# Development of Programmable

# Educational Articulated Robotic Arm

## B.E. SENIOR DESIGN PROJECT REPORT
## Electronics Specialization

**Prepared by**


| | |
|---|---|
| Safdar Mahmood | 3849 |
| Muhammad Amin Qureshi | 3837 |
| Raheel Masood | 3930 |




**College of Engineering**

**PAF – Karachi Institute of Economics and Technology**
**Karachi**

*This page was intentionally left blank*

# PAF - Karachi Institute of Economics & Technology

PAF Base Korangi Creek, Karachi-75190, Pakistan

## SDP COMPLETION DECLARATION

We the students of College of Engineering, batch Fall 2007, hereby declare that we have completed our final year project titled "Articulated Educational Robotic Arm" and have achieved all targets set forth in the project proposal.

It is requested that we may be examined.

| | NAME | REGISTRATION ID | SIGNATURE |
|---|---|---|---|
| 1. | Muhammad Amin Qureshi | 3837 | _____ |
| 2. | Raheel Masood | 3930 | _____ |
| 3. | Safdar Mahmood | 3849 | _____ |

**Confirmation by Project Supervisor:**

I _____, as project supervisor for the above mentioned group confirm that they have completed all requirements of the final project including the number of minimum visits for supervision, submission of all deliverables etc.

They may now be allowed to present and defend their work.

_____

Date & signature

**FOR OFFICE USE ONLY**

| | | | Remarks (If any) |
|---|---|---|---|
| 1. | Report submitted | ☐ Yes ☐ No | _____ |
| 2. | Project hardware & software fully functional | ☐ Yes ☐ No | _____ |
| 3. | All other deliverables received | ☐ Yes ☐ No | _____ |
| 4. | Presentation Date & Time | _____ | |

**Approved by Project Committee** ☐ Yes ☐ No

| | NAME | SIGNATURE |
|---|---|---|
| 1. Chairperson | Asst. Prof. Muhammad Abdul Aleem | _____ |
| 2. Member | Asst. Prof. Rehan Adil | _____ |
| 3. Member | Asst. Prof. MohazzabJaved | _____ |
| 4. Member | Lecturer ObaidRehmani | _____ |

**Approved by the Dean/Director:**

I certify, to the best of my knowledge, that the required procedures have been followed and the preparation criteria have been met for this project work.

Dated: _____ _____

**(Dean/Director CoE)**

**PAF - Karachi Institute of Economics & Technology**

PAF Base Korangi Creek, Karachi-75190, Pakistan

## SDP CERTIFICATE OF COMPLETION

This is to certify that the following students of College of Engineering have completed their Senior Design Project under the title **"*Articulated Educational Robotic Arm*"** in partial fulfillment of the requirement of the Bachelor of Engineering in Electronics with Telecommunications/Computer Systems/Industrial Electronics/Avionics specialization.

GROUP MEMBERS:

| | |
|---|---|
| Muhammad Amin Qureshi | 3837 |
| Raheel Masood | 3930 |
| Safdar Mahmood | 3849 |

**Project Advisor:**

_____

(Name &*Signature)*

**Senior Design Project Committee:**

| | | | |
|---|---|---|---|
| _____ | _____ | _____ | _____ |
| **Asst. Professor** | **Asst. Professor** | **Asst. Professor** | **Lecturer** |
| **M. Abdul Aleem** | **Rehan Adil** | **MohazzabJaved** | **ObaidRahmani** |
| **(Chairperson)** | **(Member)** | **(Member)** | **(Member)** |

_____

**Mr. S. NajeebHaiderJafri**

HoD

Dept. of Telecom.Engg.

College of Engineering

PAF – KIET

*To Our Parents*

*…..and Mentors*

# ABSTRACT

*Educational and training equipment cost too much when imported. We proposed a design of a 5-axis Articulated Robotic Arm Trainer (named IES Edu.bot v1), which is produced locally at relatively lower costs than those imported. We used sophisticatedly designed links made out of acrylic and high torque servo motors. This is relatively a different and an indigenous design. The mathematical modeling is done in MATLAB using standard forward kinematic convention of Denavit-Hartenberg parameters. These parameters are used to calculate the position of the end-effector when the joint angles are known. The desired joint angles are given as commands through the MATLAB GUI over the serial port to the Atmega128 microcontroller which then processes them and sends the output accordingly to the PWM channels to rotate the servo motors.Lab manuals have been compiled for giving students a clear understanding of the working of robotic arms. The text will guide students to perform operation efficiently, guiding them though each procedure.*

# KEYWORDS

Robotic Arm Trainer

5 DOF (Degrees of Freedom)

Link Coordinate Diagram

Home Position

Transformation Matrices

Forward Kinematics

Articulated Robotic Arm

Revolute Joint

Wrist Pitch and Roll

D-H Parameters (Denavit&Hartenberg Parameters)

Training Equipment

RC/PWM Servo Motor (Radio Control/Pulse Width Modulation)

Serial Communication

FBD (Free body Diagram)

# TABLE OF CONTENTS

# TABLE OF FIGURES

# LIST OF TABLES

# PROJECT OBJECTIVES

## Motivation

We have always had great passion for robotics. Our previous work indicates this passion, as we have made a Path Finding Robot indigenously, which has made us win and remain runners-up in several national level competitions.

The motivation for this very project for our Senior Design Project was given by our respected and beloved Internal Advisor, Asst. Prof. Mr. Rehan Adil, who not only invited us to accomplish Lab Trainers for the university's Robotics laboratory, but also helped us throughout in our tough times.

## Aim

Our aim, for this project, was to develop a Robotic Arm Trainer for the Robotics Laboratory which would be a low cost alternative to its counterparts. And another vision for this project is to initiate a step for high quality robotic works in Pakistan.

## Objectives

The objectives are as follows:

- Design and development of a 5-DOF, Programmable, Articulated Robotic Arm Trainer.
- To interface the Robotic Arm with MATLAB through serial port for user programming and matrix solving for the link coordinates.
- To use the servo motors for link joints.
- Compilation of User Guide.

- Creation of lab manuals for undergraduates to demonstrate the working of an Articulated Robotic Arm and providing them tasks to perform and observe its principals of operation.

## Results

The results have been as expected. We have successfully made an indigenous design for the Robotic Arm using acrylic sheets. This design is not only much cheaper than its current counterparts of the same price range, but also it is quite beautiful, since the cost did not trade off for its aesthetics, we have taken a very special care for that.

As for the working of the Robotic Arm, it is fully functional and working according to the expectations, and in fact, it has reached par our expectations. We expected a lot of jittering of servos and instability of the structure due to the use of thin sheets for joints, which made it lighter, but it is fair enough to demonstrate the working principals of Articulated Robotic Arms to the undergraduates.

The lab procedures provide undergrad students with the complete operation guidance and simple tasks to perform and then observe the principals on which the most versatile robotic arms work. This is because Articulated Robotic Arms make the most utilized type of robotic arms in the industry.

## Methodology Adapted

This Senior Design Project started from vagueness and travelling from a lot of turns and tough terrains, led us to the completion of our BE degree. We learned a lot of new things during these phases. This learning was not only academic in nature but we also came to know how to get a work done any way. We came to know about the real engineering workshops and wholesale markets in Karachi which are hidden in the

mist and glitter of this vibrant city. We met a lot of people from different fields and utilized their experiences and suggestions for achieving our goal. This helped us broadening our visions and enhancing our skills and capabilities.

We travelled more than a thousand kilometer on bike and car for purchasing components and parts for the project throughout the year. Having said all this, this effort is worth putting in. This effort is the essence of academic life, which sharpened us and made our hearts shine with the glare and illumination of knowledge, practice and hard work. The detailed break-down of different phases is given under.

**Initial/Proposal Phase**

This was the very start of our project and we were asked to submit a project proposal with the title and slight details of what we are going to do in it. At first, we were looking forward to go ahead with our previous semester project which was a Path Planning Robot, but we could not manage to find a sponsor for it eventually.

Then very soon we were approached by Asst. Prof. Mr. Rehan Adil who gave us the idea of Robotic Arm Trainer for PAF-KIET. We were amused by the idea because this is really something we were going to do – Robotics.

We proposed it to the SDP Committee of College of Engineering, who accepted it and we moved one phase ahead – to the literature review phase.

**The Literature Review Phase**

Since robotics is nothing easy and piece of cake, we had already realized this fact during our previous project, thus started reading rigorously and seriously.

We first downloaded project theses and research papers from **IEEE, Science Direct and E-brary,** some of the largest online research resources. We extracted a lot of great information related to our project from these research papers. We understood how people achieved their goals and how they play with equations and matrices.

We also searched for the data about commercially available Robotic Arm Trainers like **Scorbot** and pioneering Robotic Arms **like Rhino SR-3** and **SR-4**. The final and complete learning was gained through the books of Robotics, from which we learned about link coordinates and D-H parameters.

**The Planning, Procurement & Testing Phase**

After we had enough learning and knowledge to implement whatever we have got from the research papers and books, we started to plan about the servo motors we were going to use in the robotic arm. We searched for high-torque servo motors used in RC planes and ships, keeping in mind the cost should be low. We selected our motors and ordered them online. As soon as we received the motors through courier, we started testing their angle responses and repeatability. After this testing was done, we paid attention to the mechanical design. We chose acrylic sheets for the links because it is cheap and light. After getting the servo motors and the sheet for the arm links, we moved towards the mid-implementation phase.

**Mid-Implementation Phase**

We made AUTOCAD design for the cutting of acrylic sheets to make the links of the robotic arm, with the precise dimension of servo motors. We found a workshop where they cut the acrylic sheets by LASER. After getting that work done, we felt a need of gearing on the shafts of servo motors, so that our torque increases and shafts of the servo motors remain safe from breaking and bending under load conditions.

We developed some sketches of gears and found a place where they only make gears for industries and large machines. The suitable material for gears was chosen to be nylon, since its light-weight, cheap and can accommodate enough loads for our design specifications. The mid-implementation was a time consuming phase, because we had to change our design every now and then according to the developments and progress and also under the suggestions of technical experts we met. However, we successfully got what we needed and thought about. Everything was just about perfect, more than we expected.

Previously in our testing phase, we decided to run the servo motors using three Attiny2313 microcontrollers and one main controller Atmega8 as the brain and arbitrator. This was because we could not find a single microcontroller having 6-channels of 16-bit PWMs. This was not only bulky and complex but also it made our coding a monster-like thing which was nearly killing us. Luckily we got an Atmega128 from an individual. This controller comforted us and is now used as the only controller for PWMs and all other things which we were going to use in the project.

**The Final Implementation**

Now this was the phase which seemed very interesting to us, because we enjoyed the fitting things at their places and it was again as we expected it to be. We fitted the links, motors and gear. We got a box made out of wood for the platform and for the sake of carrying the actual robotic arm. There was also the need for extension wires for servo motors, we made them. The PCB of Atmega128's breakout board, the PCB of the main circuit board, the I/O module and the power supplies were made. We also got some proper tools and parts to make it look like a finished product.

## Market Adaptability

As we have mentioned earlier, Articulated Robotic Arms are integral for industries, thus the development of a robotic arm locally can be proved vital for providing the Pakistan industry a boost to manufacture robotic arms locally.

Also, it is not useless to mention about the Laboratory Trainers' industry in Pakistan, since we do not currently manufacture some good quality trainers here, this quality product-cum-project would be an initiative for manufacturing Lab Trainers locally.

We are planning to set up an enterprise for the production and manufacture of such trainers, though not being the pioneers.

