch is brought into a close proximity of the magnet, the switch will active and vice versa.

6.2.3 Sensitivity Evaluation of Magnetic Contact

In order to test the adopted magnetic contact, the reed switch was connected to a digital multimeter with the lowest selected resistance range (ohms). At the beginning, the multimeter indicated 1 mega ohms. As the magnet was brought into proximity of the switch, the reed which is ferromagnetic was closed and the digital multimeter indicated 0.2 ohms. The sensitivity of the switch has been examined due to the different distance and orientation, which confirms that the maximum distance between the magnet and reed switch has to be less than 15 mm.

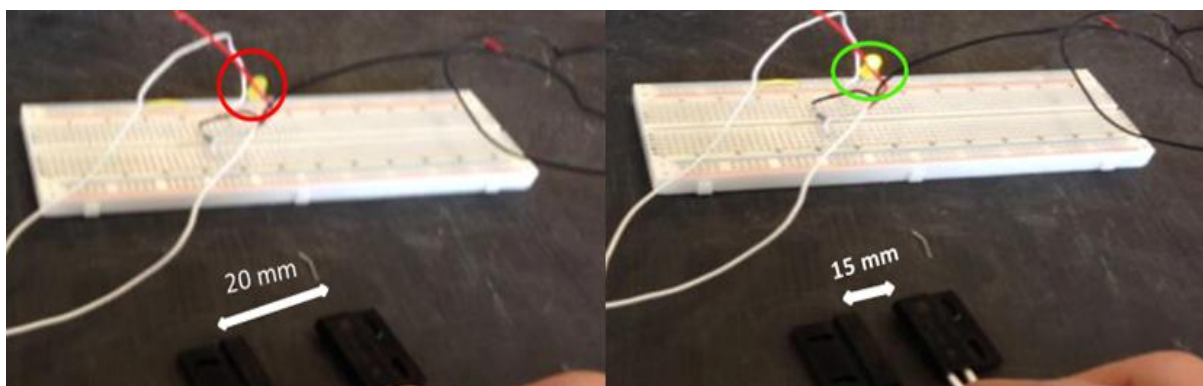


Figure 59: Reed switch sensitivity evaluation

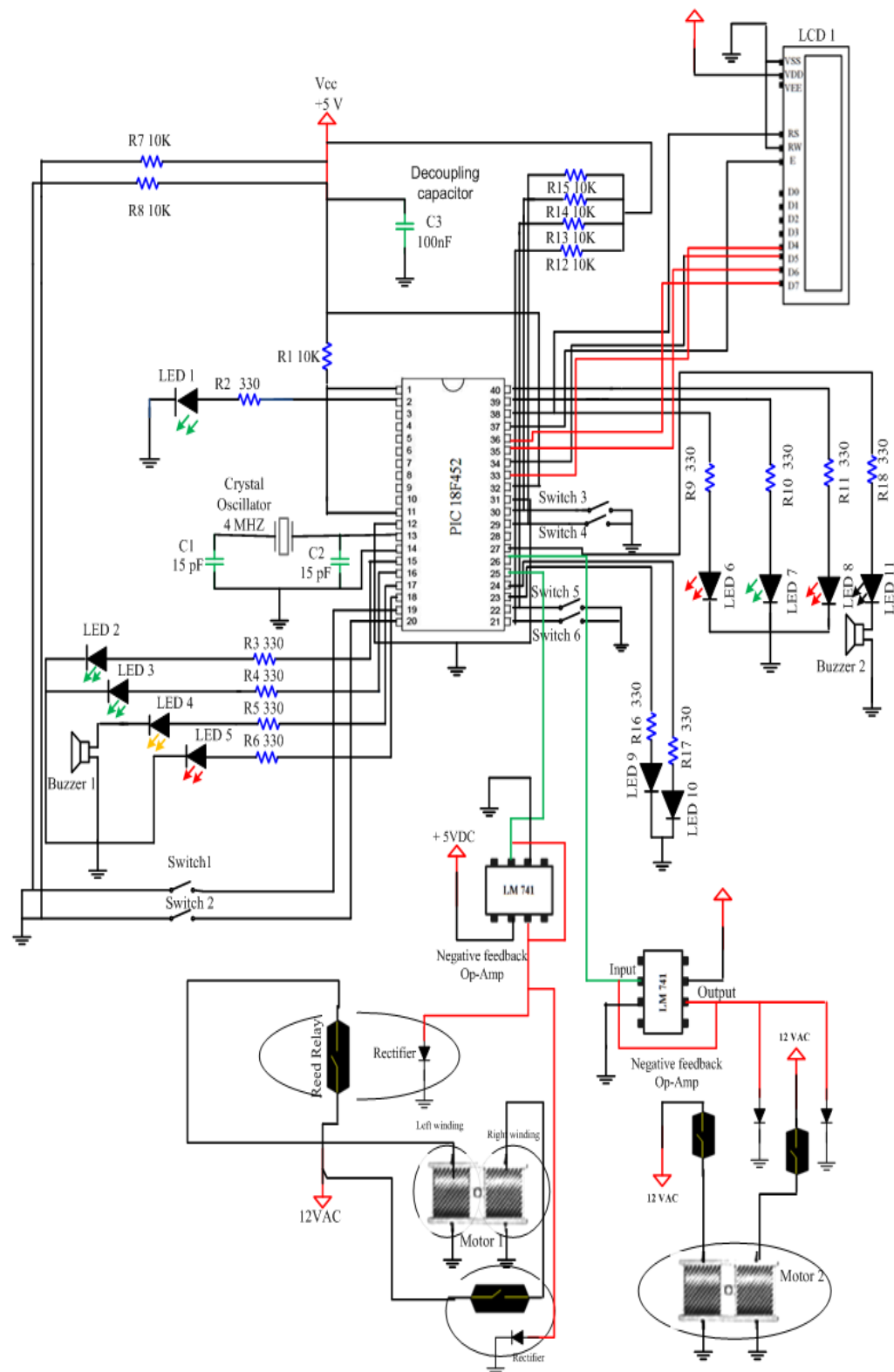


Figure 60: The layout of the circuit

6.2.4 Switch Bounce and Filter Circuit

The switches are connected to the input pins of the microcontroller. As the switch locates in a close proximity to the magnet, the internal contacts will activate. Hence, there is a finite amount of time (up to 20 mS) due to microcontroller oscillation, which is called switch bounce.

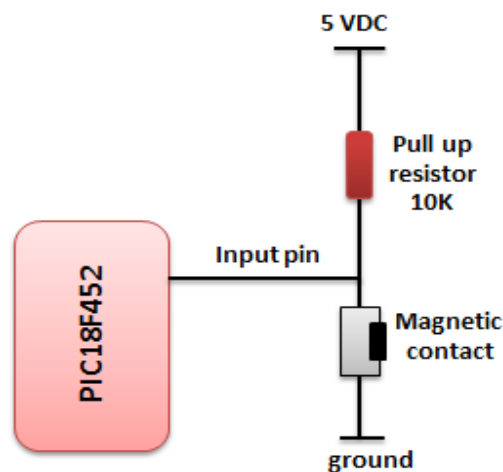


Figure 61: Pull-up connection

During this time the contacts will operate very fast and generate a lot of on and off transitions, which introduce more than one output pulse to the system. These pulses are not visible in the electrical circuit. But, the fast digital circuit will respond to all unwanted pulses, which will cause mis-operation.

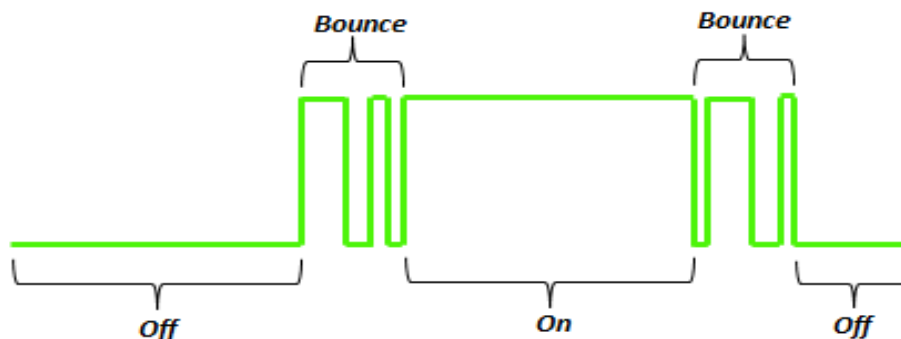


Figure 62: Switch bounce

To eliminate these false pulses, an alternative is to use delay routine inside the microcontroller program. Accordingly, the status of the switch will be checked after 10 mS delay and due to that either logic 1 or logic 0 will generate as returning value to the microcontroller.

```

//*****
//***   This routine is to examine that switch 2 is pressed or not   ***//
//***                                     No input                      ***//
//***   If sw2 is pressed=> output=1 if not output=0                 ***//
//*****
unsigned char is_sw2_pressed(void)
{
    if (PORTDbits.RD1==0)          // is switch 2 pressed?
    {
        // if yes
        delay_ms(10);              // wait 10miliseconds
        if (PORTDbits.RD1==0)      // check the state of switch after 10 mS delay
        {
            // if it pressed
            return 1;              //return value1
        }
    }
    return 0; // return 0, if the switch is not pressed
}

```

Avoid switch bounce (indicated by an arrow pointing to the `delay_ms(10);` line)

Figure 63: Delay routine

Another method to overcome switch bounce within the circuit is to use filter circuit which consists of a resistor and a capacitor. The capacitor will be charged and discharged through a 10K resistor based on activation and deactivation of the magnetic contact. This will remove bounces within the output pulse and a clean single pulse will be achievable.

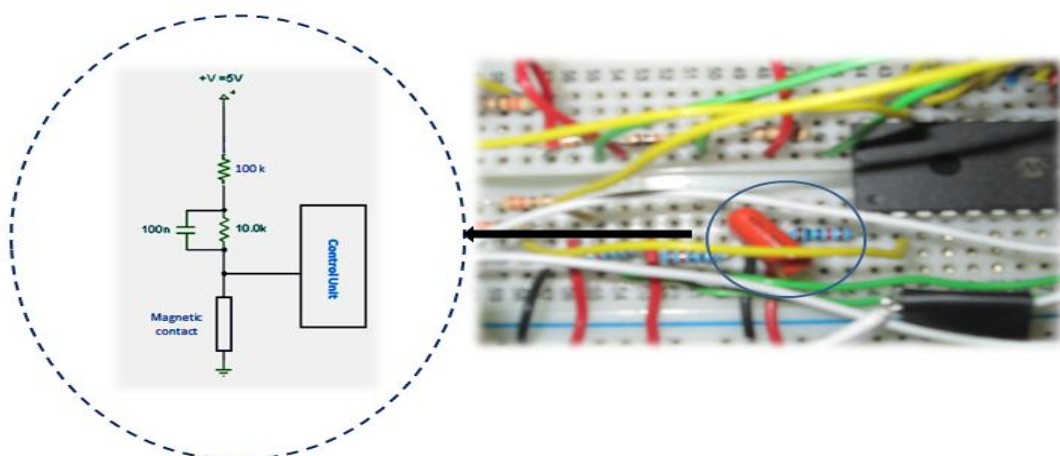


Figure 64: Filter circuit

6.2.5 Real Time Data Acquisition

As mentioned earlier in this chapter, the magnets and reed switches have been used to extract information about the status of the levers from the interlocking box. The microcontroller operates as a control protocol between LEDs and magnetic contacts, which effectively guarantee accurate real time communication between the mechanical interlocking lever frame and the electrical circuit. The following figures demonstrate 2- aspects of signalling based on obtained data from the interlocking lever frame.

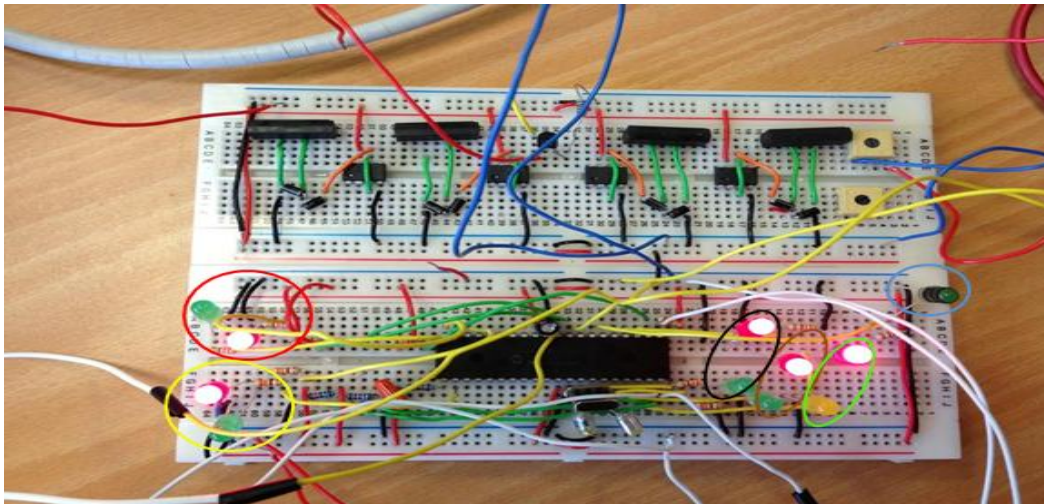


Figure 65: Two aspects of signalling (first shot)

Figure 65 shows associated LEDs to each lever. According to the figure, the black circle indicates the associated LEDs to lever 2, which is controlled by first magnetic contact. The orange circle shows associated LEDs to lever 3, which is governed by second magnetic contact. The green and the red circles show the corresponding signals to third and fourth levers respectively. The LEDs inside the blue and yellow circles are controlled by the fifth and sixth levers.



Figure 66: Interlocking (top-view)

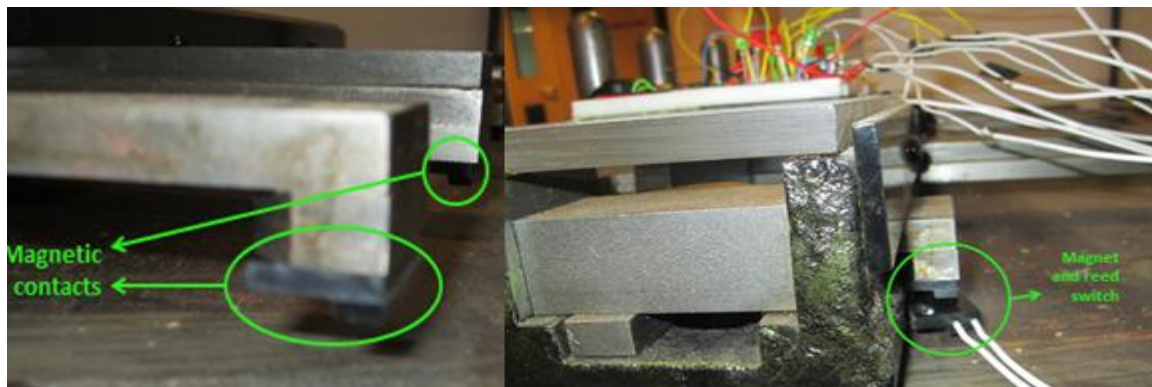


Figure 67: Position of magnetic contact

In order to control train operation concerning the straight route, three signals need to get clear through activation of first three levers. The first step is to clear home signal by reversing lever 2. Figure 68 illustrates home signal due to activation of lever 2 (In here lever 1 is locked).

