**2 ROTATION MANIPULATOR (8051)**

by

VISHWATH KUMAR B S 18BEC1289

YOGEESHWAR S 18BEC1343

A project report submitted to

**Dr. M. JAGANNATH**

**SCHOOL OF ELECTRONICS ENGINEERING**

in partial fulfilment of the requirements for the course of

**ECE3003 – MICROCONTROLLER AND ITS APPLICATIONS**

in

**B.Tech. ELECTRONICS AND COMMUNICATION ENGINEERING**

****

**Vandalur – Kelambakkam Road**

**Chennai – 600127**

**JUNE 2020**

**BONAFIDE CERTIFICATE**

Certified that this project report entitled “**2 ROTATION MANIPULATOR”** is a bonafide work of  **VISHWATH KUMAR B S -18BEC1289, YOGEESHWAR S – 18BEC1343** who carried out the Project work under my supervision and guidance for **ECE3003 – MICROCONTROLLERS AND ITS APPLICATIONS.**

**Dr. M. JAGANNATH**

Associate Professor Senior

School of Electronics Engineering (SENSE),

VELLORE INSTITUTE OF TECHNOLOGY

Chennai – 600 127.

**ABSTRACT**

Manipulators with free joints are typically actuated mechanisms. In this project, we analyse the 2 link manipulator’s end effect and implement it with the help of 8051, keil uvision, proteus and scilab and find the deviation between the calculated and the actual values of the end effect.

**ACKNOWLEDGEMENT**

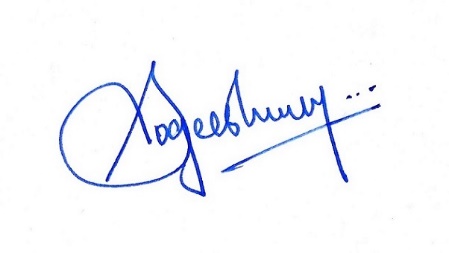
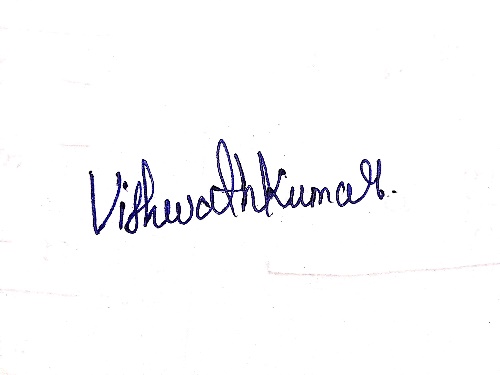
We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. M. JAGANNATH,** Associate Professor Senior, School of Electronics Engineering, for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. A. Sivasubramanian,** Dean of School of Electronics Engineering, VIT Chennai, for extending the facilities of the School towards our project and for her unstinting support.

We express our thanks to our Head of the Department **Dr. P. Vetrivelan** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.



**VISHWATH KUMAR B S YOGEESHWAR S**

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **SL. NO.** |  | **TITLE** | **PAGE NO.** |
|  |  | ABSTRACT | 3 |
|  |  | ACKNOWLEDGEMENT | 4 |
|  |  |  |  |
| 1 |  | INTRODUCTION | 7 |
|  |  |  |  |
|  | 1.1 | OBJECTIVES | 7 |
|  | 1.2 | BENEFITS | 7 |
|  | 1.3 | FEATURES | 7 |
|  |  |  |  |
| 2 |  | 2 ROTATION MANIPULATOR - DESIGN  FLOWCHART OF 2R MANIPULATOR | 8-16 |
|  | 2.1 | BLOCK DIAGRAM OF SYSTEM DESIGN | 8-9 |
|  | 2.2  2.3 | HARDWARE SPECIFICATIONS  SOFTWARE SPECIFICATIONS | 9-16  16-16 |
|  |  |  |  |
| 3 | 3.1  3.2  3.3 | ROTATION MANIPULATOR - IMPLEMENTATION AND ANALYSIS  SYSTEM IMPLEMENTATION  APPROACH  SOURCE CODE  PROTEUS IMPLEMENTATION | 17-24  17-18  19-23  24-25 |
|  |  |  |  |
| 4 |  | CONCLUSION AND FUTURE WORK | 26 |
|  | 4.1 | CONCLUSION | 26 |
|  | 4.2 | FUTURE WORK | 26 |

REFERENCES 27

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
| 1 | FLOWCHART OF 2R MANIPULATOR | 8 |
| 2 | BLOCK DIAGRAM FOR 2R MANIPULATOR IN SCILAB | 9 |
| 3 | ATMEL AT89C52 CHIP | 10 |
| 4 | 8051 PIN DIAGRAM | 11 |
| 5 | PIN DIAGRAM OF MG995 SERVO | 12 |
| 6  7  8  9  10  11  12  13  14  15  16  17 | DUTY CYCLE OF THE PWM FOR THE SERVO  4X4 KEYPAD  INTERNAL STRUCTURE OF THE 4X4 KEYPAD  16X2 LCD DISPLAY  INVERSE KINEMATIC FOR 2 LINK  LOOKUP TABLE FOR COORDINATES VS ANGLES  LOOKUP TABLE FOR DUTY CYCLE (ON PERIOD IN MS).  LOOKUP TABLE FOR TIMER MODE CALCULATION (Y) (ON TIME IN DECIMAL).  PROTEUS SETUP OF 2R MANIPULATOR  OUTPUT OF CORRECT CASE (SERVO ACTUATED).  SERVO READINGS  OUTPUT OF INCORRECT CASE (SERVO NOT ACTUATED) | 13  13  15  15  18  18  19  19  25  25  26  26 |

**List of Tables**

|  |  |  |
| --- | --- | --- |
| **Table No..** | **Title** | **Page No.** |
| 1  2  3 | PIN CONFIGURATION OF MG995 SERVO  PIN CONFIGURATION OF THE 4X4 KEYPAD.  PIN CONFIGURATION OF 16X2 LCD DISPLAY. | 11  13-14  15-16 |

**CHAPTER 1**

**INTRODUCTION**

**1.1 OBJECTIVES**

* Detect the input from the keypad (input points(X,Y))
* Parallelly display the given point on the LCD panel.
* To actuate the servos according to the given input points.
* To observe the end effect of the 2R manipulator and hence calculate the deviation from the calculated value.