# 1.0 INTRODUCTION

## 1.1 The Project

Universities in Pakistan are facing a lot of problems in introducing new engineering disciplines. Most of the problems are related to the setting-up of new labs in the related department. Training Equipment cost too much and need a lot of investment. Most of the training equipment is either costly or not available easily in Pakistan. For that reason, one has to look for it in the international market, which needs a huge amount of funds.

The problem areas which we have identified are mainly:

- High cost of training equipments, so new lab-setups need a lot of investment
- Equipments are rarely manufactured in Pakistan
- Imported training equipments are expensive

The project undertaken is to make lab training equipment for robotics and automation laboratory. The project was divided in several phases, as discussed in a later chapter.

In this project we tried to focus on constructing the robot with all kinematic modeling parameters and their analyses. Forward Kinematics and Link co-ordinate designing is the basis of constructing our robotic arm. All the basic parameters like Denavit-Hartenbergparametersare fully considered and have been put in an adequate table. Forward kinematics is done in the software made for basic controlling of our robotic arm. We took help from the textbooks of robotics for the mathematical modeling of the robotic arm and other design parameters.

Forward kinematics is the calculation of the position and orientation of the robot's manipulator as a function of its joint angles. This essentially contains transformation matrices between the frames of different links. The Denavit-Hartenberg convention for describing the forward kinematic matrices is quite famous. [1]

In this report we have presented adequate material about our project and its working. The chapter 2 deals with the project objectives and the issues we faced during the period. The chapter 3 includes complete design specifications from literature review to process flow and block diagram to schematics and PCB layouts. This chapter also presents Gantt chart of the project along with mechanical drawings along with design parameters and link coordinate diagrams. The chapter 4 is where we put all the tests and their results, and what we concluded from them. These tests were performed on the servo motors we are using and we calculated the PWM signal register values for these motors, and designed a formula for using in the code. There is an annexure or appendix too, which contains the source code of the project.

# 2.0 DESIGN OBJECTIVES, ISSUES&THEIR ANALYSES

## 2.1 Design Objectives

As discussed in previous section of project objectives, our design objectives were:

- Selection of high-torque, low-cost servo motors.
- Selection of suitable, light and cheap material for link joints.
- Designing of a mechanical structure which would demonstrate the working of an Articulated Robotic Arm efficiently, without any jittering.
- Selection of right microcontroller for configuring 16-bit PWM to get the best available resolution of servo motors.
- Implementation of the DH-parameters on the robotic arm.

## 2.2 Issues

The issues arose from time to time in the project, from planning to testing and to the final implementation phases. These issues are listed in bulleted form here:

- The motors were jittering and were non-repeatable. This issue was later resolved by replacing the servo motors.
- The idea of using 4 microcontrollers in the case of unavailability of a single microcontroller capable of handling all the desired I/Os seemed quite complex and obstructing. The pace of the project was slowed down until we got Atmega128.
- Implementation of PCB breakout board of Atmega128 SMT package (QFN). We made this PCB using the PCB prototyping machine in R&D lab of College of Engineering.
- Development of main board's PCB was quite expensive from market, so we made it on our own, using FeCl3 etching method.

# 3.0 DESIGN SPECIFICATIONS

## 3.1 Literature Review

In Pakistan, local companies produce a very less number of trainer equipment. One or two international companies were operating in Pakistan few years back but those companies have discontinued their work in the related field in Pakistan. One example is RIMS technology which is an international company. In Pakistan, none or very less number of companies are indulged in the development and production of trainer/educational lab equipments.

Internationally, there are many companies that are involved in this work. When talking specifically about programmable robotic arm trainers, there are many in the list making this equipment. We have studied some of them. Few of them produce low cost, less featured robotic arms for educational purpose and training; but few produce expensive and high quality richly featured robotic arms. The details are given in the following paragraphs.

Intelitek® produces two types of robotic arm trainer systems. One is called SCORBOT-ER-9pro and the other is SCOROBOT-ER-4U. SCORBOT-ER-4U has the following features.

**i) SCORBOT-ER-4U:** [2]

The SCORBOT-ER 4u robot is a versatile and reliable system for educational use. The Scorbot-ER4u robot arm can be mounted on a tabletop, pedestal or linear slidebase.

The robot's speed and repeatability make it highly suited for both stand-alone operations and integrated use in automated workcell applications such as robotic welding, machine vision, CNC machine tending and other FMS operations.

**Figure 3-1: SCORBOT-ER-4U**

The robot is supported by SCORBASE robotics programming and control software. Optional RoboCell 3D graphic software lets students design, create and control simulated industrial work cells, and provides dynamic simulation of the robot and work cell devices during position teaching and program execution.

SPECIFICATIONS

***SCORBOT-ER-4U:***

| | |
|---|---|
| Mechanical structure | • Vertically articulated |
| Degrees of freedom | • 5 rotational axes + gripper |
| Payload capacity | • 2.1 kg (4.6 lb) |
| Axis Range | • Axis 1: Base rotation: 310°<br>• Axis 2: Shoulder rotation: +130° / -35°<br>• Axis 3: Elbow rotation: ±130°<br>• Axis 4: Wrist pitch: ±130°<br>• Axis 5: Wrist roll: Unlimited (mechanically); ±570° (electrically) |
| Reach | • 610mm (24") end of gripper |
| Speed | • 700 mm/sec (27.6"/sec) |
| Standard gripper | • Servo motor, parallel fingers |
| Gripper opening | • 75 mm (3") without pads<br>• 65 mm (2.6") with pads |
| Feedback | • High resolution incremental optical encoder on each axis |

| | |
|---|---|
| | and gripper |
| Actuators | • 12 VDC servo motor on all axes and gripper |
| Transmission | • Gears, timing belts, lead screw |
| Weight | • 10.8 kg (23.8 lb) |
| Ambient operating temperature | • 2° - 40°C (36° - 104°F) |
| Additional features | • Roller bearing support on all axes<br>• Anti-backlash gearing system on base axis<br>• Robot connects to controller through single 50-pin cable<br>• Built-in pneumatic cabling enables use of pneumatic end effectors |

**Table 3-1: SCORBOT ER-4 specifications table**

**ii) SCORBOT-ER-9PRO:** [3]

The SCORBOT-ER 9Pro offers advanced robotic path control, speed and accuracy in an affordable, industrial grade robot.

With the new state of the art USB-Pro controller, the ER9Pro is provided with multi-tasking, real-time control and synchronization of up to 8 axes, 16 digital inputs, 16 digital outputs, 4 analog inputs and 2 analog outputs. The SCORBOT-ER 9Pro supports both stand-alone applications as well as sophisticated automated work-cells.

**Figure 3-2: Scorbot-ER-9 PRO**

SPECIFICATIONS

***SCORBOT-ER-9PRO:***

| | |
|---|---|
| Mechanical structure | • Vertically articulated<br>• Enclosed casting |
| Maximum Payload | • 4kg (8.8 lb) (with reduced acceleration)<br>• 2kg (4.4 lb)(Full speed) |

Axis Movement

| | Range | Effective Speed | Maximum Speed |
|---|---|---|---|
| Axis 1: Base rotation | 276° | 80°/sec | 140°/sec |
| Axis 2: Shoulder rotation | 153° | 69°/sec | 123°/sec |
| Axis 3: Elbow rotation | 214° | 78°/sec | 140°/sec |
| Axis 4: Wrist pitch | 202° | 103°/sec | 166°/sec |
| Axis 5: Wrist roll | 737° | 185°/sec | 300°/sec |

| | |
|---|---|
| Speed | • 1.9 m/sec (74.8"/sec) |
| Operating radius | • 691 mm (27.2") without gripper |
| Number of Axes | • 5 rotational axes and gripper |
| Repeatability | • ± 0.05 mm (0.002") |
| Position Feedback | • Incremental optical encoders with index pulse on each axis |

| | |
|---|---|
| Actuators | • 24VDC servo motor on each axis |
| Transmission | • Harmonic drives and timing belts |
| Weight | • 51 kg (112.5 lb) |

**Table 3-2: SCORBOT ER-9PRO's specifications table**

**iii) ARISTO 6XT:** [4]

Another company MTAB produces similar products. MTAB is an Indian company involved in the making of educational equipment and trainers. They have three types of robotic arm. One of them is SCARA. The articulated robotic arms are named as ARISTO 6XT and MINI robot. Since our programmable robotic arm will be an articulated one, so we reviewed only those two related products. The summarized detail according to their product specification is reproduced below:


**Figure 3-3: ARISTO 6XT**

| **SPECIFICATIONS** | **ARISTO** | 6XT |
|---|---|---|
| Gripper | : | Pneumatic / Electrical |
| Link 1 | : | 300 mm |
| Link 2 | : | 400 mm |
| Joint Actuators | : | DC Servo geared motors |
| Vertical Height | : | 400 mm |
| Transmission | : | Joint 1 : Gear Train<br>Joint 2 & 3 : Ball screw<br>Joint 4, 5 & 6 :Timing belt drive |
| Position Feedback | : | Optical Encoder (HP 2 phase 500 PPR) |

| | | |
|---|---|---|
| Gravity Compensation | : | 100% (Non-back drivable ball screw) |
| No. of Axes | : | 6 (3axes waist-shoulder-elbow manipulator with 3 axes Roll- Pitch-Roll spherical wrist) |

**Mechanical**

| | | |
|---|---|---|
| Payload | : | 3 Kg including gripper |

**Joint Motion**

| | | |
|---|---|---|
| Joint 1 | : | 300 degree |
| Joint 2 | : | 60 degree |
| Joint 3 | : | 60 degree |
| Joint 4 | : | 300 degree |
| Joint 5 | : | 180 degree |
| Joint 6 | : | 300 degree |

**Table 3-3: ARISTO 6XT's specifications table**

**iv) RHINO XR-4 Educational Robotic Arm:**[5]

A five axis robot arm popular with those educators who need a rugged semi-enclosed design suitable for students at university level. The XR-4 robotic arm, developed by Rhino Robotics, is a semi-enclosed five axis design with all completely independent axes that can be controlled simultaneously. Powered by six permanent-magnet, direct-current (PMDC) servo motors with integral gearboxes and incremental encoders, the arm executes real time closed loop operations.

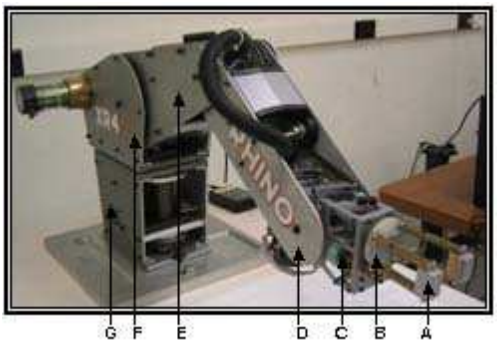

**Figure 3-4: RhinoXR-4 links**

In addition, the XR-4 is pre-programmed to return to its home position every time upon startup. This function is used to initialize the

incremental encoders which provide feedback information to the controller. The following describe the mechanical parts and controllers of the robotic arm.

**Parts of the XR-4:**

The basic components are:

- The manipulator linkage
- Actuators and transmissions
- Sensors
- Controllers
- User interface

**The manipulator linkage**

There are five manipulator linkages including the end effector which is a gripper connected through rotational joints. The regional and orientational structure of the manipulator produces a roughly spherical workspace. As shown in above, letters A, B, C, D, E, F, and G represent the different links of the manipulator.

**Actuators and transmissions**

The XR-4 uses electric actuators which are suitable for high speed, low load applications such as surface polishing, force control applications and movements requiring precision.

**Sensors**

Each motor is equipped with an incremental encoder which provides accurate joint position information to the feedback controller. Also, limit switches protect the XR-4 from moving its end effector to a position outside of its workspace. The end effector's limit switch also prevents damage to the object gripped.

