UNIVERSITI MALAYSIA PAHANG

BORANG PENGESAHAN STATUS TESIS

JUDUL: DEVELOPMENT CONTROLLER FOR 2 AXIS MECHANISM MACHINE

SESI PENGAJIAN: 2011/2012

Saya, **MUHAMMAD HAYYUL BIN SOHAIMI (881205-07-5157)**
(HURUF BESAR)

mengaku membenarkan tesis Projek Tahun Akhir ini disimpan di perpustakaan dengan syarat-syarat kegunaan seperti berikut:

1. Tesis ini adalah hakmilik Universiti Malaysia Pahang (UMP).
2. Perpustakaan dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. **Sila tandakan (√)

	SULIT	(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)
	TERHAD	(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi / badan di mana penyelidikan dijalankan)
√	**TIDAK TERHAD**	

Disahkan oleh:

_____ _____

(TANDATANGAN PENULIS) (TANDATANGAN PENYELIA)

Alamat Tetap:

No 2, Jalan 14/1D ,
Taman Cheras Jaya,
43200 Balakong Selangor
Tarikh:

KHAIRUL FIKRI BIN MUHAMMAD
(NamaPenyelia)

Tarikh:

CATATAN: * Potong yang tidak berkenaan.

** Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali tempoh tesis ini perlu dikelaskan sebagai SULIT atau TERHAD.

DEVELOPMENT CONTROLLER FOR 2 AXIS MECHANISM MACHINE

MUHAMMAD HAYYUL BIN SOHAIMI

A report submitted in partial fulfilment of the requirements

for the award of the degree of

Bachelor of Manufacturing Engineering

Faculty of Manufacturing Engineering

UNIVERSITY MALAYSIA PAHANG

JUNE 2012

SUPERVISOR'S DECLARATION

We hereby declare that we have checked this project and in our opinion this project is satisfactory in terms of scope and quality for the award of the degree of Bachelor of Manufacturing Engineering.

Name of Supervisor:
Position:
Date:

Name of Panel:
Position:
Date:

STUDENT'S DECLARATION

I hereby declare that the work in this thesis is my own except for quotations and Summaries which have been duly acknowledged. The thesis has not been accepted for any degree and is not concurrently submitted for award of other degree.

Signature:
Name: MUHAMMAD HAYYUL BIN SOHAIMI
ID Number: FA09071
Date:

DEDICATION

To my parents Sohaimi Pawanchi and Siti Zaharah Zakaria

and those made it possible

ACKNOWLEDGEMENT

In the name of Allah, the Most Gracious and the Most Merciful Alhamdulillah, all praises to Allah for the strengths and His blessing in completing this thesis First and foremost I offer my sincerest gratitude to my supervisor, En. Khairul Fikri, who has supported me throughout my thesis with his patience and knowledge whilst allowing me the room to work in my own way. I attribute the level of my degree to his encouragement and effort and without him this thesis, too, would not have been completed or written. One simply could not wish for a better or friendlier supervisor.

In the various laboratories and workshops I have been given instruction by En.Ismayuzi, a fine technical engineer who kept me through this project. The smooth running of the Bunggard Machine is much more a testament to his efforts than my own. Mr. Mohsein, undergraduate student who that give an idea and teach me how to use visual basic programming without him there maybe have a problem during to program the motor.

Sincere thanks to all my friends especially for first batch Manufacturing Engineering for their kindness and moral support during my study. Thanks for the friendship and memories.

Last but not least, my deepest gratitude goes to my beloved parents; Mr. Sohaimi Pawanchi and My mother Siti Zaharah Binti Zakaria that give me support during my three years journey study at UMP and also to my brothers Quyyum, Soffian and my younger's sister Farhana Najwa for their endless love, prayers and encouragement.

ABSTRACT

In engineering application most of part is machining to make it realize, an example plastic product. Moulds are machine by using Milling machine, turning machine and EDM machine. This entire machine is using a same concept that was 2 axis mechanisms to move the table. This thesis is focus on controlling stepper motor in axis during machining process in term of positioning motion control. All of the task would be control by using Visual Basic Programming. This project is limited to low cost automation and low servo speed. The device or 3 axis machine has been develop before this, this machine are using stepper motor to move the table, this is because step motor provides efficiently and precise movement that required during application in machining. During this development of controller there are no manufacturer product catalogues, outlining characteristic and rating of actuators that can helps engineer. However, because of lack information, engineers resolve to choose based an experience. From this project, the criteria in the selection aid focus in type of nature of control requirement by the application such as position control, speed control and torque control requirement.

ABSTRAK

Dalam aplikasi kejuruteraan sebahagian besar daripada bahagian mesin akan dimesin untuk merealisasikannya , produk plastik misalnya. Acuan akan dimesin dengan menggunakan mesin raut, mesin larik dan mesin EDM. Keseluruhan mesin ini menggunakan konsep yang sama iaitu mekanisme 3 paksi untuk menggerakkan meja kerja. Tesis ini memfokuskan kepada pengendalian motor semasa proses memesin ,didalam maksud posisi kawalan pergerakkan. Semua kawalan ini akan menggunakan Visual Basic program. Projek ini terbatas untuk automasi kos murah dan kecepatan servo rendah. Mesin 3 paksi telah dibangunkan sebelum ini, mesin ini menggunakan stepper motor untuk menggerakkan meja kerja, ini kerana motor ini memberikan gerakan yang cekap dan tepat yang diperlukan semasa proses memesin dilakukan . Semasa pembangunan kawalan untuk motor stepper ini, tiada katalog disediakan oleh pembuat produk yang dapat menghuraikan ciri-ciri dan penarafan penggerak yang dapat membantu jurutera. Namun, kerana kekurangan maklumat, jurutera memutuskan untuk memilih berdasarkan pengalaman. Daripada projek ini, ciri-ciri pemilihan tumpuan bantuan jenis sifat keperluan kawalan oleh permohonan seperti kawalan kedudukan dan kawalan kelajuan.

TABLE OF CONTENT

LIST OF FIGURE'S

LIST OF TABLE'S

LIST OF SYMBOLS

F	Force
Ω	Ohm
Lw	Inductor
Rw	Resistor
V	Volt
A	Ampere
θ	Rotation Angle
θs	Step Angle
A	Pulse Number
N	Speed of The Motor
f	Pulse speed in Hz

LIST OF ABBREVIATIONS

PTP	Point to Point
DC	Direct Current
AC	Accumulative Current
PLC	Programmable Logic Controller
CNC	Computer Numerical Control
RPM	Rotation per Minute
CW	Clock Wise
PCB	Printed Circuit Board
EDM	Electrical Discharge Machine

CHAPTER 1

INTRODUCTION

1.1 Project Background.

Nowadays lots of machine has been produce to use for making parts such as in automotive industries, food industries, plastic industries and etc [4]. Behind these industries all machine has been produce by using turning machine and milling machine. These two machines are a basic machine that use in all machining processes to produce parts machines and make the engineering process easier and efficiencies. [2]

Material removing process can be divided into mainly two groups which are conventional and non conventional machine. Conventional machining processes mostly remove material in the form of chips by applying forces on the work material using a wedge shaped cutting tool that is harder than the work material under machining condition, example milling, turning, grinding etc. Non conventional machining usually known as advance machining. This type of machining the tool does not to be harder than the work piece material. For example, in EDM, copper is used as the tool material to machine hardened steels, non conventional machining also may not represent physical tool example in laser jet machining, machining is carried out by laser beam. This non-conventional machine mostly integrated with Computer Numerical Control (CNC) [7].

The advantages using CNC in machining process is its easier to control, precise machining and complex parts can be machine using three axis platform. Three axis machines is a basic machine in manufacturing in producing small part to the large size of any part in machine, it starts to increase the automation Industries in manufacturing field. In addition, three axis machines are more flexibility in time to produce different components.

In this project, control system for two axis mechanism machine will be developed by using Visual Basic as software, this control system will be used to control the movement of this mechanism, control system is a device or set of the device to control, manage, command, direct or regulated behavior of other device or system. [3]

1.2 Project objectives

1. To design a control system for two axis mechanism machine.
2. To test and analyze the parameters and characteristics of tracking control of feed drive in machine.

1.3 Scope of project

1. Design a control system for two axis mechanism machine using Visual Basic.
2. Testing and programming for machining setup.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction to control system

Control system is a one of discipline in engineering field. “Control system” is a device or set of the device to control, manage, command, direct or regulated behavior of other device or system. A control system consist of subsystem and process (or plants) assembled for the purpose of obtaining a desired output with desired performance, given a specified input. [3]

Modern manufacturing such as machine tool is supported by servo system that provides two fundamental functions that was regulation and maneuvering. The regulation function is required to maintain the controlled object at a desired position in the presences of external distribution in maneuvering control, the motion can be classified in two cases point to point (PTP) and countering (or tracking) when controlled object is moved along a prescribed trajectory. In PTP control the final positioning accuracy and the transition time are important while the transient path is of secondary importance. Positioning a drill bit in machining center is a typical example of PTP controller. In tracking control, the controlled object must be moved along a desired trajectory and the error during transient must also be minimized. Example of tracking control is can be found in milling operation such as machining in circular work-piece. [8]

In this system, a DC motor is used as the power set of the feeding mechanism. Compared with AC motor control, DC motor's drive circuit is more simple and reliable. Moreover, the speed adjusting accuracy and dynamic response characteristics of the DC motor are more ideal with a greater speed range and higher speed adjusting accuracy.

2.2 Components in Two Axis Machine

Basically in two axis machine there don't have more than 10 components in the machine to make a movement. Axis are a direction of a motion control by the servo motor. The number of axis a machine has determine s it's machining capabilities, the number of axis in a certain movement machine is depends on the application example in automotive industries to produce cylinder block for car engine there need more angle for machine this part so maybe they need 5 axis machine to meet the requirement for machining process.

In order to understanding a two axis mechanism machine, there is a need to determine which the components have to utilize so that it can perform smoothly and efficiency. The components in 2 axis machine are:-

i). Linear slides
ii). Ball bearing slides
iii). Dove tail slides
iv). Machine slides
v). Roller slides
vi). Roller tables
vii). XY table

2.2.1 Linear Motion System

Linear motion system is a system that provides movement to the three axis machine, usually linear motion system using for straight movement of slider to move one point to another point by a system such as slider. A modern motion

system typically consist of a motion controller, a motor drive or amplifier, an electrical motor, and feed back sensor. The system might also contain other components such as one or more belt, ballscrew or lead screw that function to drive linear guide or axis stages. [1]

A motion controller today can be stand alone programmable controller, a personal computer containing a motion control card, or a programmable logic controller (PLC). All of the component of a motion control system must work together so that their can performed their assigned functions. All of their selection must be based on both engineering and economic consideration. Figure 2.1 below illustrate linear motion control that have 3 axis machine.

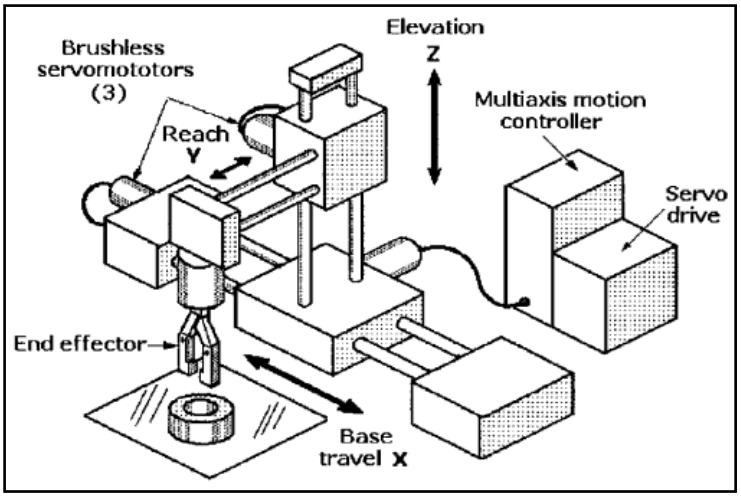


Figure 2.1: Linear Motion in 3 axis machine [1]

With additional mechanical electro mechanical components on each axis, rotation on three axis can provide up to six degree of freedom show in Figure 2.2 below.

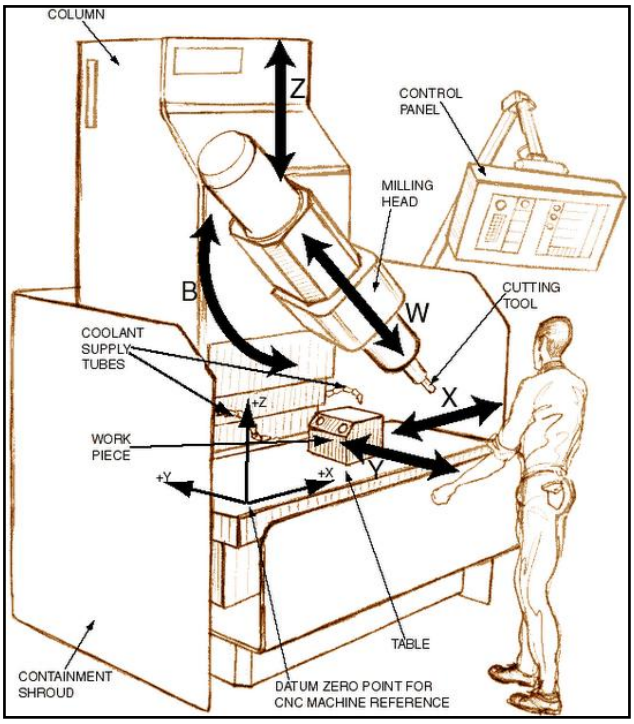


Figure 2.2: 6 axis milling machine

Each of axis that provide in three axis machine have their own function, this axis develop base on the application that need in machining process, table 2.1 show machine tool axis definition.

Table 2.1:ISO Machine tool axis definition

<table>
<tr><td>AXIS</td><td colspan="3">MACHINE TOOL WITH SPINDLE</td><td>MACHINE TOOL WITH NO SPINDLE</td></tr>
<tr><td>Z</td><td colspan="3">axis of spindle,
(+Z) as tool goes away from the work piece</td><td>perpendicular to work holding surface, (+Z) as tool goes away from the workpiece</td></tr>
<tr><td rowspan="3">X</td><td>MACHINE TOOL WITH ROTATING WORKPIECE</td><td colspan="2">MACHINE TOOL WITH ROTATING TOOL</td><td></td></tr>
<tr><td></td><td>HORIZONTAL AXIS</td><td>VERTICAL AXIS</td><td></td></tr>
<tr><td>radial and parallel to cross slide, (+X) when tool goes away from the axis of spindle</td><td>horizontal and parallel to work holding surface, (+X) to the right when viewed from spindle towards work piece</td><td>horizontal and parallel to the work holding surface, (+X) to the right when viewed from spindle towards column</td><td>parallel to and positive in the principal direction of cutting (primary motion)</td></tr>
<tr><td>Y</td><td colspan="4">apply right hand rules</td></tr>
</table>

2.2.2 Linear slides

linear slides are designed for robust applications that demand high thrust along with high-precision accuracy and stiffness. This configuration of drive and guide is just one of several pre-assembled, ready-to-install linear slides available to mechanical motion engineers. All provide low friction and smooth, accurate motion for a wide range of moment or normal loading configurations.

Application of the linear slides is to move mounted mechanisms across a given axis either in one direction or combine of three or more directions. Complete linear slides normally consist of at least a base, a saddle, adjusting screws and a straight rib. Linear slides are resistant to contamination, extremely durable in shock load conditions and run smoothly on lightweight frames .Figure 2.3 below shows the example of linear slide that usually have in market.

Figure 2.3 : Linear slide [12]

There are some advantages of using linear slide. Some of them are: -

i. Products that having a wide range of weights, from lightweight miniatures to payloads of several hundred pounds can be move easily.
ii. Products can be move in distances that range from as little as 2.5 millimetres to 1.5 meters.
iii. Rapidly position their loads.
iv. They position their loads so precisely, that the final positioning can be measured in microns (millionths of a meter).