**1.2 BENEFITS**

Our project proves that even 8051 being low on specifications, can be used as a powerful tool for plotting and graphing purposes. This automated actuation of the arms helps us swiftly plotting the points in a 2-D space using just 2 links and 2 rotations. Further improvement can prove much benefits in the area of robotics.

**1.3 FEATURES**

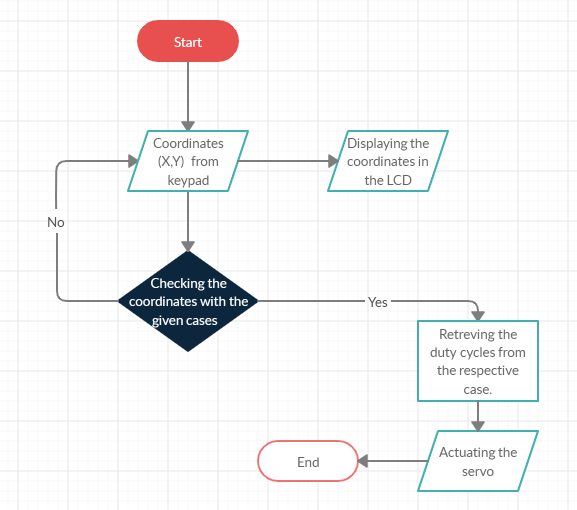
* MG995 High Torque Servo used for better performance and better accuracy.
* Keypad and LCD used for easy access and effective user interface.
* Microcontroller used is 8051.
* The efficient coding in the software tool called as Keil uVision5.

**CHAPTER 2**

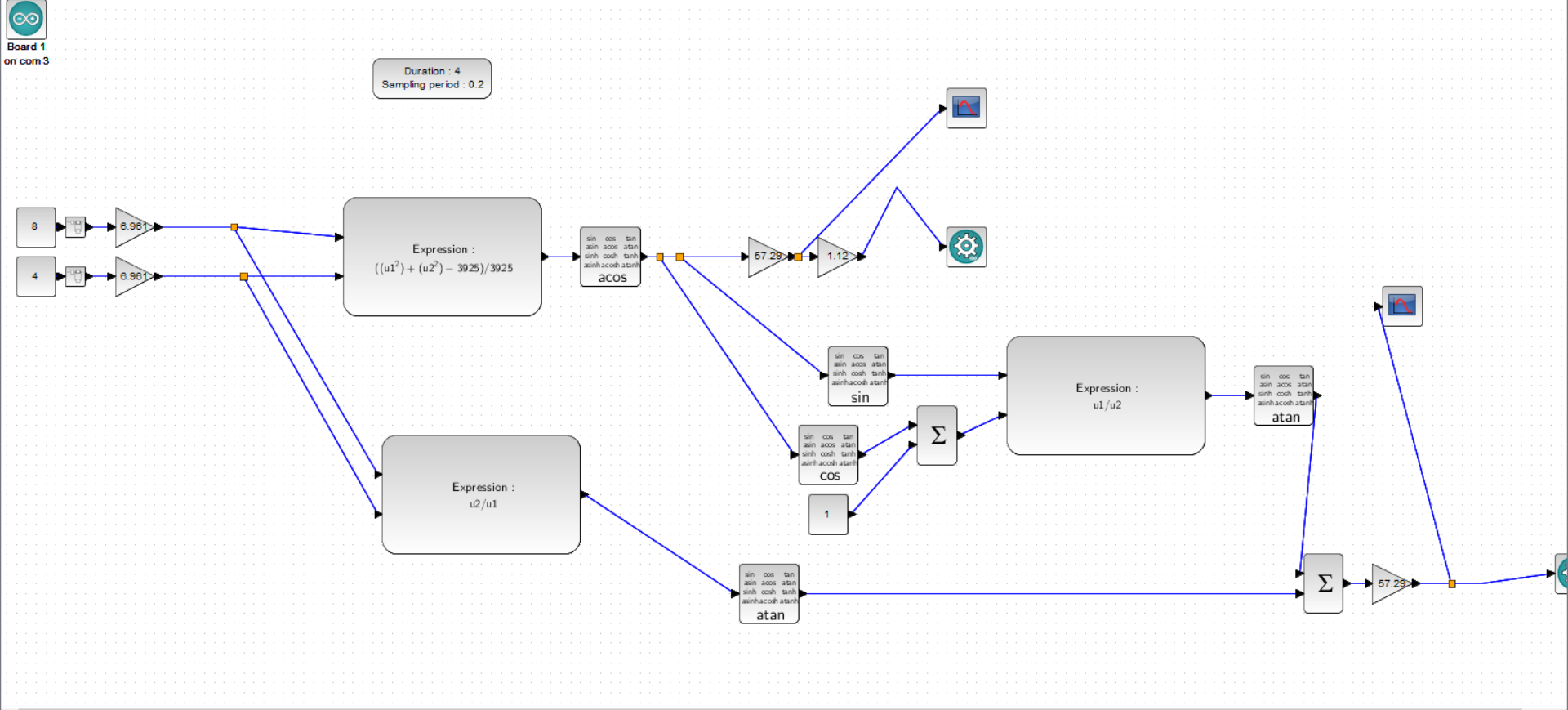
**2 ROTATION MANIPULATOR – DESIGN**

**2.1.1 FLOWCHART**

The work flow of the 2R manipulator using 8051 in assembly language is given in **Figure 1**.