**Controllers**

The Mark IV Controller, which is also developed by Rhino Robotics Ltd., integrates the power supplies, communication, and microprocessor logic, teach pendant support,

input/output capability and a software language. Connected to the XR-4, the controller uses proportional, integral and derivative (PID loop) control algorithms for full speed control of the robotic arm encoders.


**Figure 3-5: Mechanical links corresponding to Mark IV Controller labels**

Two main input ports allow a computer and a teach pendant to be connected to the controller as shown in Figure.

There are also eight line input pairs, eight switches and eight output line pairs to allow external devices to be connected to the controller. The line pairs are labeled 1 thru 8, while the switches are from 9 thru 16. Each input pair is connected to a LED bulb that lights up when input current is detected.

**User Interface**

RoboTalk is the programming software used to send commands to the XR-4 robotic arm and interpret signals that are received from each encoder. Electrical current signals from the line pairs and switches, caused by interaction between the XR-4 robotic arm and other external devices, are also received by RoboTalk. Also, the software provides built in commands that move the XR-4 end effector to desired positions, specified by either Cartesian positions or encoder counts for each of the manipulator joints.

Also, RoboTalk provides a feature to record the encoder counts for each mechanical link for a particular position within the XR-4 workspace. Hence the exact position desired for the end effector can be easily determined for each movement by using the teach pendant to position the end effector at the desired location.

The teach pendant, attached to the Mark IV Controller allows users to operate the XR-4 robotic arm without writing code with a programming software. This is extremely useful to maneuver the end effector to a specific location without requiring the user to know the Cartesian positions or the encoder counts for each mechanical link, prior to the move.

## 3.2 Process Flow Diagram

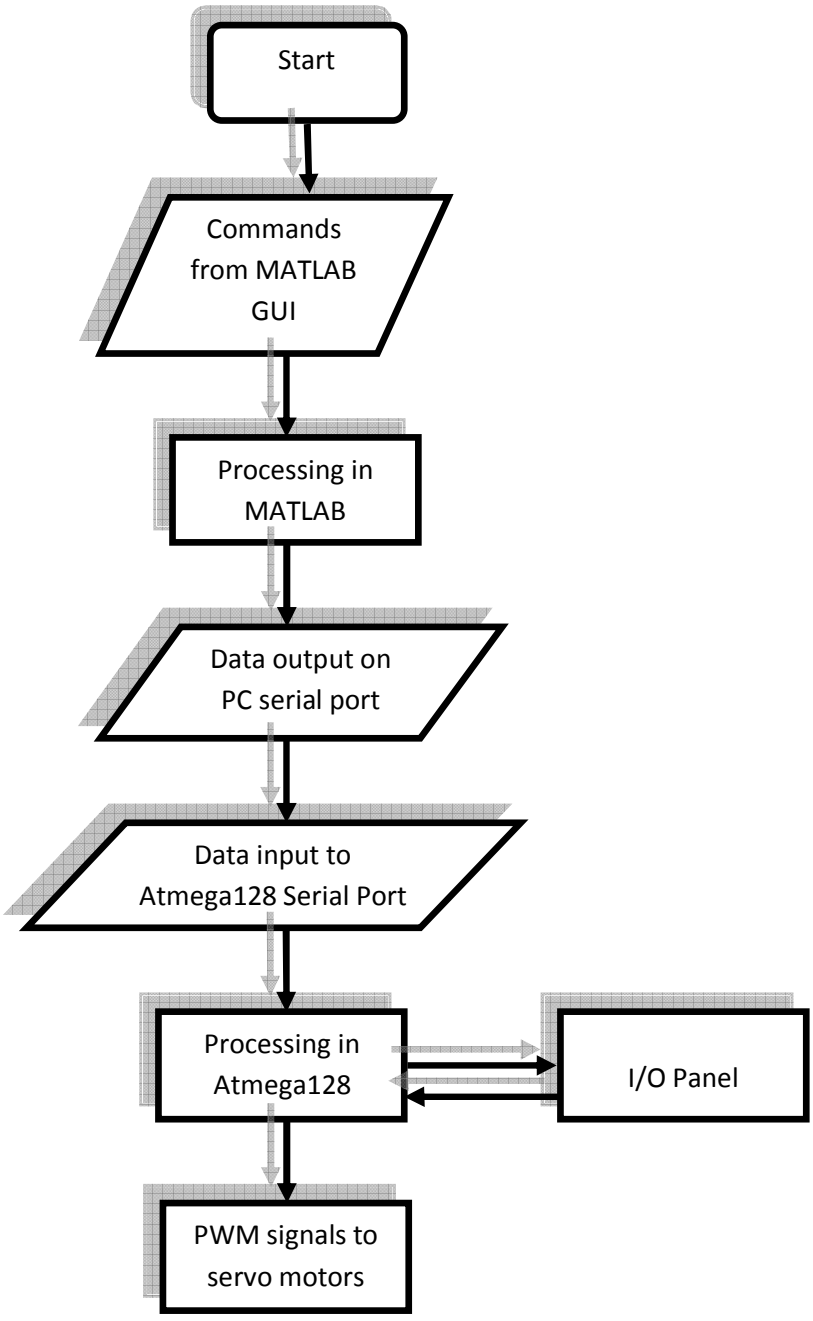


**Figure 3-6: Process flow chart for the robotic arm**

## 3.3 Block Diagrams

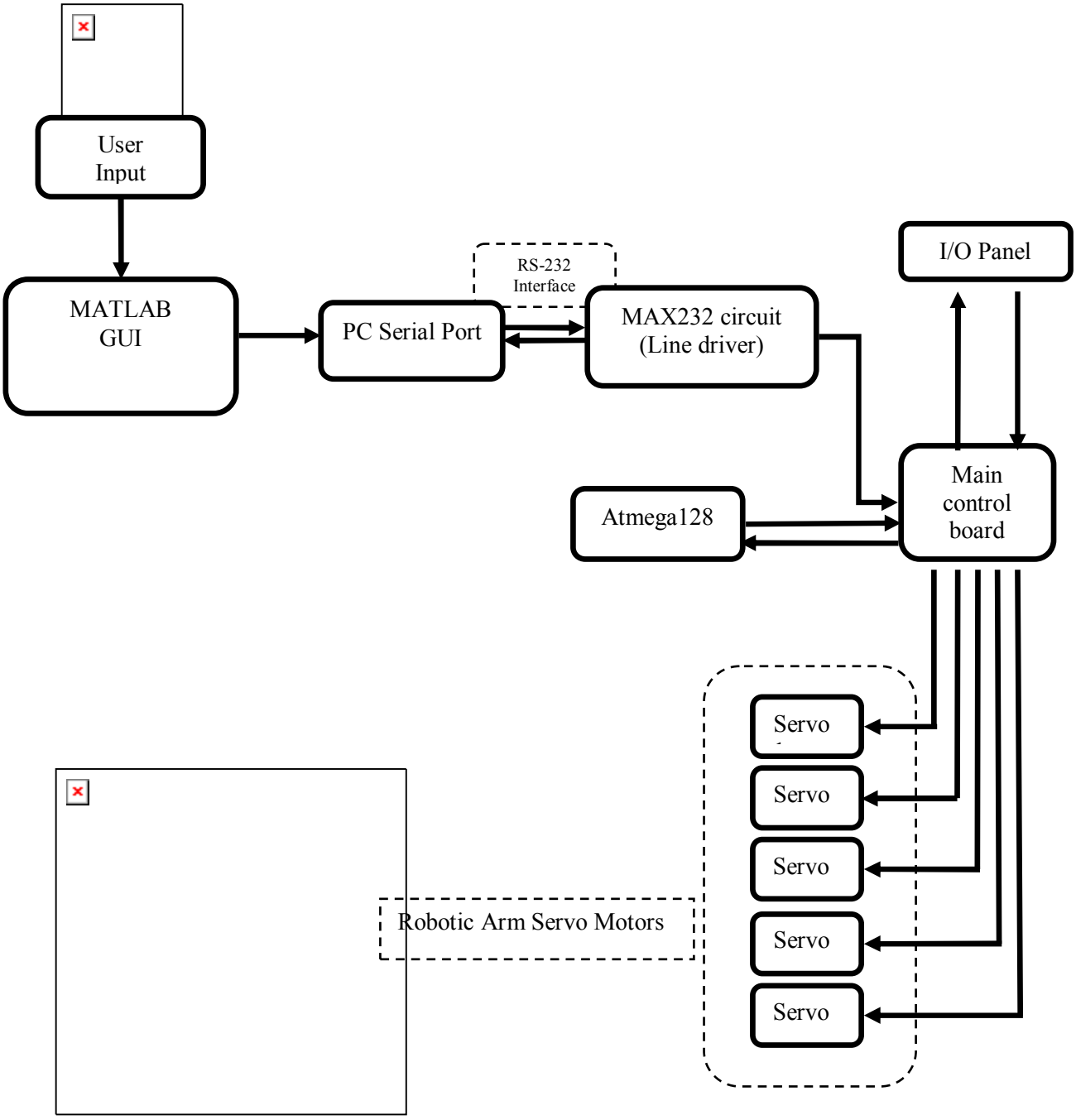


**Figure 3-7: Block diagram of the robotic arm trainer**

## 3.4 Design Parameters

5-AXIS ARTICULATED ROBOTIC ARM
**IES Edu.bot v1.0**

### 3.4.1 Link Coordinate Diagram

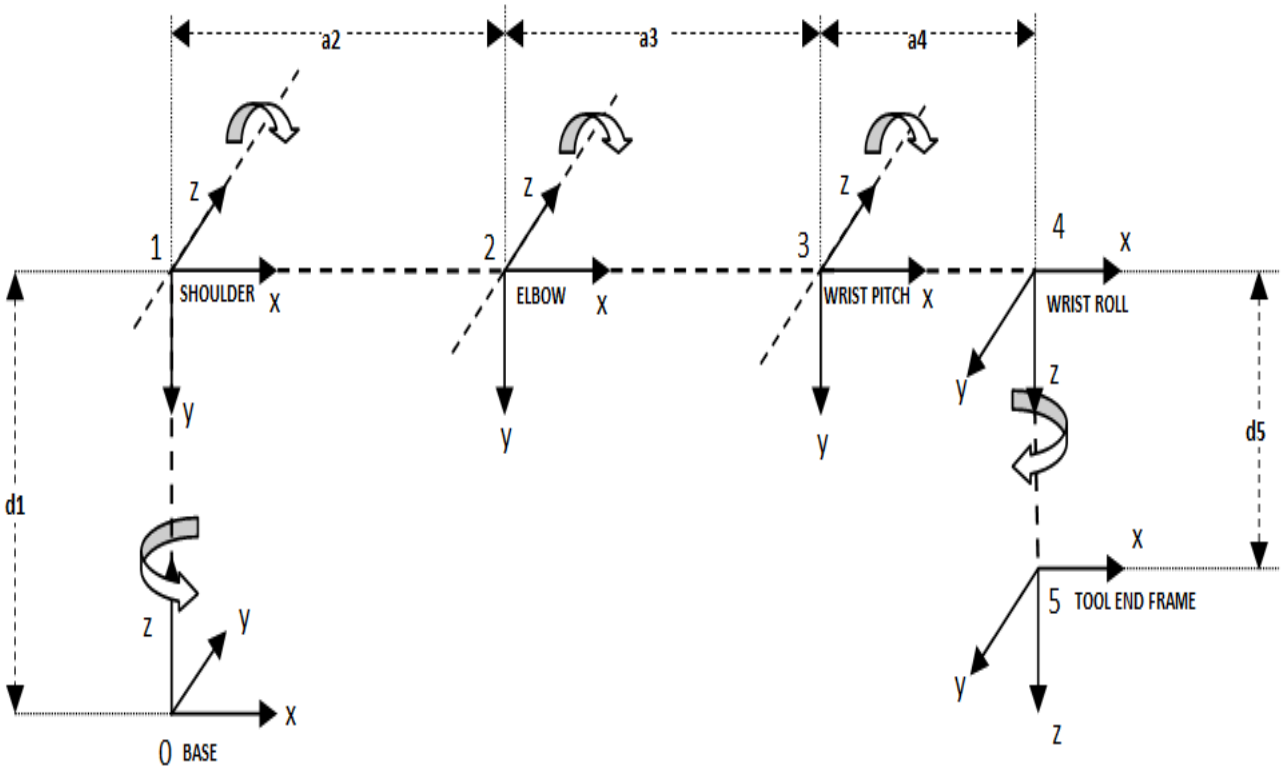


**Figure 3-8: Link coordinate diagram of 5-axis articulated robotic arm**

This Link co-ordinate diagram is the primary diagram with all the links rotated at 0 degrees. The Axis of rotation is taken at z-axis for each frame in all revolute joints. The arm starts from the frame # 0 as the fixed frame at base. Frame # 01 can be rotated along z-axis of the base frame. Similarly, the link # 02 can be rotated along z-axis i.e. the shoulder frame's z–axis. Link # 03 can be rotated along z-axis of the Elbow frame, and so on.