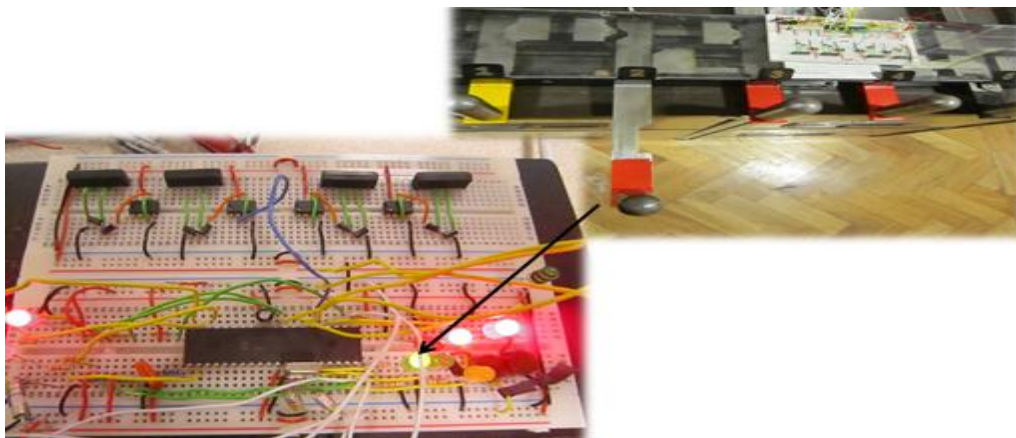


Figure 68: Home signal

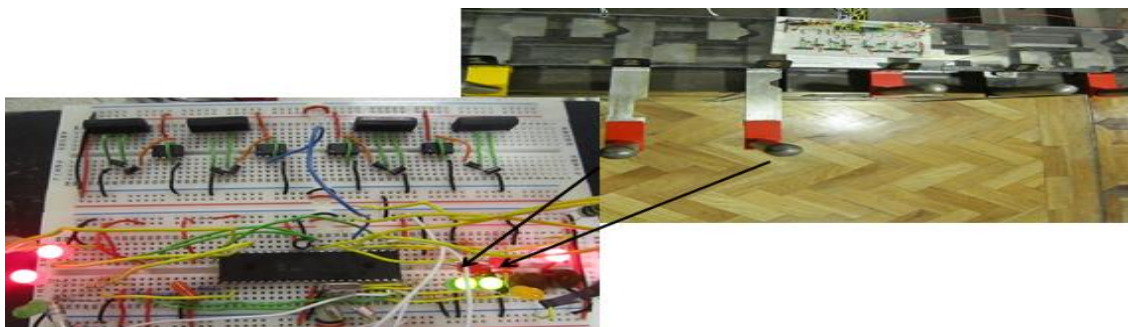


Figure 69: Station started signal

The next step is to clear station-started-signal, which is control by lever 3. Figure 69 shows home and station-started signal due to activation of lever 2 and lever 3 (lever 1 can get reverse now).

The last step is to clear distant signal, which is supervised by lever 1.

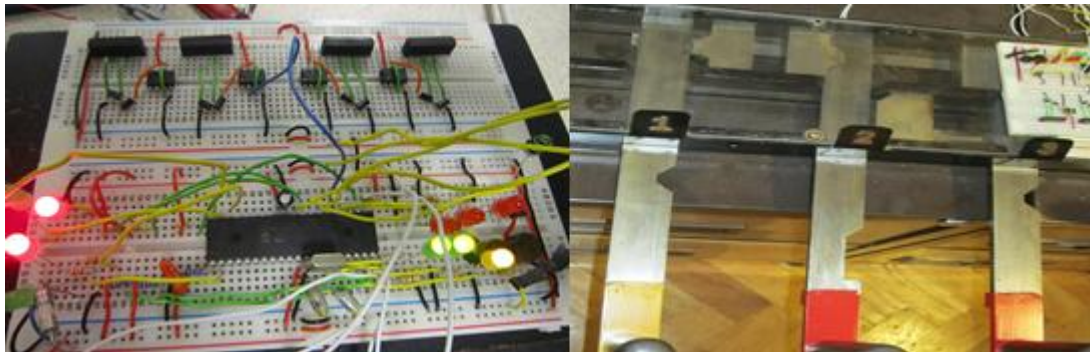


Figure 70: Distant signal

Train operation in the reverse route supervises by 4 signals that are control by the last four levers. Accordingly, the first step is to clear station-started-signal, which means the third lever has to get reverse.

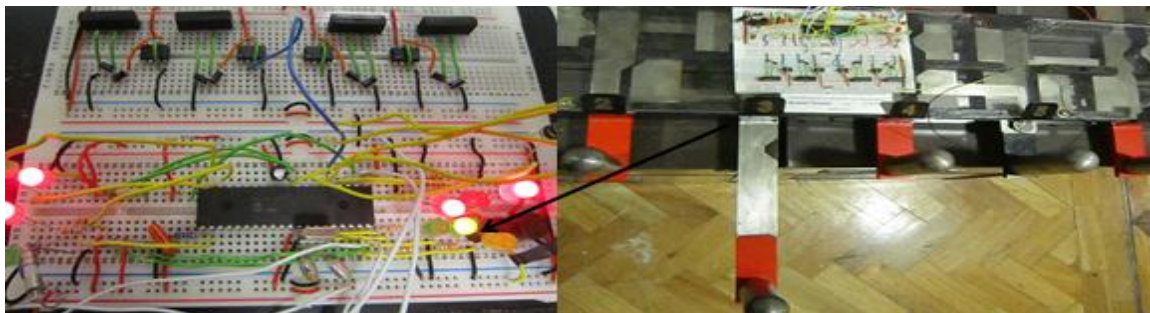


Figure 71: Station-started signal for the reverse route

Then it requires shunting signal to get clear before the points start to operate and change the direction of the track.

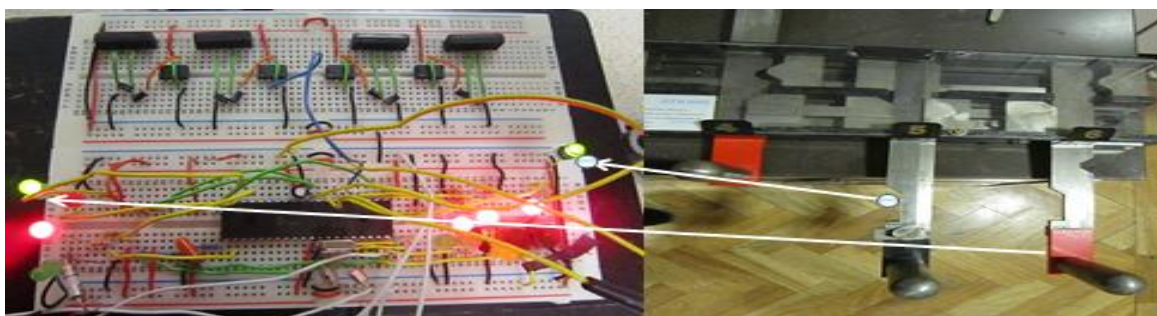


Figure 72: Shunting signal and point

The shunting signal (control by lever six) and station-started signal cannot get clear at the same time because of the safety issues. Hence, as one lever gets lock, the other one will return back to its normal state.

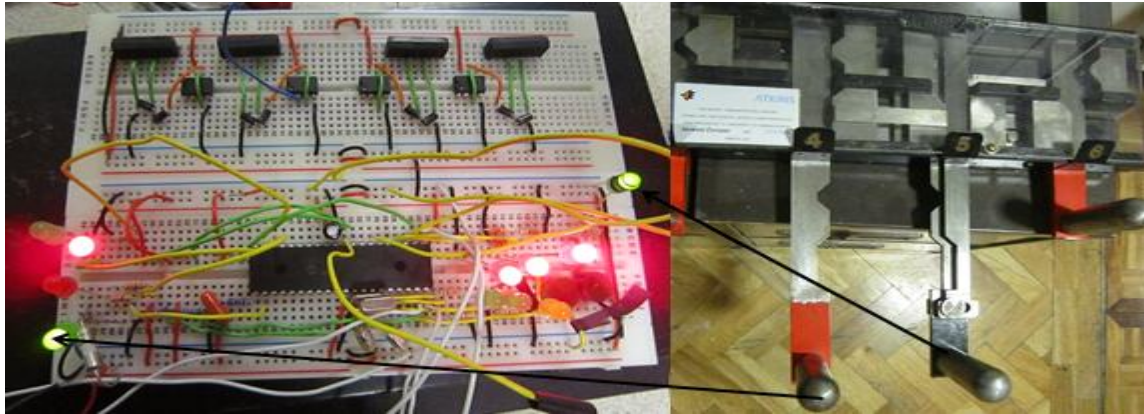


Figure 73: Shunting signal and point #2

Subsequently, as the points 5 get active and the track's direction changed, the shunting signal will clear due to activation of lever 4 (before lever 4 gets reverse, the lever 6 needs to return back to its normal state).

6.2.6 Circuit Enhancement

The provided interlocking operates by considering all the safety issues and existing standards within railway signalling. Hence, levers cannot operate individually, as some of them are internally locked by other levers and these locks will not allow the lever to move freely. The working mechanism of the levers concerning the straight route is different from that in reverse route. Secure train operation will be achievable through the operation of first three levers. The distant signal (double yellow signal) will be clear due to operation of the first lever. However, it is restricted by the second and third levers, which control the home and platform started signal respectively. But, in the reverse route, safe operation will be controlled through the operation of the last four levers (lever 3, lever 6, lever 5 and lever 4). Signal operation due to the reverse route is separate and completely different from that of the straight route. It was suggested by the author to provide an LCD unit to indicate the status of the route due to the operated signals. The applied LCD unit has been programmed through MPLAB software. The LCD routine will be called twice inside the code. Firstly, the distant signal gets clear and secondly, the direction of the switches has been changed due to activation of motor 1 and motor 2. The LCD will be activated due to these two events and a message will be displayed, which indicates that the "route is clear". The LCD code hex file has been exported from the MPLAB programming software and imported into the

corresponding library inside the simulating software (Proteus ISIS). Figure 74 demonstrates the simulation result through the activation of the LCD.

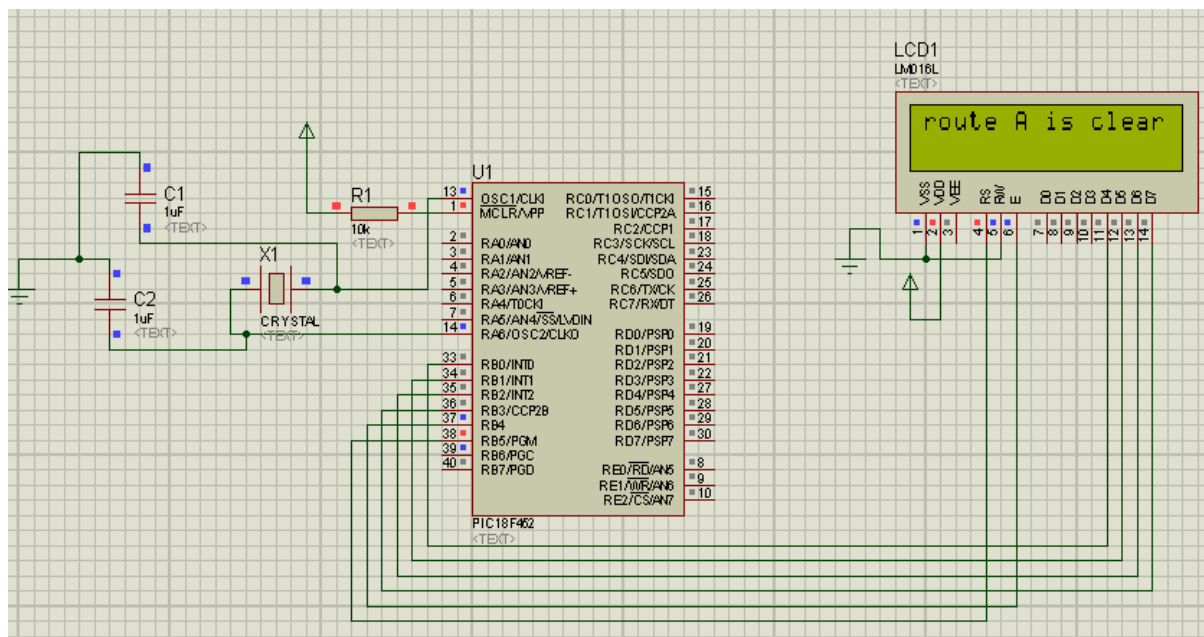


Figure 74: LCD simulation result

CHAPTER SEVEN

7 Conclusion

7.1 Findings

All parts of this project were practically implemented and tested and they work as expected. The adopted strategy to run the project has been performed step by step and the expected results have been obtained and presented through the report. The performed tasks are as follows:

- A comprehensive literature review about the railway signalling operating system
- Visualisation of the interlocking system through LabVIEW and logical simulation concerning signalling within both straight and reverse routes
- Introducing the operational logic-based status of each single lever within the interlocking working mechanism through both straight and reverse routes.
- System optimisation by considering the safety issues, restriction and existing standards
- Investigating solutions for real-time data acquisition
- Designing an electronic circuit to use the extracted data from the mechanical unit and generate the desired output as electrical signals
- Extending the code in order to energise the intended motor, which has a prototype switching function at the junctions, in parallel with two aspects of signalling
- Enhancing the designed circuit by utilising LCD and buzzers, which operate as two additional events as each route get clear by its corresponding control signals.

7.2 Recommendation

All the mentioned tasks have been performed successfully through three different software: LabVIEW, MPLAB and Proteus ISIS. The most important problem during the work was circuit isolation as two parts of the circuit were supplied by two different voltage suppliers. The motor part of the project required 12 VAC, whilst the signalling part was supplied by 5 VDC. The designed circuit accomplished only one PIC, which was in charge of controlling all the events within the entire system. Hence, it is obvious that the control logic for the energising motor came from the PIC output, which operates with a 5VDC and a 25 mA current, which was not enough to energise the motor's windings. To overcome this issue and guarantee fairly safe communication between circuit components, a PNP transistor was used firstly. Low-speed switching frequency of the intended PNP transistor confirmed that it cannot provide the required voltage for the motor due to each oncoming pulse from PIC, because the switching frequency of the intended PIC was more than that of in transistor. The

general study on the other electrical components introduced operational amplifier as an alternative solution. The adopted amplifier module (LM 741) is connected in non-inverting manner to the output of the PIC in order to provide enough amount of current for activating the corresponding relay module for each motor. The motor's winding will then activated for a short period of time and provide linear motion (due to the bar movement)

7.3 Review of Approach

The overall objective of this project was to understand the control logic and effectively analyse the working mechanism of the interlocking lever frame and design a virtual simulation-based signalling control system, which is operated in parallel with the behaviour of each lever and all existing restrictions within the interlocking. This work utilises 6 magnetic sensors, which are placed underneath each lever to extract data about state of each lever. Accordingly, this project has greatly benefited from an electronic circuit, which has been applied to read the status of each lever in binary format. The proposed circuit was designed using a high-performance PIC microcontroller. The intended microcontroller analyses the oncoming data from the mechanical unit (interlocking) and generates the corresponding output through the status of each lever in the form of electrical signals. The generated signal from PIC has enough current to turn on a LED as the maximum sunk current from each output pin of the PIC is 25 mA and the required operating current for the intended LED is 15 mA. Apart from that, a track layout with two switches has been designed. The direction of each switch is governed by two motors. Subsequently, as each motor energises for a short period of time, a quick linear motion will be created due to each accomplished winding. As the system starts to run, the motor will be activated twice for a short period of time at the same time. Once when the distant signal is cleared, and the other time when the shunting signal is cleared and the corresponding signal to point which is controlled by lever 5 will indicate that proceeding in shunt neck is hazard-free. Ultimately, the designed circuit and the provided electromechanical modelling approach can effectively show the working mechanism and control logic of the entire system through 2-aspects of railway signalling, which can be used as a new training electronic-based tool for students who are willing to know about railway signalling operating systems.