2.2.3 Ball bearing slide

Also known as "ball slides", ball bearing slides are the most common type of linear slide. Ball bearing slides offer smooth precision motion along a single-axis linear design, aided by ball bearings housed in the linear base, with self-lubrication properties that increase reliability. Ball bearing slide applications include delicate instrumentation, robotic assembly, cabinetry, high-end appliances and clean room environments, which primarily serve the manufacturing industry but also the furniture, electronics and construction industries. For example, a widely used ball bearing slide in the furniture industry is a ball bearing drawer slide.

Commonly constructed from materials such as aluminium, hardened cold rolled steel and galvanized steel, ball bearing slides consist of two linear rows of ball bearings contained by four rods and located on differing sides of the base, which support the carriage for smooth linear movement along the ball bearings. This low-friction linear movement can be powered by either a drive mechanism, inertia or by hand. Ball bearing slides tend to have a lower load capacity for their size compared to other linear slides because the balls are less resistant to wear and abrasions. In addition, ball bearing slides are limited by the need to fit into housing or drive systems. Ball bearing slides mostly use for delicate instrumentation, the structure of ball screw is shown in figure 2.4 next page.

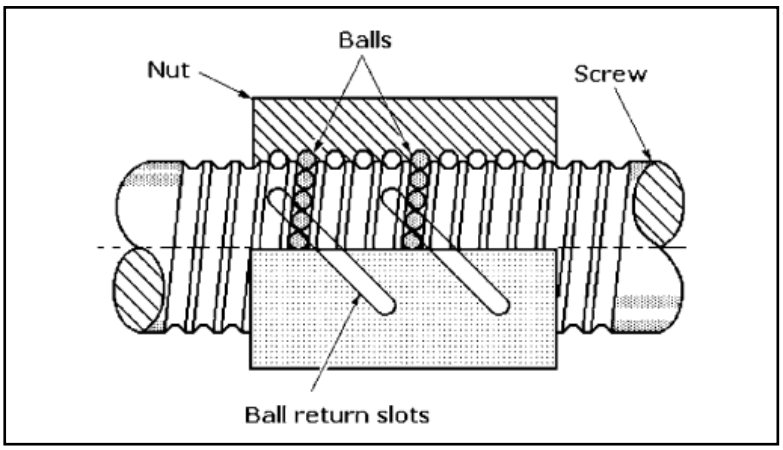


Figure 2.4 :Ball Screw drive: ballscrew use recirculating balls to reduce friction and gain higher efficiency than conventional lead screws. [1]

2.2.4 Dovetail Slide

Dovetail slides or dovetail was a slide are typically made from cast iron, but can also be constructed from hard-coat aluminium, acetyl or stainless steel. Like any bearing, a dovetail slide is composed of a stationary linear base and a moving carriage. A Dovetail carriage has a v-shaped, or dovetail-shaped protruding channel which locks into the linear base's correspondingly shaped groove. Once the dovetail carriage is fitted into its base's channel, the carriage is locked into the channel's linear axis and allows free linear movement. When a platform is attached to the carriage of a dovetail slide, a dovetail table is created, offering extended load carrying capabilities. Figure 2.5 below illustrate traditional dovetail.

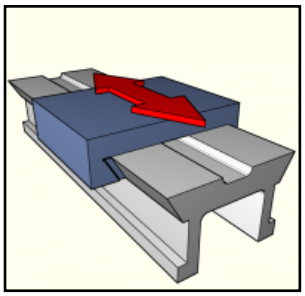


Figure 2.5: Traditional dovetail slide

Since dovetail slides have such a large surface contact area, a greater force is required to move the saddle than other linear slides, which results in slower acceleration rates. Additionally, dovetail slides have difficulties with high-friction but are advantageous when it comes to load capacity, affordability and durability. Capable of long travel, dovetail slides are more resistant to shock than other bearings, and they are mostly immune to chemical, dust and dirt contamination.

Dovetail slides can be motorized, mechanical or electromechanical. Electric dovetail slides are driven by a number of different devices, such as ball screws, belts and cables, which are powered by functional motors such as stepper motors, linear motors and handwheels. Dovetail slides are direct contact systems, making them fitting for heavy load applications including CNC machines, shuttle devices, special machines and work holding devices. Mainly used in the manufacturing and laboratory science industries, dovetail slides are not ideal for high-precision applications.

2.2.5 Machine Slide

Machine slide mostly use in CNC machine for linear movement, to move the machine slide several part include of AC/DC motor, ball bearing slide, and roller slide need for that function. By using machine slide higher rigidity of the machine can be retained, which create accurate linear motion for the machine for all application include roughing and drilling in CNC milling machine. For machine slides it’s have adjustable ribs in order to make up for any irregular movement that maybe develop during application.

Machine slide can be single, double or multi axis, in this project three axis will be use. Some of standard machine slide include dovetail slide, hardened way slide, linear guide slide and more. Figure 2.6 show in the next page is machine slide for CNC milling.

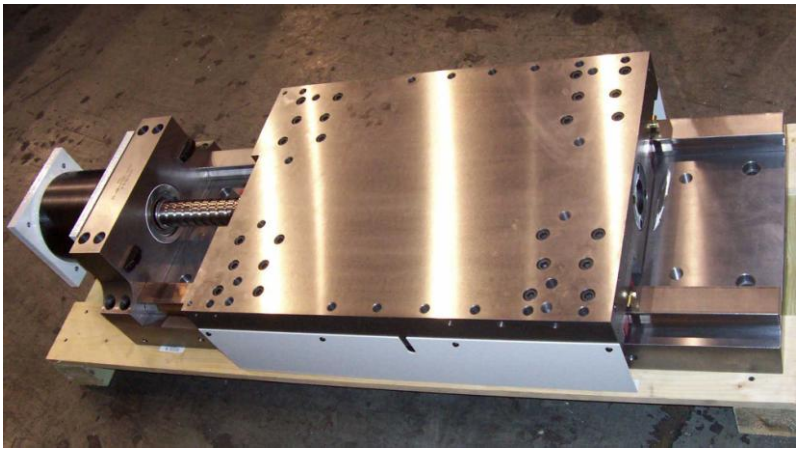


Figure 2.6: Machine slide for CNC milling [13]

2.2.6 Roller slide

Roller slide that also known as crossed roller slides, this equipment are non-motorized linear slides that provide low-friction linear movement for equipment powered by inertia or by hand. Roller slides are based on linear roller bearings, which are frequently criss-crossed to provide heavier load capabilities and better movement control. The example of roller slide is shown in figure 2.7 below.

Figure 2.7: Roller slide made from stainless steel [13]

Its utilize rollers that crisscross each other at a 90° angle and move between the four semi-flat and parallel rods that surround the rollers. The rollers are between "V" grooved bearing races, one being on the top carriage and the other on the base. The design of' crossed roller slides allows them to carry up to

twice the load of ball bearing slides and to absorb larger impacts or stackable to create multi-axis linear motion.

Serving industries such as manufacturing, photonics, medical and telecommunications, roller slides are versatile and can be adjusted to meet numerous applications which typically include clean rooms, vacuum environments, material handling and automation machinery.

2.2.7 Roller Tables

The other part in three axis mechanism is roller table. It functions to front sliding surface and rear sliding surface that longitudinally aligned. For secure to rear supporting group, lifting levers are pivoted on a bearing bar. The levers have feeler pins engaged in sliding manners along guiding grooves, which are shaped so that when the front and rear supporting groups are moved away from each other.

2.2.8 XY table

There are various type of XY table being used in machine technology. The apparatus is applied in positioning element mechanism of lathe, milling and other machine. The most was applied in positioning element mechanism of lathe, milling and other machine. The most was applied in Computer Numerical Controller (CNC) machining centre. The XY table is use for moves to work of marking, cutting, drilling and others. The name of XY table is because of the prime activity X and Y axis. Then, there is also Z axis which is for the vertical axis. Figure 2.8 show the example of XY table machine.

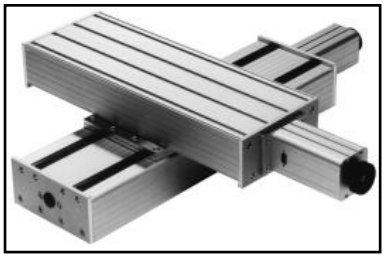


Figure 2.8: XY table machine [14]

Variations among XY tables include the ways and drive mechanism. While the drive mechanism determine smoothness and speed, the ways determine load capacity, straight line accuracy, repeatability, and resolution required, as well as the appropriate motor for the application and whether or not an encoder is need.

XY table usually mounted horizontally to the machine, usually XY table use in manufacturing application example in machining, assembly, laser cutting, water jet and also in automation in factory.

2.3 Introduction to the Motor.

Motor is one of mechanical part use to give a power to make a movement in three axis machine. It is the component that delivers power from electrical, hydraulic, or pneumatic source convert to mechanical application. Most of motor operate through the interaction of magnetic fields and current carrying conductors to generate force. The input to the motor is in the form of the voltage and current, and the output is mechanical torque and speed. The key physical phenomenon in this process is different for various motor. [5]

In motor application there have two main types of motor. Usually are used direct current motor (DC) and alternative current (AC). Each of this motor have rotor on a shaft, stator (stationary component), housing and bearing to support rotor in the housing with allowance for some axial play between shaft and the housing. Figure 2.9 below show the typical motor control function.

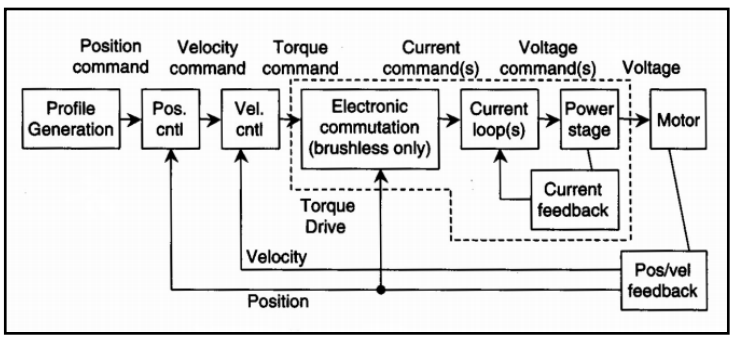


Figure 2.9: Typical motor control function. [1]

2.3.1 DC motor

DC motor now widely use in engineering application that was need a accurate speed control, example in this project movement of table XY need to measure so it can be precise during application. In DC motor, electric current passes through a coil in a magnetic field, the magnetic force produces a torque which turns the DC motor it can be seen in figure 2.10 below.

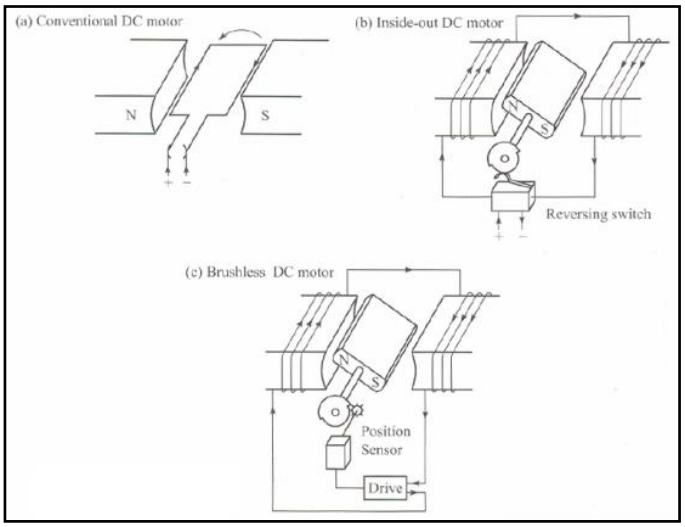


Figure 2.10: (a) Conventional DC motor, (b) Inside-out DC motor, (c) Brushless DC motor [5]

DC motor are divided to two typeswhich are brush and brushless. In the brushed DC motor it generates torque directly from DC power supplied to the motor by using internal commutation, stationary magnets and rotating electrical magnets. Like all electric motors or generators, torque is produce by principle of Lorentz force, which state that any current carrying conductor placed within an external magnetic field experiences a torque of force known as Lorentz force. Advantage of this type DC motor is high reliability, initial cost and simple control of motor speed.

In brushless DC motor, it use a rotating permanent magnet or soft magnetic core in the rotor, and stationary electrical magnets on the motor housing as shown in figure 2.10 a. A motor controller converts DC to AC. The advantage of brushless is long life span, high efficiency and little or no maintenance.

2.3.2 AC motor

AC motor also known as alternating/alternative current is one type one electric motor typically use for application that need accurate and precise function. There are two main types of AC motors, depending on the type of rotor used. The first type is the induction motors, which run slightly slower than the supply frequency. The magnetic field on the rotor of this motor is created by an induced current. The second type is synchronous motor, which does not rely on induction and as a result, can rotate exactly at the supply frequency or a sub-multiple of the supply frequency. The magnetic field on the rotor is either generated by current delivered through slip rings or by a permanent magnet.

For three phase, which also called polyphase are usually found in industrial setting. The three phase AC induction motor has a squirrel cage rotor in which aluminium conductors or bars are shorted together at both ends of the rotor by cast aluminium end rings. When three currents flow through the three symmetrically placed windings, a sinusoidally distributed air gap flux generating the rotor current is produced. The interaction of the sinusoidally distributed air gap flux and induced rotor currents produces a torque on the rotor. Figure 2.11 on the next page shows the working principle of AC motor.

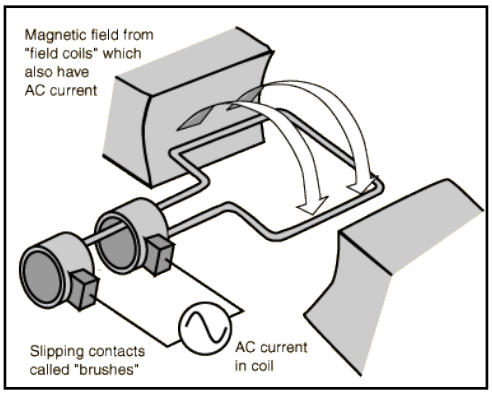


Figure 2.11: Motion for AC motor [5]

2.3.3 Linear Motor

From the linear system, the mechanical energy that provide from the motor can take the form of rotary-to-linear or direct-to-linear motion. Most of rotary motor being use in linear motion is the stepper motor and servo motor

.

Stepper motor work on basically the same principle with brushless DC motors, except that the stator winding distribution is different. A given stator excitation state defines a stable rotor position as a result of the attraction between electromagnetic poles of the stator and permanent magnets of the rotor. The rotor moves to minimize the magnetic reluctance. At a stable rotor position of a step motor, two magnetic fields are parallel. It can be incorporated into open loop (without position feedback) or closed loop system.

A servomotor is an electromechanical device in which an electrical input determines the position of the armature of a motor; Servos are used extensively in robotics and radio-controlled cars, airplanes, and boats. The key characteristic of a servomotor is the ability to provide precise torque control. Ideally, the output torque of a servo system should be highly responsive and independent of motor position and of speed across the systems entire operating speed range. Servomotor generally has greater torque capabilities than stepper motor and run up to 7500 RPM. [5]

By applying closed system to servomotor, positioning information can be obtained from the output of the motor to provide feedback for the control

system, it also have obvious advantage of greater accuracy than open-loop system. They are less sensitive to noise, disturbances, and changes in the environment [3]. From figure 2.12 below show the example of closed system for CNC servomotor closed system.