### 3.4.2 Link Coordinate Diagram (Home Position Convention)

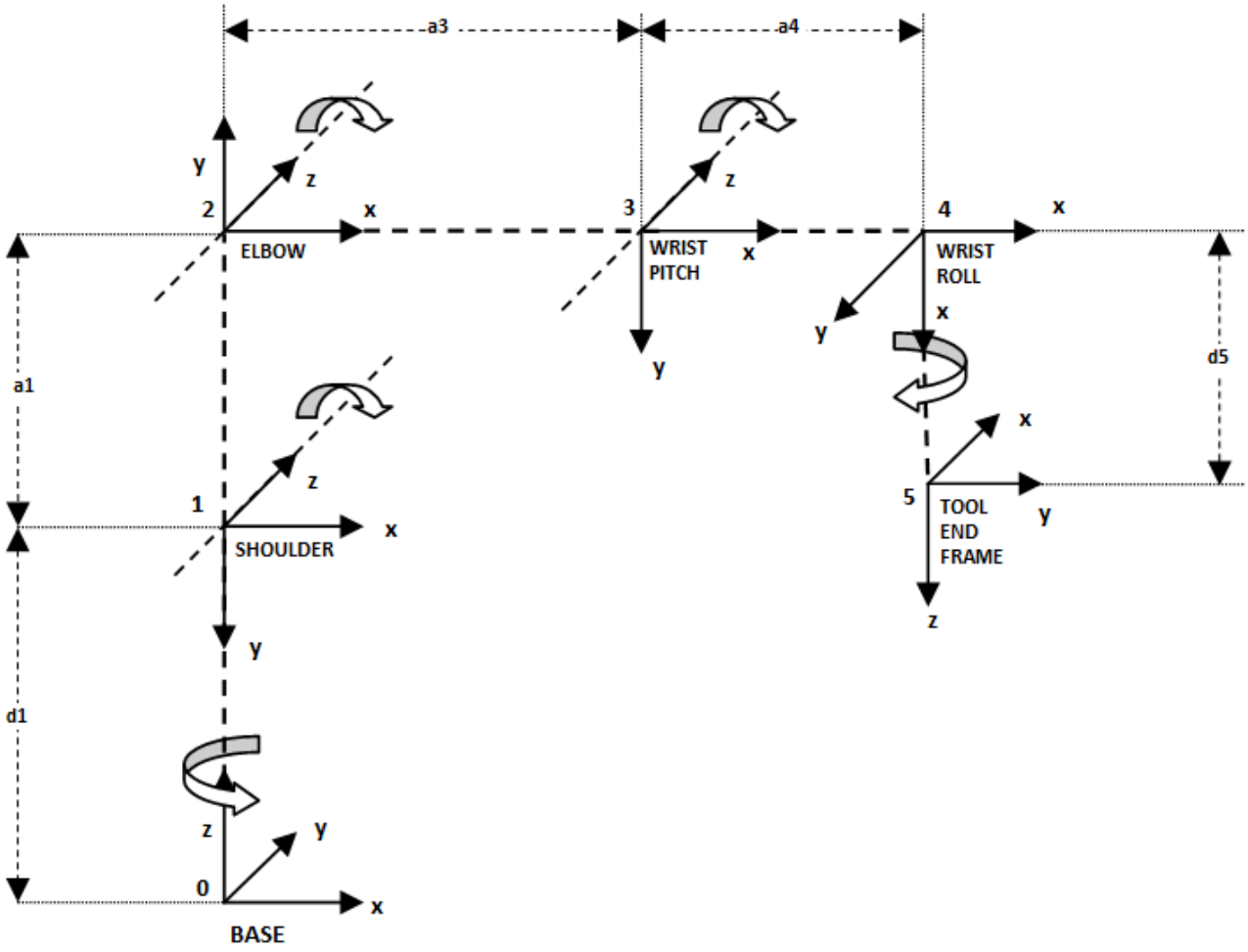


**Figure 3-9: Link coordinate diagram for home position convention**

In this diagram, the home position of the edu.bot v1.0 is shown. The arm's link # 02 has been rotated at $-\pi/2$ along the z-axis of the shoulder, the link # 03 is rotated at $\pi/2$ along the z-axis of the elbow joint. Similarly, the tool has been rotated at $-\pi/2$ along the z-axis of the wrist roll joint. The resulting diagram is thus shown above.

### 3.4.5 Denavit-Hartenberg Parameters for Edu.bot:

| Axis | ø | d (mm) | A (mm) | α | Home |
|---|---|---|---|---|---|
| 1 | q1 | 117 | 0 | -π/2 | 0 |
| 2 | q2 | 0 | 196.00 | 0 | -π/2 |
| 3 | q3 | 0 | 148.9 | 0 | π/2 |
| 4 | q4 | 0 | 2.5 | -π/2 | 0 |
| 5 | q5 | 76 | 0 | 0 | -π/4 |

**Table 3-4: D-H parameters table according to the link co-ordinate diagram**

## 3.5 The Transformation Matrices

### 3.5.1 General Transformation Matrix

The following matrix is a general transformation matrix between two frames. K-1 is the joint or frame previous to the joint k.

$$T_{k-1}^{k} = \begin{bmatrix} C\theta_k & -C\alpha_k S\theta_k & S\alpha_k S\theta_k & a_k C\theta_k \\ S\theta & C\alpha_k C\theta_k & -S\alpha_k C\theta_k & a_k S\theta_k \\ 0 & S\alpha_k & C\alpha_k & d_k \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where

$\alpha$ = twist angle

$\theta$ = joint angle

*d, a =link length parameters*

*where: d= length (distance) between two parallel **x-axis** of two frames*

*a= length (distance) between two parallel **z-axis** of two frames*

In specific terms, for example: Transformation matrix for representing shoulder frame on base frame will be like following:

$$T_{base}^{shoulder} = \begin{bmatrix} C\theta_k & -C\alpha_k S\theta_k & S\alpha_k S\theta_k & a_k C\theta_k \\ S\theta & C\alpha_k C\theta_k & -S\alpha_k C\theta_k & a_k S\theta_k \\ 0 & S\alpha_k & C\alpha_k & d_k \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

By substituting the parameter values in the above matrix we will get:

OR $$T_{base}^{shoulder} = \begin{bmatrix} C\theta_1 & 0 & -S\theta_1 & 0 \\ S\theta & 0 & C\theta_1 & 0 \\ 0 & -1 & 0 & 117 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

*Where **k** represents the parameters assigned for the **link k** which is between the fixed base frame and moveable shoulder frame w.r.t. base frame.*

### 3.5.2 Transformation matrices of EDU.BOTv1.0

Now if we consider our EDU.BOTv1.0 the whole transformation from the tool end with respect to the fixed base frame will be like:

$$T_{base}^{tool} = T_{base}^{shoulder} * T_{shoulder}^{elbow} * T_{elbow}^{wristpitch} * T_{wristpitch}^{wristroll} * T_{wristroll}^{tool}$$

Where:

d=[117, 0, 0, 0, 76+tool_length];
a=[0 196 148.9 2.5 0];
alpha=[-90 0 0 -90 0];

$$T_{base}^{shoulder} = \begin{bmatrix} C\theta_1 & 0 & -S\theta_1 & 0 \\ S\theta_1 & 0 & C\theta_1 & 0 \\ 0 & -1 & 0 & 117 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$T_{shoulder}^{elbow} = \begin{bmatrix} C\theta_k & -S\theta_k & 0 & 196*C\theta_k \\ S\theta_k & C\theta_k & 0 & 196*S\theta_k \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$T_{elbow}^{wristpitch} = \begin{bmatrix} C\theta_k & -S\theta_k & 0 & 148.9*C\theta_k \\ S\theta_k & C\theta_k & 0 & 148.9*S\theta_k \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$T_{wristpitch}^{wrisroll} = \begin{bmatrix} C\theta_k & 0 & -S\theta_k & 2.5*C\theta_k \\ S\theta_k & 0 & C\theta_k & 2.5*S\theta_k \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$T_{base}^{shoulder} = \begin{bmatrix} C\theta_k & -S\theta_k & 0 & 0 \\ S\theta & C\theta_k & 0 & 0 \\ 0 & 0 & 1 & 76+d_k \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

So, the final transformation matrix can be found by multiplying these matrices in order of the frames assigned, as follows:

$$T_{base}^{tool} = T_{base}^{shoulder} * T_{shoulder}^{elbow} * T_{elbow}^{wristpitch} * T_{wristpitch}^{wristroll} * T_{wristroll}^{tool}$$

## 3.6 Design Matrix


**SENIOR DESIGN PROJECT-II**

**Student #1: Safdar Mahmood** **Student #2: Muhammad Amin Qureshi** **Student #3: Raheel Masood**

---

### Programmable Robotic Arm

**Table 3-5: The design matrix**

| S.No. | Objectives | Modules | Components | Alternatives | Justifications | Functionality | Input | Output |
|---|---|---|---|---|---|---|---|---|
| 1 | **Structural Design of Robotic Arm and assembling motors** | N / A | PWM Servo Motors, Structural Material | DC Motors with Quad-encoders + PID motion and position controlling | DC motors with encoders are not used because this will add another module to be made. So designing control circuit module for motors will cost more time. | The revolute joints of the arm will be able to move due to these motors. The motors are actually the actuators of all revolute axis of the arm. | PWM signal according to the desired angle response | Revolute Actuation of each axis in terms of angular degrees according to the PWM signal |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **2** | **Testing Linearity and Repeatability of motors.** | Angle response testing module | Attiny2313, Breadboard based testing module, ISP programmer, PWM servo motors | Off-the-shelf Servo Angle Tester | Motors should be tested for linearity because the output should be consistent with input. In robotic arms angular movements are mainly important. | The module outputs PWM signal from the controller and angle response is measured by a 360 degree protector mounted on motor's shaft. | Predefined angle values set in the controller | Angular motion output from the motor |
| **3** | **Development of Servo motors modules** | Motor control module | Atmega128, Max232 for RS-232 communication | PIC microcontrollers can be used instead | Atmel chips are relatively low cost and I.Cs. Moreover we have hand-on experience with ATMEL 8-bit microcontrollers. | The module will basically generate the PWM signals for servo motors according to the input angle value. | Control signals from PC | PWM signal |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **4** | **Physical structure** | N/A | Gears, Acrylic based links, bearings and shaft assemblies and other | Ungeared links with no bearings could have been used | Gears and bearing are used for shifting the robotic arm's weight to the gears and bearing from motors' shaft and gears. If this would have not designed like that, the motors' wear and tear rate would be much increased. | Motors output shaft are fitted with the driving gear and the other gears are fitted with the link. The respective shaft gears and link gears are then coupled to make the moveable assembly. | Servo motors' mechanical force | Mechanical output in terms of angles. |
| **5** | **Development of External I/O module** | User I/O module | 2n2222 transistors, LEDs, connectors, DPDT switches | Buffers ICs could have been used in place of transistors | Making of I/O modules with buffer ICs could have increase the scope. The transistor based 4-bit I/O panel is not intended for use at long distance I/O functions but for on-trainer interfaces. | The module will provide the interface between the PC software and any sensors' assembly used on workspace of robotic arm. This will actually be connected to the Control Board. | Parallel 4-bit input | Parallel 4-bit output |
| | | | | | | | | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **6** | **PC Software Design** | N / A | Matlab R2009a | Labview | Mathematical calculations and computations can easily be performed on MATLAB, complex computation of matrices are real easy and fast to be made on MATLAB | It will provide the GUI and programming interface of the robotic arm. | User programming commands | Commands to Control Board |
| **7** | **Compilation of USER GUIDE and LAB Practical** | Booklets containing USER GUIDE and LAB Practical | N/A | N/A | User's guide is always necessary for professional equipment so that it can be more user friendly and easy to use. Lab practical will be provided as a part of this project since this is a Lab trainer. | Students using the trainer will be performing lab practical according to it. These very basic practical will be helpful for the teacher to practically demonstrate the theoretical lectures covered in class. | N/A | N/A |