7.4 Areas for Further Work

The next stage for this project is to extend the adopted approach to a new microprocessor-based electronic circuit. The proposed circuits will be formed by two processor units and one comparator unit, which determines a fail-safe comparison of the output signals of two processors. The obtained data from the interlocking will be stored in the first processor in

binary format. The stored data in the first processor and the random data from the second processor will be transmitting to a comparator unit. Subsequently, the first input from the advance processor (first processor) will be compared to the first generated data packet from the second processor and so on. Accordingly, any mismatch between generated data packets from the first processor and the second processor will trigger the comparator to provide a safety command to check the signals that affect system safety. Such a system will require tightly synchronised architecture as two processors needs to be checked at the same time (A fai-safe interlocking system for railways, 1985) .

7.5 Word count

There are 11484 words from chapter one to the end of chapter seven.

8 References

8.1 Documents

A fail-safe interlocking system for railways. **Verma M.R** 1985, fail-safe system design, pp.58-65.

Adams B.B The block system of signalling on american railroads. *The Railroad Gazette*. 1901, pp.173-226.

al.Ryland et *Interlocking for a railway system.* 6,308,117 B1 United state, 2001.

Automation of Railway Gate Control Using Microcontroller. **Dewangan A.K** 3, 2012, International Journal of engineering research & technology, Vol. 1, pp.1-7.

Calvert J.B Railway history, signalling, engineering. [Online] 2008. [Cited: 22 04 2013.] <http://www.railway-technical.com>.

Changing track moving-block railway signalling . **Lockyear M.J** 1996.

Chapman S.J *Electric machinery fundamentals.* s.l., MC Graw Hill, third, 1999.

Cribbens *Railway signalling systems.* 5,242,136 United kingdom, 1993.

electronics Labcentre *Intelligent schematic input system .* 2002.

GWR Great western railway 3-bar tappet lever frame. [Online] 2013. [Cited: 21 07 2013.] <http://www.s-r-s.org.uk/research.html>.

In on the fixed signals of railways. **Rapier R.C** New York, Institution of civil engineers, 1874.

Instuments National *LabVIEW user manual.* Austin, Texas, National Instruments corporate Headquarters, 2003.

Johnson A.H The development of fixed signals on railways. *Railroad Gazette.* 7 April 1893 , pp.255-258.

Lawrence Andy An Introduction to Railway Signalling and Equipments. [Online] 18 August 2011. [Cited: 24 5 2014.] <http://cs.swan.ac.uk/~csal/Talks/Railwaytalk1.pdf>.

Macdougall *A railway signalling system .* EP0341826GB, 1994.

Macmillan B.E *Amplifier circuit and method for providing negative feedback .* wo 2001071905 A3 United state, 2002.

Microchip *28/40/44-Pin Enhanced Flash Microcontrollers*. s.l., Microchip Technology Inc, 2004.

—. *MPLAB C18 c compiler libraries*. s.l., Microchip Technology Inc, 2005.

Newman G.D *Railway signalling system*. 5,437,422 United State, 1995.

Principle of interlocking . [Online]2004. [Cited: 25 7 2013.]<http://mysite.du.edu/~etuttle/rail/lock.html>.

Signalling UK *The Development and Principles of UK Signalling*. [Online]2013. [Cited: 26 02 2013.]<http://www.railway-technical.com/sigtxt1.shtml>.

Silicon-germanium-base heterojunction bipolar transistors. **PATTONG.L**1988, Vol. 9, pp.165-167.

Solid-state interlocking (SSI): an integrated electronic signalling system for mainline railways. **Cribbens A** may 1987, 1987, *Railway electrification and transportation, signalling, Electronic circuits*, pp.148 - 158.

Waller J *Solid state interlocking . Railway control systems*. london, A & C Black, 1991 , pp.21-31.

8.2 Web Site Links

Calvert, J.(2008). *Railway history, signalling, engineering*. [Online]
Available at: <http://www.railway-technical.com>
[Accessed 22 04 2013].

GWR, (2013). *Great western railway 3-bar tappet lever frame*. [Online]
Available at: <http://www.s-r-s.org.uk/research.html>
[Accessed 21 07 2013].

Principle of interlocking. (2004). [Online]
Available at: <http://mysite.du.edu/~etuttle/rail/lock.html>
[Accessed 25 7 2013].

Railway systems, technologies and operation across the world. (2013) [Online]
Available at: <http://www.railway-technical.com/sigtxt1.shtml>
[Accessed 26 02 2013].

An Introduction to Railway Signalling and Equipment. (2011)[Online]

Available at: <http://cs.swan.ac.uk/~csal/Talks/Railwaytalk1.pdf>

[Accessed 24 05 2014].

8.3 Bibliography

A fail-safe interlocking system for railways. **Verma M.R** 1985, fail-safe system design, pp.58-65.

Adams B.B The block system of signalling on american railroads. *The Railroad Gazette*. 1901, pp.173-226.

al. Ryland et *Interlocking for a railway system.* 6,308,117 B1 United state, 2001.

Automation of Railway Gate Control Using Microcontroller. **Dewangan A.K** 3, 2012, International Journal of engineering research & technology, Vol. 1, pp.1-7.

Calvert J.B Railway history, signalling, engineering. [Online] 2008. [Cited: 22 04 2013.] <http://www.railway-technical.com>.

Changing track moving-block railway signalling . **Lockyear M.J** 1996.

Chapman S.J *Electric machinery fundamentals.* s.l., MC Graw Hill, third, 1999.

Cribbens *Railway signalling systems.* 5,242,136 United kingdom, 1993.

electronics Labcentre *Intelligent schematic input system .* 2002.

GWR Great western railway 3-bar tappet lever frame. [Online] 2013. [Cited: 21 07 2013.] <http://www.s-r-s.org.uk/research.html>.

In on the fixed signals of railways. **Rapier R.C** New York, Institution of civil engineers, 1874.

Instuments National *LabVIEW user manual.* Austin, Texas, National Instruments corporate Headquarters, 2003.

Johnson A.H The development of fixed signals on railways. *Railroad Gazette*. 7 April 1893, pp.255-258.

Lawrence Andy An Introduction to Railway Signalling and Equipments. [Online] 18 August 2011. [Cited: 24 5 2014.] <http://cs.swan.ac.uk/~csal/Talks/Railwaytalk1.pdf>.

Macdougall *A railway signalling system .* EP0341826GB, 1994.

MacmillanB.E *Amplifier circuit and method for providing negative feedback* . wo 2001071905 A3United state, 2002.

Microchip *28/40/44-Pin Enhanced Flash Microcontrollers* . s.l., Microchip Technology Inc, 2004.

—. *MPLAB C18 c compiler libraries*. s.l., Microchip Technology Inc, 2005.

NewmanG.D *Railway signalling system*. 5,437,422United State, 1995.

Principle of interlocking . [Online]2004. [Cited: 25 7 2013.]<http://mysite.du.edu/~etuttle/rail/lock.html>.

SignallingUKThe Development and Principles of UK Signalling. [Online]2013. [Cited: 26 02 2013.]<http://www.railway-technical.com/sigtxt1.shtml>.

Silicon-germanium-base heterojunction bipolar transistors. **PATTONG.L**1988, Vol. 9, pp.165-167.

Solid-state interlocking (SSI): an integrated electronic signalling system for mainline railways. **CribbensA**may 1987, 1987, Railway electrification and transportation, signalling, Electronic circuits, pp.148 - 158.

WallerJSolid state interlocking . *Railway control systems*. london, A & C Black, 1991 , pp.21-31.

Appendix A

- Codes and comments

```

////////////////////////////////////
//////// Author: M.Maksabi //////////
//////// Function: pulse generating //////////
//////// Device: PIC 18F2550 //////////
////////////////////////////////////

#include <p18f2550.h>
#include <timers.h>

//*****Configuration bits*****//
#pragma config FOSC = XT_XT // Crystal oscillator
#pragma config PWRT = OFF // Power-up timer disabled
#pragma config BOR = OFF // Brown-out reset disabled
#pragma config LVP = OFF // Low-voltage programming disabled
#pragma config WDT = OFF // Watch-dog timer disabled
#pragma config DEBUG = OFF // Background debugger disabled
//*****Code*****//

#pragma code

#define ON 0
#define OFF 1
#define left 1
#define right 2

#define OutputLeft PORTBbits.RB7 // output pin
#define OutputRight PORTBbits.RB6 // output pin

void direction (int dir)
{
if (dir==left) // check the current direction
{
OutputLeft = ON; // RB7 gets logic 1
Delay1KTCYx (50); // for 50 milliseconds
OutputLeft = OFF; // returns back to logic 0
} else

```

```
{
OutputRight = ON;    // RB6 gets logic 1
Delay1KTCYx (50);    // for 50 milliseconds
OutputRight = OFF;    // returns back to logic 0
}
void main (void)
{
TRISB=0;              // // PORTB the output port
while(1)
{
direction (left);
Delay10KTCYx(200);
Delay10KTCYx(200);
direction (right);
Delay10KTCYx(200);
Delay10KTCYx (200);
}
}
```

```

////////////////////////////////////
/////////                Author: M.Maksabi                //////////
/////////    Levers mechanism due to the straight route    //////////
/////////                Device: PIC 18F452                //////////
////////////////////////////////////

#include<p18f452.h>

//*****Configuration Bits*****//

#pragma config PWRT =OFF    // Power-up timer disabled
#pragma config BOR  =OFF    // Brown-out reset disabled
#pragma config LVP  =OFF    //Low-voltage disabled
#pragma config WDT  =OFF    //Watch-dog timer disabled
#pragma config DEBUG =OFF   // Background debugger disabled
//*****Codes and Definitions *****//

#pragma code

#define Lever1LowerLimit PORTBbits.RB0    //input
#define Lever1UpperLimit PORTBbits.RB1    //input
#define Lever1GoUp PORTBbits.RB2          //input
#define Lever1GoDown PORTBbits.RB3        //input
#define Lever1IsUp PORTBbits.RB4          //output
#define Lever1IsDown PORTBbits.RB5        //output

#define Lever2LowerLimit PORTBbits.RB6    //input
#define Lever2UpperLimit PORTBbits.RB7    //input
#define Lever2GoUp PORTCbits.RC0          //input
#define Lever2GoDown PORTCbits.RC1        //input
#define Lever2IsUp PORTCbits.RC2          //output
#define Lever2IsDown PORTCbits.RC3        //output

#define Lever3LowerLimit PORTCbits.RC4    //input
#define Lever3UpperLimit PORTCbits.RC5    //input
#define Lever3GoUp PORTCbits.RC6          //input
#define Lever3GoDown PORTCbits.RC7        //input
#define Lever3IsUp PORTDbits.RD0          //output
#define Lever3IsDown PORTDbits.RD1        //outpu

```

/*

The code works as next:

if Lever2UpperLimit = 1 and Lever3UpperLimit = 1

that is lever2 and lever3 are lifted, then lever 1 can be lifted.

More over if upper limit is reached Go up button is invalid and vice versa

*/

void main ()

{

 TRISB=0xCF;

 TRISC=0xF3;

 TRISD=0xFC;

 PORTB=PORTC=PORTD=0; //reset all the ports

 while (1)

 {

 if(Lever2UpperLimit && Lever3UpperLimit)

 {

 if (! Lever1UpperLimit && Lever1GoUp)

 {

 Lever1IsUp=1;

 Lever1IsDown=0;

 }

 else

 Lever1IsUp=0;

 if(!Lever1LowerLimit && Lever1GoDown)

 {

 Lever1IsUp=0;

 Lever1IsDown=1;

 }

 else

 Lever1IsDown=0;

 }

 else


```
{  
    Lever1IsUp=Lever1IsDown=0;  
}  
  
if(!Lever3UpperLimit && Lever3GoUp)  
{  
    Lever3IsUp=1;  
    Lever3IsDown=0;  
}  
else  
    Lever3IsUp=0;  
if(!Lever3LowerLimit && Lever3GoDown)  
{  
    Lever3IsUp=0;  
    Lever3IsDown=1;  
}  
else  
    Lever3IsDown=0;  
  
if(!Lever2UpperLimit && Lever2GoUp)  
{  
    Lever2IsUp=1;  
    Lever2IsDown=0;  
}  
else  
    Lever2IsUp=0;  
if(!Lever2LowerLimit && Lever2GoDown)  
{  
    Lever2IsUp=0;  
    Lever2IsDown=1;  
}  
else  
    Lever2IsDown=0;
```

```

    }
}

////////////////////////////////////
/////////          Author: M.Maksabi          //////////
/////////    Levers mechanism due to the reverse route    //////////
/////////          Device: PIC 18F452          //////////
////////////////////////////////////

#include<p18f452.h>
//*****Configuration Bits*****//
#pragma config OSC = XT
#pragma config PWRT =OFF    // Power-up timer disabled
#pragma config BOR  =OFF    // Brown-out reset disabled
#pragma config LVP  =OFF    //Low-voltage disabled
#pragma config WDT  =OFF    //Watch-dog timer disabled
#pragma config DEBUG =OFF   // Background debugger disabled
//*****Codes and Definitions *****//
#pragma code

#define Lever4LowerLimit PORTBbits.RB0    //input
#define Lever4UpperLimit PORTBbits.RB1    //input
#define Lever4GoUp PORTBbits.RB2          //input
#define Lever4GoDown PORTBbits.RB3        //input
#define Lever4IsUp PORTBbits.RB4          //output
#define Lever4IsDown PORTBbits.RB5        //output

#define Lever5LowerLimit PORTBbits.RB6    //input
#define Lever5UpperLimit PORTBbits.RB7    //input
#define Lever5GoUp PORTCbits.RC0          //input
#define Lever5GoDown PORTCbits.RC1        //input
#define Lever5IsUp PORTCbits.RC2          //output
#define Lever5IsDown PORTCbits.RC3        //output

#define Lever6LowerLimit PORTCbits.RC4    //input
#define Lever6UpperLimit PORTCbits.RC5    //input
#define Lever6GoUp PORTCbits.RC6          //input

```

```
#define Lever6GoDown PORTCbits.RC7    //input
#define Lever6IsUp PORTDbits.RD0      //output
#define Lever6IsDown PORTDbits.RD1    //output
/*
```

The code works as next:

if Lever5UpperLimit = 1 and Lever4lowerLimit = 1

that is lever5 is lifted and lever4 is in normal state "locked" , then lever 6 can be lifted.