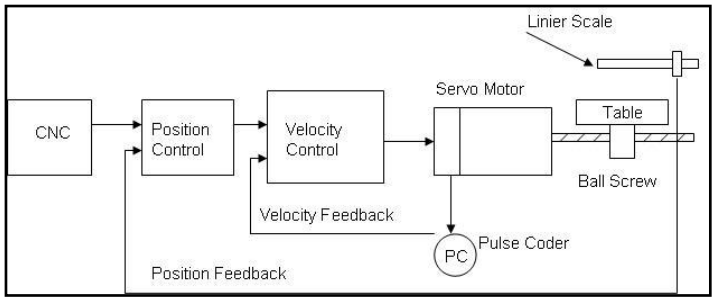


Figure 2.12: Closed system for CNC servomotor[1]

2.3.4 AC servo motor

These types of servo motors are basically have two phase, reversible, induction motors modified for servo application. Ac servo motors are used in operation requiring rapid and accurate response characteristics. To achieve these characteristics, these ac servo motors have small diameter, high resistance rotors. The ac servo motor's small diameter provides low inertia for fast starts, stops, and reversals. High resistance provides nearly linear speed-torque characteristics for accurate servo motor control.

2.3.5 DC servo motor

DC servo motors are normally used as prime movers in application such as computers, numerically controlled machinery, or other applications where starts and made quickly and accurately. Because of servomotors have lightweight, low inertia armatures so it can respond quickly to excitation-voltage changes. In addition, servomotors have a low electrical time constant that further sharpens servomotor response to command signals. The servo motor features a field, which is provided by cast Alnico magnets whose magnetic axis is radial.

There are some characteristics of DC servo motor there are inertia, physical shape, costs, shaft resonance, shaft configuration, speed, and weight. Although these DC motors have similar torque ratings, their physical and electrical constants vary

2.3.6 Stepper Motor

A stepper motor and also known as step motor is a brushless, it mean that stepper motor have some type of rotor that position sensor for electronic commutation of the current. This motor can be position without any feed back by using open loop controller. This motor can be used in various areas of any microcontroller project such as making robots, robotic arm, and automatic door lock system.

Figure 2.13 below illustrates one complete rotation of a stepper motor. At position 1, the rotor is beginning at the upper electromagnet, which is currently active (has voltage applied to it). To move clockwise (CW), the upper electromagnet is deactivated and the right electromagnet is active, causing the rotor to move 90 degrees CW, aligning itself with the active magnet. This process is repeated in the same manner at the south and west electromagnets until we once again reach the starting position.

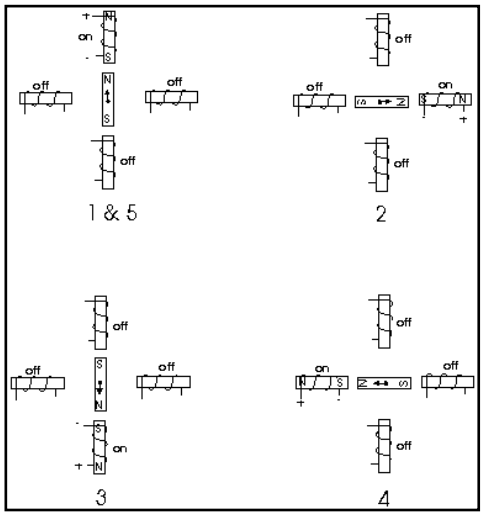


Figure 2.13 : Movement of stepper motor [5]

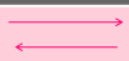
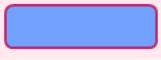
2.4 Introduction to Visual Basic

Programmers have undergone a major change in many years of programming various machines. For example what could be created in minutes with Visual Basic could take days in other languages such: as "C" or "Pascal". Visual Basic provides many interesting sets of tools to aid you in building exciting applications. Visual Basic provides these tools to make your life far easier because all the real hard code is already written for you.

With controls like this, user can create many applications which use certain parts of windows. For example, one of the controls could be a button, which we have demonstrated in the "Hello World" program below. First create the control on the screen, then write the code which would be executed once the control button is pressed. With this sort of operation in mind, simple programs would take very little code. Why do it like the poor old "C" programmer who would have to write code to even display a window on the screen, when Visual Basic already has this part written for user.

Before starting the programming, flow chart should be design in which standard graphical symbols are used to represent the logical flow of data through a function. In graphical symbols, each symbol represent a function for the next step show in table 2.2 below, and from this symbol can develop flow chart that show work flow how the system function from the beginning to the end.

Table 2.2: Flow Chart symbol

Symbol	Name	Meaning
	Flow lines	Direction of process
	Terminal (START/END)	Start /end of a program.
	Input/output	Read data/ print result
	Process	Statements that performs computations or data manipulations.
	Decision/selection	Compare the rules
	Connector	Connecting points in an algorithm

The advantage by using C++ as programming language in this project is it almost as efficient as C, Appropriate for many design methodologies, Large base of programmers, and ANSI/ISO standard.

2.4 Parallel Port

In many applications of control circuits, most of engineers use the interface port of PC to communicate with circuits. Generally more common and ready to use devices are named parallel port. For special application, there are dozens of parallel-port devices for use in data collection, testing and control system. And the parallel port is the interface of choice for many one of a kind and small scale project that require communication between the computer and external device. The Parallel Port allows the input of up to 9 bits or the output of 12 bits at any one given time, thus requiring minimal external circuitry to implement many simpler tasks.

Usually port are founded on the rear of computer and are of following type that was male ports (having pins coming out of port) and female port (having holes for pins), parallel port generally have a 25 pin female connector with which a printer is usually attached (figure 2.14). Parallel port pins are grouping to the three groups:

i). Data Port
ii). Status Port
iii). Control Port

In data port it includes pin 2 to pin 9 with pin names data-0 to data-9, it is usually for data output according to old “standard parallel port” standard. Status port include pin 10 to pin 15, is an input only port, data can’t be output on this port but it can be read. And lastly is control port, it is a read/write port include pin 14, 16, 14 and 1. Figure 2.14 on the next page shows pin configuration of parallel port.

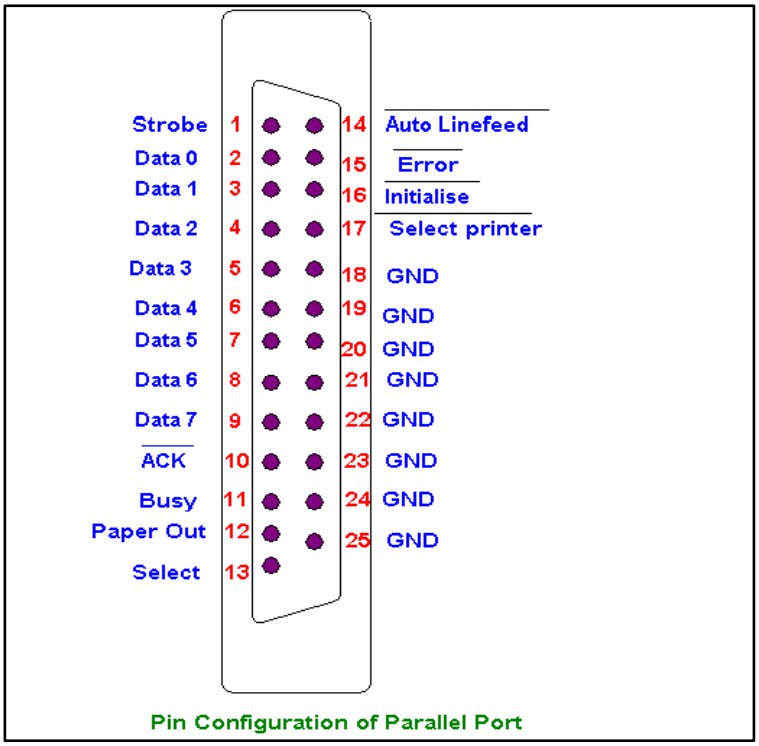


Figure 2.14: Pin Configuration of Parallel port. [6]

2.5.1 Parallel Port Interfacing

An interface is a tool and concept that refers to a point of interaction between components, and is applicable at the level of both hardware and software. This allows a component, whether a piece of hardware such as a graphics card or a piece of software such as an Internet browser, to function independently while using interfaces to communicate with other components via an input/output system and an associated protocol.

Hardware interfaces exist in computing systems between many of the components such as the various buses, storage devices, other I/O devices, etc. A hardware interface is described by the mechanical, electrical and logical signals at the interface and the protocol for sequencing them (sometimes called signalling). A standard interface, such as SCSI, decouples the design and introduction of computing hardware, such as I/O devices, from the design and introduction of other components of a computing system, thereby allowing users

and manufacturer's great flexibility in the implementation of computing systems. Hardware interfaces can be parallel where performance is important or serial where distance is important.

CHAPTER 3

RESEARCH AND METHODOLOGY

3.1 INTRODUCTION

In this chapter will discuss how the flow to develop this controller, I have divided to two parts. The first part is about project flow chart.It follows by the second part which flow to develop controller for two axis machine using Visual Basic programming.

The first flow chart is shown in figure 3.1 on the next page. It is about how the process flows of this project start with briefing to the final year project by our dean until the last part, thesis submission that was final report submission.

The research methodology of this study is shown in figure 3.2. Its starts with understand the visual basic program language, this is most important part in the process, because understanding how to use visual basic code can give advantage during “communicate” to the stepper motor so it can perform smoothly and follow my instruction, after thatit will continue with understanding what is parallel port because this is device use to communicate with stepper motor. And after this two process, basic prototype will be develop follow by control stepper motor, analysis distance can be control and finally result from the analysis. Figure 3.3 show the machine 3 axis that will use in this project.