**Figure 1.** **FLOWCHART OF 2R MANIPULATOR**

**2.1.2 BLOCK DIAGRAM**

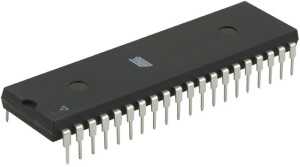
The detailed working of the 2R manipulator using 8051 in assembly language is given in **Figure 2**:

**Figure 2.** **Block diagram for 2R manipulator in SciLab**

**2.2 HARDWARE SPECIFICATIONS**

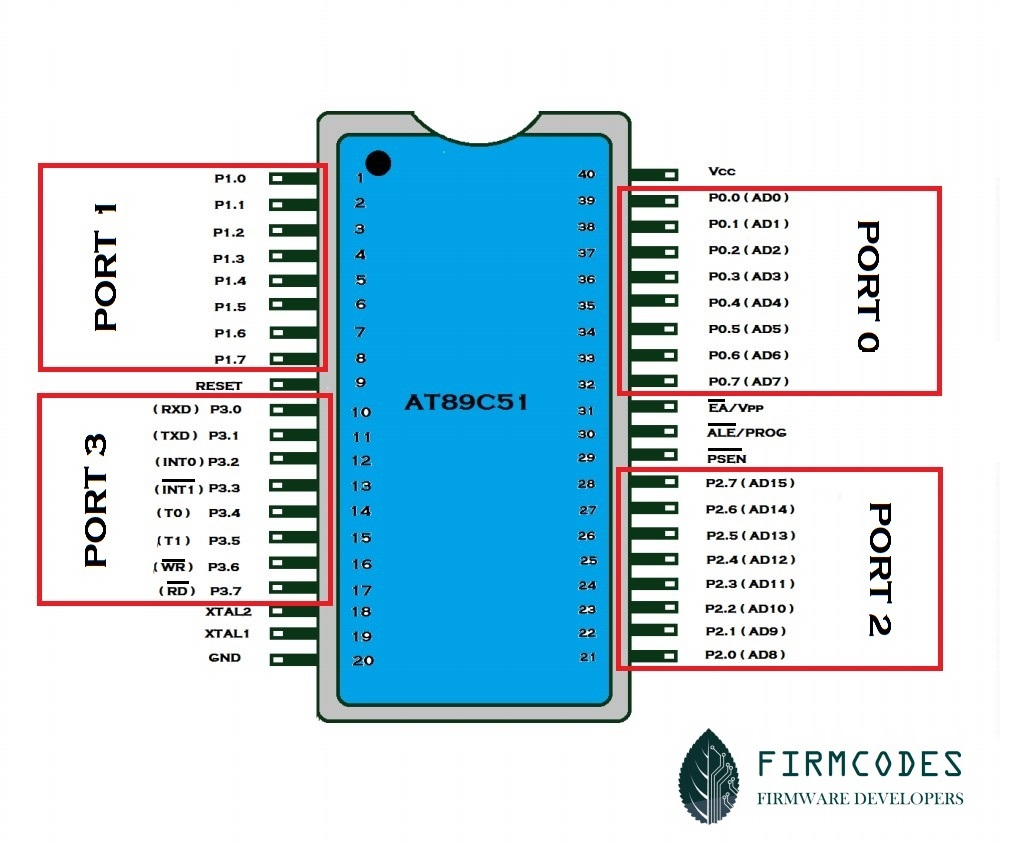
The prototype consists of :

* + The power supply,
  + Microcontroller 8051
  + 8051 Programmer
  + Actuating Rods (2nos)
  + MG995 15kg-cm high torque servo (2 nos)
  + 4x4 Matrix keypad
  + 16x2 LCD display panel
  + Push buttons
  + Capacitors, Resistors and Jumper wires

* + 1. **8051 MICROCONTROLLER**
* The AT89C51 [Refer **Figure 3**] is a low-power, high-performance CMOS 8-bit microcomputer with 4 Kbytes of Flash Programmable and Erasable Read Only Memory (EEPROM). The device is manufactured using Atmel’s high density non-volatile memory technology and is compatible with the industry standard MCS-51 instruction set and pin-out.[12]

**Figure 3.** **Atmel AT89c51 chip [8]**

* The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional non-volatile memory programmer. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C51 is a powerful microcomputer which provides a highly flexible and cost effective solution to many embedded control applications.[12]
* The AT89C51 provides the following standard features: 4Kbytes of Flash, 128 bytes of RAM, 32 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture [Refer **Figure 4**], a full duplex serial port, on-chip oscillator and clock circuitry. In addition, the AT89C51 is designed with static memory.[12]



**Figure 4.** **8051 pin diagram [9].**

**2.2.2 MG995 HIGH TORQUE SERVO**

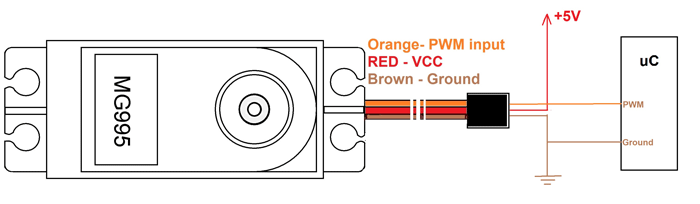
**MG995**is a **servo motor** that is popular for its performance and low price. The motor is used in many applications mainly being robotics and drones.[5]