More over if upper limit is reached Go up button is invalid and vice versa

```
*/
```

```
void main ()
```

```
{
    TRISB=0xCF;
    TRISC=0xF3;
    TRISD=0xFC;
    PORTB=PORTC=PORTD=0; // reset all the ports
    while (1)
    {
        if(Lever5UpperLimit && Lever4LowerLimit)
        {
            if(!Lever6UpperLimit && Lever6GoUp)
            {
                Lever6IsUp=1;
                Lever6IsDown=0;
            }else
                Lever6IsUp=0;
            if(!Lever6LowerLimit && Lever6GoDown)
            {
                Lever6IsUp=0;
                Lever6IsDown=1;
            } else
                Lever6IsDown=0;
        } else
        {
            Lever6IsUp=Lever6IsDown=0;
        }
    }
}
```

```
    }

    if (Lever5UpperLimit && Lever6LowerLimit)
    {
        if(!Lever4UpperLimit && Lever4GoUp)
        {
            Lever4IsUp=1;
            Lever4IsDown=0;
        } else
            Lever4IsUp=0;
        if(!Lever4LowerLimit && Lever4GoDown)
        {
            Lever4IsUp=0;
            Lever4IsDown=1;
        } else
            Lever4IsDown=0;
    }
```

```

////////////////////////////////////
/////////      Author: M.Maksabi      //////////
/////////      LED and Switches      //////////
/////////      Device: PIC 18F452      //////////
////////////////////////////////////

#include <p18f452.h>
#include <delays.h>

//***** Configuration bits *****//
#pragma config OSC   = XT      // 4MHz Crystal, (XT oscillator)
#pragma config PWRT =OFF      // Power-up timer disabled
#pragma config BOR   =OFF      // Brown-out reset disabled
#pragma config LVP   =OFF      //Low-voltage disabled
#pragma config WDT   =OFF      //Watch-dog timer disabled
#pragma config DEBUG =OFF      // Background debugger disabled
//*****Codes and Definitions *****//

#pragma code
#define ON  0
#define OFF 1

#define left 1      //Left winding motor1
#define right 2     //Right winding motor1
#define left1 3     //Left winding motor2
#define right1 4    //Right winding motor2

void delay_ms(unsigned int duration);
unsigned char is_sw1_pressed(void);
unsigned char is_sw2_pressed (void);
unsigned char is_sw3_pressed(void);
unsigned char is_sw4_pressed(void);
unsigned char is_sw5_pressed(void);
unsigned char is_sw6_pressed(void);

void direction (int dir)
{
if (dir==left)
{

```

```
PORTCbits.RC6 = ON;
Delay1KTCYx(50);
PORTCbits.RC6 = OFF;
}
else
{
PORTCbits.RC7 = ON;
Delay1KTCYx(50);
PORTCbits.RC7 =OFF;
}
if (dir==left1)
{
PORTDbits.RD4= ON;
Delay1KTCYx(50);
PORTDbits.RD4 =OFF;
}
else
{
PORTDbits.RD5 = ON;
Delay1KTCYx(50);
PORTDbits.RD5=OFF;
}
}

void main()
{
TRISCbits.TRISC0=0;    // PortC as output, LED1 is connected to this pin
TRISCbits.TRISC1=0;    // PortC as output, LED2 is connected to this pin
TRISCbits.TRISC2=0;    //PortC as output, LED3 is connected to this pin
TRISCbits.TRISC3=0;    // Set wise LED1
TRISCbits.TRISC4=0;    // Set wise LED2
TRISCbits.TRISC5=0;    // Set wise LED3
TRISAbits.TRISA0=0;    // LED 4
TRISBbits.TRISB5=0;    // wise LED4
```

```
TRISBbits.TRISB6=0; // LED5
TRISBbits.TRISB7=0; // wise LED5
TRISEbits.TRISE0=0; // LED6
TRISEbits.TRISE1=0; // wise LED6

TRISCbits.TRISC6=0;    // MOTOR1 output Left
TRISCbits.TRISC7=0;    // MOTOR1 output Right
TRISDbits.TRISD4=0;    // MOTOR2 output Left
TRISDbits.TRISD5=0;    // MOTOR2 output Right

PORTCbits.RC0=0;       // turn off LED1
PORTCbits.RC1=0;       // turn off LED2
PORTCbits.RC2=0;       // turn off LED3
PORTCbits.RC3=0;       // WISE LED1
PORTCbits.RC4=0;       // WISE LED2
PORTCbits.RC5=0;       // WISE LED3
PORTAbits.RA0=0;       // turn off LED4
PORTBbits.RB5=0;       // turn off wise LED4
PORTBbits.RB6=0;       // turn off LED5
PORTBbits.RB7=0;       // turn off wise LED5
PORTEbits.RE0=0;       // turn off LED6
PORTEbits.RE1=0;       // turn off wise LED5

PORTCbits.RC6=1;       // the winding 1 for motor1 is on
PORTCbits.RC7=1;       // the winding 2 for motor1 is on
PORTDbits.RD4=1;       // the winding 1 for motor2 is on
PORTDbits.RD5=1;       // the winding 2 for motor2 is on

TRISDbits.TRISD0=1;    //set SWITCH1 pin as input
TRISDbits.TRISD1=1;    // set SWITCH2 pin as input
TRISDbits.TRISD2=1;    // set SWITCH3 pin as input
TRISDbits.TRISD3=1;    // set SWITCH4 pin as input
TRISDbits.TRISD6=1;    // set SWITCH5 pin as input
```

```
TRISDbits.TRISD7=1;    // set SWITCH6 pin as input
    while(1)
    {
    if(is_sw1_pressed()==1) // check switch #1
    {
        PORTCbits.RC0=1;    // turn ON LED1
        PORTCbits.RC3=0;

        } else
        {
            PORTCbits.RC0=0; //turn off LED1
            PORTCbits.RC3=1;

        }
    if(is_sw2_pressed()==1)    // check switch 2
    {
        PORTCbits.RC1=1; // turn on LED
        PORTCbits.RC4=0;

        } else
        {
            PORTCbits.RC1=0; //turn off LED
            PORTCbits.RC4=1;

        }
    if(is_sw3_pressed()==1) // check the switch 3
    {
        PORTCbits.RC2=1;    // turn ON LED
        PORTCbits.RC5=0;    // turn OFF LED
        } else
        {
            PORTCbits.RC2=0; //turn off LED2
            PORTCbits.RC5=1;

            {
                direction (left);
                Delay10KTCYx(20);
```



```
    Delay10KTCYx(20);
    direction (left1);
    Delay10KTCYx(20);
    Delay10KTCYx(20);
}
}
if(is_sw4_pressed()==1)    // check switch 4
{
    PORTAbits.RA0=1; // turn off LED6
    PORTBbits.RB5=0; // turn off wise LED5
    } else
{
    PORTAbits.RA0=0; // turn off LED6
    PORTBbits.RB5=1; //
        }
    if(is_sw5_pressed()==1)    // is switch 5 pressed=> if yes
        {
            PORTBbits.RB6=1; // turn off LED5
            PORTBbits.RB7=0; // turn off wise LED5
            }    else
            {
                PORTBbits.RB6=0; // turn off LED5
                PORTBbits.RB7=1; // turn off wise LED5
                }

    if(is_sw6_pressed()==1)    // is switch 6 pressed
    {
        PORTEbits.RE0=1;
        PORTEbits.RE1=0;    // turn ON LED1

        }
        else    //if not
        {
            PORTEbits.RE0=0;    //turn off LED1
```

```

    PORTEbits.RE1=1;

    }

}

}

//*****//
//***          delay routine          ***//
// ***          Delay in milliseconds is required      ***//
// ***          No Output                ***//
//*****//

void delay_ms(unsigned int duration) // delay in milliseconds for 4.0MHZ crystal
{
    unsigned int i;
    for(;duration!=0;duration--)
    {
        for(i=0;i<=50;i++)
        {
            _asm
                nop
                nop
                nop
            _endasm
        }
        _asm
            nop
            nop
        _endasm
    }
}

```

```

//*****
//***      This routine is to examine that switch 2 is pressed or not      ***//
//***                                     No input                             ***//
//***      If sw2 is pressed=> output =1 if not output= 0                    ***//
//*****

unsigned char is_sw1_pressed(void)
{
    if (PORTDbits.RD0==0)
    {
        delay_ms(10);
        if (PORTDbits.RD0==0)
        {
            return 1;
        }
    }
    return 0;// return 0, if the switch is not pressed
}

//*****
//***      This routine is to examine that switch 2 is pressed or not      ***//
//***                                     No input                             ***//
//***      If sw2 is pressed=> output =1 if not output= 0                    ***//
//*****

unsigned char is_sw2_pressed(void)
{
    if (PORTDbits.RD1==0)
    {
        delay_ms(10);
        if (PORTDbits.RD1==0)
        {
            return 1
        }
    }
    return 0;// return 0, if the switch is not pressed
}

```

```

//*****
//***      this routine is to examine that switch 3 is pressed or not      ***//
//***                                     No input                             ***//
//***      If sw2 is pressed=> output =1 if not output= 0                    ***//
//*****

unsigned char is_sw3_pressed(void)
{
    if (PORTDbits.RD2==0)
    {
        delay_ms(10);
        if (PORTDbits.RD2==0)
        {
        }
    }
    return 0;// return 0, if the switch is not pressed
}

//*****
//***      This routine is to examine that switch 4 is pressed or not      ***//
//***                                     No input                             ***//
//***      If sw2 is pressed=> output =1 if not output= 0                    ***//
//*****

unsigned char is_sw4_pressed(void)
{
    if (PORTDbits.RD3==0)
    {
        delay_ms(10);
        if (PORTDbits.RD3==0)
        {
            return 1;
        }
    }
    return 0;// return 0, if the switch is not pressed
}

```

```

//*****
//***      this routine is to examine that switch 5 is pressed or not      ***//
//***                                     No input                             ***//
//***      If sw2 is pressed=> output =1 if not output= 0                    ***//
//*****

unsigned char is_sw5_pressed(void)
{
    if (PORTDbits.RD6==0)
    {
        delay_ms(10);
        if (PORTDbits.RD6==0)
        {
            return 1;
        }
    }
    return 0;// return0, if the switch is not pressed
}

//*****
//***      This routine is to examine that switch 6 is pressed or not      ***//
//***                                     No input                             ***//
//***      If sw2 is pressed=> output =1 if not output= 0                    ***//
//*****

unsigned char is_sw6_pressed(void)
{
    if (PORTDbits.RD7==0)
    {
        delay_ms(10);
        if (PORTDbits.RD7==0)
        {
            return 1;
        }
    }
    return 0;// return 0, if the switch is not pressed
}

```

```

////////////////////////////////////
//      Author: Mersedeh Maksabi      //
//      Name: Design Optimization      //
//      Date: 12.07.2013 at 20:12      //
////////////////////////////////////

#include <p18f452.h>
//*****Configuration Bits*****//

#pragma config PWRT =OFF    // Power-up timer disabled
#pragma config BOR  =OFF    // Brown-out reset disabled
#pragma config LVP  =OFF    // Low-voltage disabled
#pragma config WDT  =OFF    // Watch-dog timer disabled
#pragma config DEBUG =OFF   // Background debugger disabled

//*****Codes and Definitions *****//
#pragma code

#define Lever1UpperLimit PORTBbits.RB0 //input
#define Lever1GoUp PORTBbits.RB1      //input
#define Lever1IsUp PORTBbits.RB2      //output

#define Lever2UpperLimit PORTBbits.RB3 //input
#define Lever2GoUp PORTBbits.RB4      //input
#define Lever2IsUp PORTBbits.RB5      //output

#define Lever3UpperLimit PORTBbits.RB6 //input
#define Lever3GoUp PORTBbits.RB7      //input
#define Lever3IsUp PORTCbits.RC0      //output

#define Lever4LowerLimit PORTCbits.RC1 //input
#define Lever4UpperLimit PORTCbits.RC2 //input
#define Lever4GoUp PORTCbits.RC3      //input
#define Lever4GoDown PORTCbits.RC4    //input
#define Lever4IsUp PORTCbits.RC5      //output
#define Lever4IsDown PORTCbits.RC6    //output

#define Lever5UpperLimit PORTCbits.RC7 //input
#define Lever5GoUp PORTDbits.RD0      //input
#define Lever5IsUp PORTDbits.RD1      //output

#define Lever6LowerLimit PORTDbits.RD2 //input
#define Lever6UpperLimit PORTDbits.RD3 //input
#define Lever6GoUp PORTDbits.RD4      //input
#define Lever6GoDown PORTDbits.RD5    //input
#define Lever6IsUp PORTDbits.RD6      //output
#define Lever6IsDown PORTDbits.RD7    //output

```

```
void main ()
{
    TRISB=0xDB;
    TRISC=0x9E;
    TRISD=0x3D;
    PORTB=PORTC=PORTD=0;

    while(1)
    {
        if ( Lever2UpperLimit && Lever3UpperLimit)
        {

            if (!Lever1UpperLimit && Lever1GoUp)
            {
                Lever1IsUp=1;
            }
            else
                Lever1IsUp=0;
        }

        if (!Lever3UpperLimit && Lever3GoUp)
        {
            Lever3IsUp=1;
        }
        else
            Lever3IsUp=0;
        if (!Lever2UpperLimit && Lever2GoUp)
        {
            Lever2IsUp=1;
        }
        else
            Lever2IsUp=0;

        if (Lever5UpperLimit && Lever4LowerLimit)
        {
            if (!Lever6UpperLimit && Lever6GoUp)
            {
                Lever6IsUp=1;
                Lever6IsDown=0;
            }
            else
                Lever6IsUp=0;
            if(!Lever6LowerLimit && Lever6GoDown)
            {
                Lever6IsUp=0;
                Lever6IsDown=1;
            }
        }
    }
}
```

```

        }
        else
            Lever6IsDown=0;
    }
    else
    {
        Lever6IsUp=Lever6IsDown=0;
    }
    if (!Lever5UpperLimit && Lever5GoUp)
    {
        Lever5IsUp=1;
    }
    else
        Lever5IsUp=0;

    if (Lever5UpperLimit && Lever6LowerLimit)
    {
        if(!Lever4UpperLimit && Lever4GoUp)
        {
            Lever4IsUp=1;
            Lever4IsDown=0;
        }
        else
            Lever4IsUp=0;
        if(!Lever4LowerLimit && Lever4GoDown)
        {
            Lever4IsUp=0;
            Lever4IsDown=1;
        }
        else
            Lever4IsDown=0;
    }
    else
    {
        Lever4IsUp=Lever4IsDown=0;
    }
}
}

```



```

////////////////////////////////////
//      Author: Mersedeh Maksabi    //
//      Name: LCD CODE              //
//      Date: 24.08.2013 at 20:12    //
////////////////////////////////////
#include <p18f452.h>
#include <delays.h>
#include <xlcd.h>

#ifdef UPPER
#undef UPPER
#endif
#ifdef BIT8
#undef BIT8
#endif
void main ()
{
    const char arr[16]=" route is clear  \0";
    OpenXLCD( FOUR_BIT & LINES_5X7 );

    putsXLCD(arr);

    while(1);
}

void DelayPORXLCD()
{
    Delay100TCYx(250);//delay for 2.5 ms on 40 MHz
    Delay100TCYx(250);
    Delay100TCYx(250);
    Delay100TCYx(250);
    Delay100TCYx(250);
    Delay100TCYx(250);
    Delay100TCYx(250);//delay for 2.5 ms on 40 MHz
    Delay100TCYx(250);
    Delay100TCYx(250);
    Delay100TCYx(250);
    Delay100TCYx(250);
    Delay100TCYx(250);

}

void DelayFor18TCY()
{
    Delay100TCYx(250);
}