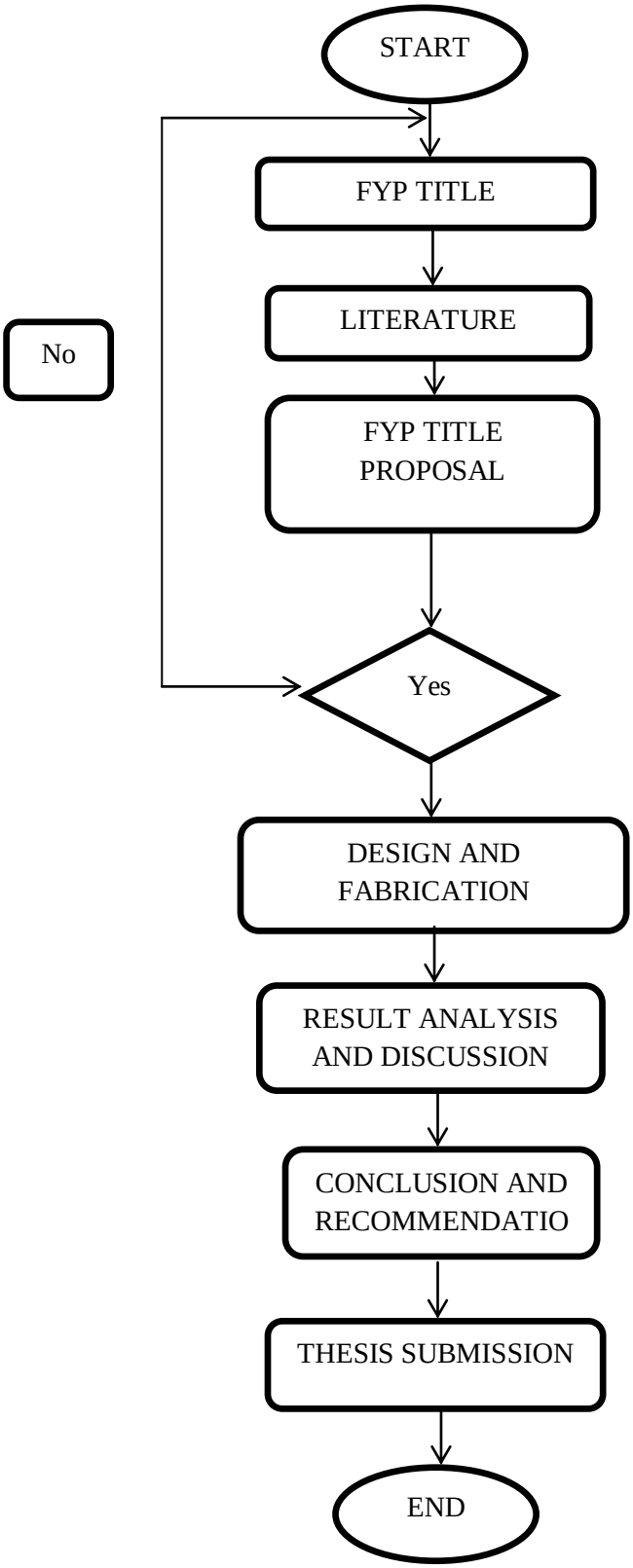


Figure 3.1: Project flow chart

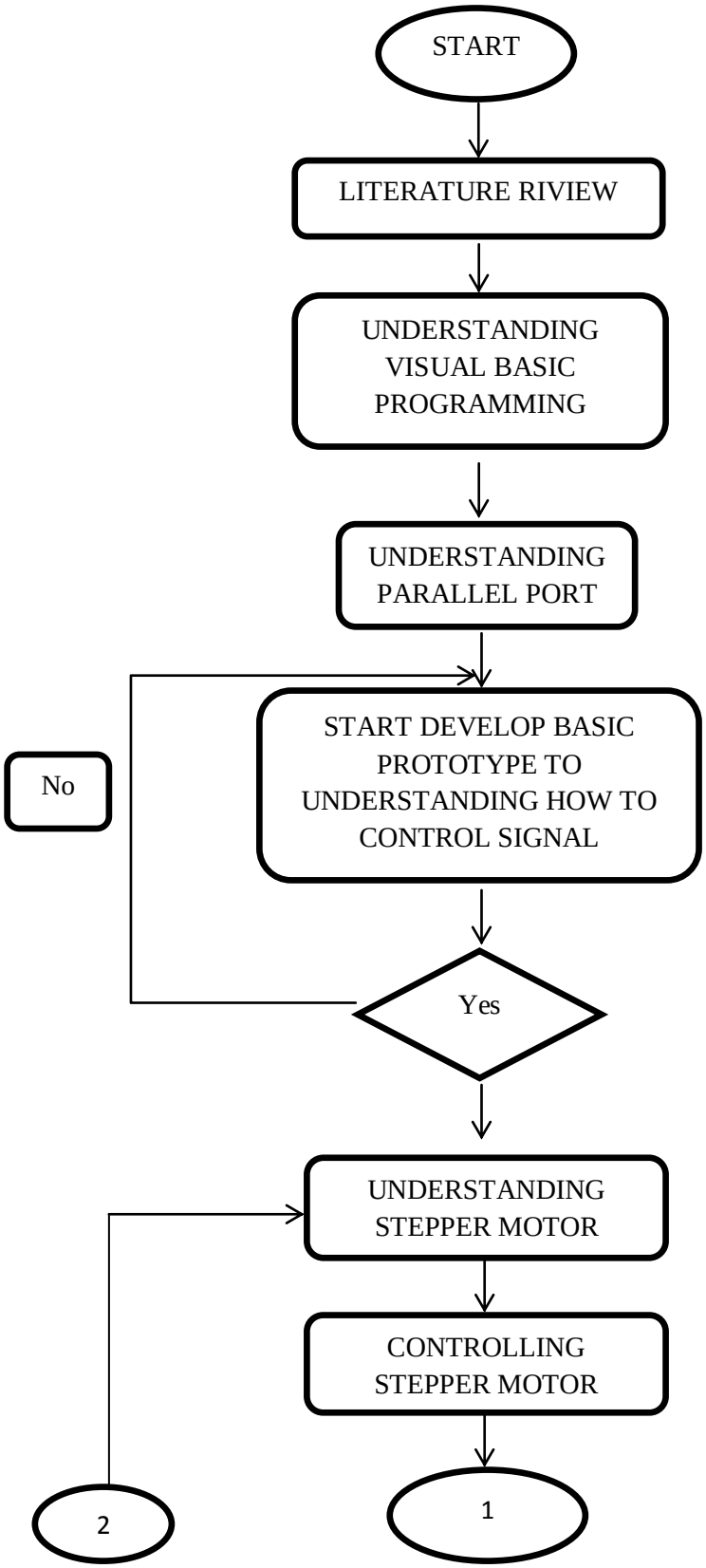


Figure 3.2 (a): Process flow chart for development controller

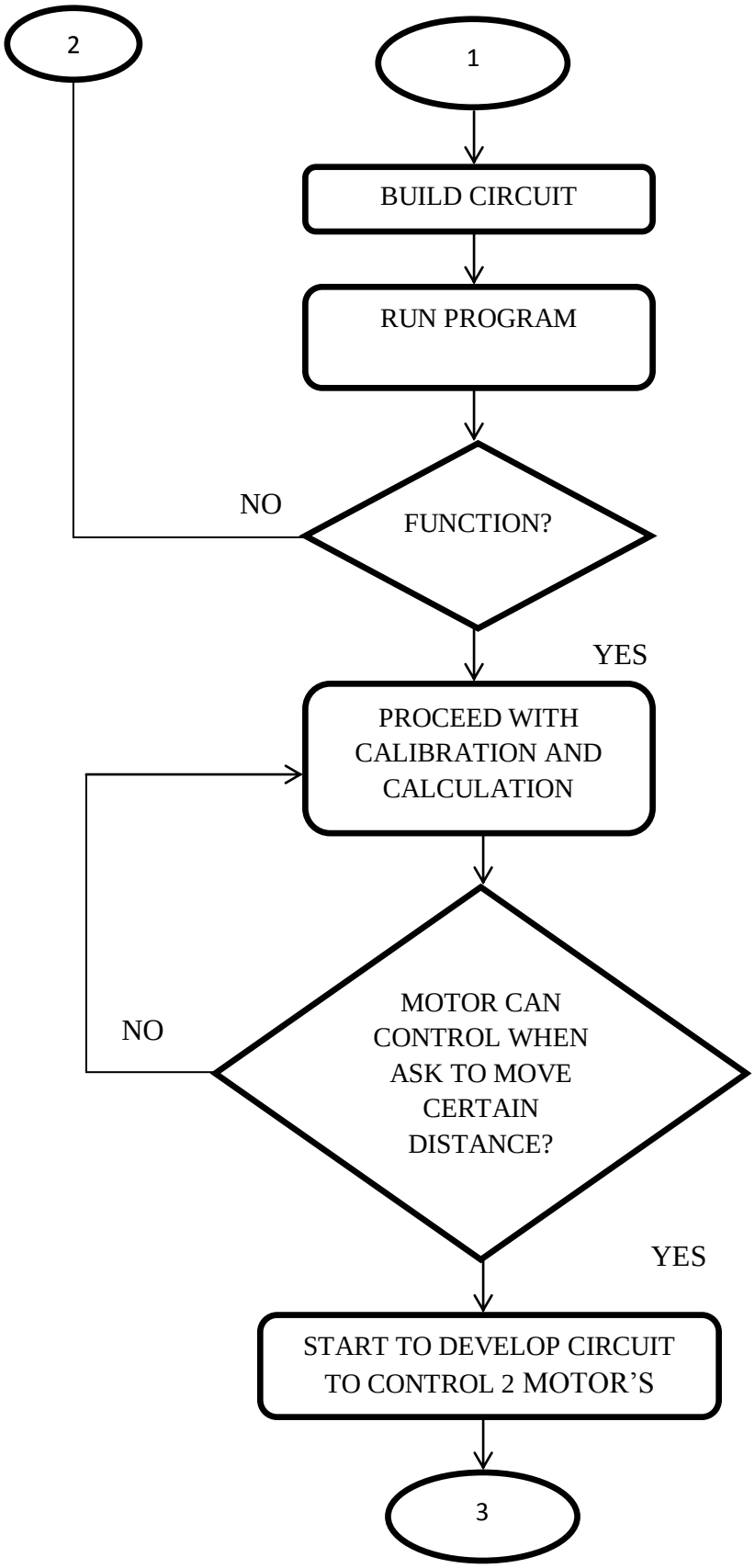


Figure 3.2 (b): Process flow chart for development controller

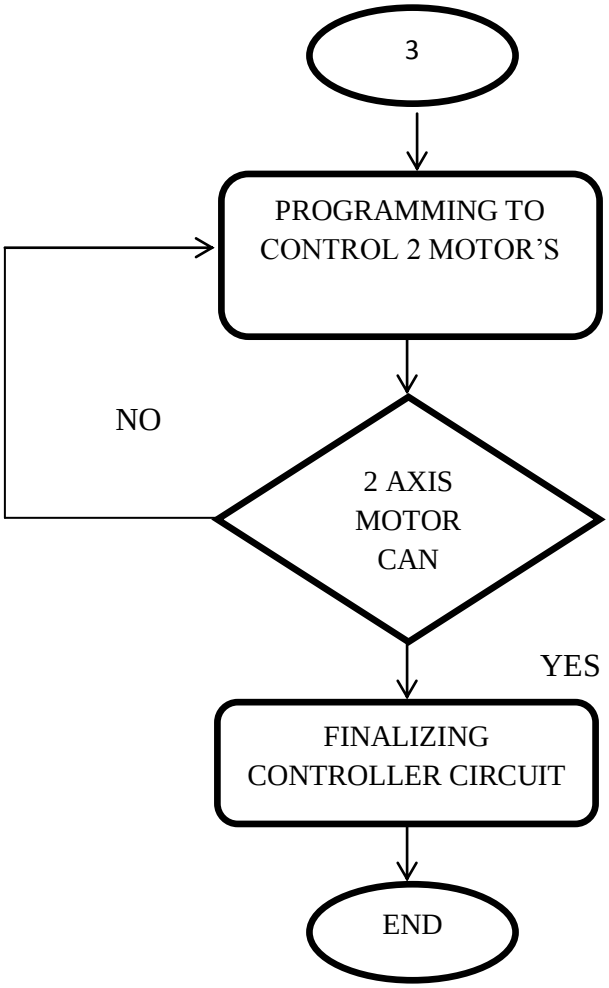


Figure 3.2 (c): Process flow chart for development controller

The real picture of machine used for this project is shown in figure 3.3 below.

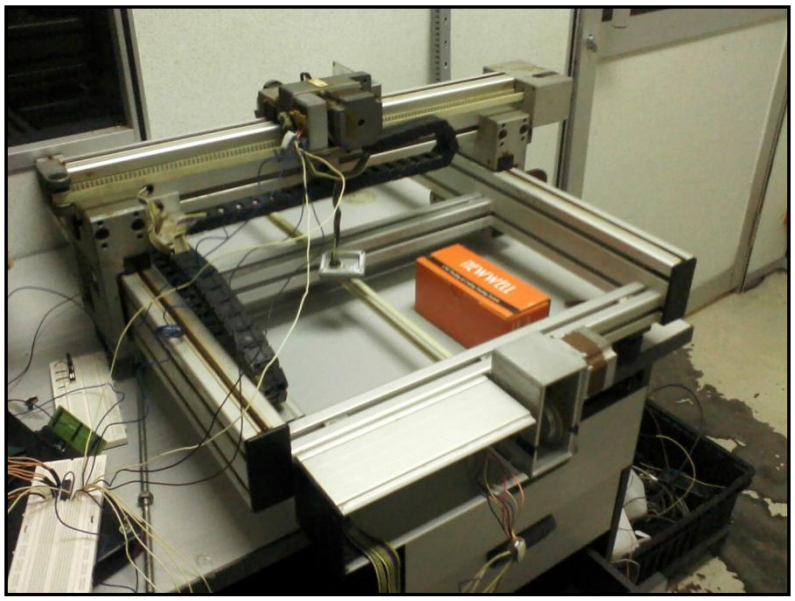


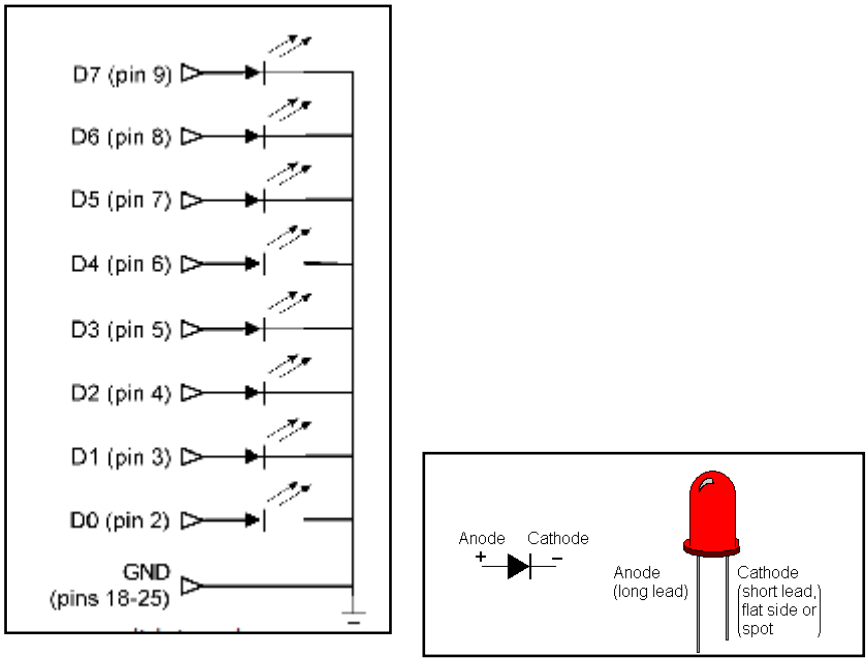
Figure 3.3: 3 Axis machine use for this project

3.2 Building Basic prototype

Before starting to set up stepper motor, basic prototype will be develop in this project. This prototype is developing to understand how to control the motion of stepper by using visual basic as programming language. This prototype function to control LED so it can blink and off followed by the program, it's same when to turn step motor on and off.

A set of eight LEDs, one connected to each data bit from the parallel port, when a data pin is set to "0" it will find 0 V on it. When it set to "1", it will find 5 V on it. This is enough to turn LEDs, but not enough to turn the step motor because step motor need is a heavier devices. All eight LEDs connect to status port (pin no 2- no 9) and get one ground pin (any pin from 19 -25) to connect to the cathode terminal (negative terminal) from all LEDs. From figure 2.15 from previous chapter show schematic for using parallel port.

Since LEDs have polarity, it should pay attention to correctly locate its anode (positive) and cathode (negative) terminals, figure 3.4 (b). As for building circuits, proto board will be use this is because proto board allow to assemble LEDs without needing to solder. After connection LEDs to proto board, printing cable is use to connect proto board to parallel port.



(a) (b)

Figure 3.4:(a) Schematic for using parallel port, (b) Terminals from LEDs [6]

3.3 Circuit Development

During this stage the circuit will be develop to make sure this machine can be functionally. For the first motor is the base circuit to build the circuit for the second motor, it just a same circuit but it control with different pin at the parallel port. For first motor it control by using pin number 2 to pin number 5, and second motor using pin number 6 to pin number 9.

In the program, it has justified each pin to control each motor base on the value at each pin

3.3.1 Wiring Stepper Motor

In this project a 6-wire 4-phase unipolar motor will use, this motor have two “common” wires there was north and south common wire. To determine this two common wire by measure the resistance between the wire using multimeter. Firstly pick any wire randomly, this wire will become the first wire of pair one. Then pick another wire at random and measure resistance, if got a resistance, that’s a pair. In this stepper motor the value of resistance between pair is 1.8Ω. Finally to determine the common wire or power supply wire, pick pair that gives the higher resistance that is the common wire.

Figure 3.6 below show the schematic of 6 wire unipolar stepper motor a-b is a pair and number1 and number 2 is the common wire.

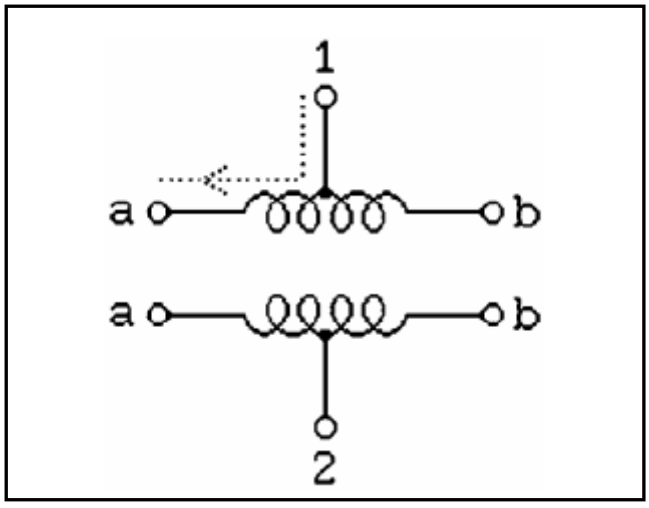


Figure 3.5: 6 wires Unipolar Stepper Motor [10]

3.3.2 Circuit to Control First Motor

First motor is control by using pin no 2 until pin no 5, this pin will give the current pulse to stepper motor obey by the programming. The pulse given to motor base on the type that need by the stepper motor, usually there have full step and half step to move the stepper motor. In this project the programmer will use half step. This is because half step is more accurate during operation compare with full step. Figure 3.5 illustrate the circuit to control first motor.

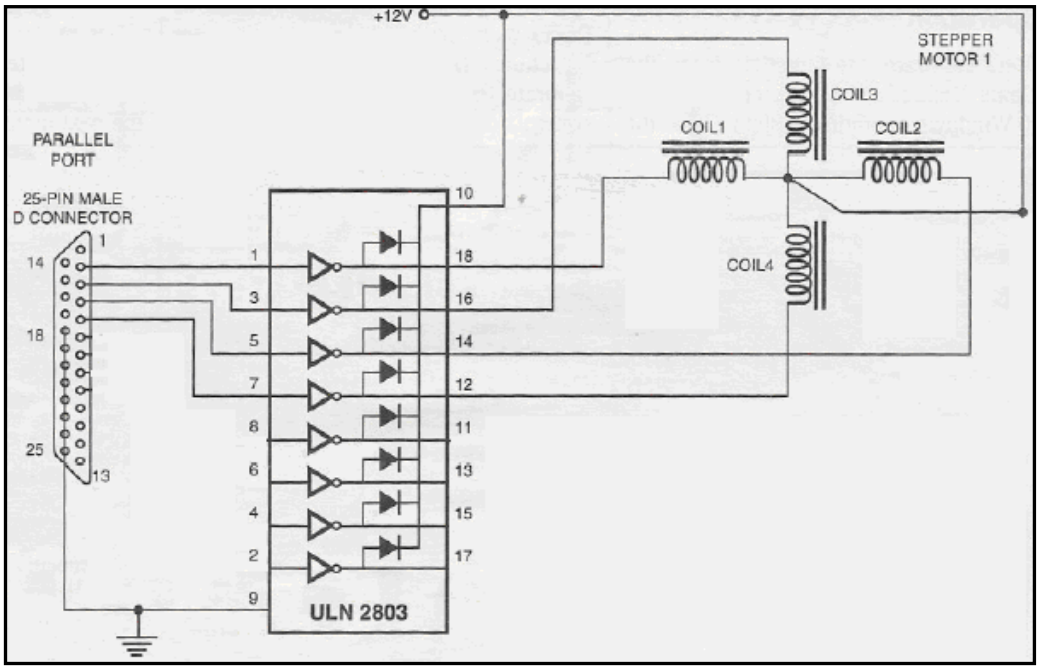


Figure 3.6: Circuit to control First motor

This project use ULN 2803. The function of this IC is to converts the signal from the parallel pot to a specific winding energizing sequence to step the motor. The output signal from ULN 2803 is sent each winding. ULN 2803 generates pulse signal as and when it receives the control data from the computer/programme [9]. The pulses in (from high to low state) activate the circuit and send signals to two sets of winding on the stepper motor. The first set

of signal changes the first winding polarity and the second set of signal changes the second winding polarity, causing the motor to rotate one step.

3.3.3 Circuit To Control Second Motor

The second circuit is same with the first method, but it connected to pin number 6 to pin number 9 to control the motor. Figure 3.7 below show the development circuit by using Eagle software version 6.1.0.