## 3.7 Schematics

### 3.7.1 The Serial Communication Module

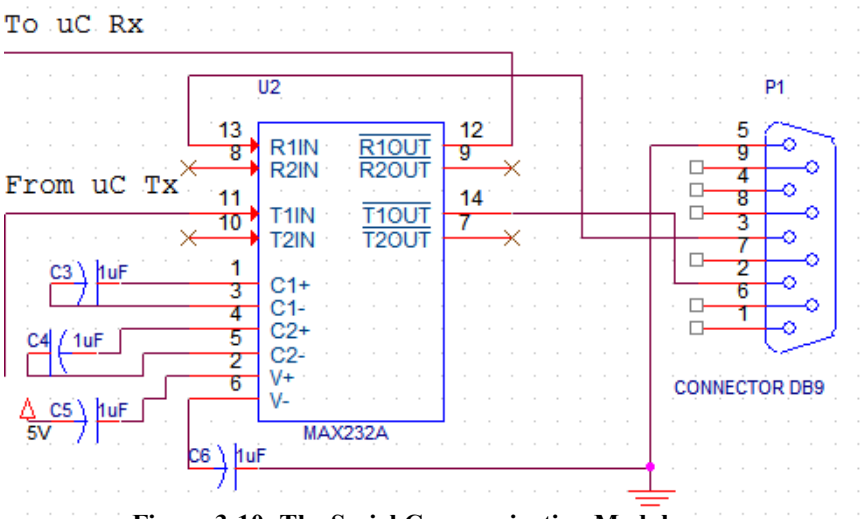


**Figure 3-10: The Serial Communication Module**

### 3.7.2 The I/O Panel

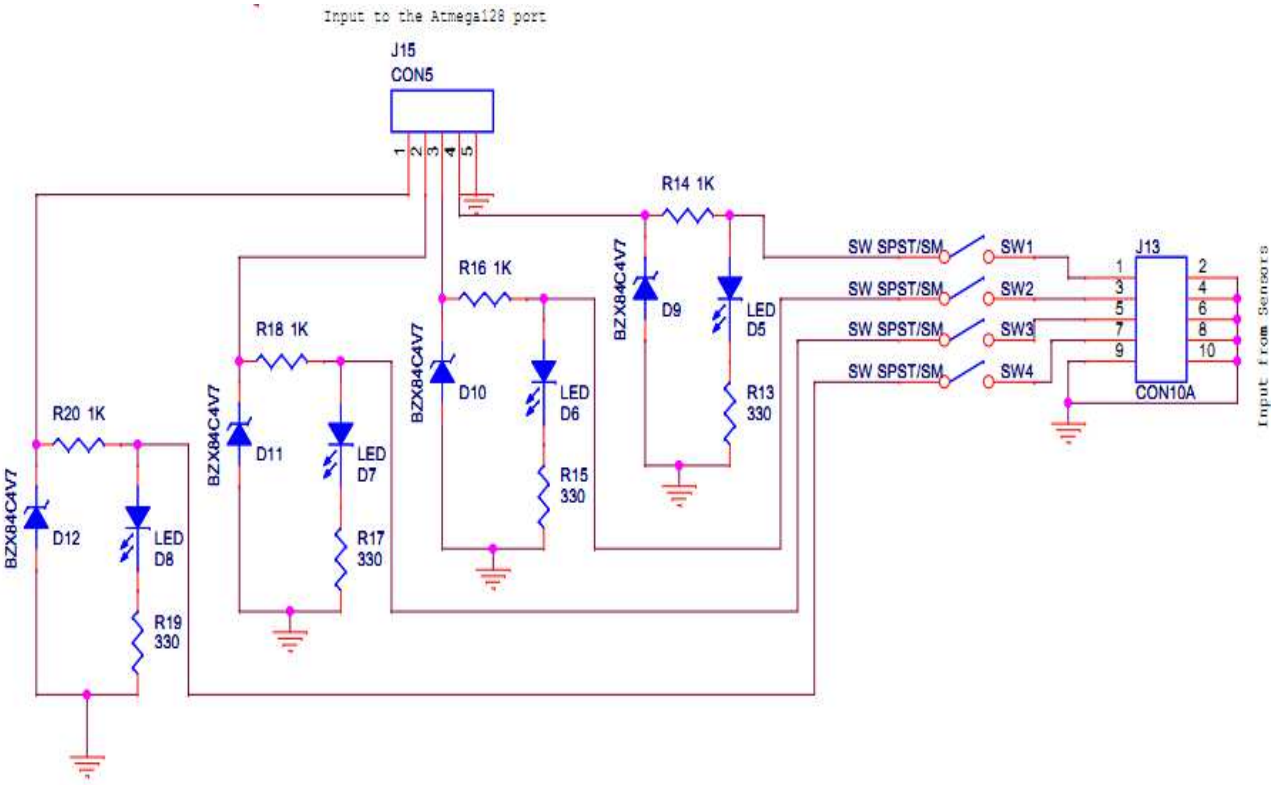


**Figure 3-12: The inputunit schematic**

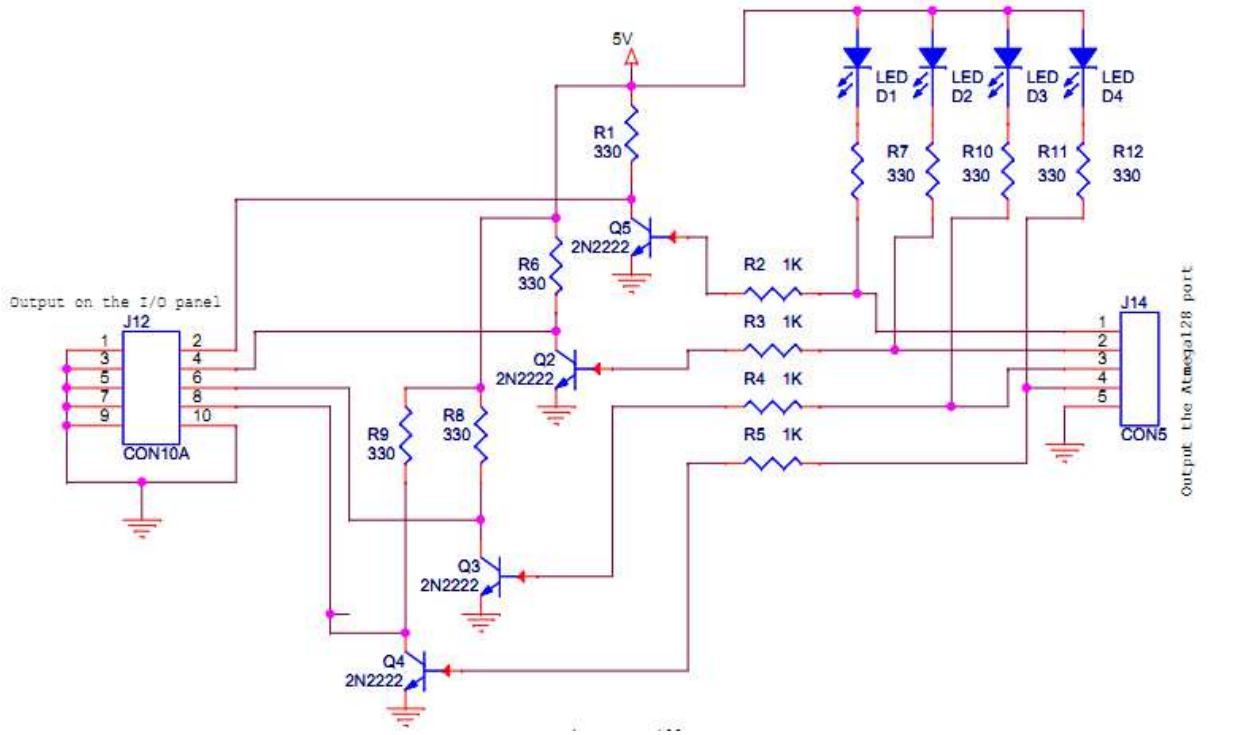


**Figure 3-11: The output unit schematic**

### 3.7.3 The Main Control Board

[See the next page for the schematic]

## 3.8 Algorithms

### 3.7.1 Smoothing the motion of all servo motors simultaneously

1. Start
2. Take input for angles: base, shoulder, elbow, wrist pitch, wrist roll;
3. Mark flag variable=0 for each joint angle
    a. Where
        i. flag=0 means angle transformation incomplete;
        ii. flag=1 means angle transformation complete;
4. Total number of flags = 5;
5. Compare each joint angle with its current value;
6. If
    a. Required Value is greater than “>” current value
7. Increase current value by one for each joint angle
    a. Else if Required Value is less than “<” current value
8. decrease current value by one for each joint angle
    a. Else mark flag variable for a completely transformed joint angle as ‘1’ i.e. flag = 1;
9. Put a very small delay from 1 ms to 10 ms;
10. Goto ‘5’ step and loop until total flags marked are equal to 5;
11. End

### 3.7.2 Token Segmentation from string command-line for Robotic Arm in Control Board

Before Start

Let's assume a string command received in the control board as:

"s2100,1400,1500,1300,780;"

Where

- *s stands for set angles*
- *Numbers separated by comma ',' are values for microcontroller to set the motor angles*
- *';' is the terminator*

1. Start
2. Declare an external character array variable, say rxdata[ ];
3. Store the string sequence received from USART in the character array variable;
4. Read the first character from the received character array or string;
5. If 's' or any other character is found jump to its sub-routine (in this case 's')
6. Declare a temporary character array temp[ ] with a length of 4 characters or more;
7. Declare integers 'i' and 'k', increment 'k';
8. Equate temp[i] with rxdata[k]
9. Increment 'i' and 'k';
10. Goto and loop until rxdata[k]=',' (comma);
11. If rxdata[k] = ",", then break and,
    a. equate i=0
    b. q1….n(joint angle) = typecast (temp[ ] string);
    c. equate temp [ ] =" ";
12. Goto '' and loop until ';' is found
13. End