### **Pin Configuration**

**MG995** has three terminals as mentioned in pin diagram [Refer **Figure 5**] and the function of each pin is given in the **Table 1**.

|  |  |  |
| --- | --- | --- |
| **Pin** | **Name** | **Function** |
| 1 | Signal pin (Orange pin) | The PWM signal which states the axis position is given through this pin. |
| 2 | VCC (Red pin) | Positive power supply for servo motor is given to this pin. |
| 3 | Ground(Brown pin) | This pin is connected to ground of circuit or power supply. |

**Table 1.** **Pin configuration of MG995 servo.[5]**



**Figure 5.** **Pin diagram of MG995 servo.[5]**

### **MG995 FEATURES AND ELECTRICAL CHARACTERISTICS [5]**

* Constant torque throughout the servo travel range
* Operating voltage range: 4.8 V to 7.2 V
* Stall torque: 9.4kg/cm (4.8v); 11kg/cm (6v)
* Operating speed: 0.2 s/60º (4.8 V), 0.16 s/60º (6 V)
* Rotational degree: 180º
* Dead band width: 5 μs
* Operating temperature range: 0ºC to +55ºC
* Current draw at idle: 10mA
* No load operating current draw: 170mA
* Current at maximum load: 1200mA

**MG995 Servo Motor Overview**

Since MG995 is a servo motor providing precise rotation over 180º range its applications are many and in them a few are stated below

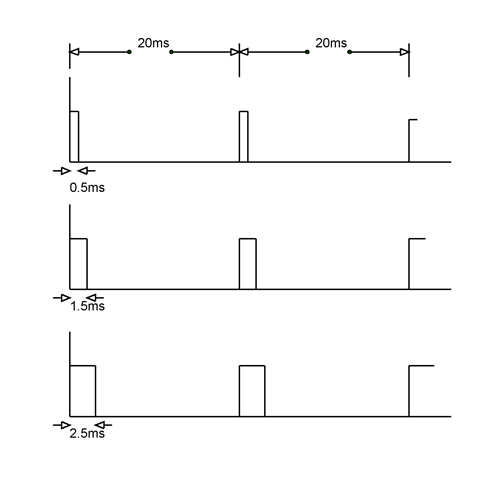
* The servo is suited for designing robotic arm in which wear and tear of motor is high. Being metal geared, the servo has long life and can be installed on system like robotic arm were motor work is huge.
* The servo is also suited to be used in drones and toy planes. Having a satisfying torque which is enough to overcome air resistance and control wings of plane, the servo is preferred in toy planes and drones which need precision control no matter the condition.[5]

**CONTROLLING MG995 WITH PWM**

* **Frequency of PWM:** MG995 takes in PWM signal of frequency 50Hz and any higher and lower frequency PWM will lead to error. As shown in figure the every single cycle of PWM needs to be 20ms width for 50Hz frequency.
* **Duty cycle of PWM:**The duty cycle of PWM (or ratio of ON time to total cycle time) determines the position of servo axis. If we provide a PWM signal of 0.5ms ON time over 20mS complete cycle, the servo axis will move to 0º.

And if we provide a PWM signal of 1.5ms ON time over 20mS complete cycle, the servo axis will move to 90º. At last if we provide a PWM signal of 2.5ms ON time over 20mS complete cycle, the servo axis will move to 180º. Refer **Figure 6** for the pictorial representation.

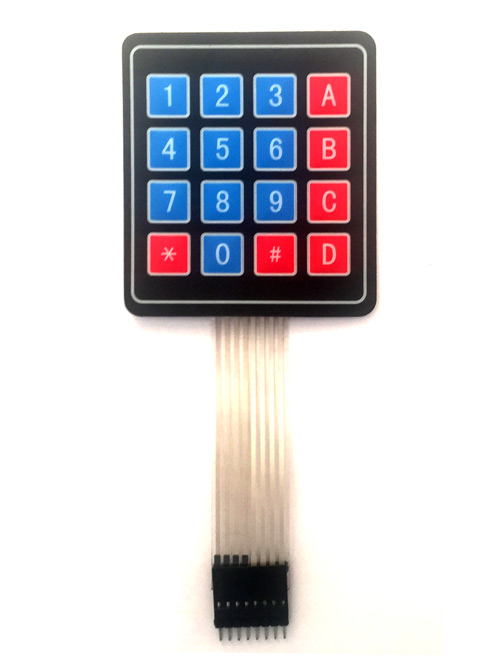
Based on these standard values we can also calculate any other degree of rotation. After calculation we just have to adjust the duty cycle of the PWM for the servo to read the signal and change to that stated position.[5]



**Figure 6****. Duty cycle of the PWM for the servo.[5]**

In this way we can control the servo position using PWM signal of any microcontroller or processor. In [arduino](https://components101.com/microcontrollers/arduino-uno)and other development boards we will have libraries readily available to make it even easier to control the servo. If we use those libraries we can directly state the servo position instead of adjusting the PWM duty cycle every single time.[5]