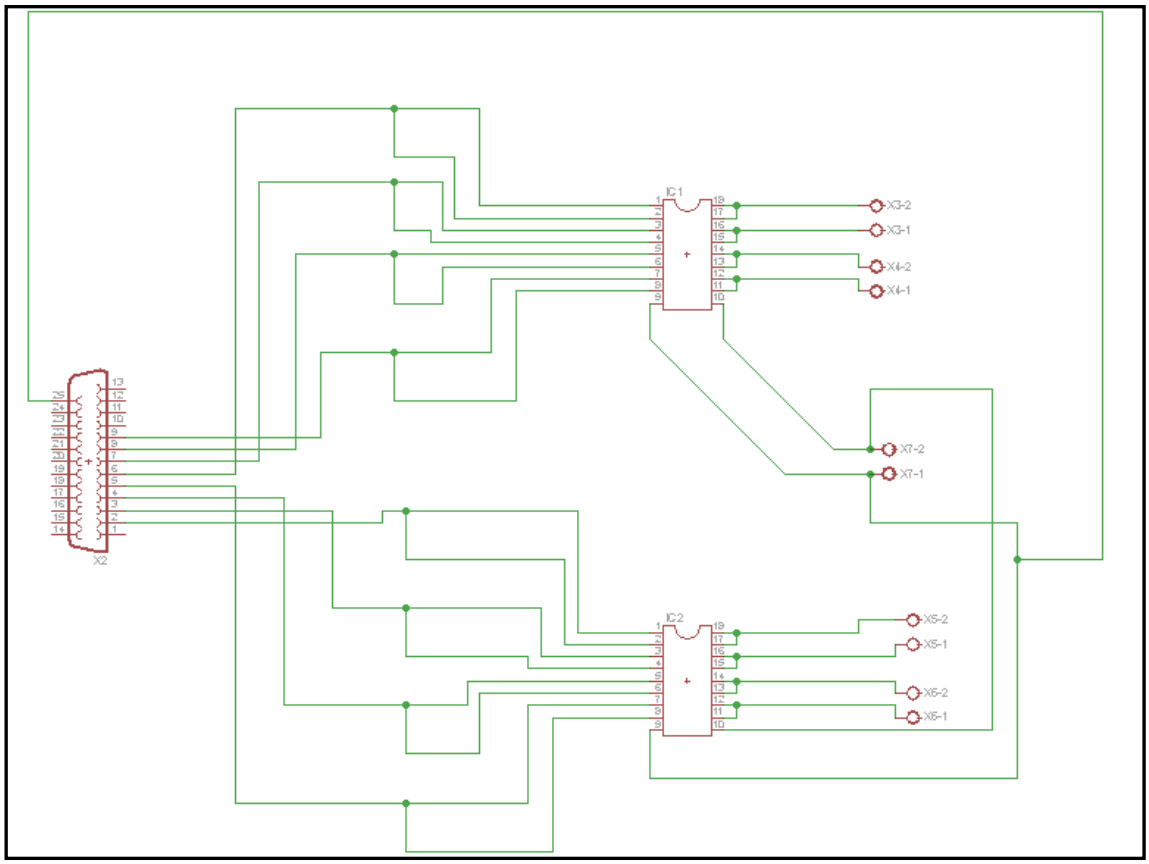


Figure 3.7: Schematic Drawing of Circuit to control 2 axis motor

3.3.4 Pull down Current

Stepper motors feature many special qualities that make them economical for control and measurement uses. Various software and hardware techniques

can optimize specific stepper features [9]. For this project pull down current concept were used to increase the torque of motor, simple modifications to the ULN 2803 motor drive minimize current use while the system operates at voltages above manufacturer-specified levels, The result is improved torque-speed qualities in the stepper.

Consider a simple stepper-motor-drive circuit controlled using a Visual Basic. The general-purpose I/O peripheral ports A and C on the MCU control the switching of high-power in ULN2803 configuration that powers the motor windings.

In this circuit, each winding of a stepping motor is connected to its own ULN 2803. The motor is driven in unipolar mode. The two motor windings or phases (winding A and winding B) are represented in the schematic by both an inductor (L_W) and resistor (R_W) that model the actual behaviour of the coil. The values specified in the electrical ratings of the manufacturer datasheet. Other specifications gleaned from the data sheet include a rated dc-supply voltage of 2.1 V and a maximum dc current of 3 A. To sink the current series resistor current limiting (figure) are apply.

3.4 Flow Chart to control Stepper motor

A VB programme was develop to control two motors in this project. This section will explained how this VB program is set to achieve the objective of this project.

3.4.1 ControllingFirst Motor

Figure 3.8 next page show the flow chart to control the first motor. Firstly operator will put the value in mm into the two text box. First box is for positive movement and the second box is for negative movement, then the program would times the value with 5.4 this is because during calibration process the output data show that 5.4 pulse will turn the motor 1mm. Third

stage, programme will calculate the value if the value can calculate the motor will move base on the value in the text box.

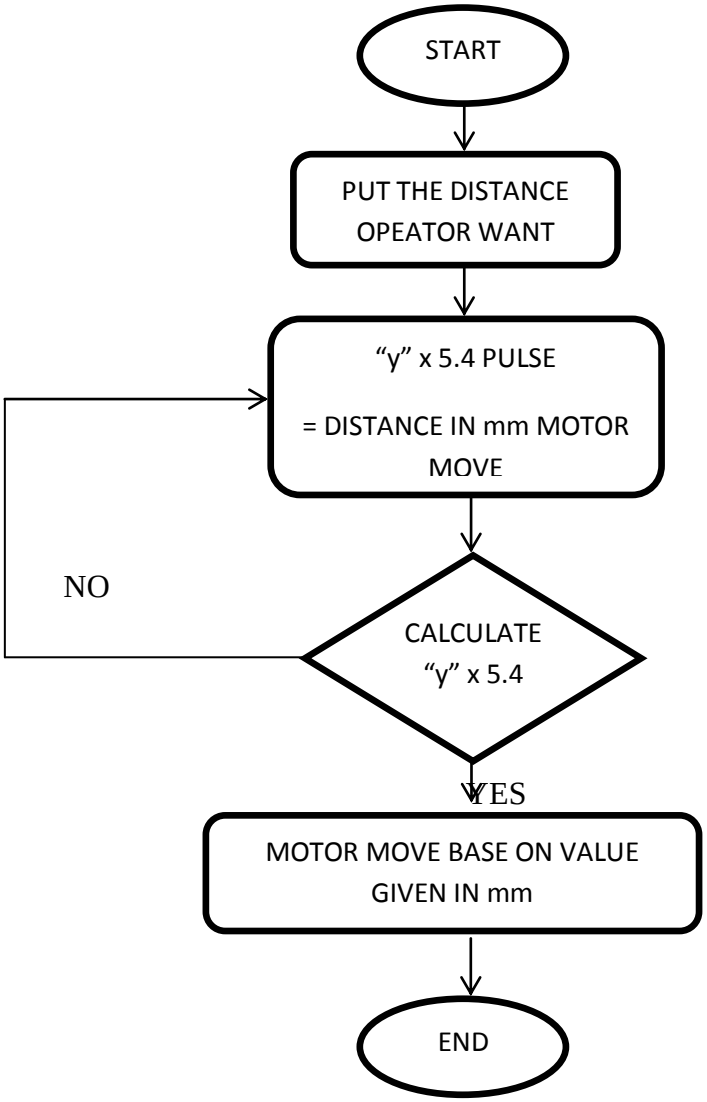
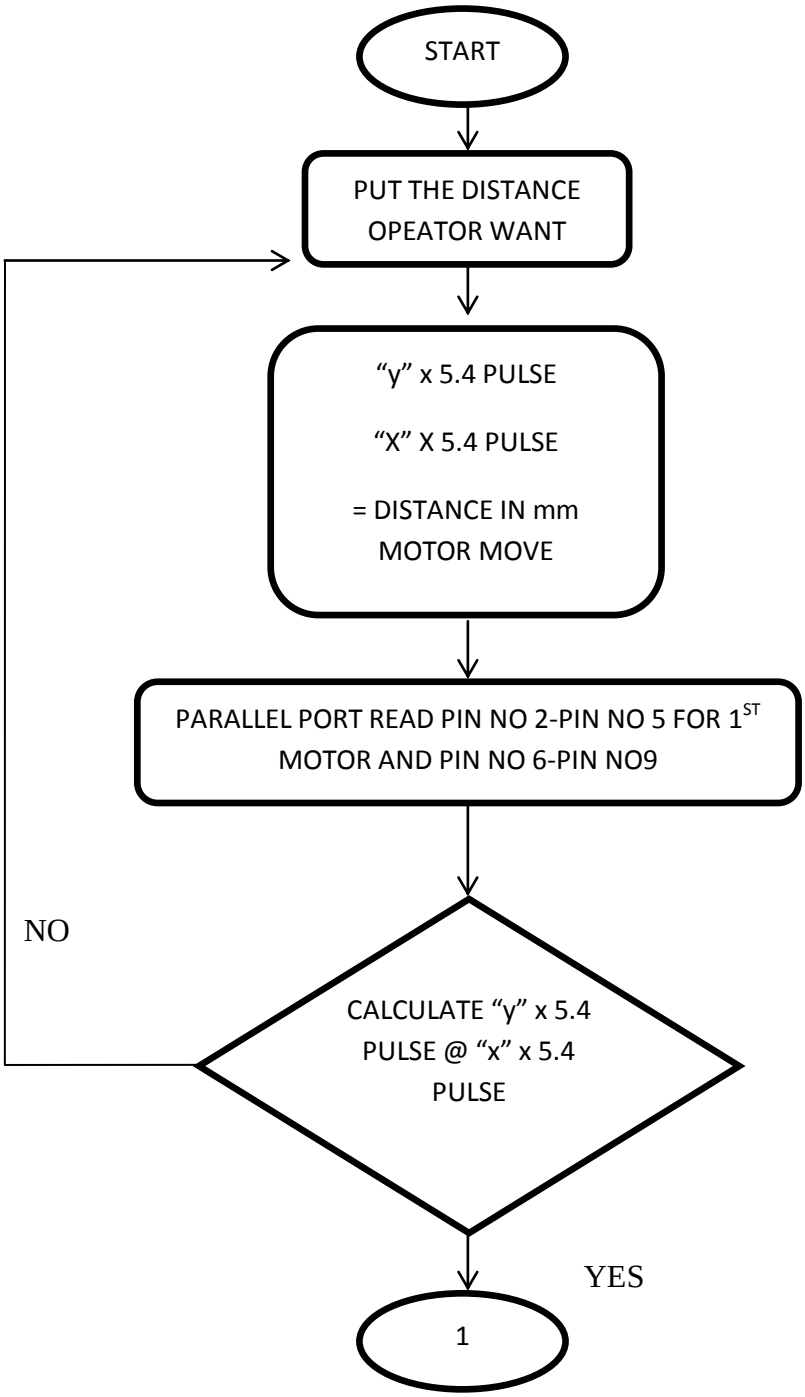


Figure 3.8: Flow chart to control 1 axis

.

3.4.2 Controlling Both of Motor

Controlling both of motor is important so the machine can move continuously follow by the program. This project uses another motion for another axis. Same concepts with same motor specification and same VB programs are apply in this project. Figure 3.9 below show the process of programming.



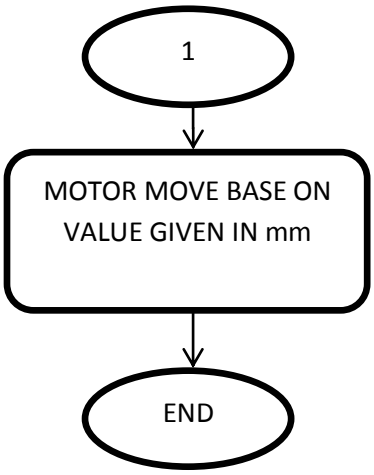


Figure 3.9: Flow chart to control both of motor

In controlling both of motor it still use the same concept with controlling first motor, the different is operator must put the value in 4 text box that was 2 text box for X axis and 2 more for Y axis.

3.5 Positioning And Motion Control Experiment

The stepper motor uses the theory of operation for magnets to make the motor shaft turn a precise distance when a pulse of electricity is provided. Figure 3.10 on the next page shows a typical cross-sectional view of the rotor and stator of a stepper motor. From this diagram the stator (stationary winding) has four poles, and the rotor has six poles (three complete magnets).

The rotor will require 12 pulses of electricity to move the 12 steps to make one complete revolution. Thus, the rotor will move precisely 30° for each pulse of electricity that the motor receives. The number of degrees the rotor will turn when a pulse of electricity is delivered to the motor can be calculated by dividing the number of degrees in one revolution of the shaft (360°) by the number of poles (north and south) in the rotor. In this type of stepper motor 360° is divided by 12 to get 30°.

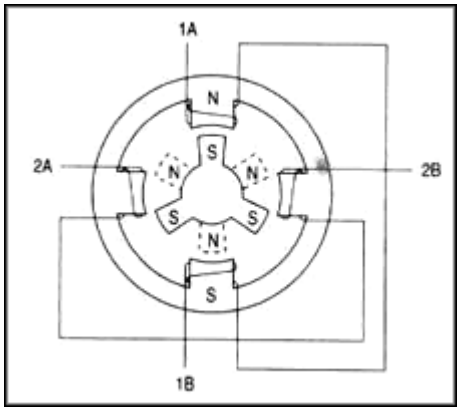


Figure 3.10: The position of the six-pole rotor and four-pole stator of a typical stepper motor [10]

Half step positioning are use to control the motion of the motor. In half step sequence motor, step angle reduces to half the angle in full mode. So the angular resolution is also increased and it becomes double the angular resolution in full mode. In the other hand, in half mode sequence the number of steps gets doubled as that of full mode. Half mode is usually preferred over full mode. The sequences of rotation are given on table 3.2 (a) and (b) below.

Table 3.2(a): Half step sequence for Clockwise

Clockwise Rotation			
Winding A	Winding B	Winding C	Winding D
1	0	0	1
1	0	0	0
1	1	0	0
0	1	0	0
0	1	1	0
0	0	1	0
0	0	1	1
0	0	0	1

Table 3.2(b): Half step sequence for anticlockwise rotation.

Anticlockwise Rotation			
Winding A	Winding B	Winding C	Winding D
0	0	0	1
0	0	1	1
0	0	1	0
0	1	1	0
0	1	0	0
1	1	0	0
1	0	0	0
1	0	0	1

Each of motor has their pin at parallel port to control the motion of the motor, for X axis motor pin that use is no2, no3, no4 and no5 and for Y axis pin no6, no7, no8, and no9. From this pin the pulse given from parallel port to ULN2803 converts the signal from the parallel pot to a specific winding energizing sequence to step the motor.

Table 3.3: Pin use to control two axis stepper motor

PIN USE TO CONTROL TWO AXIS STEPPER MOTOR			
Motor X		Motor Y	
Pin Value	Binary value	Pin Value	Binary Value
128	10000000	8	1000
144	10010000	9	1001
16	10000	1	1
48	110000	3	1100
32	100000	2	10
96	1100000	6	110
64	1000000	4	100
192	11000000	12	1100

Table 3.3 show the value of each pin at parallel port from pin number 2 to pin number 9. From this pin each data can transfer to ULN 2803 and control the motion of stepper motor. This value for motor X and motor Y is for clockwise movement, for anticlockwise it take value from bottom to upper at the program.

CHAPTER 4

RESULT AND ANALYSIS

4.1 INTRODUCTION

After completing all method from the previous chapter, this machine will be analyzed so the objective of this project can be achieved. In this chapter the machine has been move or other word it can be move, so analysis process can be made. The main objective of this chapter is analyzed how much pulse must be given to drive this machine certain distance. The next section will discuss the results of this study.

4.2 RESULTS

The machines is move when pulse from parallel port is send to ULN 2803 and transmit the pulse to stepper motor, each value of pulse is the important to rotate the stepper motor certain degree. Pulse or step are calculate to move the stepper motor to certain distance, where 1 pulse given it could be 1.8° in full-stepping and 0.9° for half stepping. Previous chapter state that this

stepper motor are move using half stepping method, it mean each step the motor will turn 0.9° and have 400 steps/revolution.