## 3.9 Simulations

### 3.8.1 Testing PWM Channels OCnA, B & C

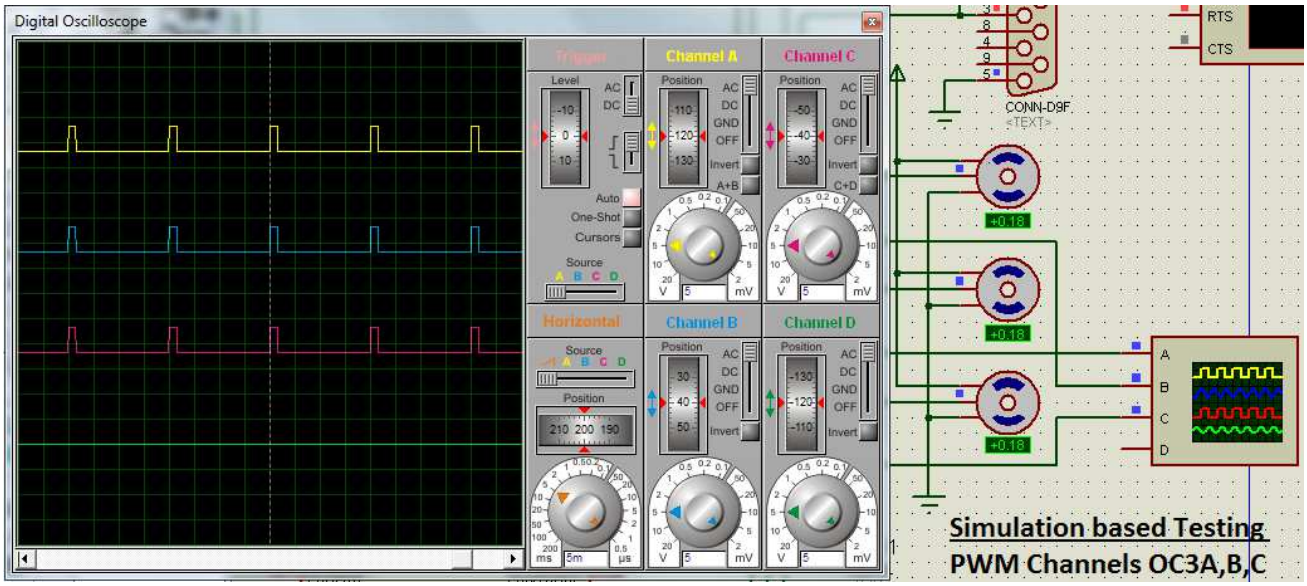


**Figure 3-13: Simulation of PWM channels OC1A,B& C**

### 3.8.2 Customized Robotic Arm Command Protocol

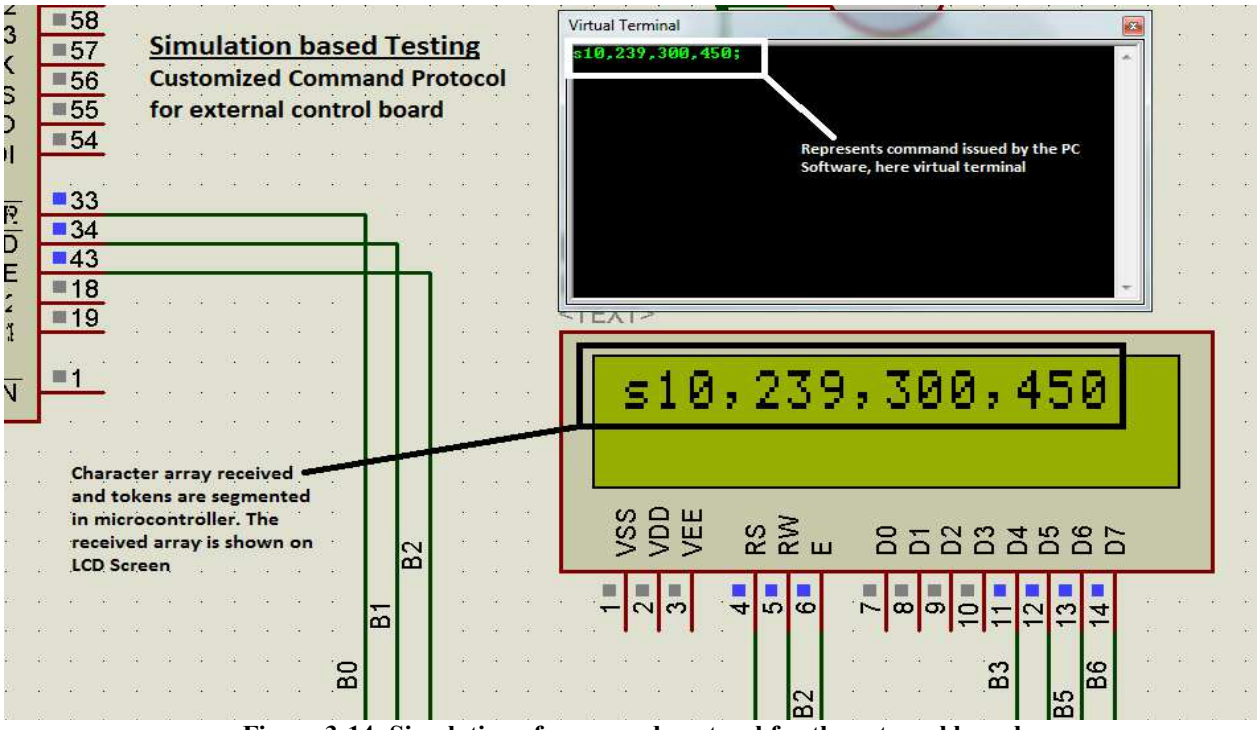


**Figure 3-14: Simulation of command protocol for the external board**

## 3.10 PCB Designs

### 3.10.1 Main Board PCB Layout

This is the main board PCB layout. Atmega128's breakout board fits on it.

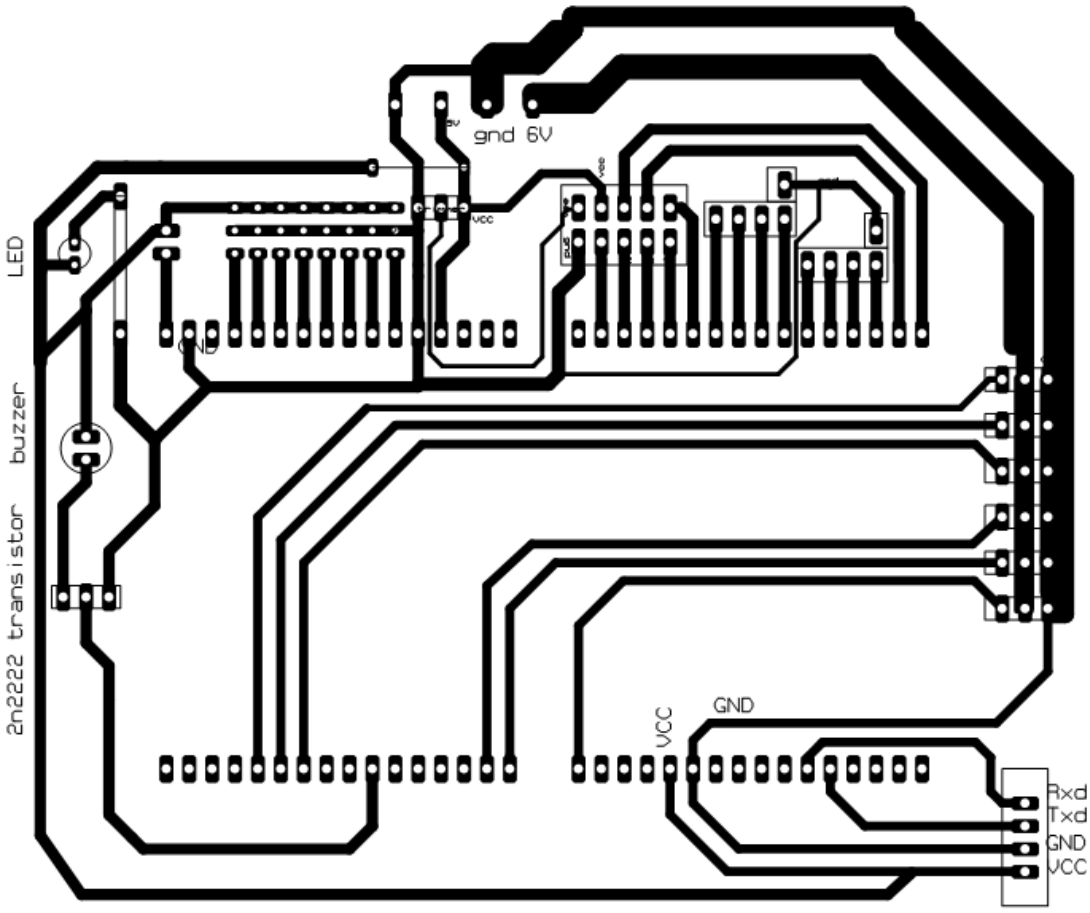


**Figure 3-15: PCB layout design of the main board**

### 3.10.2 Atmega128's Breakout Board

Here is the PCB layout of Atmega128's breakout board with the header pin holes. The IC in the middle is the TQPK package IC of Atmega128 microcontroller.