* + 1. **4X4 MATRIX KEYPAD**



**Figure 7****. 4x4 Keypad.[10]**

### **4X4 KEYPAD Pin Configuration:**

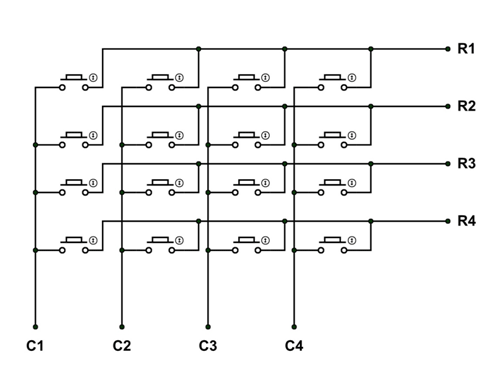
4X4 KEYPAD MODULES are available in different sizes and shapes. They all have same pin configuration. 16 buttons is arranged in 4X4 KEYPAD in matrix formation as you can see in the **Figure 7**.[10]

|  |  |
| --- | --- |
| **Pin Number** | **Description** |
| **ROWS** | |
| 1 | PIN1 is taken out from 1st  ROW |
| 2 | PIN2 is taken out from 2nd  ROW |
| 3 | PIN3 is taken out from 3rd  ROW |
| 4 | PIN4 is taken out from  4th ROW |
| **COLUMN** | |
| 5 | PIN5 is taken out from 1st  COLUMN |
| 6 | PIN6 is taken out from 2nd  COLUMN |
| 7 | PIN7 is taken out from 3rd  COLUMN |
| 8 | PIN8 is taken out from 4th COLUMN |

**Table 2.** **Pin configuration of the 4x4 Keypad.[10]**

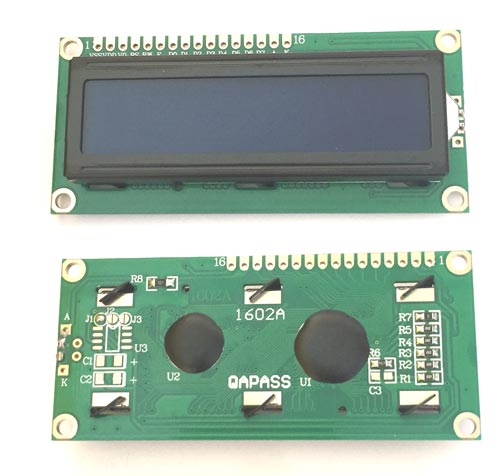
As given in **Table 2** a **4X4 KEYPAD** will have **EIGHT TERMINALS**. In them four are **ROWS of MATRIX** and four are **COLUMNS of MATRIX**. These 8 PINS are driven out from 16 buttons present in the MODULE. Those 16 alphanumeric digits on the MODULE surface are the 16 buttons arranged in MATRIX formation.[10]

The **internal structure of 4X4 KEYPAD MODULE** is shown in **Figure 8**.



**Figure 8.** **Internal structure of the 4x4 Keypad.[10]**

* + 1. **16x2 LCD DISPLAY PANEL**



**Figure 9.** **16x2 LCD display.[11]**

LCD modules [Refer **Figure 9**] are very commonly used in most embedded projects, the reason being its cheap price, availability and programmer friendly. Most of us would have come across these displays in our day to day life, either at PCO’s or calculators. The appearance and the pinouts have already been visualized above now let us get a bit technical. The pin configuration of the 16x2 LCD panel is shown in **Table 3**.[11]

### **Pin Configuration:**

|  |  |  |
| --- | --- | --- |
| **Pin No:** | **Pin Name:** | **Description** |
| 1 | Vss (Ground) | Ground pin connected to system ground |
| 2 | Vdd (+5 Volt) | Powers the LCD with +5V (4.7V – 5.3V) |
| 3 | VE (Contrast V) | Decides the contrast level of display. Grounded to get maximum contrast. |
| 4 | Register Select | Connected to Microcontroller to shift between command/data register |
| 5 | Read/Write | Used to read or write data. Normally grounded to write data to LCD |
| 6 | Enable | Connected to Microcontroller Pin and toggled between 1 and 0 for data acknowledgement |
| 7 | Data Pin 0 | Data pins 0 to 7 forms a 8-bit data line. They can be connected to Microcontroller to send 8-bit data.  These LCD’s can also operate on 4-bit mode in such case Data pin 4,5,6 and 7 will be left free. |
| 8 | Data Pin 1 |  |
| 9 | Data Pin 2 |  |
| 10 | Data Pin 3 |  |
| 11 | Data Pin 4 |  |
| 12 | Data Pin 5 |  |
| 13 | Data Pin 6 |  |
| 14 | Data Pin 7 |  |
| 15 | LED Positive | Backlight LED pin positive terminal |
| 16 | LED Negative | Backlight LED pin negative terminal |

**Table 3.** **Pin configuration of 16x2 LCD display.[11]**

**2.3 SOFTWARE SPECIFICATIONS**

* + Keil uVision5
  + Proteus Professional
  + SciLab

**CHAPTER 3**

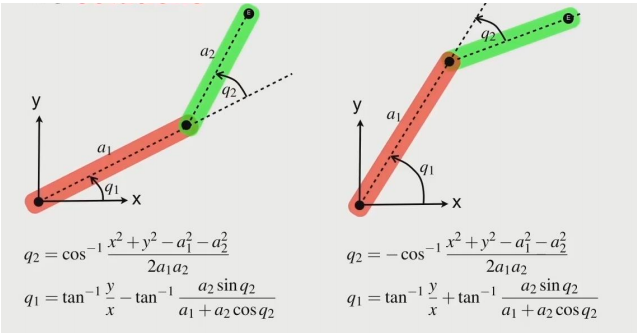
**SYSTEM IMPLEMENTATION AND ANALYSIS**

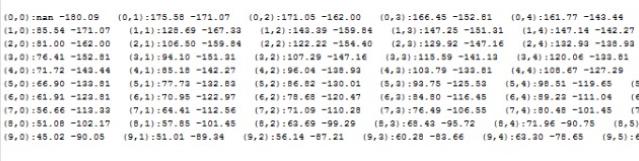
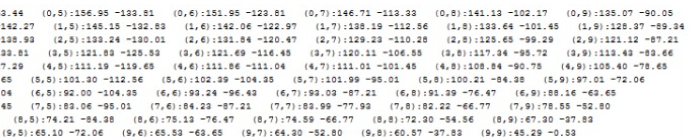
This section describes system implementation and results with inferences.