```

```

#include <p18f452.h>
#include <xlcd.h>
/*****
*   Function Name: BusyXLCD                                     *
*   Return Value:  char: busy status of LCD controller         *
*   Parameters:    void                                         *
*   Description:   This routine reads the busy status of the   *
*                  Hitachi HD44780 LCD controller.             *
*****/
unsigned char BusyXLCD(void)
{
    RW_PIN = 1;          // Set the control bits for read
    RS_PIN = 0;
    DelayFor18TCY();
    E_PIN = 1;           // Clock in the command
    DelayFor18TCY();
#ifdef BIT8              // 8-bit interface
    if(DATA_PORT&0x80)    // Read bit 7 (busy bit)
    {
        // If high
        E_PIN = 0;        // Reset clock line
        RW_PIN = 0;       // Reset control line
        return 1;         // Return TRUE
    }
    else                  // Bit 7 low
    {
        E_PIN = 0;        // Reset clock line
        RW_PIN = 0;       // Reset control line
        return 0;         // Return FALSE
    }
#else                    // 4-bit interface
#ifdef UPPER             // Upper nibble interface
    if(DATA_PORT&0x80)
#else                   // Lower nibble interface
    if(DATA_PORT&0x08)
#endif
#endif
    {
        E_PIN = 0;        // Reset clock line
        DelayFor18TCY();
        E_PIN = 1;        // Clock out other nibble
        DelayFor18TCY();
        E_PIN = 0;
        RW_PIN = 0;       // Reset control line
        return 1;         // Return TRUE
    }
    else                  // Busy bit is low
    {
        E_PIN = 0;        // Reset clock line

```

```
        DelayFor18TCY();
        E_PIN = 1;          // Clock out other nibble
        DelayFor18TCY();
        E_PIN = 0;
        RW_PIN = 0;         // Reset control line
        return 0;           // Return FALSE
    }
#endif
}
```

```

#include <p18f452.h>
#include "delays.h"
#include <xlcd.h>

/*****
*   Function Name: OpenXLCD                                     *
*   Return Value: void                                           *
*   Parameters:   lcdtype: sets the type of LCD (lines)         *
*   Description:  This routine configures the LCD. Based on     *
*                 the Hitachi HD44780 LCD controller. The       *
*                 routine will configure the I/O pins of the     *
*                 microcontroller, setup the LCD for 4- or      *
*                 8-bit mode and clear the display. The user    *
*                 must provide three delay routines:             *
*                 DelayFor18TCY() provides a 18 Tcy delay       *
*                 DelayPORXLCD() provides at least 15ms delay   *
*                 DelayXLCD() provides at least 5ms delay       *
*****/
void OpenXLCD(unsigned char lcdtype)
{
    // The data bits must be either a 8-bit port or the upper or
    // lower 4-bits of a port. These pins are made into inputs
#ifdef BIT8                // 8-bit mode, use whole port
    DATA_PORT = 0;
    TRIS_DATA_PORT = 0x00;
#else                      // 4-bit mode
#ifdef UPPER                // Upper 4-bits of the port
    DATA_PORT &= 0x0f;
    TRIS_DATA_PORT &= 0x0F;
#else                      // Lower 4-bits of the port
    DATA_PORT &= 0xf0;
    TRIS_DATA_PORT &= 0xF0;
#endif
#endif
    TRIS_RW = 0;           // All control signals made outputs
    TRIS_RS = 0;
    TRIS_E = 0;
    RW_PIN = 0;           // R/W pin made low
    RS_PIN = 0;           // Register select pin made low
    E_PIN = 0;            // Clock pin made low

    // Delay for 15ms to allow for LCD Power on reset
    DelayPORXLCD();
}

```

```

//-----reset procedure through software-----
    WriteCmdXLCD(0x30);
        Delay10KTCYx(0x05);

    WriteCmdXLCD(0x30);
        Delay10KTCYx(0x01);

    WriteCmdXLCD(0x32);
    while( BusyXLCD() );
//-----

// Set data interface width, # lines, font
while(BusyXLCD());          // Wait if LCD busy
WriteCmdXLCD(lcdtype);      // Function set cmd

// Turn the display on then off
while(BusyXLCD());          // Wait if LCD busy
WriteCmdXLCD(DOFF&CURSOR_OFF&BLINK_OFF);    // Display
OFF/Blink OFF
while(BusyXLCD());          // Wait if LCD busy
WriteCmdXLCD(DON&CURSOR_ON&BLINK_ON);        // Display
ON/Blink ON

// Clear display
while(BusyXLCD());          // Wait if LCD busy
WriteCmdXLCD(0x01);        // Clear display

// Set entry mode inc, no shift
while(BusyXLCD());          // Wait if LCD busy
WriteCmdXLCD(SHIFT_CUR_LEFT); // Entry Mode

// Set DD Ram address to 0
while(BusyXLCD());          // Wait if LCD busy
SetDDRamAddr(0x80);         // Set Display data ram address to 0

return;
}

```

```

#include <p18f452.h>

#include <xlcd.h>

/*****
 *   Function Name:  putsXLCD                               *
 *   Return Value:   void                                    *
 *   Parameters:     buffer: pointer to string               *
 *   Description:    This routine writes a string of bytes to the *
 *                   Hitachi HD44780 LCD controller. The user *
 *                   must check to see if the LCD controller is *
 *                   busy before calling this routine. The data *
 *                   is written to the character generator RAM or *
 *                   the display data RAM depending on what the *
 *                   previous SetxxRamAddr routine was called. *
 *****/
void putsXLCD(const char *buffer)
{
    unsigned char val=0;
    while(*buffer)      // Write data to LCD up to null
    {
        while(BusyXLCD());    // Wait while LCD is busy
        SetDDRamAddr(val);
        WriteDataXLCD(*buffer); // Write character to LCD
        buffer++;              // Increment buffer
        val++;
    }
    return;
}

```

```

#include <p18F452.h>
#include <xlcd.h>

/*****
*   Function Name: putsXLCD                                     *
*   Return Value:  void                                         *
*   Parameters:    buffer: pointer to string                   *
*   Description:   This routine writes a string of bytes to the *
*                  Hitachi HD44780 LCD controller. The user    *
*                  must check to see if the LCD controller is  *
*                  busy before calling this routine. The data  *
*                  is written to the character generator RAM or *
*                  the display data RAM depending on what the  *
*                  previous SetxxRamAddr routine was called.    *
*****/
void putsXLCD(char *buffer)
{
    while(*buffer)        // Write data to LCD up to null
    {
        while(BusyXLCD()); // Wait while LCD is busy
        WriteDataXLCD(*buffer); // Write character to LCD
        buffer++;           // Increment buffer
    }
    return;
}

```

```

#include <p18F452.h>
#include <xlcd.h>

/*****
*
*      Function Name: ReadAddrXLCD
*      Return Value:  char: address from LCD controller
*      Parameters:    void
*      Description:   This routine reads an address byte from the
*                    Hitachi HD44780 LCD controller. The user
*                    must check to see if the LCD controller is
*                    busy before calling this routine. The address
*                    is read from the character generator RAM or
*                    the display data RAM depending on what the
*                    previous SetxxRamAddr routine was called.
*****/

/
unsigned char ReadAddrXLCD(void)
{
    char data;          // Holds the data retrieved from the LCD

#ifdef BIT8             // 8-bit interface
    RW_PIN = 1;         // Set control bits for the read
    RS_PIN = 0;
    DelayFor18TCY();
    E_PIN = 1;          // Clock data out of the LCD controller
    DelayFor18TCY();
    data = DATA_PORT;  // Save the data in the register
    E_PIN = 0;
    RW_PIN = 0;         // Reset the control bits
#else
    // 4-bit interface
    RW_PIN = 1;         // Set control bits for the read
    RS_PIN = 0;
    DelayFor18TCY();
    E_PIN = 1;          // Clock data out of the LCD controller
    DelayFor18TCY();
#ifdef UPPER            // Upper nibble interface
    data = DATA_PORT & 0xf0; // Read the nibble into the upper nibble of data
#else
    // Lower nibble interface
    data = (DATA_PORT << 4) & 0xf0; // Read the nibble into the upper nibble of
data
#endif
    E_PIN = 0;          // Reset the clock

```



```
    DelayFor18TCY();
    E_PIN = 1;           // Clock out the lower nibble
    DelayFor18TCY();
#ifdef UPPER             // Upper nibble interface
    data |= (DATA_PORT>>4)&0x0f; // Read the nibble into the lower nibble of
data
#else                     // Lower nibble interface
    data |= DATA_PORT&0x0f;    // Read the nibble into the lower nibble of data
#endif
    E_PIN = 0;
    RW_PIN = 0;           // Reset the control lines
#endif
    return (data&0x7f);    // Return the address, Mask off the busy bit
```

```

#include <p18F452.h>
#include <xlcd.h>

/*****
*   Function Name: ReadDataXLCD                                     *
*   Return Value:  char: data byte from LCD controller             *
*   Parameters:    void                                           *
*   Description:   This routine reads a data byte from the        *
*                   Hitachi HD44780 LCD controller. The user      *
*                   must check to see if the LCD controller is    *
*                   busy before calling this routine. The data    *
*                   is read from the character generator RAM or    *
*                   the display data RAM depending on what the    *
*                   previous SetxxRamAddr routine was called.     *
*****/
char ReadDataXLCD(void)
{
    char data;

#ifdef BIT8                // 8-bit interface
    RS_PIN = 1;            // Set the control bits
    RW_PIN = 1;
    DelayFor18TCY();
    E_PIN = 1;            // Clock the data out of the LCD
    DelayFor18TCY();
    data = DATA_PORT;    // Read the data
    E_PIN = 0;
    RS_PIN = 0;          // Reset the control bits
    RW_PIN = 0;
#else                      // 4-bit interface
    RW_PIN = 1;
    RS_PIN = 1;
    DelayFor18TCY();
    E_PIN = 1;            // Clock the data out of the LCD
    DelayFor18TCY();
#ifdef UPPER              // Upper nibble interface
    data = DATA_PORT&0xf0; // Read the upper nibble of data
#else                    // Lower nibble interface
    data = (DATA_PORT<<4)&0xf0; // read the upper nibble of data
#endif
    E_PIN = 0;            // Reset the clock line
    DelayFor18TCY();
    E_PIN = 1;            // Clock the next nibble out of the LCD
    DelayFor18TCY();
#ifdef UPPER              // Upper nibble interface
    data |= (DATA_PORT>>4)&0x0f; // Read the lower nibble of data
#else                    // Lower nibble interface
    data |= DATA_PORT&0x0f; // Read the lower nibble of data
#endif
}

```

```
#endif
    E_PIN = 0;
    RS_PIN = 0;           // Reset the control bits
    RW_PIN = 0;
#endif
    return(data);         // Return the data byte
}
```

```

#include <p18F452.h>
#include <xlcd.h>

/*****
*   Function Name: SetCGRamAddr                                     *
*   Return Value: void                                             *
*   Parameters:   CGaddr: character generator ram address         *
*   Description:  This routine sets the character generator       *
*                 address of the Hitachi HD44780 LCD              *
*                 controller. The user must check to see if       *
*                 the LCD controller is busy before calling       *
*                 this routine.                                    *
*****/
void SetCGRamAddr(unsigned char CGaddr)
{
#ifdef BIT8                // 8-bit interface
    TRIS_DATA_PORT = 0;          // Make data port output
    DATA_PORT = CGaddr | 0b01000000; // Write cmd and address to port
    RW_PIN = 0;                 // Set control signals
    RS_PIN = 0;
    DelayFor18TCY();
    E_PIN = 1;                 // Clock cmd and address in
    DelayFor18TCY();
    E_PIN = 0;
    DelayFor18TCY();
    TRIS_DATA_PORT = 0xff;      // Make data port inputs
#else                          // 4-bit interface
#ifdef UPPER                  // Upper nibble interface
    TRIS_DATA_PORT &= 0x0f;    // Make nibble input
    DATA_PORT &= 0x0f;        // and write upper nibble
    DATA_PORT |= ((CGaddr | 0b01000000) & 0xf0);
#else                          // Lower nibble interface
    TRIS_DATA_PORT &= 0xf0;    // Make nibble input
    DATA_PORT &= 0xf0;        // and write upper nibble
    DATA_PORT |= (((CGaddr | 0b01000000)>>4) & 0x0f);
#endif
#endif
}

```

```

    RW_PIN = 0;                // Set control signals
    RS_PIN = 0;
    DelayFor18TCY();
    E_PIN = 1;                  // Clock cmd and address in
    DelayFor18TCY();
    E_PIN = 0;

#ifdef UPPER                    // Upper nibble interface
    DATA_PORT &= 0x0f;         // Write lower nibble
    DATA_PORT |= ((CGaddr<<4)&0xf0);
#else                            // Lower nibble interface
    DATA_PORT &= 0xf0;         // Write lower nibble
    DATA_PORT |= (CGaddr&0x0f);
#endif

    DelayFor18TCY();
    E_PIN = 1;                  // Clock cmd and address in
    DelayFor18TCY();
    E_PIN = 0;

#ifdef UPPER                    // Upper nibble interface
    TRIS_DATA_PORT |= 0xf0;     // Make inputs
#else                            // Lower nibble interface
    TRIS_DATA_PORT |= 0x0f;     // Make inputs
#endif
#endif
    return;
}

```

```

#include <p18F452.h>
#include <xlcd.h>

```

```

/*****
*   Function Name: SetDDRamAddr
*   Return Value: void
*   Parameters:   CGaddr: display data address
*   Description:  This routine sets the display data address
*                 of the Hitachi HD44780 LCD controller. The
*                 user must check to see if the LCD controller
*                 is busy before calling this routine.
*****/
void SetDDRamAddr(unsigned char DDaddr)
{
#ifdef BIT8 // 8-bit interface
    TRIS_DATA_PORT = 0; // Make port output
    DATA_PORT = DDaddr | 0b10000000; // Write cmd and address to port
    RW_PIN = 0; // Set the control bits
    RS_PIN = 0;
    DelayFor18TCY();
    E_PIN = 1; // Clock the cmd and address in
    DelayFor18TCY();
    E_PIN = 0;
    DelayFor18TCY();
    TRIS_DATA_PORT = 0xff; // Make port input
#else // 4-bit interface
#ifdef UPPER // Upper nibble interface
    TRIS_DATA_PORT &= 0x0f; // Make port output
    DATA_PORT &= 0x0f; // and write upper nibble
    DATA_PORT |= ((DDaddr | 0b10000000) & 0xf0);
#else // Lower nibble interface
    TRIS_DATA_PORT &= 0xf0; // Make port output
    DATA_PORT &= 0xf0; // and write upper nibble
    DATA_PORT |= (((DDaddr | 0b10000000)>>4) & 0x0f);
#endif
#endif
    RW_PIN = 0; // Set control bits
    RS_PIN = 0;
    DelayFor18TCY();
    E_PIN = 1; // Clock the cmd and address in
    DelayFor18TCY();
    E_PIN = 0;
#ifdef UPPER // Upper nibble interface
    DATA_PORT &= 0x0f; // Write lower nibble
    DATA_PORT |= ((DDaddr<<4)&0xf0);
#else // Lower nibble interface
    DATA_PORT &= 0xf0; // Write lower nibble
    DATA_PORT |= (DDaddr&0x0f);
#endif
    DelayFor18TCY();
    E_PIN = 1; // Clock the cmd and address in