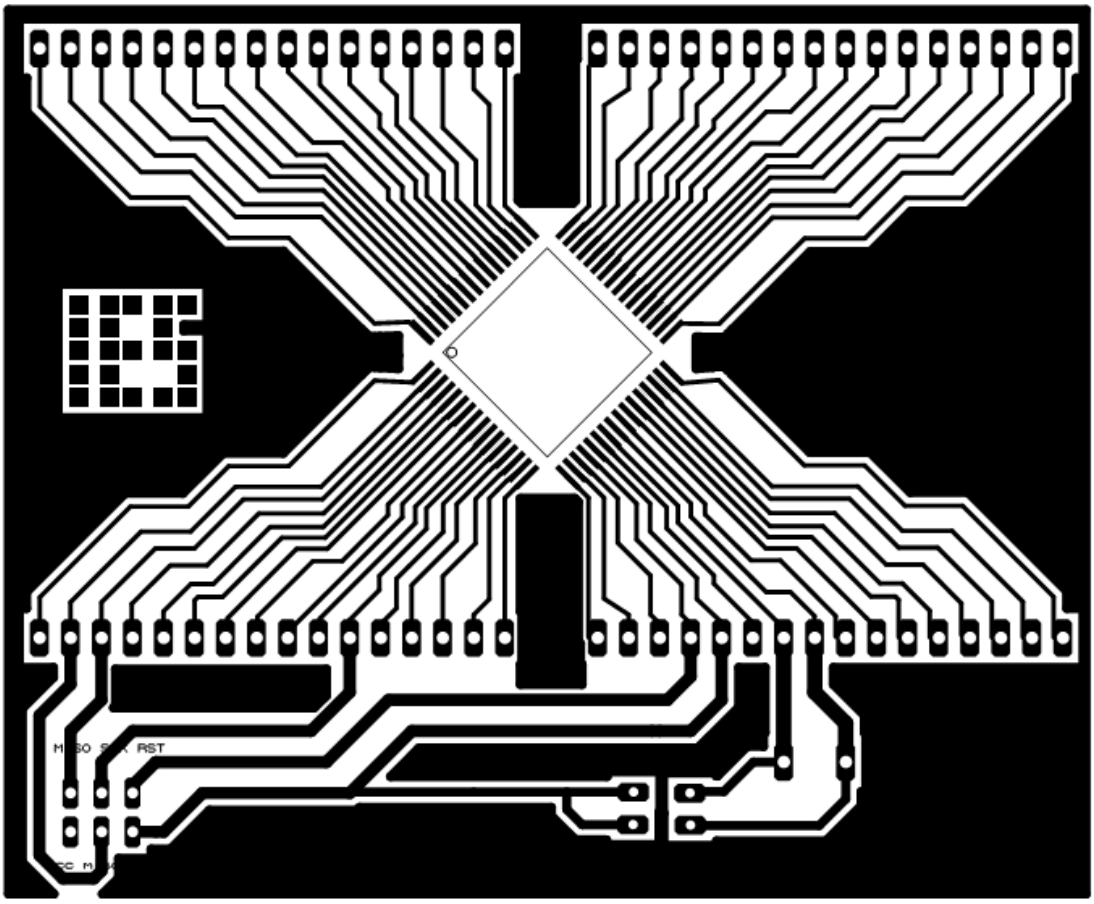

**Figure 3-16: PCB layout for Atmega128 breakout board**

## 3.11 Free Body Diagram (FBD)

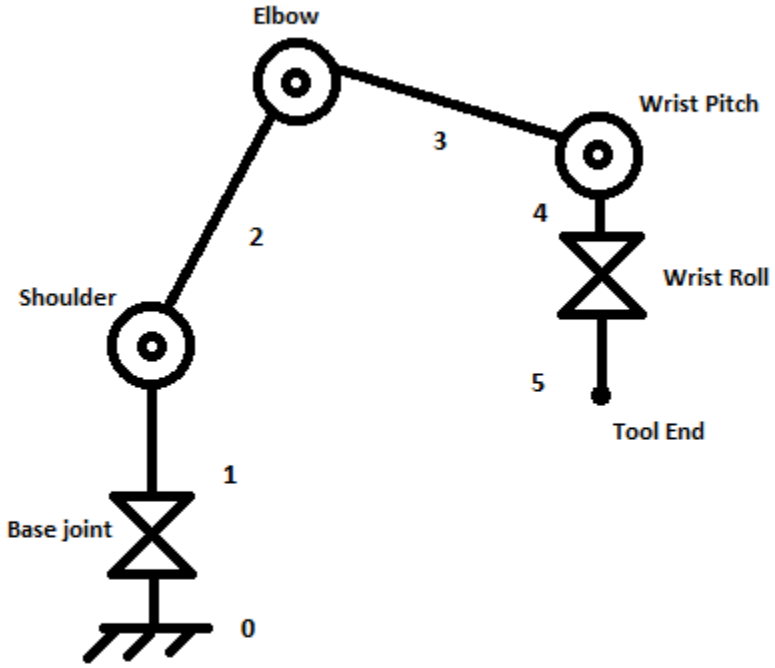


**Figure 3-17: Free Body Diagram of 5-axis articulated robotic arm, Edu.bot**

## 3.12 Mechanical Designs

In this section, we depict the different sections and parts of our mechanical design. Since it is a robotic arm, it contains several joints and parts. All dimensions mentioned are in centimeters.

**Base:**

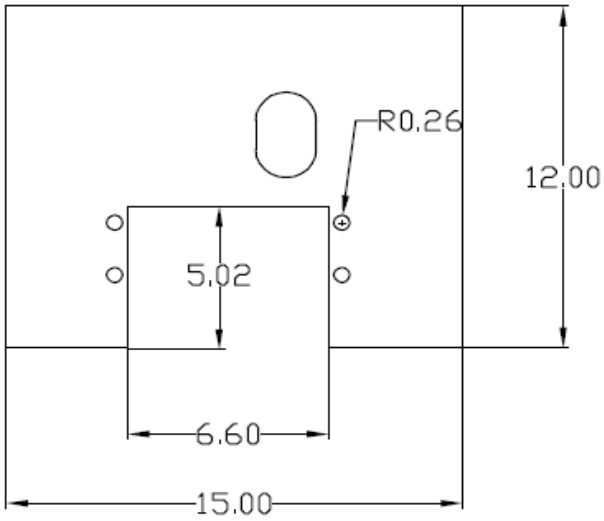


**Figure 3-18: Part of the base with the space for motor**

**Elbow**

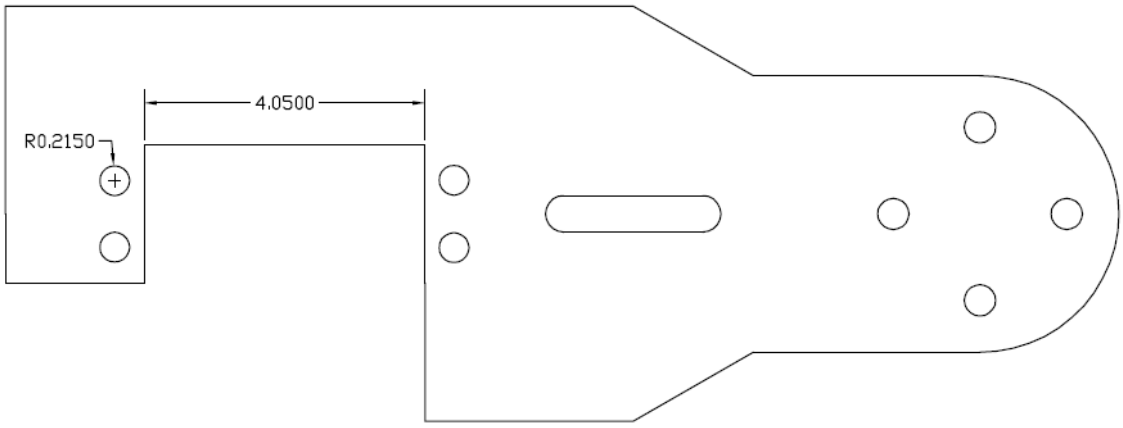


**Figure 3-19: Elbow link with the space for motor**

## Link A & B

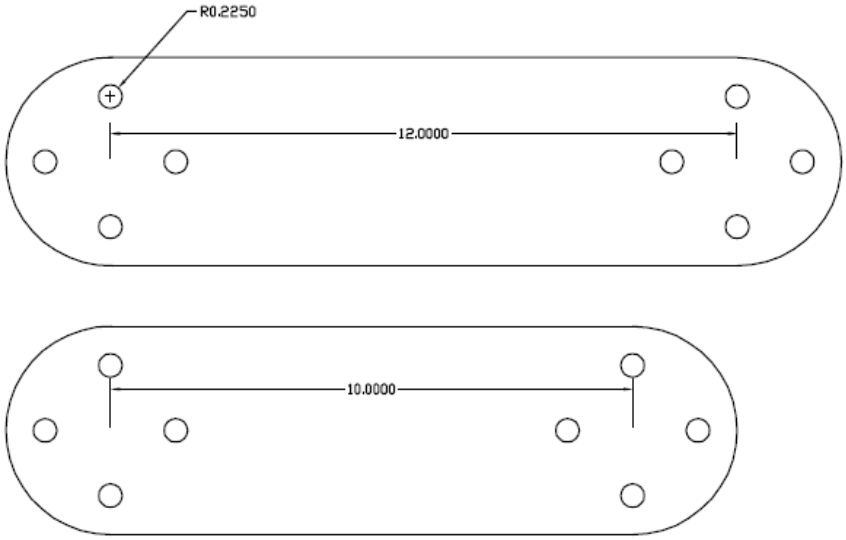


**Figure 3-20: Links A& B for shoulder and elbow**

## Shoulder

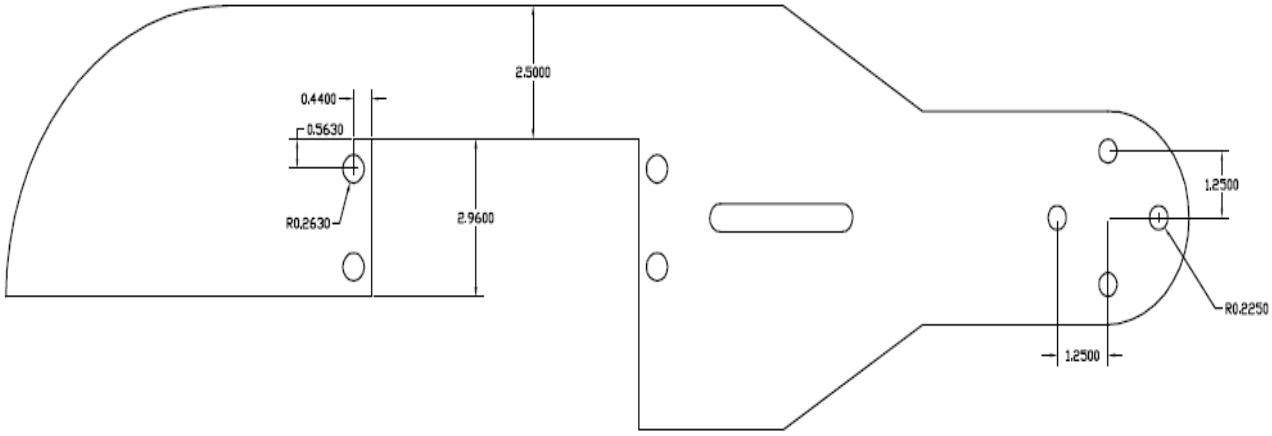


**Figure 3-21: Shoulder link with the space for motor visible**

**Wrist Roll Clamp:**

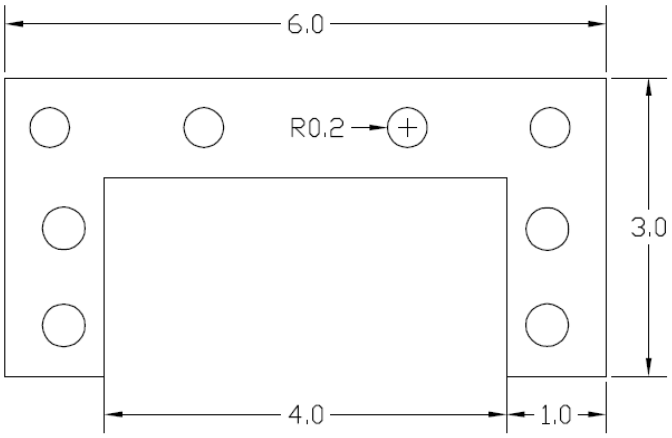


**Figure 3-22: Wrist roll clamp to move with the servo motor placed in it**

**Wrist Roll:**

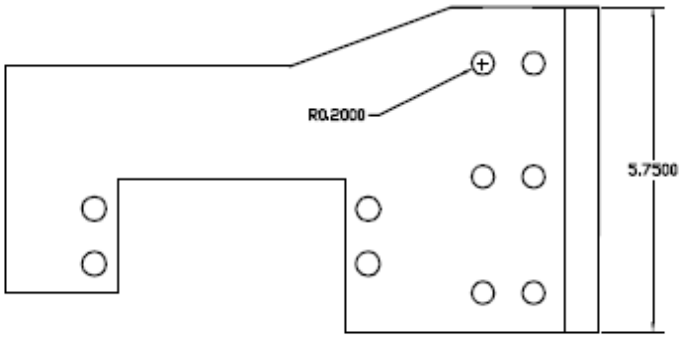


**Figure 3-23: Wrist roll part with the space for motor**

These mechanical links were joined together with spacers and fitted with gears as shown in the picture. The spaces for servo motors are for fitting the servo motors. These servo motors do not move, rather they move the whole link while being static. All the joints use bearing support. The base joint uses dual ball bearing support.