**3.1 APPROACH**

**3.1.1 INVERSE KINEMATICS**

The inverse kinematic equations are shown in **Figure 10.**



**Figure 10.** **Inverse kinematic equation for 2 link.[1]**

**Figure 11****. Lookup tables for coordinates vs angles.**

The respective angles for the given coordinates in the input are looked up from the

**Figure 11.**

**3.1.2. DUTY CYCLE CALCULATION**

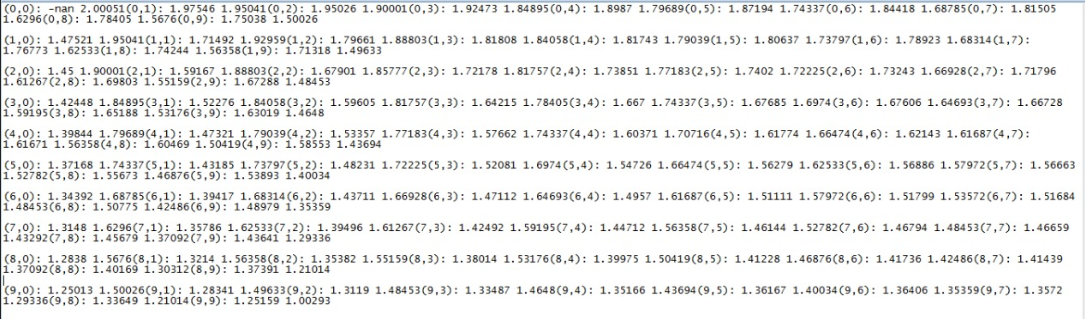
Duty cycle for 1st servo =[(q1/18)+2.5] [Refer **Figure 12** for further details]

Duty cycle for 2nd servo =[(q1+q2)/18+2.5] [Refer **Figure 12** for further details]

Delay for 1st servo = (Duty cycle for 1st servo\*20)/100 ms

Delay for 2nd servo = (Duty cycle for 2nd servo\*20)/100 ms

Range of the delay= 2.5ms-12.5ms (0-180 degrees)

Refresh rate – 20ms

**Figure 12****. Lookup table for duty cycle (ON period in ms).**

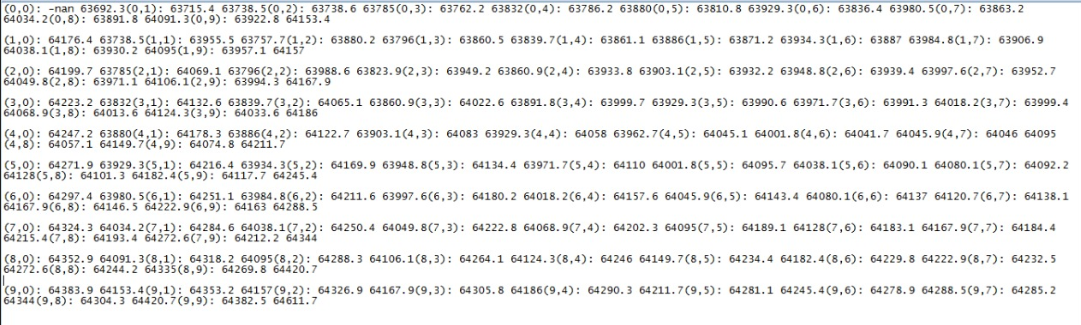
**3.1.3. INTIALISING TIMER PARAMETERS**

Timer mode -1

Step1: Delay/1.080806us = X

Step2: P=65536-X=Y

Step3: Y converted to hexadecimal =ABCD

Step4: TH1=0xAB; TL1=0xCD; [Refer **Figure 13** for further details]

**Figure 13.** **Lookup table for timer mode calculation (Y) (ON time in decimal).**

**3.2 SOURCE CODE**

**Assembly language code (in keil) :**

ORG 0

MOV DPTR,#MYCOM

C1: CLR A

MOVC A,@A+DPTR

ACALL COMNWRT

ACALL DELAY

JZ SEND\_DAT

INC DPTR

SJMP C1

SEND\_DAT:MOV DPTR,#LINE1

D1: CLR A

MOVC A,@A+DPTR

ACALL DATAWRT

ACALL DELAY

INC DPTR

JZ AGAIN

SJMP D1

AGAIN:MOV DPTR,#LINE2

MOV A,#0C0H

ACALL COMNWRT

X1: CLR A

MOVC A,@A+DPTR

ACALL DATAWRT

ACALL DELAY

INC DPTR

JZ AGAIN1

SJMP X1

AGAIN1:

MOV R7,#'X'

ACALL CHECK

MOV A,R7

MOV 60H,A

ACALL DATAWRT

MOV R7,#'X'

ACALL DELAY

MOV A,#','

ACALL DATAWRT

ACALL CHECK

MOV A,R7

MOV 61H,A

ACALL DATAWRT

MOV A,60H

CJNE A,#'6',CN1

MOV A,61H

CJNE A,#'1',CN2

ACALL CASE1 //6,1

CN1:

CN2:

MOV A,60H

CJNE A,#'9',CN3

MOV A,61H

CJNE A,#'3',CN4

ACALL CASE2 //9,3

CN3:

CN4:

MOV A,60H

CJNE A,#'2',CN5

MOV A,61H

CJNE A,#'8',CN6

ACALL CASE3 //2,8

CN5:

CN6:

MOV DPTR,#MYCOM

C12: CLR A

MOVC A,@A+DPTR

ACALL COMNWRT

ACALL DELAY

JZ SEND\_DAT1

INC DPTR

SJMP C12

SEND\_DAT1:

MOV DPTR,#FAIL

D12: CLR A

MOVC A,@A+DPTR

ACALL DATAWRT

ACALL DELAY

INC DPTR

JZ AGAIN12

SJMP D12

AGAIN12:SJMP AGAIN12

CASE1:

MOV 10H,#0FAH

MOV 11H,#0FBH

MOV 12H,#0F9H

MOV 13H,#0F1H

MOV 14H,#0C3H

MOV 15H,#014H

ACALL CASE\_PRIME

CASE2:

MOV 10H,#0FBH

MOV 11H,#032H

MOV 12H,#0FAH

MOV 13H,#0BAH

MOV 14H,#0C2H

MOV 15H,#014H

ACALL CASE\_PRIME

CASE3:

MOV 10H,#0F9H

MOV 11H,#0E3H

MOV 12H,#0FAH

MOV 13H,#06AH

MOV 14H,#0C3H

MOV 15H,#0B3H

ACALL CASE\_PRIME

CHECK:

LOOP1:

ACALL KEY1

ACALL KEY2

ACALL KEY3

ACALL KEY4

CJNE R7,#'X',LOOP

SJMP LOOP1

LOOP:

RET

KEY1:

clr p3.4

MOV A,p3

ANL A,#0FH

MOV r2,A

cjne r2,#14,n1

MOV r7,#'7'

n1: cjne r2,#13,n2

mov r7,#'4'

n2: cjne r2,#11,n3

mov r7,#'1'

n3: cjne r2,#7,n4

mov r7,#'X'

n4: lcall delay

SETB P3.4

RET

; Checking for Key Press on the Second Column of 4x4 Matrix

KEY2:

clr p3.5

MOV A,p3

ANL A,#0FH

MOV r2,A

cjne r2,#14,q1

mov r7,#'8'

q1: cjne r2,#13,q2

mov r7,#'5'

q2: cjne r2,#11,q3

mov r7,#'2'; A=65

q3: cjne r2,#7,q4

mov r7,#'0'

q4: lcall delay

SETB p3.5

RET

; Checking for Key Press On The Third Column of 4x4 Matrix

KEY3:

clr p3.6

MOV A,p3

ANL A,#0FH

MOV r2,A

cjne r2,#14,w1

mov r7,#'9'

w1: cjne r2,#13,w2

mov r7,#'6'

w2: cjne r2,#11,w3

mov r7,#'3'

w3: cjne r2,#7,w4

mov r7,#'X'

w4: lcall delay

SETB p3.6

RET

; Checking for Key Press on the Fourth Column of 4x4 Matrix

KEY4:

clr p3.7

MOV A,p3

ANL A,#0FH

MOV r2,A

cjne r2,#14,e1

mov r7,#'X'

e1: cjne r2,#13,e2

mov r7,#'X'

e2: cjne r2,#11,e3

mov r7,#'X'

e3: cjne r2,#7,e4

mov r7,#'X'

e4: lcall delay

SETB p3.7

RET

CASE\_PRIME:

HERE:

SETB P2.3

ACALL ON1

CLR P2.3

SETB P2.4

ACALL ON2

CLR P2.4

ACALL OFF

SJMP HERE

ON1:

MOV TMOD,#01H

MOV TH0,10H

MOV TL0,11H

SETB TR0

LP: JNB TF0,LP

CLR TR0

CLR TF0

RET

ON2:

MOV TMOD,#01H

MOV TH0,12H

MOV TL0,13H

SETB TR0

LP1: JNB TF0,LP1

CLR TR0

CLR TF0

RET

OFF:

MOV TMOD,#01H //1ST OFF-2ND ON

MOV TH0,14H // FFFF-2ND ON + 1ST OFF

MOV TL0,15H

SETB TR0

LOOP2: JNB TF0,LOOP2

CLR TR0

CLR TF0

RET

COMNWRT:

MOV P1,A

CLR P2.1

CLR P2.2

SETB P2.0

ACALL DELAY

CLR P2.0

RET

DATAWRT:

MOV P1,A

SETB P2.1

CLR P2.2

SETB P2.0

ACALL DELAY

CLR P2.0

RET

DELAY: MOV R3,#250

HERE2: MOV R4,#255

HE: DJNZ R4,HE

DJNZ R3,HERE2

RET

ORG 300H

MYCOM: DB 38H,0EH,01,06,80H,0

LINE1: DB "2R MANIPULATOR",0

LINE2: DB "(X,Y)-",0

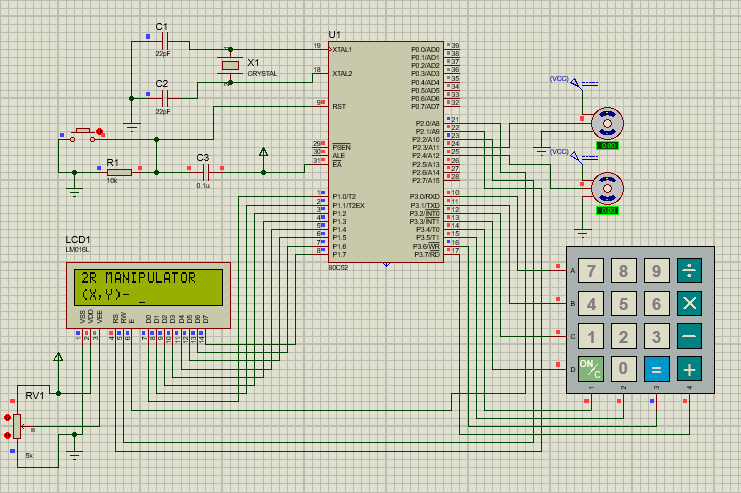
FAIL: DB "TRY AGAIN!",0

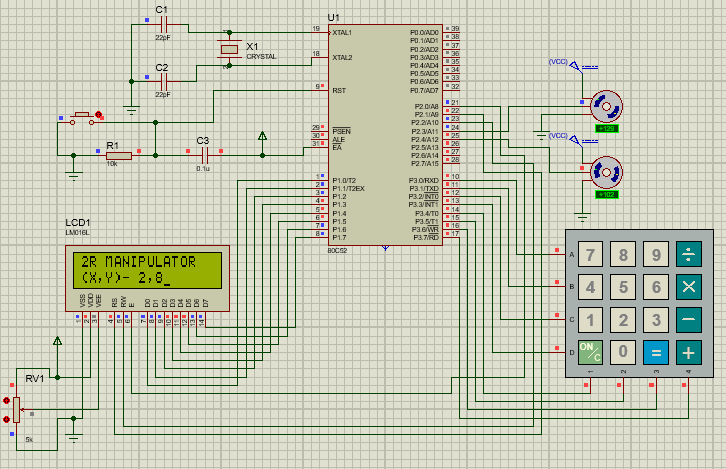
END

**3.3 PROTEUS IMPLEMENTATION**

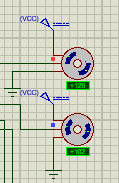
**Given Cases: (9,3), (2,8),(6,1)**

A specific case (2,8) is taken from the user input and is implemented through Proteus 8 Professional as shown in **Figure 14, Figure 15, Figure 16** and a failed case is shown in **Figure 17.**

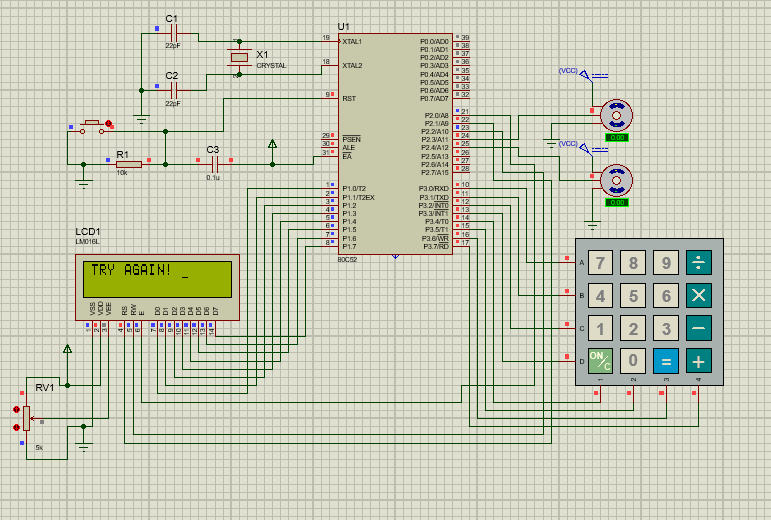
****

**Figure 14.** **Proteus setup of 2R Manipulator.**

**Figure 15.** **Output of correct case (Servo actuated).**



**Figure 16.** **Servo Readings.**



**Figure 17.** **Output of incorrect case (Servo NOT actuated).**

**CODE ANALYSIS**

* **Number of Lines: 266**

**ERROR ANALYSIS**

* **Error between the observed angle and the calculated angle in the proteus – 2.36%**

**CHAPTER 4**

**CONCLUSION AND FUTURE WORK**

**4.1 CONCLUSION**

Therefore for a given input from the keypad (X,Y),the specified point is displayed in the LCD panel and the corresponding duty cycle is calculated and hence the servo is actuated using Pulse Width Modulation, with the required delays being assigned to the timer parameters.

Thereby the servos are actuated and as the end effect, the rod being pointed to the specified input point.

**4.2 FUTURE WORK**

This project will be further improved by directly implementing the inverse kinematic equations in High level language thereby reducing the error significantly. We can even use high processing power microcontrollers or microprocessor to reduce the error.

This idea of the project can also be implemented for high-end manipulating devices as robotic arms (2 link).

**REFERENCES**

[1] Dr. M. JAGANNATH, Associate professor, VIT Chennai, 2020

[2] <http://www.sml.ee.upatras.gr/UploadedFiles/InverseKinematics.pdf>

[3] <https://link.springer.com/article/10.1186/s12938-017-0383-2>

[4]<https://www.researchgate.net/publication/224244534_A_new_geometrical_approach_to_solve_inverse_kinematics_of_hyper_redundant_robots_with_variable_link_length>

[5] <https://components101.com/motors/mg995-servo-motor>

[6] <https://www.electronicwings.com/sensors-modules/4x4-keypad-module>

[7] [https://en.wikipedia.org/wiki/Inverse\_kinematics - :~:text=In%20computer%20animation%20and%20robotics,the%20start%20of%20the%20chain.](https://en.wikipedia.org/wiki/Inverse_kinematics%20-%20:~:text=In%20computer%20animation%20and%20robotics,the%20start%20of%20the%20chain.)

[8] <https://www.indiamart.com/proddetail/8051-microcontroller-18928716988.html>

[9] <http://firmcode.blogspot.com/2014/12/introduction-of-8051.html>

[10]<https://components101.com/misc/4x4-keypad-module-pinout-configuration-features-datasheet>

[11] <https://components101.com/16x2-lcd-pinout-datasheet>

[12] <https://www.elprocus.com/8051-microcontroller-architecture-and-applications/>