```

```
    DelayFor18TCY();
    E_PIN = 0;
#ifdef UPPER                // Upper nibble interface
    TRIS_DATA_PORT |= 0xf0;  // Make port input
#else                        // Lower nibble interface
    TRIS_DATA_PORT |= 0x0f;  // Make port input
#endif
#endif
    return;
}
```

```

#include <p18F452.h>
#include <xlcd.h>

/*****
*   Function Name: WriteCmdXLCD
*   Return Value: void
*   Parameters:   cmd: command to send to LCD
*   Description:  This routine writes a command to the Hitachi
*                 HD44780 LCD controller. The user must check
*                 to see if the LCD controller is busy before
*                 calling this routine.
*****/
void WriteCmdXLCD(unsigned char cmd)
{
#ifdef BIT8           // 8-bit interface
    TRIS_DATA_PORT = 0;           // Data port output
    DATA_PORT = cmd;           // Write command to data port
    RW_PIN = 0;                 // Set the control signals
    RS_PIN = 0;                 // for sending a command
    DelayFor18TCY();
    E_PIN = 1;                 // Clock the command in
    DelayFor18TCY();
    E_PIN = 0;
    DelayFor18TCY();
    TRIS_DATA_PORT = 0xff;       // Data port input
#else
    // 4-bit interface
#ifdef UPPER           // Upper nibble interface
    TRIS_DATA_PORT &= 0x0f;
    DATA_PORT &= 0x0f;
    DATA_PORT |= cmd&0xf0;
#else
    // Lower nibble interface
    TRIS_DATA_PORT &= 0xf0;
    DATA_PORT &= 0xf0;
    DATA_PORT |= (cmd>>4)&0x0f;
#endif
#endif

    RW_PIN = 0;                 // Set control signals for command
    RS_PIN = 0;
    DelayFor18TCY();
    E_PIN = 1;                 // Clock command in
    DelayFor18TCY();
    E_PIN = 0;

#ifdef UPPER           // Upper nibble interface
    DATA_PORT &= 0x0f;
    DATA_PORT |= (cmd<<4)&0xf0;
#else
    // Lower nibble interface
    DATA_PORT &= 0xf0;

```



```
    DATA_PORT |= cmd&0x0f;
#endif

    DelayFor18TCY();
    E_PIN = 1;           // Clock command in
    DelayFor18TCY();
    E_PIN = 0;
#ifdef UPPER             // Make data nibble input
    TRIS_DATA_PORT |= 0xf0;
#else
    TRIS_DATA_PORT |= 0x0f;
#endif
#endif
    return;
}
```

```

#include <p18F452.h>
#include <xlcd.h>
/*****
*   Function Name: WriteDataXLCD                                     *
*   Return Value:  void                                             *
*   Parameters:   data: data byte to be written to LCD             *
*   Description:  This routine writes a data byte to the           *
*                 Hitachi HD44780 LCD controller. The user         *
*                 must check to see if the LCD controller is       *
*                 busy before calling this routine. The data       *
*                 is written to the character generator RAM or     *
*                 the display data RAM depending on what the       *
*                 previous SetxxRamAddr routine was called.        *
*****/
void WriteDataXLCD(char data)
{
#ifdef BIT8                // 8-bit interface
    TRIS_DATA_PORT = 0;    // Make port output
    DATA_PORT = data;     // Write data to port
    RS_PIN = 1;           // Set control bits
    RW_PIN = 0;
    DelayFor18TCY();
    E_PIN = 1;            // Clock data into LCD
    DelayFor18TCY();
    E_PIN = 0;
    RS_PIN = 0;           // Reset control bits
    TRIS_DATA_PORT = 0xff; // Make port input
#else                      // 4-bit interface
#ifdef UPPER              // Upper nibble interface
    TRIS_DATA_PORT &= 0x0f;
    DATA_PORT &= 0x0f;
    DATA_PORT |= data&0xf0;
#else                    // Lower nibble interface
    TRIS_DATA_PORT &= 0xf0;
    DATA_PORT &= 0xf0;
    DATA_PORT |= ((data>>4)&0x0f);
#endif
#endif
    RS_PIN = 1;           // Set control bits
    RW_PIN = 0;
    DelayFor18TCY();
    E_PIN = 1;            // Clock nibble into LCD
    DelayFor18TCY();
    E_PIN = 0;
#ifdef UPPER              // Upper nibble interface
    DATA_PORT &= 0x0f;
    DATA_PORT |= ((data<<4)&0xf0);
#else                    // Lower nibble interface

```

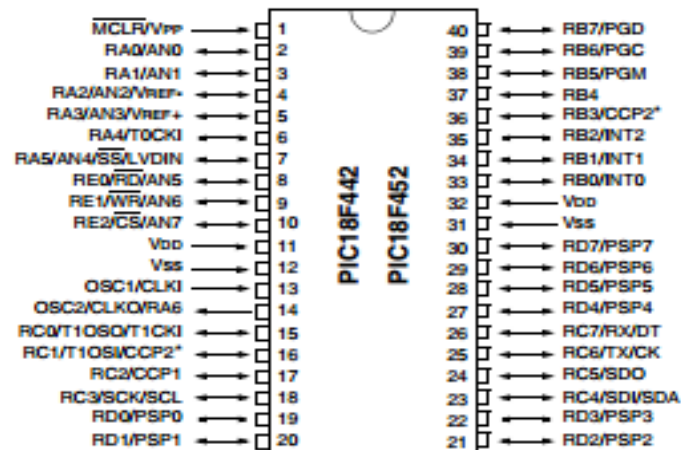
```
    DATA_PORT &= 0xf0;
    DATA_PORT |= (data&0x0f);
#endif
    DelayFor18TCY();
    E_PIN = 1;           // Clock nibble into LCD
    DelayFor18TCY();
    E_PIN = 0;
#ifdef UPPER             // Upper nibble interface
    TRIS_DATA_PORT |= 0xf0;
#else                    // Lower nibble interface
    TRIS_DATA_PORT |= 0x0f;
#endif
#endif
    return;
}
```

Appendix B

Electrical components datasheets

- **Microcontroller (PIC18f452)**

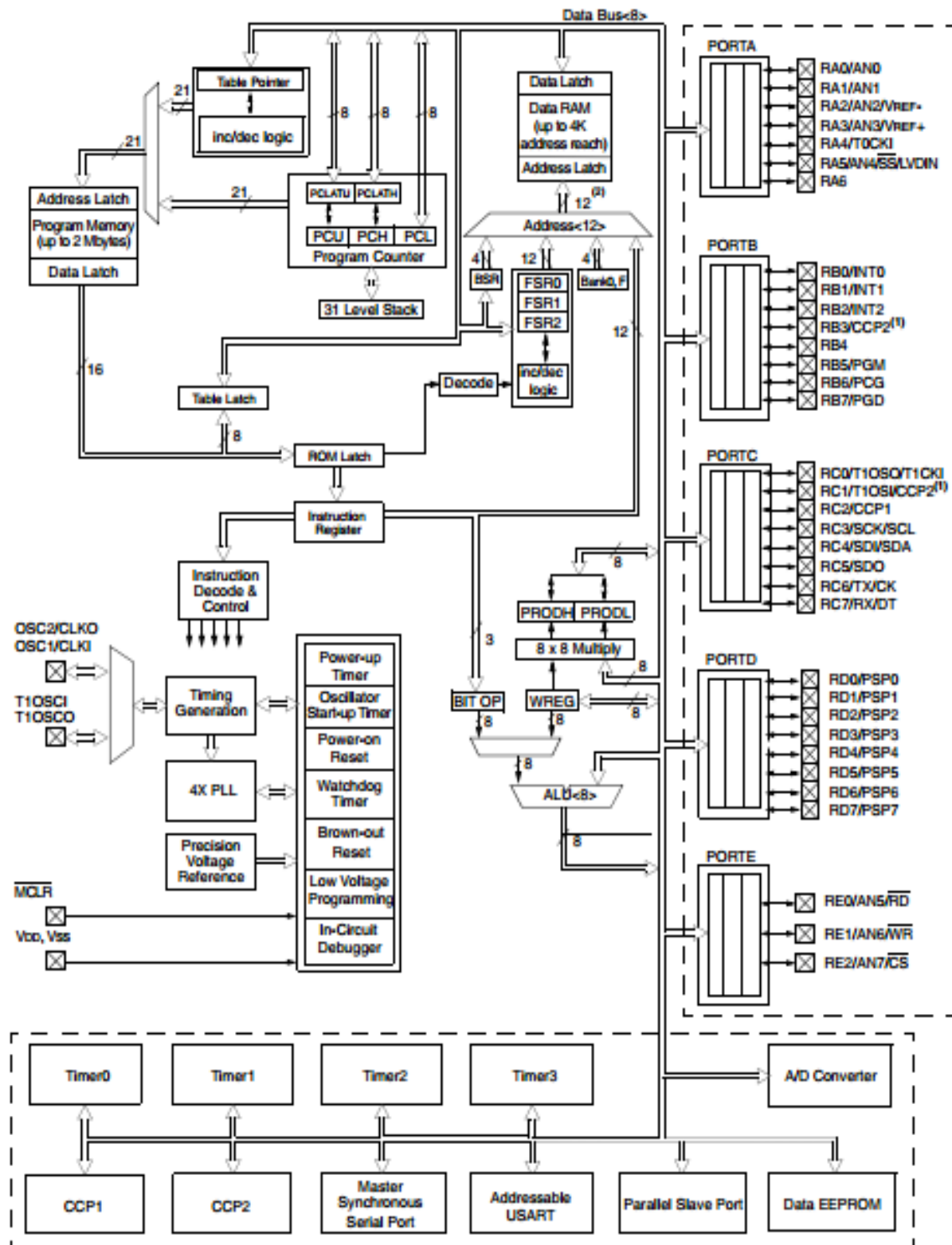
Pin connection (Top view)



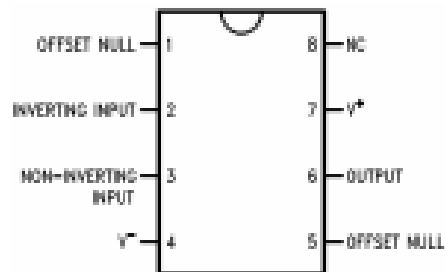
Absolute maximum rating

Ambient temperature under bias.....	-55°C to +125°C
Storage temperature	-65°C to +150°C
Voltage on any pin with respect to Vss (except VDD, $\overline{\text{MCLR}}$, and RA4)	-0.3V to (VDD + 0.3V)
Voltage on VDD with respect to Vss	-0.3V to +7.5V
Voltage on $\overline{\text{MCLR}}$ with respect to Vss (Note 2)	0V to +13.25V
Voltage on RA4 with respect to Vss	0V to +8.5V
Total power dissipation (Note 1)	1.0W
Maximum current out of Vss pin	300 mA
Maximum current into VDD pin	250 mA
Input clamp current, I _{IK} (V _I < 0 or V _I > VDD).....	±20 mA
Output clamp current, I _{OK} (V _O < 0 or V _O > VDD)	±20 mA
Maximum output current sunk by any I/O pin.....	25 mA
Maximum output current sourced by any I/O pin	25 mA
Maximum current sunk by PORTA, PORTB, and PORTE (Note 3) (combined).....	200 mA
Maximum current sourced by PORTA, PORTB, and PORTE (Note 3) (combined).....	200 mA
Maximum current sunk by PORTC and PORTD (Note 3) (combined).....	200 mA
Maximum current sourced by PORTC and PORTD (Note 3) (combined).....	200 mA

Block diagram



- **LM741**

Pin connection (Top view)

Absolute maximum rating

	LM741A	LM741	LM741C
Supply Voltage	$\pm 22\text{V}$	$\pm 22\text{V}$	$\pm 18\text{V}$
Power Dissipation ⁽⁴⁾	500 mW	500 mW	500 mW
Differential Input Voltage	$\pm 30\text{V}$	$\pm 30\text{V}$	$\pm 30\text{V}$
Input Voltage ⁽²⁾	$\pm 15\text{V}$	$\pm 15\text{V}$	$\pm 15\text{V}$
Output Short Circuit Duration	Continuous	Continuous	Continuous
Operating Temperature Range	-55°C to $+125^\circ\text{C}$	-55°C to $+125^\circ\text{C}$	0°C to $+70^\circ\text{C}$
Storage Temperature Range	-65°C to $+150^\circ\text{C}$	-65°C to $+150^\circ\text{C}$	-65°C to $+150^\circ\text{C}$
Junction Temperature	150°C	150°C	100°C
Soldering Information			
P0008E-Package (10 seconds)	260°C	260°C	260°C
NAB0008A or LMC0008C-Package (10 seconds)	300°C	300°C	300°C
M-Package			
Vapor Phase (60 seconds)	215°C	215°C	215°C
Infrared (15 seconds)	215°C	215°C	215°C
ESD Tolerance ⁽⁶⁾	400V	400V	400V

Electrical characteristics

Parameter	Test Conditions	LM741A			LM741			LM741C			Units
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Input Offset Voltage	$T_A = 25^\circ\text{C}$										mV
	$R_S \leq 10\text{ k}\Omega$					1.0	5.0		2.0	6.0	
	$R_S \leq 50\Omega$		0.8	3.0							
Average Input Offset Voltage Drift	$T_{AMN} \leq T_A \leq T_{AMAX}$										mV
	$R_S \leq 50\Omega$			4.0							
	$R_S \leq 10\text{ k}\Omega$						6.0			7.5	
Average Input Offset Voltage Drift				15							$\mu\text{V}/^\circ\text{C}$

- Reed Relay: SIL05-1A72-71D

SIL Series**Single-In-Line
Reed Relays****MEDER electronic****DESCRIPTION**

Single-In-Line Reed Relays reduce the required space to a minimum. Requiring only half the PCB area of the DIP or DIL series, the SIL relays offer all the advantages of Reed Technology.