In full round of stepper motor there have 360° or 360°/revolution, so to fill 360°/rotation there need 400 steps. Stepper motor takes one step per pulse, so to complete 400 steps/revolution it need 400 pulse current. Equation 3.1 below shows.

$$pitch \times \ num.of\ tooth \ = \ 1revolution\, distance\, motor\, move\ (x)$$

(Eqn 3.1)

The distance travelled is the number of teeth times the belt pitch, an example with 8 tooth with T5 pulleys it would have 40 steps/mm. The relationship of the stepping motor's rotation and input pulses is expressed as the following Equation 3.2 below.

$$\theta \ = \ \theta s \ \times A \qquad \text{(Eqn 3.2)}$$

where θrotation angle of the motor output shaft in degree is, θs is step angle and A is pulse number.

The speed of the rotation is then proportional to the speed of the pulses. The relationship of the pulse speed (Hz) and motor speed (r/min) is expressed as the following equation 3.3 below.

$$N \ = \ \frac{\theta s}{360} \times f \ \times 60 \qquad \text{(Eqn 3.3)}$$

Where N speed of the motor is output shaft in r/min and f is pulse speed in Hz.

Table 4.1: Analysis result from actual calibration

STEPS	DISTANCE MOTOR MOVE (mm)	STEPS/mm	OUTPUT (V)
40	7	5.71429	0.731
60	11	5.45455	0.732
80	15	5.33333	0.747
100	19	5.26316	0.813
120	22	5.45455	0.913
140	27	5.18519	0.949

Table 4.1 show the result from the analysis that have been made, this analysis is determine the distance motor move when j value or step are given. This result will be use in the program to turn the motor to distance that required. The value of steps/mm times with the distance ask so the motor will move certain mm, equation below show how this value are use.

$$Distance\ required\ \times 5.4005\ = Motor\ move \qquad \text{(Eqn 3.4)}$$

Value 5.4005 is average values that get from the data, this value are use in the calculation to move the motor. Two of motor that use are using the same equation because this two motor have a same specification.

4.3 ANALYSIS

In this analysis calibration process are use to determine the step use to move the motor. In the previous program that is use as a base program, the users need to put the step to move the motor. However, users don't know the movement distance move, so the program needs an alteration. The results are more accurate and users friendly controller can be produced.

4.3.1 ULN 2803

The ULN2803 output board allows higher voltage and current devices to be interfaced to the low level logic [11]. Ideally suited for interfacing between low logic level digital circuitry (such as microcontrollers) and the higher current/voltage requirements of lamps, relays, printer hammers or other similar loads for a broad range of computer, industrial, and consumer applications.

Two unit of ULN 2803 is needed to control two stepper, figure 4.2 below illustrate the circuit of two stepper motor with ULN 2803.

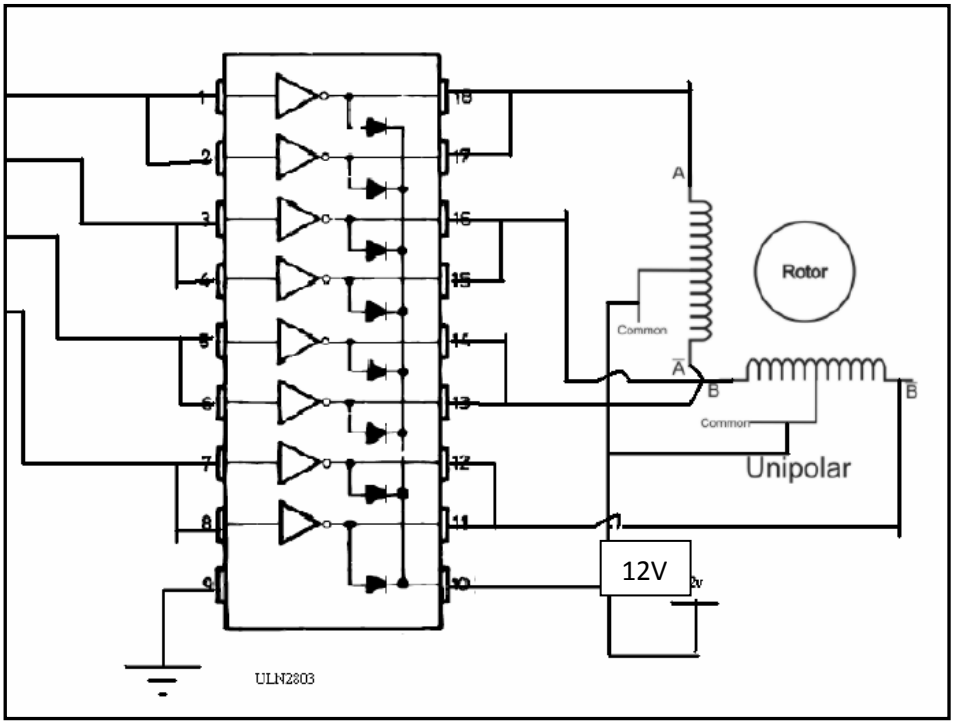


Figure 4.1: Unipolar stepper connection with ULN 2803 [10]

Understanding the characteristic of ULN 2803 is important thing to avoid this IC burn during application, the electrical characteristic of ULN 2803 shows in table 4.2 on the next page.

Table 4.2: Electrical characteristic ULN 2803

Characteristic		Symbol	Min	Typ	Max	Unit
Output Leakage Current (Figure 1)		I_{CEX}				µA
(V_O = 50 V, T_A = +70°C)	All Types		–	–	100	
(V_O = 50 V, T_A = +25°C)	All Types		–	–	50	
(V_O = 50 V, T_A = +70°C, V_I = 6.0 V)	ULN2802		–	–	500	
(V_O = 50 V, T_A = +70°C, V_I = 1.0 V)	ULN2804		–	–	500	
Collector–Emitter Saturation Voltage (Figure 2)		$V_{CE(sat)}$				V
(I_C = 350 mA, I_B = 500 µA)	All Types		–	1.1	1.6	
(I_C = 200 mA, I_B = 350 µA)	All Types		–	0.95	1.3	
(I_C = 100 mA, I_B = 250 µA)	All Types		–	0.85	1.1	
Input Current – On Condition (Figure 4)		$I_{I(on)}$				mA
(V_I = 17 V)	ULN2802		–	0.82	1.25	
(V_I = 3.85 V)	ULN2803		–	0.93	1.35	
(V_I = 5.0 V)	ULN2804		–	0.35	0.5	
(V_I = 12 V)	ULN2804		–	1.0	1.45	
Input Voltage – On Condition (Figure 5)		$V_{I(on)}$				V
(V_{CE} = 2.0 V, I_C = 300 mA)	ULN2802		–	–	13	
(V_{CE} = 2.0 V, I_C = 200 mA)	ULN2803		–	–	2.4	
(V_{CE} = 2.0 V, I_C = 250 mA)	ULN2803		–	–	2.7	
(V_{CE} = 2.0 V, I_C = 300 mA)	ULN2803		–	–	3.0	
(V_{CE} = 2.0 V, I_C = 125 mA)	ULN2804		–	–	5.0	
(V_{CE} = 2.0 V, I_C = 200 mA)	ULN2804		–	–	6.0	
(V_{CE} = 2.0 V, I_C = 275 mA)	ULN2804		–	–	7.0	
(V_{CE} = 2.0 V, I_C = 350 mA)	ULN2804		–	–	8.0	
Input Current – Off Condition (Figure 3) (I_C = 500 µA, T_A = +70°C)	All Types	$I_{I(off)}$	50	100	–	µA
DC Current Gain (Figure 2) (V_{CE} = 2.0 V, I_C = 350 mA)	ULN2801	h_{FE}	1000	–	–	–
Input Capacitance		C_I	–	15	25	pF
Turn–On Delay Time (50% E_I to 50% E_O)		t_{on}	–	0.25	1.0	µs
Turn–Off Delay Time (50% E_I to 50% E_O)		t_{off}	–	0.25	1.0	µs
Clamp Diode Leakage Current (Figure 6) (V_R = 50 V)	T_A = +25°C T_A = +70°C	I_R	–	–	50 100	µA
Clamp Diode Forward Voltage (Figure 7) (I_F = 350 mA)		V_F	–	1.5	2.0	V

4.3.2 Calibration Process

Calibrating a machine is a must in machine fabrication. Tools have to move exactly and accurately, there have two calibration methods that are usually used. The first method is use dial indicator and mach 3 (axis calibration under setting tab) to calibrate a short distance, the second method is using step/mm or step/revolution base on pitch and teeth at the stepper motor.

This project uses the second method that base on step/mm or step/revolution. In the program the j value are adjust to move the machine table, by using a pen it will sketch linear line when machine move. Figure 4.1 show the value that change and figure 4.2 shows how the process are obtained.

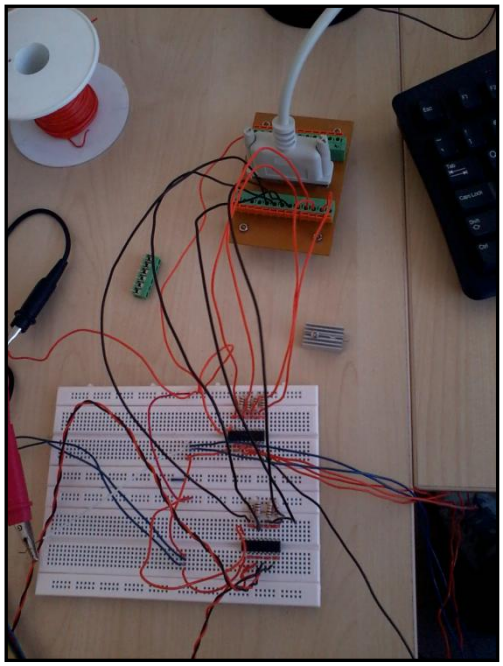


Figure 4.2: ULN connect to two stepper motor

Value j that change in the program is show in figure 4.3 below.

```
Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
    Dim j As Integer
    j = 0

    Dim y As Integer
    y = 6
    Do Until j = 150

        Out(888, 12)
        System.Threading.Thread.Sleep(y)
        Out(888, 4)
        System.Threading.Thread.Sleep(y)
        Out(888, 6)
        System.Threading.Thread.Sleep(y)
        Out(888, 2)
        System.Threading.Thread.Sleep(y)
        Out(888, 3)
        System.Threading.Thread.Sleep(y)
        Out(888, 1)
        System.Threading.Thread.Sleep(y)
        Out(888, 9)
        System.Threading.Thread.Sleep(y)
        Out(888, 8)
        System.Threading.Thread.Sleep(y)
```

Figure 4.3: Step value that change in the calibration process

Calibration process in this project is using a pen and clamp by chuck. Figure 4.4 show the calibrating process

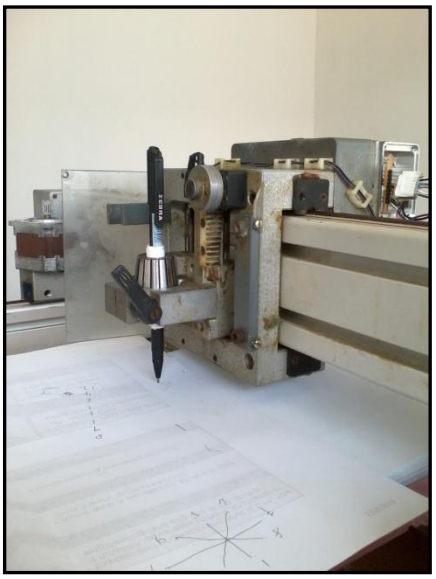


Figure 4.4: Calibration process

CHAPTER 5

CONCLUSIONS AND RECOMMENDATION

5.1 CONCLUSIONS

During journey of this project there a some problem faced especially in development of program to control stepper motor and interface I/O ports to device. However, this problem can be overcome by implementing the skill that had been learned during four year learning in engineering field.

This project starts with understanding the required parts in three axis machine. Then it follows by wiring the stepper motor, wiring is more difficult compare with other motor such as DC motor and AC motor.

In this stepper motor there have 6 wires, usually stepper motors are comes with 5 and 6 wire. Thirdly by using the basic program can be used to control the movement motor. In the basic program step and speed are adjust to ask motor move. Base on equation (3.1) the value step motor can determine, each step in half-step movement is 0.9° and has 400 steps to fill full rotation of 360°.

In calibration process, step values are set using VB program. Table move based on that step, and operator need to key in the distance. VB program will calculate and control the motor to moves desired point. An example, when the operator keys in 12mm in text box, the program calculate the movement using

equation 3.4. The results then are sent to parallel port (I/O port) and interface the data to ULN 2803. Finally ULN change it to pulse before transmit to stepper motor.

For the conclusion, this study has achieved the two objectives. A control system for two axis mechanism machine had been successfully design. Where for the second objective, to test and analyze the parameters and characteristic of tracking control of feed drive in machine are also tested and analyzed.

5.2 RECOMMENDATION

There are some improvement can be done for this controller. For current project, this machine only can make a linear movement and two of motor that use can't integrate together to move to the point that ask as an usual CNC machine that have in market now. It is strongly recommended that a program which can integrate the movement of two axis can be done. There are lots of application can be applied if it done.

In addition to improve the application:

- Using 74LS244 1 of 4 Decoder to control 3 axis stepper motor
- Tool path to roughing
- Feed during machining process
- Have a specific task in a program such as application for circular machining, plunge drilling and pocket machining.

REFERENCES

[1] Sclater. N, *Mechanisms and Mechanical Devices Sourcebook,* McGraw-Hill Professional, 2011, pp. 22-37.

[2] Kalpakjian. S , *Manufacturing Engineering,* Prentice Hall, 2006, pp. 760-771.

[3] Nise. S. N, *Control System Engineering,* John Wiley, 2011, pp. 4-30.

[4] Amilin,W.A , *Design and Analysis a Control System for 3 Axis Mechanism Machine,* Faculty of Mechanical Engineering, University Malaysia Pahang , May 2009

[5] Craig. K, "Motor for Mechatronics an Introduction"

[6] Wei U. X and Jihong. C, "Design of Servo System For 3 axis CNC drilling Machine Based on xPC –Target. ISECS International colloquium on computing, communication, control and management. 2009.

[7] Scribd, " Non-conventional Machining" December 2011, http://www.scribd.com/doc/11621837/35-Non-Conventional-Machining

[8] Smid. P, *CNC programming handbook: a comprehensive guide to practical CNC programming,* Industrial Press Inc., 2003, pp. 177-180.

[9] Shinjo.N, Buist.L and Subramanian.S. 1999. Stepper Motor Control With a PC-Based Data Acquisition Toolkit. Florida Institute of Technology, Melbourne.

[10] Singh.M , Rekha and Singh.B. January-June 2010. Microcontroller Based Clockwise/Anticlockwise Stepper Motor Controller Using PC Keyboard Via Com Port, *Intenational Journal of Compute Science & Communication.* pp. 189-191.

[11] Paul.b. *Controlling the World With Your PC*, Newnes, 1994, pp 97-96.

[12] Global Spec " How to Select Linear Slides" December 2011, http://www.globalspec.com/learnmore/motion_controls/linear_rotary_motion_components/linear_positioning_stages.