CHARACTERISTICS

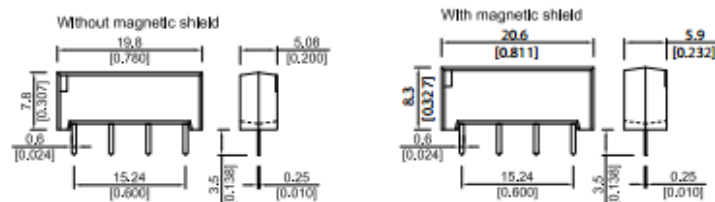
- High resistance coils of up to 2000 Ω at 12 VDC
- Breakdown voltage coil / contact of up to 4.25 kVDC
- Contact form 1A, 1B or 1C

FEATURES

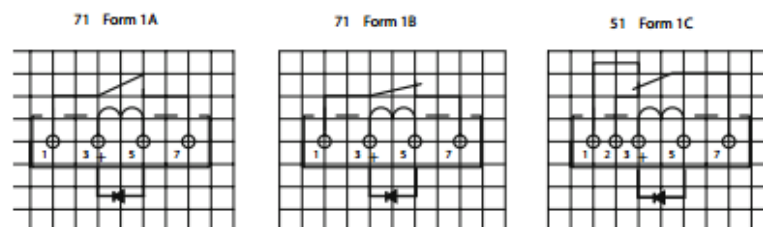
- Magnetic shield available
- High resistance version
- Other coil resistances available
- Option with coax screen for Z=50 Ohm Impedance

DIMENSIONS

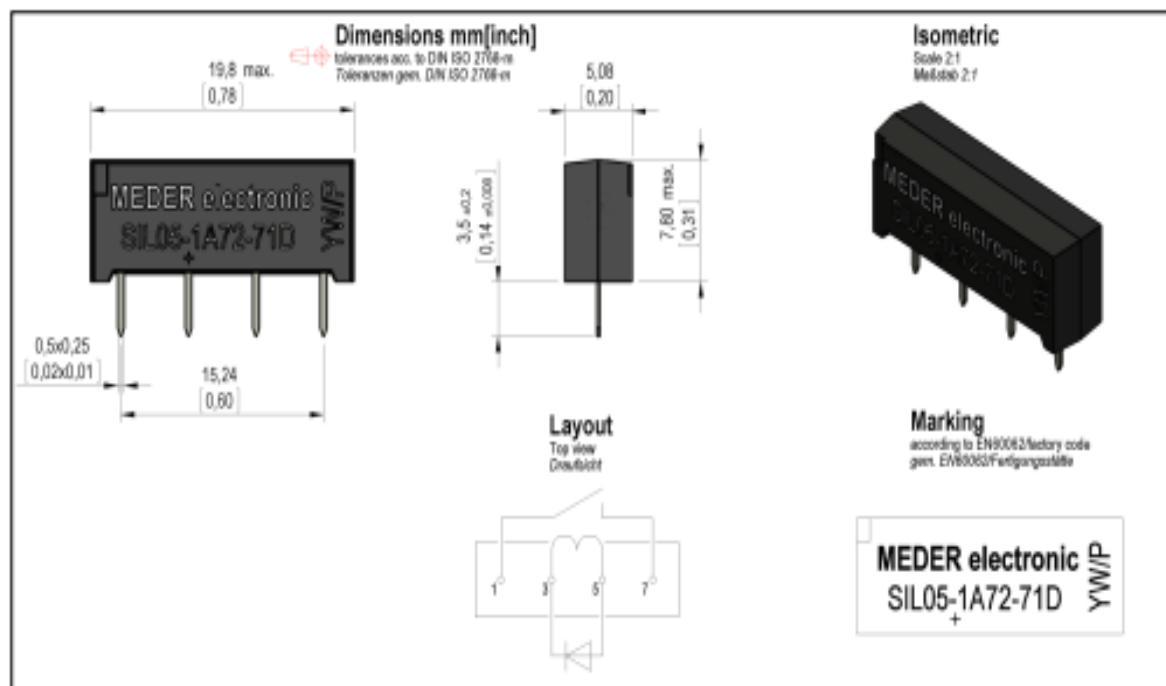
All dimensions in mm [inch]

**PIN OUT**

View from top of component, 2.54mm [0.10"] pitch grid



"+" by option with diode

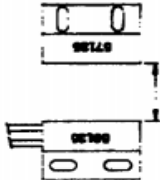
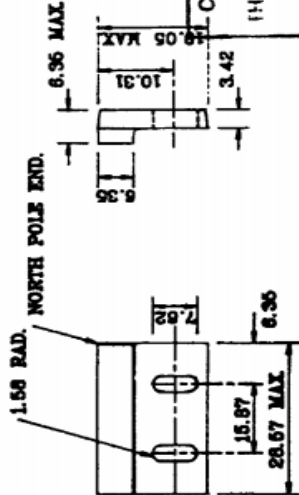



Coil Data at 20 °C	Conditions	Min	Typ	Max	Unit
Coil resistance		450	500	550	Ohm
Coil voltage			5		VDC
Rated power			50		mW
Thermal resistance	max. Relay temperature - operating temperature + self heating		109		K/W
Pull-In voltage				3,5	VDC
Drop-Out voltage		0,75			VDC

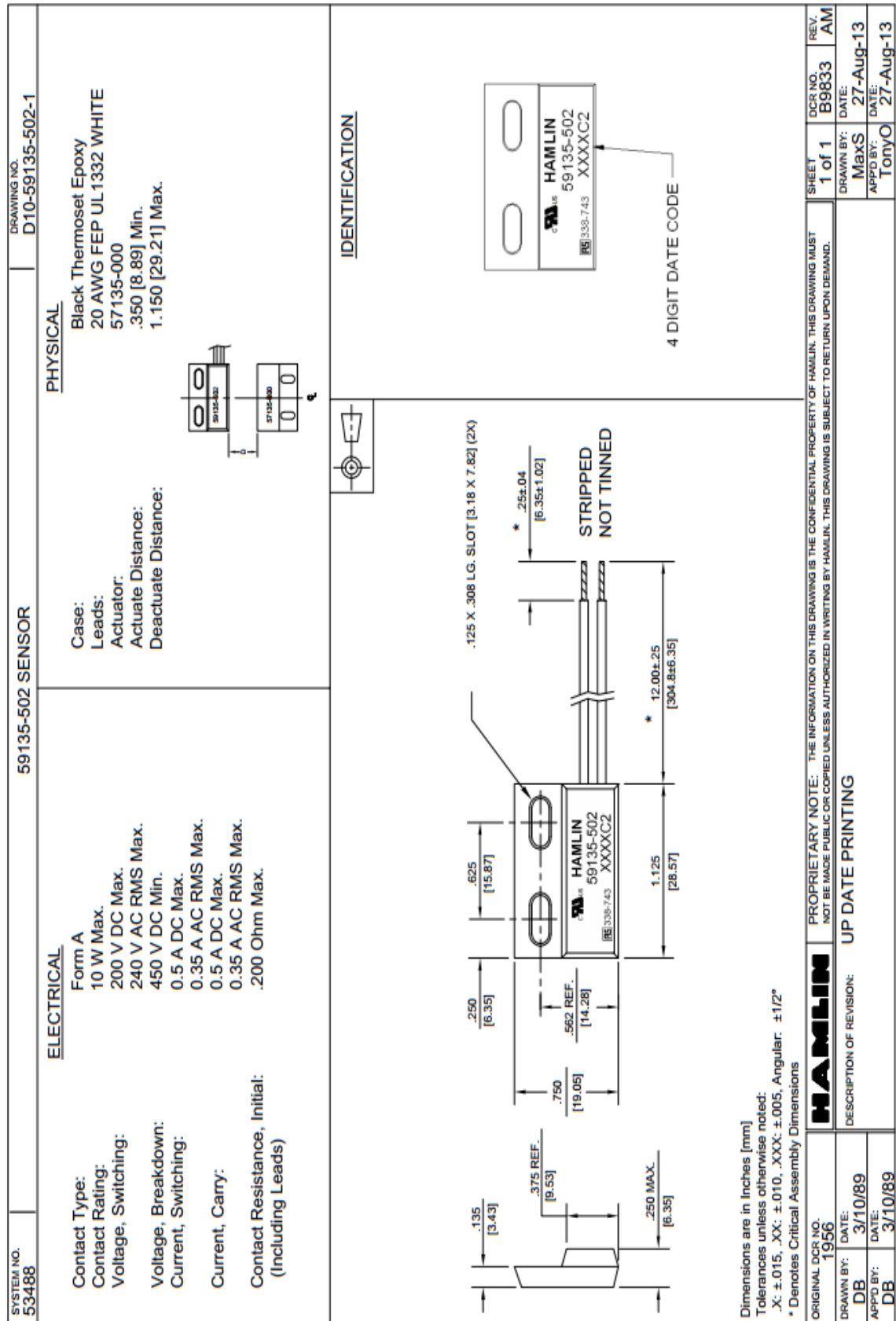
Contact data 66/3	Conditions	Min	Typ	Max	Unit
Contact rating	Any DC combination of V & A not to exceed their individual max.'s			10	W
Switching voltage	DC or Peak AC			200	V
Switching current	DC or Peak AC			0,5	A
Carry current	DC or Peak AC			1	A
Contact resistance static	Measured with 40% overdrive Start Value			150	mOhm
Contact resistance dynamic	Maximum value 1,5 ms after excitation Start Value			150	mOhm
Insulation resistance	RIH <45 %, 100 V test voltage	10			GOhm
Breakdown voltage	according to IEC 255-5	250			VDC
Breakdown voltage (> 20 AT)	according to IEC 255-5	400			VDC
Operate time incl. bounce	measured with 40% overdrive			0,7	ms
Release time	measured with no coil excitation			0,05	ms
Capacity	@ 10 kHz across open switch		0,3		pF

Special Product Data	Conditions	Min	Typ	Max	Unit
Dielectric Strength Coil/Contact	according to EN 60255-5	1,5			kV DC
Insulation resistance Coil/Contact	RIH <45%, 200 VDC measuring voltage	100			GOhm
Capacity Coil/Contact	@ 10 kHz		0,3		pF
Housing material		epoxy resin			
Connection pins		Copper alloy tin plated			
Reach / RoHS conformity		yes			

- magnetic proximity sensor actuator 10-51135-500

Drg. No.: 10-57135-500		Title: ACTUATOR		Used On: 10-57135-500	
DESCRIPTION: - MAGNETIC PROXIMITY SENSOR ACTUATOR		CUSTOMER: - RS. COMPONENTS		CUSTOMER DRG. No.: - 338-759	
CONSTRUCTION: - TRANSFER MOULDED					
<p><u>ELECTRICAL</u></p> <p>Maximum Switched Current</p> <p>Maximum Switched Voltage</p> <p>Maximum Switched Power</p> <p>Maximum Initial Contact Resistance (Inc. leadouts)</p> <p>Maximum Carry Current</p> <p>Minimum Insulation Resistance</p> <p>Minimum Voltage Holdoff</p>		<p><u>PHYSICAL</u></p> <p>Body Material: - Thermoset Generic</p> <p>Operate Distance: -</p>			
<p>FOR CONDITIONS OF TESTING REFER TO DRG. No. 30.003</p> <p>ALL DIMENSIONS ARE ± 0.25 UNLESS OTHERWISE STATED</p>					
					
<p>PRINT DETAIL</p> 					
MOD. No.	DATE	ISSUE No.	MOD. No.	DATE	ISSUE No.
A2210	20.2.92	1			
A2390	14.4.92	2			
	30.6.93	3			
<p>© HAMILTON ELECTRONICS (EUROPE) LTD. 1992</p> <p>THIS DESIGN AND PRINT IS COPYRIGHT AND IS THE PROPERTY OF HELL. IT MUST NOT BE COPIED OR REPRODUCED IN ANY FORM OR BY ANY MEANS, ELECTRONIC OR MECHANICAL, INCLUDING PHOTOCOPYING, RECORDING, OR BY ANY INFORMATION STORAGE AND RETRIEVAL SYSTEM, WITHOUT THE WRITTEN PERMISSION FROM THE COMPANY.</p>		<p>Dr. Wm. Approved</p> <p>Sheet 1 of 1 Sheet(s)</p> <p>Inspected</p> <p>Issued</p>		<p>Dimensions: mm</p> <p>Scale: 1 : 1</p> <p>Date: 13.2.92</p> <p>File: A</p>	
Drg. No.: 10-57135-500					

- Reed switch (Sensor)

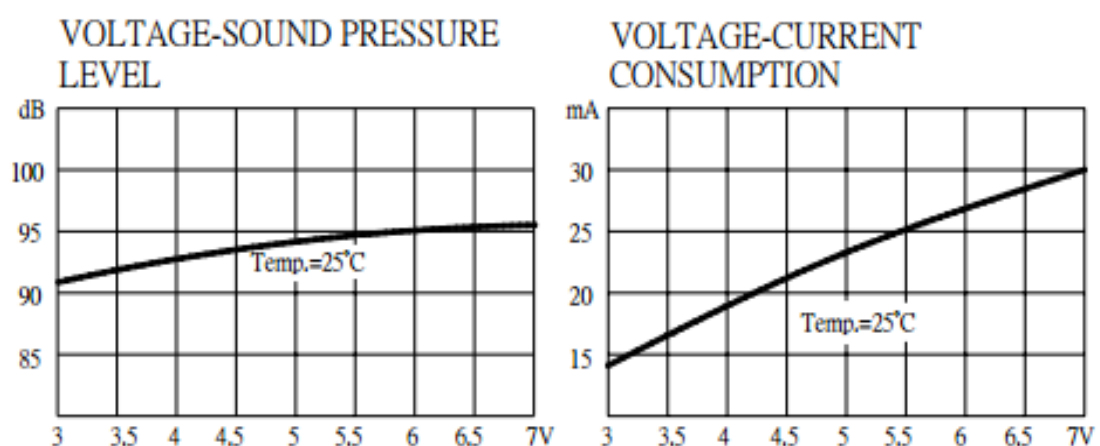


-
- **Buzzer**

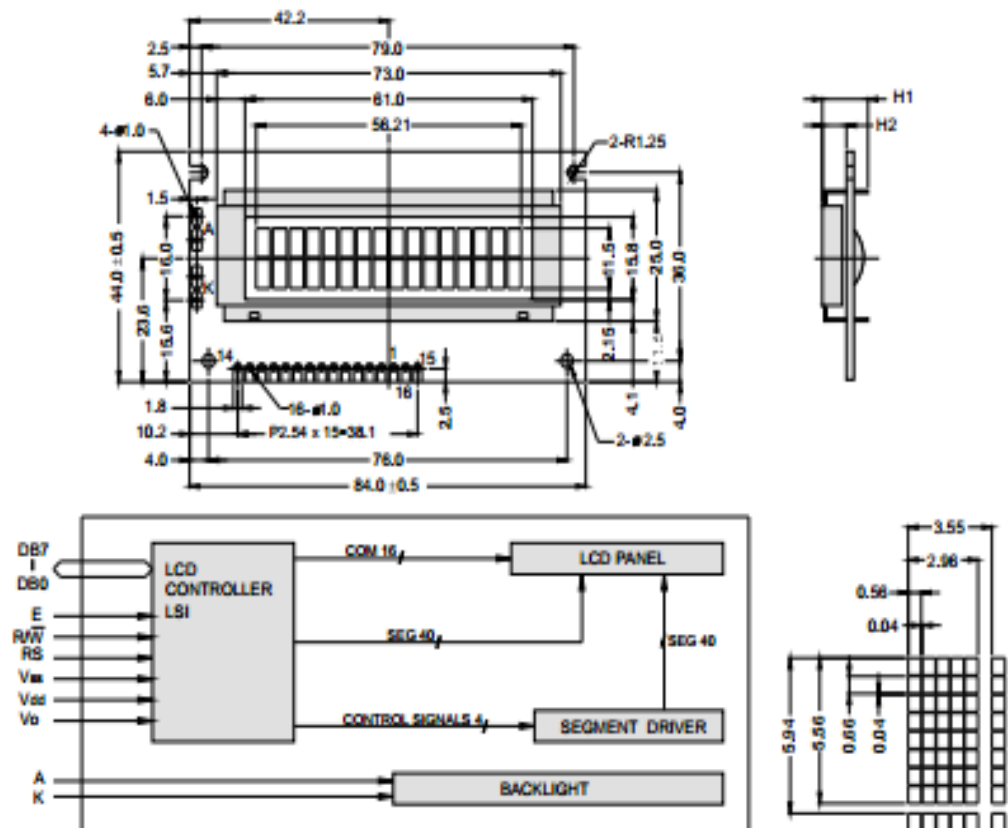
Specification

No.	Item	Unit	Specification	Condition
1	Rated Voltage 額定電壓	V _{DC}	5.0	
2	Operating Volt. 操作電壓	V _{DC}	3.0~7.0	
3	Mean Current 平均耗電流	mA	Max. 30	
4	Sound Output 輸出音壓	dBa	Min.85 (Typical 94)	Distance at 10cm(A-weight free air). Applying rated voltage. 在自由音場 10 公分處,施以額定電壓.
5	Resonant Frequency 譜振頻率	Hz	2300 ± 300	
6	Operating Temp. 操作溫度	℃	-30 ~ +70	
7	Storage Temp. 儲存溫度	℃	-40 ~ +85	
8	Dimension 尺寸	mm	φ 12.0 × H9.5	See attached drawing. 請參照外觀尺寸圖
9	Weight 重量	gram	2.0	
10	Material 材質		PBT+15%Glass (Black)	
11	Terminal 端子		Pin type (鍍化金/Plating Au)	See attached drawing. 請參照外觀尺寸圖
12	Environmental Protection Regulation 環保法規		RoHS	

Typical frequency response curve



- LCD (PC1602F B)



The tolerance unless classified $\pm 0.3\text{mm}$

MECHANICAL SPECIFICATION			
Overall Size	84.0 x 44.0	Module	H2 / H1
View Area	61.0 x 15.8	W/O B/L	5.1 / 9.7
Dot Size	0.56 x 0.66	EL B/L	5.1 / 9.7
Dot Pitch	0.60 x 0.70	LED B/L	9.4 / 14.0

PIN ASSIGNMENT		
Pin no.	Symbol	Function
1	Vss	Power supply(GND)
2	Vdd	Power supply(+)
3	Vo	Contrast Adjust
4	RS	Register select signal
5	R/W	Data read / write
6	E	Enable signal
7	DB0	Data bus line
8	DB1	Data bus line
9	DB2	Data bus line
10	DB3	Data bus line
11	DB4	Data bus line
12	DB5	Data bus line
13	DB6	Data bus line
14	DB7	Data bus line
15	A	Power supply for LED B/L (+)
16	K	Power supply for LED B/L (-)

ABSOLUTE MAXIMUM RATING					
Item	Symbol	Condition	Min.	Max.	Units
Supply for logic voltage	Vdd-Vss	25°C	-0.3	7	V
LCD driving supply voltage	Vdd-Vee	25°C	-0.3	13	V
Input voltage	Vin	25°C	-0.3	Vdd+0.3	V
ELECTRICAL CHARACTERISTICS					
Item	Symbol	Condition	Min.	Typical	Max. Units
Power supply voltage	Vdd-Vss	25°C	2.7	-	5.5 V
LCD operation voltage	Vop	Top	N	W	N W V
		-20°C	-	7.1	- 7.5 - 7.9 V
		0°C	4.5	-	5.1 - 5.3 - V
		25°C	4.1	6.1	4.7 6.4 4.9 6.7 V
		50°C	3.8	-	4.4 - 4.6 - V
LCD current consumption (No B/L)	Idd	Vdd=5V	-	2	3 mA
			-	40	- mA
Backlight current consumption	LED/edge	VB/L=4.2V	-	120	- mA
			-	-	- mA

- PNP Transistor(CBC212L)

BC212, BC212B, BC213, BC214

Amplifier Transistors

PNP Silicon

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Collector-Emitter Voltage BC212 BC213 BC214	V_{CE0}	-50 -30 -30	Vdc
Collector-Base Voltage BC212 BC213 BC214	V_{CB0}	-60 -45 -45	Vdc
Emitter-Base Voltage	V_{EB0}	-5.0	Vdc
Collector Current – Continuous	I_C	-100	mA dc
Total Device Dissipation @ $T_A = 25^\circ\text{C}$ Derate above 25°C	P_D	350 2.8	mW mW/ $^\circ\text{C}$
Total Device Dissipation @ $T_C = 25^\circ\text{C}$ Derate above 25°C	P_D	1.0 8.0	Watts mW/ $^\circ\text{C}$
Operating and Storage Junction Temperature Range	T_J, T_{stg}	-55 to +150	$^\circ\text{C}$

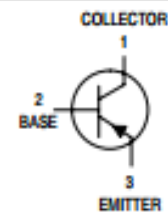
THERMAL CHARACTERISTICS

Characteristic	Symbol	Max	Unit
Thermal Resistance, Junction to Ambient	$R_{\theta JA}$	357	$^\circ\text{C/W}$
Thermal Resistance, Junction to Case	$R_{\theta JC}$	125	$^\circ\text{C/W}$



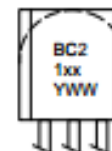
ON Semiconductor™

<http://onsemi.com>



TO-92
CASE 29
STYLE 17

MARKING DIAGRAMS



BC21xx = Specific Device Code
xx = 2, 2B, 3 or 4
Y = Year
WW = Work Week

ORDERING INFORMATION

Device	Package	Shipping
BC212	TO-92	5000 Units/Box
BC212B	TO-92	5000 Units/Box
BC212BRL1	TO-92	2000/Tape & Reel
BC212BZL1	TO-92	2000/Ammo Pack
BC213	TO-92	5000 Units/Box
BC214	TO-92	5000 Units/Box
BC214RL1	TO-92	2000/Tape & Reel

Electrical characteristics

Characteristic		Symbol	Min	Typ	Max	Unit
Collector-Emitter Breakdown Voltage ($I_C = -2.0 \text{ mA}$, $I_B = 0$)	BC212	$V_{(BR)CEO}$	-50	-	-	Vdc
	BC213		-30	-	-	
	BC214		-30	-	-	
Collector-Base Breakdown Voltage ($I_C = -10 \mu\text{A}$, $I_E = 0$)	BC212	$V_{(BR)CBO}$	-60	-	-	Vdc
	BC213		-45	-	-	
	BC214		-45	-	-	
Emitter-Base Breakdown Voltage ($I_E = -10 \mu\text{A}$, $I_C = 0$)	BC212	$V_{(BR)EBO}$	-5	-	-	Vdc
	BC213		-5	-	-	
	BC214		-5	-	-	
Collector-Emitter Leakage Current ($V_{CE} = -30 \text{ V}$)	BC212	I_{CBO}	-	-	-15	nAdc
	BC213		-	-	-15	
	BC214		-	-	-15	
Emitter-Base Leakage Current ($V_{EB} = -4.0 \text{ V}$, $I_C = 0$)	BC212	I_{EBO}	-	-	-15	nAdc
	BC213		-	-	-15	
	BC214		-	-	-15	

ON CHARACTERISTICS

DC Current Gain ($I_C = -10 \mu\text{A}$, $V_{CE} = -5.0 \text{ Vdc}$)	BC212	h_{FE}	40	-	-	-
	BC213		40	-	-	
	BC214		100	-	-	
($I_C = -2.0 \text{ mA}$, $V_{CE} = -5.0 \text{ Vdc}$)	BC212		60	-	-	
	BC213		80	-	-	
	BC214		140	-	600	
($I_C = -100 \text{ mA}$, $V_{CE} = -5.0 \text{ Vdc}$) (Note 1.)	BC212, BC214		-	120	-	
	BC213		-	140	-	
Collector-Emitter Saturation Voltage ($I_C = -10 \text{ mA}$, $I_B = -0.5 \text{ mA}$) ($I_C = -100 \text{ mA}$, $I_B = -5.0 \text{ mA}$) (Note 1.)		$V_{CE(sat)}$	-	-0.10 -0.25	- -0.6	Vdc
Base-Emitter Saturation Voltage ($I_C = -100 \text{ mA}$, $I_B = -5.0 \text{ mA}$)		$V_{BE(sat)}$	-	-1.0	-1.4	Vdc
Base-Emitter On Voltage ($I_C = -2.0 \text{ mA}$, $V_{CE} = -5.0 \text{ Vdc}$)		$V_{BE(on)}$	-0.6	-0.62	-0.72	Vdc

DYNAMIC CHARACTERISTICS

Current-Gain – Bandwidth Product ($I_C = -10 \text{ mA}$, $V_{CE} = -5.0 \text{ Vdc}$, $f = 100 \text{ MHz}$)	BC212	f_T	-	280	-	MHz
	BC214		-	320	-	
	BC213		-	360	-	
Common-Base Output Capacitance ($V_{CB} = -10 \text{ Vdc}$, $I_C = 0$, $f = 1.0 \text{ MHz}$)		C_{ob}	-	-	6.0	pF
Noise Figure ($I_C = -0.2 \text{ mA}$, $V_{CE} = -5.0 \text{ Vdc}$, $R_S = 2.0 \text{ k}\Omega$, $f = 1.0 \text{ kHz}$) ($I_C = -0.2 \text{ mA}$, $V_{CE} = -5.0 \text{ Vdc}$, $R_S = 2.0 \text{ k}\Omega$, $f = 1.0 \text{ kHz}$, $f = 200 \text{ Hz}$)	BC214	NF	-	-	2	dB
	BC212, BC213		-	-	10	
Small-Signal Current Gain ($I_C = -2.0 \text{ mA}$, $V_{CE} = -5.0 \text{ Vdc}$, $f = 1.0 \text{ kHz}$)	BC212	h_{fe}	60	-	-	-
	BC213		80	-	-	
	BC214		140	-	-	
	BC212B		200	-	400	

1. Pulse Test: T_p 300 s, Duty Cycle 2.0%.

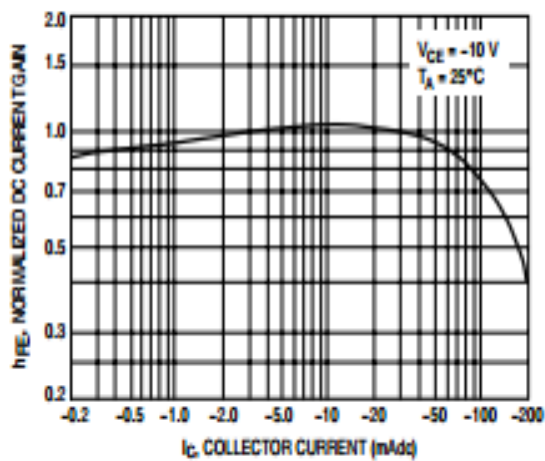


Figure 1. Normalized DC Current Gain

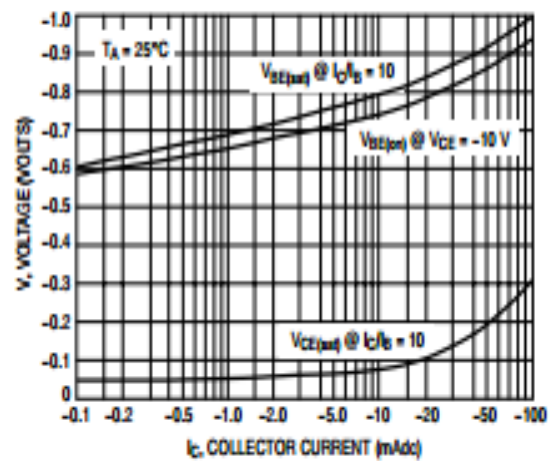


Figure 2. "Saturation" and "On" Voltages

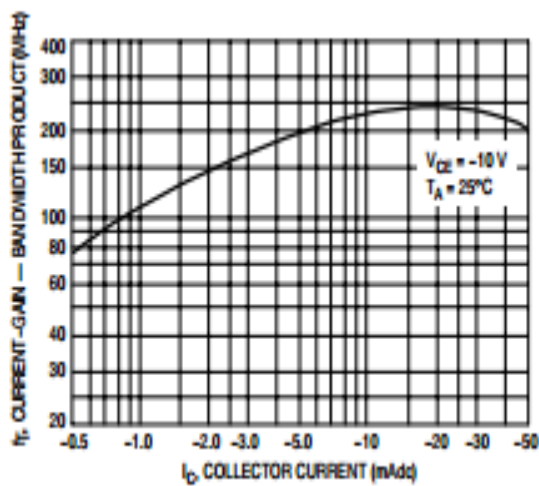


Figure 3. Current-Gain — Bandwidth Product

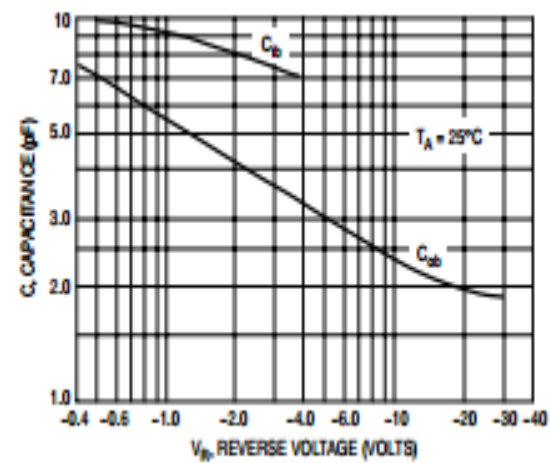


Figure 4. Capacitances

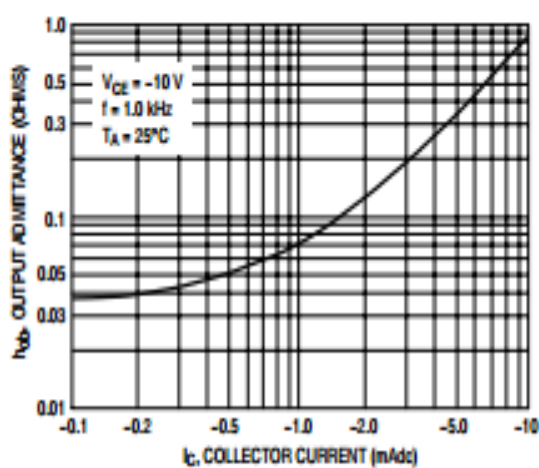


Figure 5. Output Admittance

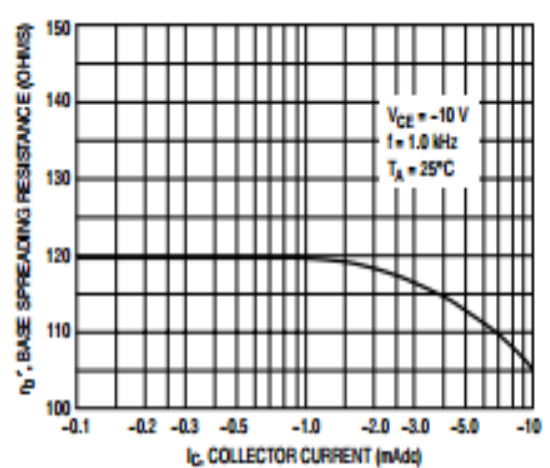


Figure 6. Base Spreading Resistance