[13] Generic Slides "Manual Positioning Slide", December 2011, http://www.genericslides.com/manual_positioning_slides.html

[14] Little Machine Shop "XY Table", December 2011, http://littlemachineshop.com/products/product_focus.php?Focus=X-Y+Tables

APPENDIX A

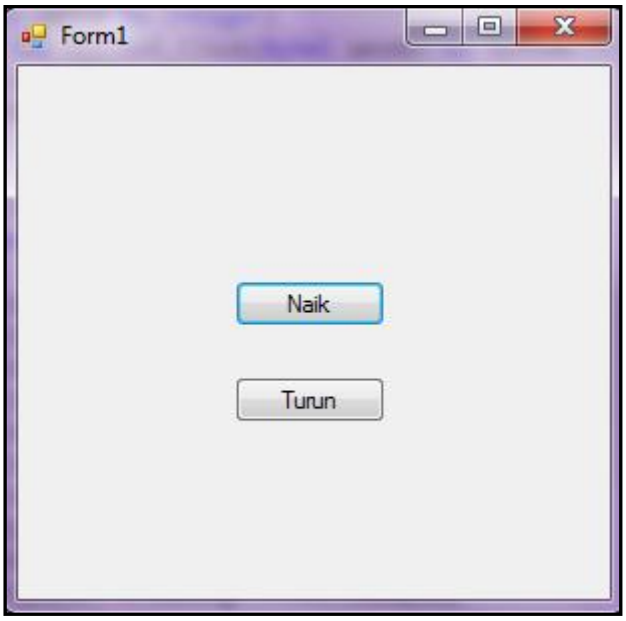
BASIC PROGRAM TO CONTROL STEPPER MOTOR

```
Public Class Form1
    Public Declare Sub Out Lib "inpout32.dll" Alias "Out32" _
      (ByVal PortAddress As Integer, _
      ByVal Value As Integer)
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
        System.EventArgs) Handles Button1.Click
      Dim i As Integer
      i = 0

      Dim x As Integer
      x = 10
      Do Until i = 150

        Out(888, 8)
        System.Threading.Thread.Sleep(x)
        Out(888, 9)
        System.Threading.Thread.Sleep(x)
        Out(888, 1)
        System.Threading.Thread.Sleep(x)
        Out(888, 3)
        System.Threading.Thread.Sleep(x)
        Out(888, 2)
        System.Threading.Thread.Sleep(x)
        Out(888, 6)
        System.Threading.Thread.Sleep(x)
        Out(888, 4)
        System.Threading.Thread.Sleep(x)
        Out(888, 12)
        System.Threading.Thread.Sleep(x)

        i = i + 1
      Loop

      Out(888, 0)


    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
        System.EventArgs) Handles Button2.Click
      Dim j As Integer
      j = 0
```

```
        Dim y As Integer
        y = 10
        Do Until j = 150

            Out(888, 12)
            System.Threading.Thread.Sleep(y)
            Out(888, 4)
            System.Threading.Thread.Sleep(y)
            Out(888, 6)
            System.Threading.Thread.Sleep(y)
            Out(888, 2)
            System.Threading.Thread.Sleep(y)
            Out(888, 3)
            System.Threading.Thread.Sleep(y)
            Out(888, 1)
            System.Threading.Thread.Sleep(y)
            Out(888, 9)
            System.Threading.Thread.Sleep(y)
            Out(888, 8)
            System.Threading.Thread.Sleep(y)



            j = j + 1
        Loop

        Out(888, 0)


    End Sub
End Class
```

APPENDIX B

OUTPUT FROM BASIC PROGRAMMING



APPENDIX C

PROGRAMMING TO CONTROL MOTOR X AND MOTOR Y

```
Public Class Form1

    Public sumX As Integer
    Public sumX2 As Integer
    Public sumX3 As Integer
    Public sumY1 As Integer
    Public sumY2 As Integer
    Public sumX4 As Integer
    Public sumY3 As Integer
    Public sumY4 As Integer
    Public sumX5 As Integer
    Public sumX6 As Integer
    Public sumY5 As Integer
    Public sumY6 As Integer
    Public sumX7 As Integer
    Public sumX8 As Integer
    Public sumY7 As Integer
    Public sumY8 As Integer
    Public sumX9 As Integer
    Public sumX10 As Integer
    Public sumY9 As Integer
    Public sumY10 As Integer


    Public en_able As Boolean

    Private Property intNumber1 As Double

    Public Declare Sub Out Lib "inpout32.dll" Alias "Out32" _
       (ByVal PortAddress As Integer, _
       ByVal Value As Integer)



    Private Sub START_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Start_cycle.Click

        ' ////////////////////////////////////////////////////////////////////////////////////////////////X
positive movement 1
        en_able = CheckBox1.Checked
        Dim i As Integer
```

```
        i = 0

        Dim x As Integer
        x = Val(TextBox10.Text) 'SPEED
        Do Until i = Val(sumX) 'DISTANCE

            If en_able Then
                Out(888, 8)
            End If

            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 9)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 1)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 3)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 2)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 6)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 4)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 12)
            End If
            System.Threading.Thread.Sleep(x)
            sy()
            i = i + 1
        Loop

        If en_able Then
            Out(888, 0)
        End If

        '////////////////////////////////////////////////////////////////////////////////////////////////////// X
negative movement 2
```

```
Dim j As Integer
j = 0

Dim y As Integer
y = Val(TextBox10.Text) 'SPEED
Do Until j = Val(sumX2) 'DISTANCE

   If en_able Then
      Out(888, 12)
   End If
   System.Threading.Thread.Sleep(y)
   If en_able Then
      Out(888, 4)
   End If
   System.Threading.Thread.Sleep(y)
   If en_able Then
      Out(888, 6)
   End If
   System.Threading.Thread.Sleep(y)
   If en_able Then
      Out(888, 2)
   End If
   System.Threading.Thread.Sleep(y)
   If en_able Then
      Out(888, 3)
   End If
   System.Threading.Thread.Sleep(y)
   If en_able Then
      Out(888, 1)
   End If
   System.Threading.Thread.Sleep(y)
   If en_able Then
      Out(888, 9)
   End If
   System.Threading.Thread.Sleep(y)
   If en_able Then
      Out(888, 8)
   End If
   System.Threading.Thread.Sleep(y)


   j = j + 1
Loop

If en_able Then


   Out(888, 0)
End If
```

```
' ////////////////////////////////////////////////////////////////////////////////////////////////Y positive movement 1

        en_able = CheckBox1.Checked
        Dim i2 As Integer
        i2 = 0

        Dim y1 As Integer
        y1 = Val(TextBox10.Text) 'SPEED
        Do Until i2 = Val(sumY1) 'DISTANCE

            If en_able Then
                Out(888, 128)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 144)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 16)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 48)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 32)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 96)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 64)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 192)
            End If
            System.Threading.Thread.Sleep(x)

            i2 = i2 + 1
        Loop

        If en_able Then
```

```
        Out(888, 0)
    End If

    ' ////////////////////////////////////////////////////////////////////////////////////////////////////Y
Negative movement 2


    en_able = CheckBox1.Checked
    Dim i3 As Integer
    i3 = 0

    Dim y2 As Integer
    y2 = Val(TextBox10.Text) 'SPEED
    Do Until i3 = Val(sumY2) 'DISTANCE

        If en_able Then
            Out(888, 192)
        End If

        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 64)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 96)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 32)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 48)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 16)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 144)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 128)
        End If
        System.Threading.Thread.Sleep(x)
```

```
        i3 = i3 + 1
    Loop

    If en_able Then
        Out(888, 0)
    End If


    '////////////////////////////////////////////////////////////////////////////////////////////////////X
positif movement 3
    Dim i1 As Integer
    i1 = 0

    Dim x1 As Integer
    x1 = Val(TextBox10.Text) 'SPEED
    Do Until i1 = Val(sumX3) 'DISTANCE

        If en_able Then
            Out(888, 8)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 9)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 1)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 3)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 2)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 6)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 4)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 12)
        End If
        System.Threading.Thread.Sleep(x)
```

```
        i1 = i1 + 1
    Loop

    If en_able Then
        Out(888, 0)
    End If

    '////////////////////////////////////////////////////////////////////////////////////////////////////// X
negative movement 4
    Dim j1 As Integer
    j1 = 0

    Dim x2 As Integer
    x2 = Val(TextBox10.Text) 'SPEED
    Do Until j1 = Val(sumX4) 'DISTANCE

        If en_able Then
            Out(888, 12)
        End If
        System.Threading.Thread.Sleep(y)
        If en_able Then
            Out(888, 4)
        End If
        System.Threading.Thread.Sleep(y)
        If en_able Then
            Out(888, 6)
        End If
        System.Threading.Thread.Sleep(y)
        If en_able Then
            Out(888, 2)
        End If
        System.Threading.Thread.Sleep(y)
        If en_able Then
            Out(888, 3)
        End If
        System.Threading.Thread.Sleep(y)
        If en_able Then
            Out(888, 1)
        End If
        System.Threading.Thread.Sleep(y)
        If en_able Then
            Out(888, 9)
        End If
        System.Threading.Thread.Sleep(y)
        If en_able Then
            Out(888, 8)
        End If
        System.Threading.Thread.Sleep(y)
```

```
        j1 = j1 + 1
    Loop

    If en_able Then


        Out(888, 0)
    End If

    ' ////////////////////////////////////////////////////////////////////////////////////////////////Y
positive movement 3

    en_able = CheckBox1.Checked
    Dim i4 As Integer
    i4 = 0

    Dim y3 As Integer
    y3 = Val(TextBox10.Text) 'SPEED
    Do Until i4 = Val(sumY3) 'DISTANCE

        If en_able Then
            Out(888, 128)
        End If

        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 144)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 16)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 48)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 32)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 96)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 64)
```

```
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 192)
            End If
            System.Threading.Thread.Sleep(x)

            i4 = i4 + 1
        Loop

        If en_able Then
            Out(888, 0)
        End If

        ' ////////////////////////////////////////////////////////////////////////////////////////////////////Y
Negative movement 4


        en_able = CheckBox1.Checked
        Dim i5 As Integer
        i5 = 0

        Dim y4 As Integer
        y4 = Val(TextBox10.Text) 'SPEED
        Do Until i5 = Val(sumY4) 'DISTANCE

            If en_able Then
                Out(888, 192)
            End If

            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 64)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 96)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 32)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 48)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 16)
```

```
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 144)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 128)
            End If
            System.Threading.Thread.Sleep(x)

            i5 = i5 + 1
        Loop

        If en_able Then
            Out(888, 0)
        End If

        '////////////////////////////////////////////////////////////////////////////////////////////////////X
positif movement 5
        Dim i6 As Integer
        i6 = 0

        Dim x3 As Integer
        x3 = Val(TextBox10.Text) 'SPEED
        Do Until i6 = Val(sumX5) 'DISTANCE

            If en_able Then
                Out(888, 8)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 9)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 1)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 3)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 2)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 6)
```

```
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
               Out(888, 4)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
               Out(888, 12)
            End If
            System.Threading.Thread.Sleep(x)

            i6 = i6 + 1
         Loop

         If en_able Then
            Out(888, 0)
         End If
         '////////////////////////////////////////////////////////////////////////////////////////////////////// X
negative movement 6
         Dim j5 As Integer
         j5 = 0

         Dim x4 As Integer
         x4 = Val(TextBox10.Text) 'SPEED
         Do Until j5 = Val(sumX6) 'DISTANCE

            If en_able Then
               Out(888, 12)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
               Out(888, 4)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
               Out(888, 6)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
               Out(888, 2)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
               Out(888, 3)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
               Out(888, 1)
            End If
```

```
            System.Threading.Thread.Sleep(y)
            If en_able Then
                Out(888, 9)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
                Out(888, 8)
            End If
            System.Threading.Thread.Sleep(y)


            j5 = j5 + 1
        Loop

        If en_able Then


            Out(888, 0)
        End If

        ' ////////////////////////////////////////////////////////////////////////////////////////////////Y
positive movement 5

        en_able = CheckBox1.Checked
        Dim i7 As Integer
        i7 = 0

        Dim y5 As Integer
        y5 = Val(TextBox10.Text) 'SPEED
        Do Until i7 = Val(sumY5) 'DISTANCE

            If en_able Then
                Out(888, 128)
            End If

            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 144)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 16)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 48)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
```

```
            Out(888, 32)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 96)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 64)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 192)
        End If
        System.Threading.Thread.Sleep(x)

        i7 = i7 + 1
    Loop

    If en_able Then
        Out(888, 0)
    End If
    ' ////////////////////////////////////////////////////////////////////////////////////////////////////Y
Negative movement 6


    en_able = CheckBox1.Checked
    Dim i8 As Integer
    i8 = 0

    Dim y6 As Integer
    y6 = Val(TextBox10.Text) 'SPEED
    Do Until i8 = Val(sumY6) 'DISTANCE

        If en_able Then
            Out(888, 192)
        End If

        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 64)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 96)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 32)
```

```
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 48)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 16)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 144)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 128)
            End If
            System.Threading.Thread.Sleep(x)

            i8 = i8 + 1
        Loop

        If en_able Then
            Out(888, 0)
        End If


        '////////////////////////////////////////////////////////////////////////////////////////////////////X
positif movement 7
        en_able = CheckBox1.Checked
        Dim i9 As Integer
        i9 = 0
        Dim x5 As Integer
        x5 = Val(TextBox10.Text) 'SPEED
        Do Until i9 = Val(sumX7) 'DISTANCE

            If en_able Then
                Out(888, 8)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 9)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 1)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
```

```
            Out(888, 3)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 2)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 6)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 4)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 12)
        End If
        System.Threading.Thread.Sleep(x)

        i9 = i9 + 1
    Loop

    If en_able Then
        Out(888, 0)
    End If
    '////////////////////////////////////////////////////////////////////////////////////////////////////// X
negative movement 8
    Dim j6 As Integer
    j6 = 0

    Dim x6 As Integer
    x6 = Val(TextBox10.Text) 'SPEED
    Do Until j6 = Val(sumX8) 'DISTANCE

        If en_able Then
            Out(888, 12)
        End If
        System.Threading.Thread.Sleep(y)
        If en_able Then
            Out(888, 4)
        End If
        System.Threading.Thread.Sleep(y)
        If en_able Then
            Out(888, 6)
        End If
        System.Threading.Thread.Sleep(y)
        If en_able Then
            Out(888, 2)
```

```
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
               Out(888, 3)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
               Out(888, 1)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
               Out(888, 9)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
               Out(888, 8)
            End If
            System.Threading.Thread.Sleep(y)


            j6 = j6 + 1
        Loop

        If en_able Then
            Out(888, 0)
        End If
        ' ////////////////////////////////////////////////////////////////////////////////////////////////Y
positive movement 7

        en_able = CheckBox1.Checked
        Dim i10 As Integer
        i10 = 0

        Dim y7 As Integer
        y7 = Val(TextBox10.Text)  'SPEED
        Do Until i10 = Val(sumY7) 'DISTANCE

            If en_able Then
               Out(888, 128)
            End If

            System.Threading.Thread.Sleep(x)
            If en_able Then
               Out(888, 144)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
               Out(888, 16)
            End If
```

```
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 48)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 32)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 96)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 64)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 192)
            End If
            System.Threading.Thread.Sleep(x)

            i10 = i10 + 1
        Loop

        If en_able Then
            Out(888, 0)
        End If
        ' ////////////////////////////////////////////////////////////////////////////////////////////////////Y
Negative movement 8
        en_able = CheckBox1.Checked
        Dim i11 As Integer
        i11 = 0

        Dim y8 As Integer
        y8 = Val(TextBox10.Text)  'SPEED
        Do Until i11 = Val(sumY8) 'DISTANCE

            If en_able Then
                Out(888, 192)
            End If

            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 64)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 96)
```

```
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 32)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 48)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 16)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 144)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 128)
            End If
            System.Threading.Thread.Sleep(x)

            i11 = i11 + 1
        Loop

        If en_able Then
            Out(888, 0)
        End If
        '////////////////////////////////////////////////////////////////////////////////////////////////X
positif movement 9
        en_able = CheckBox1.Checked
        Dim i12 As Integer
        i12 = 0
        Dim x8 As Integer
        x8 = Val(TextBox10.Text)  'SPEED
        Do Until i12 = Val(sumX9) 'DISTANCE

            If en_able Then
                Out(888, 8)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 9)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 1)
            End If
```

```
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 3)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 2)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 6)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 4)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 12)
            End If
            System.Threading.Thread.Sleep(x)

            i12 = i12 + 1
        Loop

        If en_able Then
            Out(888, 0)
        End If
        '////////////////////////////////////////////////////////////////////////////////////////////////////// X
negative movement 10
        Dim j7 As Integer
        j7 = 0

        Dim x9 As Integer
        x9 = Val(TextBox10.Text)  'SPEED
        Do Until j7 = Val(sumX10) 'DISTANCE

            If en_able Then
                Out(888, 12)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
                Out(888, 4)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
                Out(888, 6)
            End If
            System.Threading.Thread.Sleep(y)
```

```
            If en_able Then
                Out(888, 2)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
                Out(888, 3)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
                Out(888, 1)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
                Out(888, 9)
            End If
            System.Threading.Thread.Sleep(y)
            If en_able Then
                Out(888, 8)
            End If
            System.Threading.Thread.Sleep(y)


            j7 = j7 + 1
        Loop

        If en_able Then
            Out(888, 0)
        End If
        ' ////////////////////////////////////////////////////////////////////////////////////////////////Y
positive movement 9

        en_able = CheckBox1.Checked
        Dim i13 As Integer
        i13 = 0

        Dim y9 As Integer
        y9 = Val(TextBox10.Text)  'SPEED
        Do Until i13 = Val(sumY9) 'DISTANCE

            If en_able Then
                Out(888, 128)
            End If

            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 144)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
```

```
            Out(888, 16)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 48)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 32)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 96)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 64)
        End If
        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 192)
        End If
        System.Threading.Thread.Sleep(x)

        i13 = i13 + 1
    Loop

    If en_able Then
        Out(888, 0)
    End If
    ' ////////////////////////////////////////////////////////////////////////////////////////////////Y
Negative movement 10
    en_able = CheckBox1.Checked
    Dim i14 As Integer
    i14 = 0

    Dim y10 As Integer
    y10 = Val(TextBox10.Text)  'SPEED
    Do Until i14 = Val(sumY10) 'DISTANCE

        If en_able Then
            Out(888, 192)
        End If

        System.Threading.Thread.Sleep(x)
        If en_able Then
            Out(888, 64)
        End If
        System.Threading.Thread.Sleep(x)
```

```
            If en_able Then
                Out(888, 96)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 32)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 48)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 16)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 144)
            End If
            System.Threading.Thread.Sleep(x)
            If en_able Then
                Out(888, 128)
            End If
            System.Threading.Thread.Sleep(x)

            i14 = i14 + 1
        Loop

        If en_able Then
            Out(888, 0)
        End If

    End Sub



    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        CheckBox1.Checked = True

    End Sub

    Private Sub CCE_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles CCE.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(CCE.Text)
        intNumber2 = 5.4
```

```
        sumX = intNumber1 * intNumber2

    End Sub

    Private Sub ACCE_TextChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles ACCE.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(ACCE.Text)
        intNumber2 = 5.4

        sumX2 = intNumber1 * intNumber2

    End Sub


    Private Sub CCE1_TextChanged_1(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles CCE1.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(CCE1.Text)
        intNumber2 = 5.4

        sumX3 = intNumber1 * intNumber2

    End Sub

    Private Sub TextBox1_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TextBox1.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox1.Text)
        intNumber2 = 5.4

        sumY1 = intNumber1 * intNumber2

    End Sub

    Private Sub TextBox2_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TextBox2.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox2.Text)
        intNumber2 = 5.4

        sumY2 = intNumber1 * intNumber2

    End Sub
```

```
    Private Sub TextBox3_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TextBox3.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox3.Text)
        intNumber2 = 5.4

        sumX4 = intNumber1 * intNumber2

    End Sub

    Private Sub TextBox4_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TextBox4.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox4.Text)
        intNumber2 = 5.4

        sumY3 = intNumber1 * intNumber2
    End Sub

    Private Sub TextBox5_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TextBox5.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox5.Text)
        intNumber2 = 5.4

        sumY4 = intNumber1 * intNumber2
    End Sub

    Private Sub TextBox6_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TextBox6.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox6.Text)
        intNumber2 = 5.4

        sumX5 = intNumber1 * intNumber2
    End Sub

    Private Sub TextBox7_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TextBox7.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox7.Text)
        intNumber2 = 5.4

        sumX6 = intNumber1 * intNumber2
```

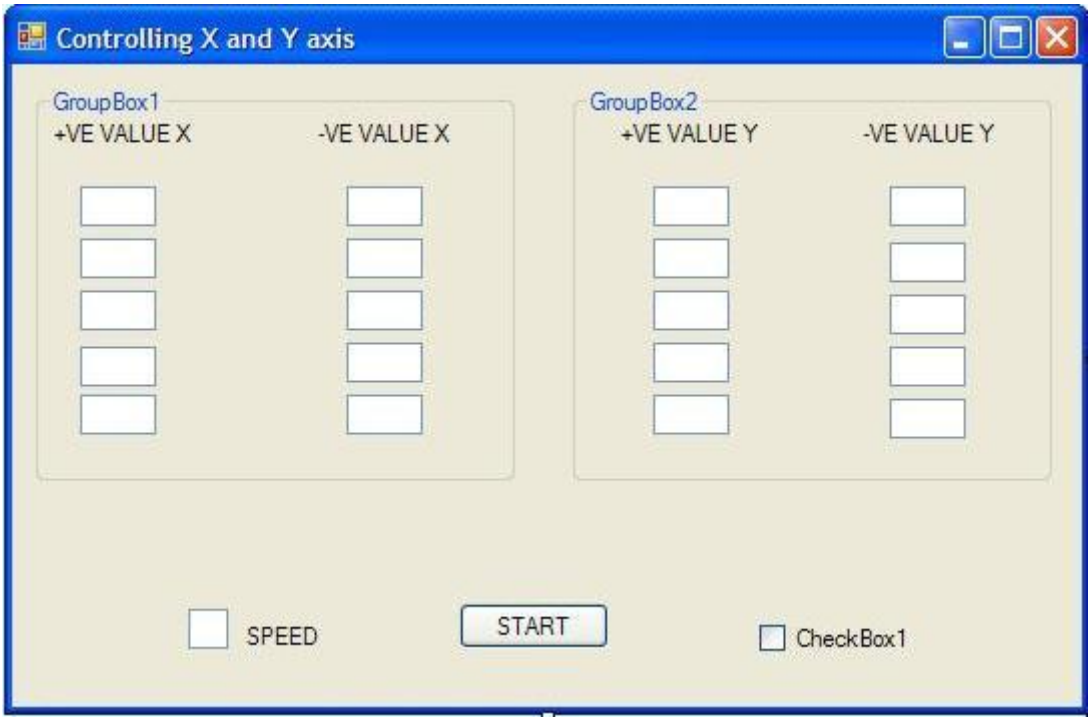
```
    End Sub

    Private Sub TextBox8_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TextBox8.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox8.Text)
        intNumber2 = 5.4

        sumY5 = intNumber1 * intNumber2
    End Sub

    Private Sub TextBox9_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles TextBox9.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox9.Text)
        intNumber2 = 5.4

        sumY6 = intNumber1 * intNumber2
    End Sub

    Private Sub TextBox10_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles TextBox10.TextChanged
        Dim intNumber1
        intNumber1 = Val(TextBox10.Text)

    End Sub

    Private Sub TextBox11_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles TextBox11.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox11.Text)
        intNumber2 = 5.4

        sumX7 = intNumber1 * intNumber2
    End Sub

    Private Sub TextBox12_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles TextBox12.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox12.Text)
        intNumber2 = 5.4

        sumX8 = intNumber1 * intNumber2
    End Sub
```

```
    Private Sub TextBox13_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles TextBox13.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox13.Text)
        intNumber2 = 5.4

        sumY7 = intNumber1 * intNumber2
    End Sub

    Private Sub GroupBox1_Enter(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles GroupBox1.Enter

    End Sub

    Private Sub TextBox14_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles TextBox14.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox14.Text)
        intNumber2 = 5.4

        sumY8 = intNumber1 * intNumber2
    End Sub

    Private Sub TextBox15_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles TextBox15.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox15.Text)
        intNumber2 = 5.4

        sumX9 = intNumber1 * intNumber2
    End Sub

    Private Sub TextBox16_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles TextBox16.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox16.Text)
        intNumber2 = 5.4

        sumX10 = intNumber1 * intNumber2
    End Sub

    Private Sub TextBox17_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles TextBox17.TextChanged
        Dim intNumber1
        Dim intNumber2
```

```
        intNumber1 = Val(TextBox17.Text)
        intNumber2 = 5.4

        sumY9 = intNumber1 * intNumber2
    End Sub

    Private Sub TextBox18_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles TextBox18.TextChanged
        Dim intNumber1
        Dim intNumber2
        intNumber1 = Val(TextBox18.Text)
        intNumber2 = 5.4

        sumY10 = intNumber1 * intNumber2
    End Sub

    Private Sub Label3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label3.Click

    End Sub

    Private Sub Label1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label1.Click

    End Sub
End Class



'       i = i + 1
'  Loop

'        If en_able then Out(888, 0)


'    End Sub


'End Class
```

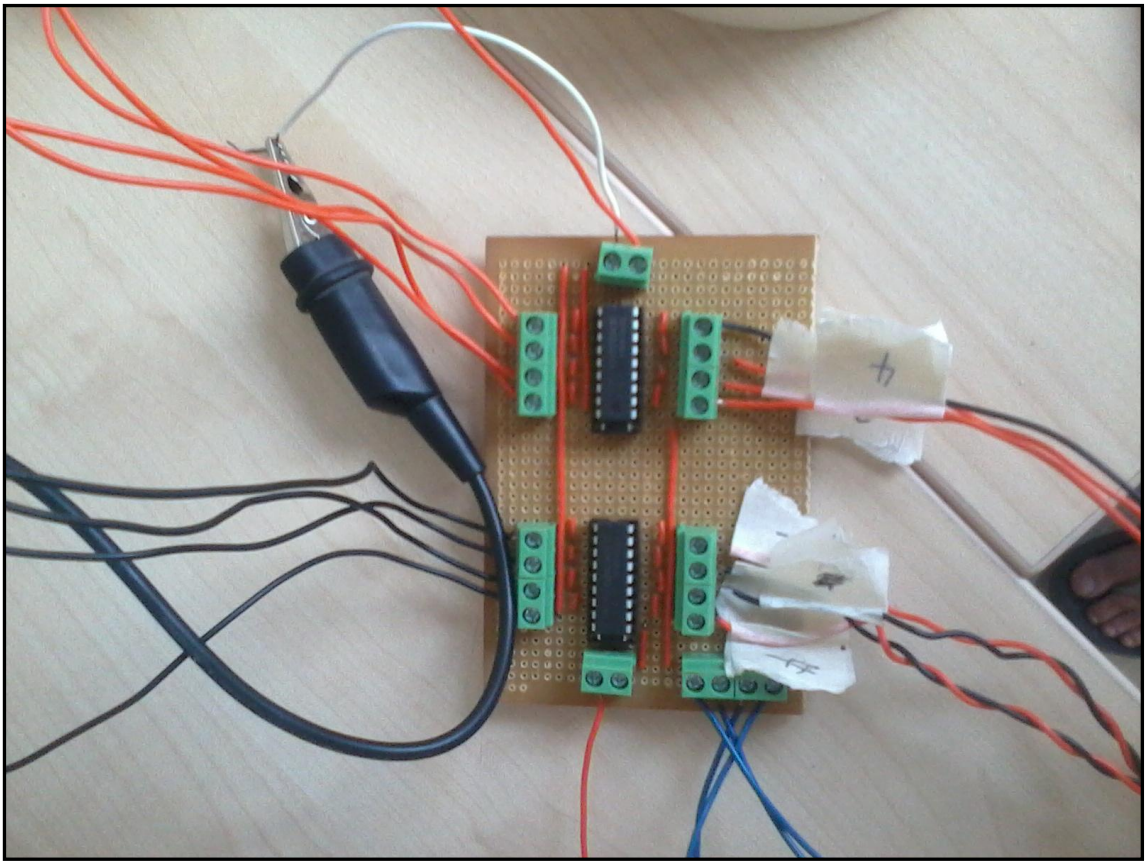
APPENDIX D

OUTPUT PROGRAMME TO CONTROL MOTOR X AND MOTOR Y



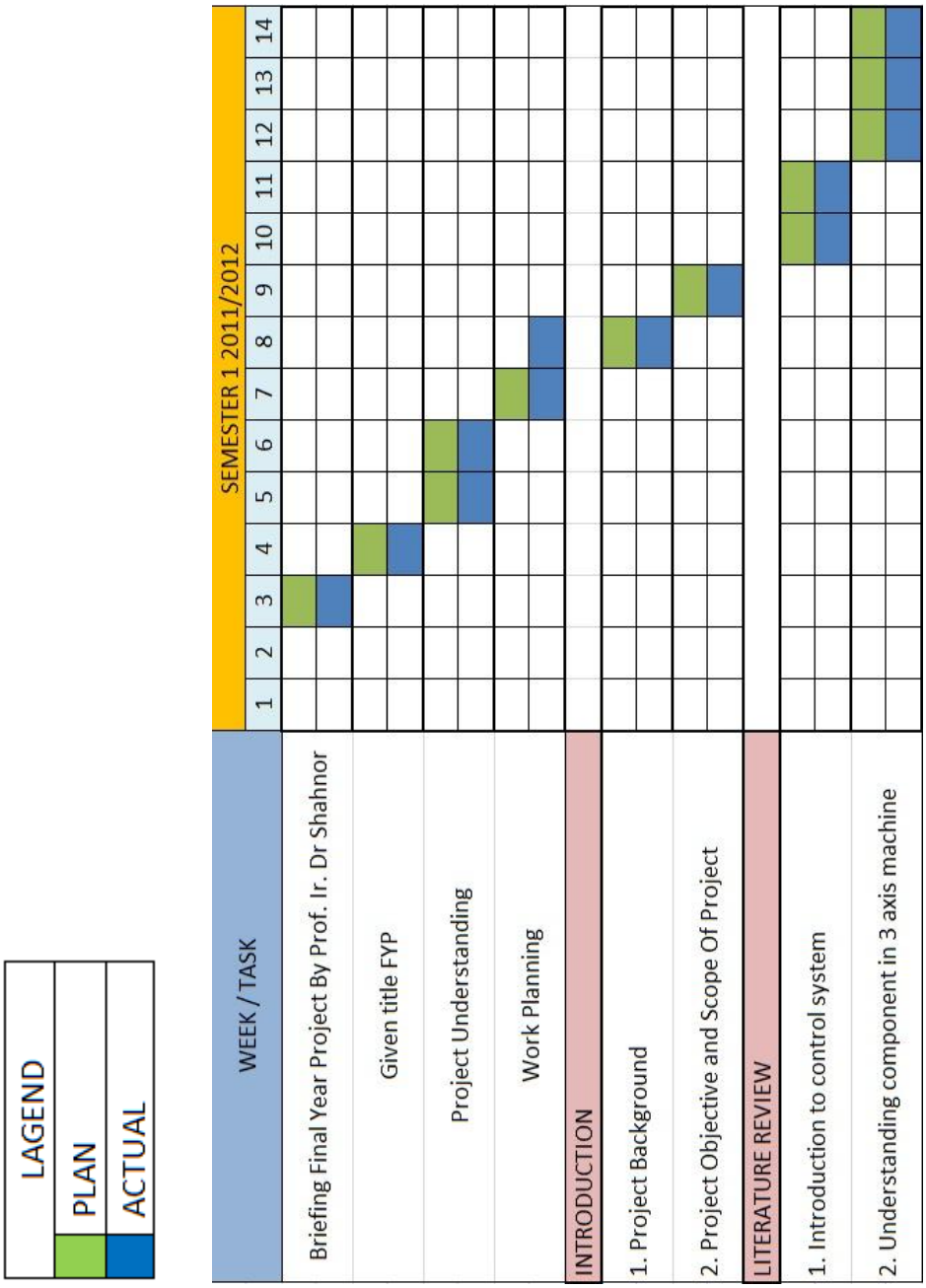
APPENDIX E

CIRCUIT TO CONTROL X AND Y AXIS



APPENDIX F1

GANTT CHART PSM 1



APPENDIX F2

GANTT CHART PSM 2