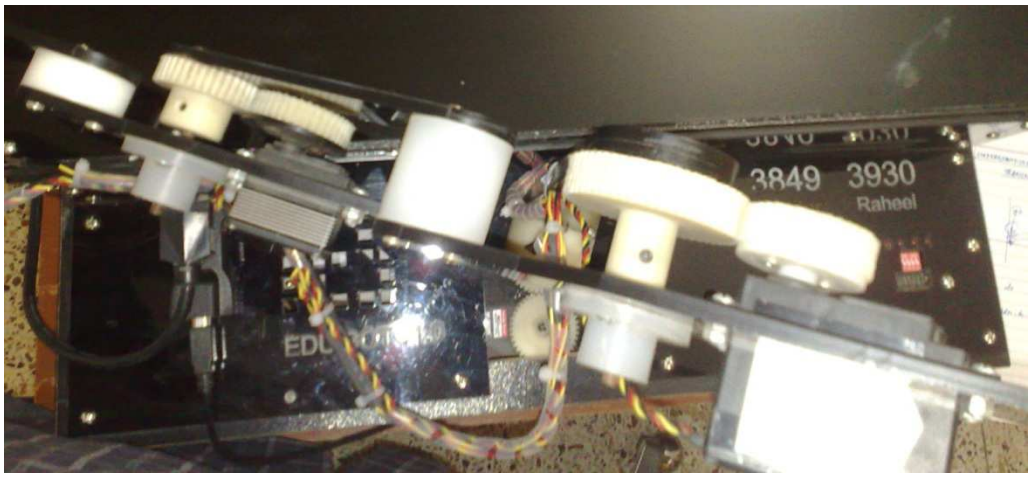


**Figure 3-24: Gear assembly from the top**

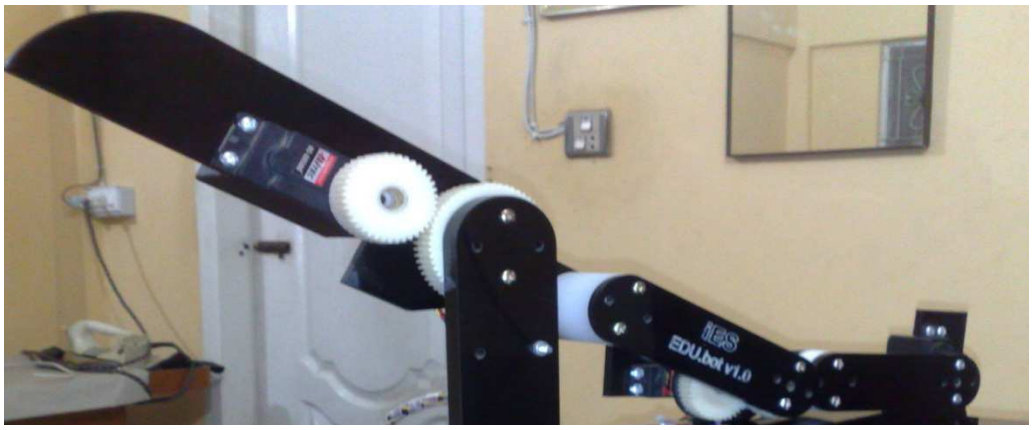


**Figure 3-25: Links and joints with cylinders**

## 3.13 Sample Lab Manual

### 3.12.1 Lab #1: UNDERSTANDING THE BASIC TERMINOLOGIES USED WITH ROBOT ARMS

FREEBODY DIAGRAM

The freebody diagram of robotic arm is a rough visualization diagram that how a robot will look like with given joints and given joint types. A freebody diagram does not follow any specific rules, but symbols for joint types are conventionally used. The EDU.BOTv1.0 is a five axis robot arm and the freebody diagram for this robot arm is shown in the figure:

DEGREES OF FREEDOM

The EDU.BOTv1.0 is an articulated robot arm. The degrees of freedom of Robotic arm are the number of moveable axis attached to it. In case of this articulated arm, all joint are revolute.

The EDU.BOTv1.0 has five degrees of freedom. The degrees of freedom of an end-effector attached to the arm are never included in the DOF counts of robotic arm.

In order to practically demonstrate and observe the DOF of EDU.BOTv1.0, use the software for this Laboratory manual.

1- Press the button "__" to command the robot to move to a position look like one in the figure:
   [Real picture]
2- Press "Base" to move the robot show its revolute axis only for rotating along the fixed base frame.
   [Freebody diagram]

3- Now, the “shoulder” button will be activated. Press it to make the robot move itself along shoulder axis only. Repeat the same procedure till all axis are tested.

Total number of degrees-of-freedom are: _____

WORK ENVELOPE

The work envelope

A robot can only work in the area in which it can move. This area is called the work envelope. The work envelope is determined by how far the robot's arm can reach and how flexible the robot is. The more reach and flexibility a robot has, the larger the work envelope will be. [5]

The terminologies i.e. “**Reach**” and ‘**Stroke**” are normally used in the concept of work envelope. The reach and the stroke of a robotic manipulator help in making the rough measure of work envelope.

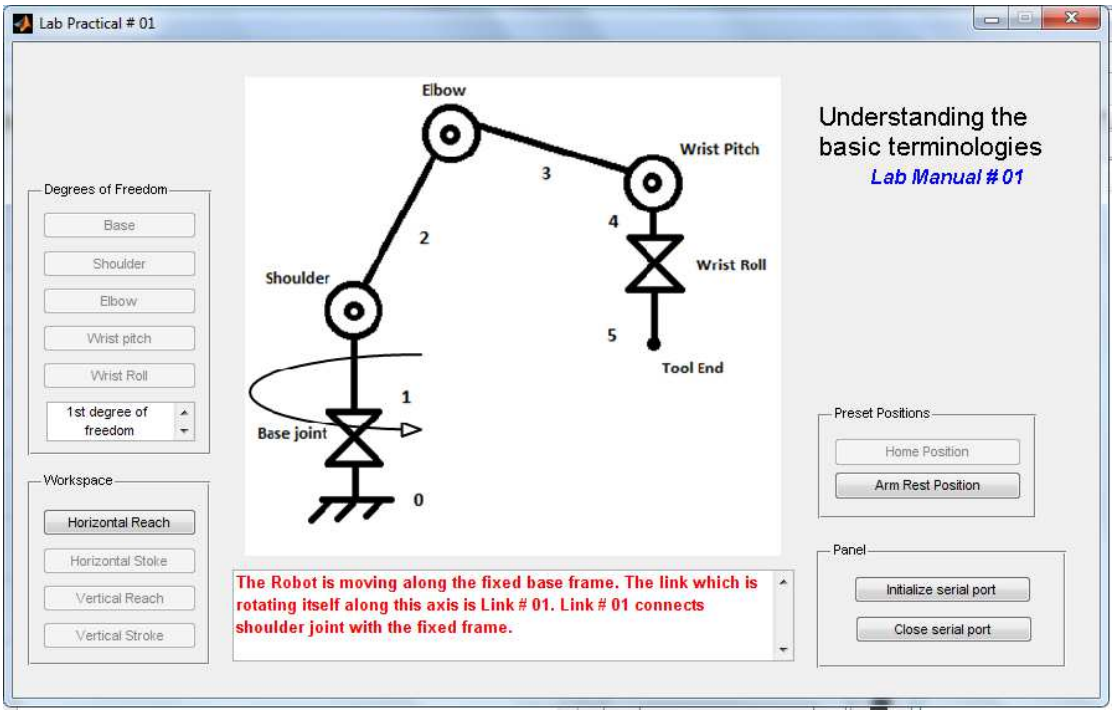


**Figure 3-26: GUI window for the Lab#1**

**Reach**

***Horizontal Reach****:* The maximum radial distance a wrist mounting flange can be positioned from the vertical axis about which the robot rotates.

***Vertical Reach:***The maximum elevation above the work surface the wrist mounting can reach.

**Stroke**

***Horizontal Stroke:*** The Horizontal stroke represents the total radial distance the wrist can travel.

***Vertical stroke***: The total vertical distance that the wrist can travel.

In order to estimate the work envelope of EDU.BOTv1.0, go through the following procedures.

1- Press "**Horizontal Reach**" along fixed XY-plane. An image showing the rough radial horizontal reach is given below. Draw a rough image in the blank space:

2- Now Press "**Horizontal Stroke**", observe it along fixed XY-plane, an image showing the rough radial horizontal stroke is given below. Draw a rough image in the blank space:

3- Now press "**Vertical Reach**" and observe the movement of the arm in XZ-plane till it reaches the top. Draw the path in the space given below:

4- Now Press "**Vertical Stroke**", observe it along XZ-plane, an image showing the rough radial vertical stroke is given below. Draw a rough image in the blank space:

Now by combining the results of movements of robotic arm in XY-plane and XZ-plane, you can visualize a rough estimate of the work envelope of the robotic arm.

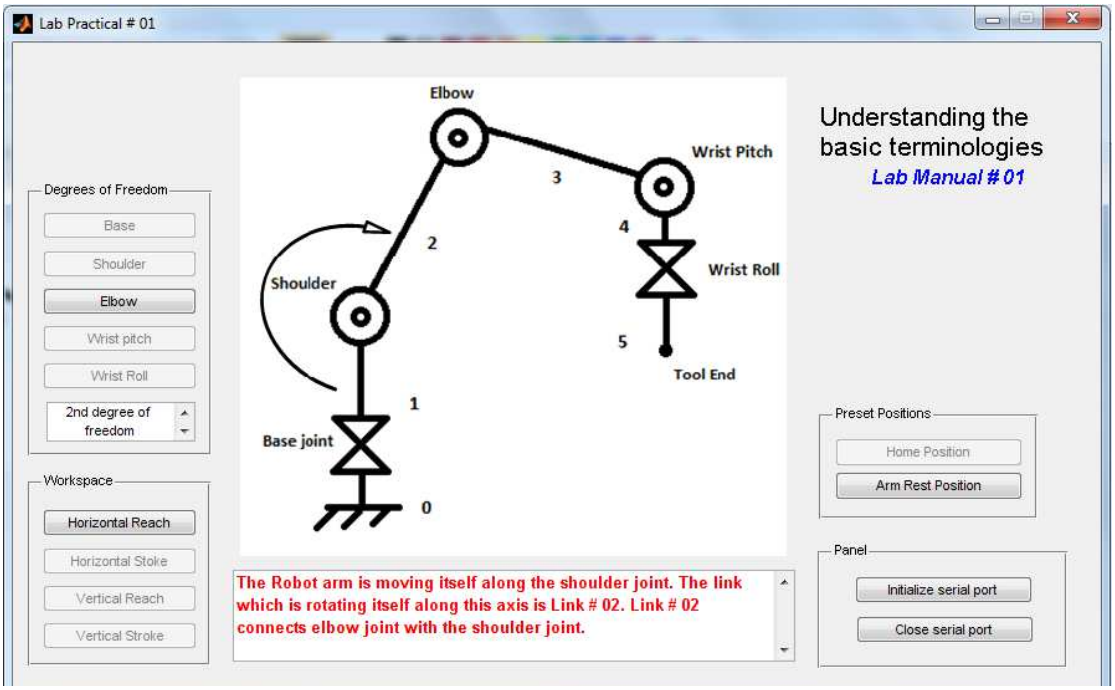


**Figure 3-27: GUI window of lab#1 showing 2nd degree of freedom**

## 3.14 Hardware and Software List

### 3.14.1 Hardware

HitecServo motors (HS-805BB, HS-7955TG, HS-5485HB)

Atmega128 microcontroller

MAX232

Capacitors

Crystal

Acrylic sheets for links and panel

Nylon gears

Wooden box

USB to serial cable

PCB

Vero board

Cables, connectors, headers

### 3.14.2 Software

MATLAB

Visual C#

AVR Studio

SinaProg

Dev C++

MS Word

MS Powerpoint

### 3.14.3 Cost Analysis

This is our total cost, while other robotic arm trainers that are available in the international market are way expensive. Since this is our first prototype, it is still little bit expensive than what we were expecting, the cost will surely go down significantly if we start to build another similar educational robot or some other group follows our footsteps.

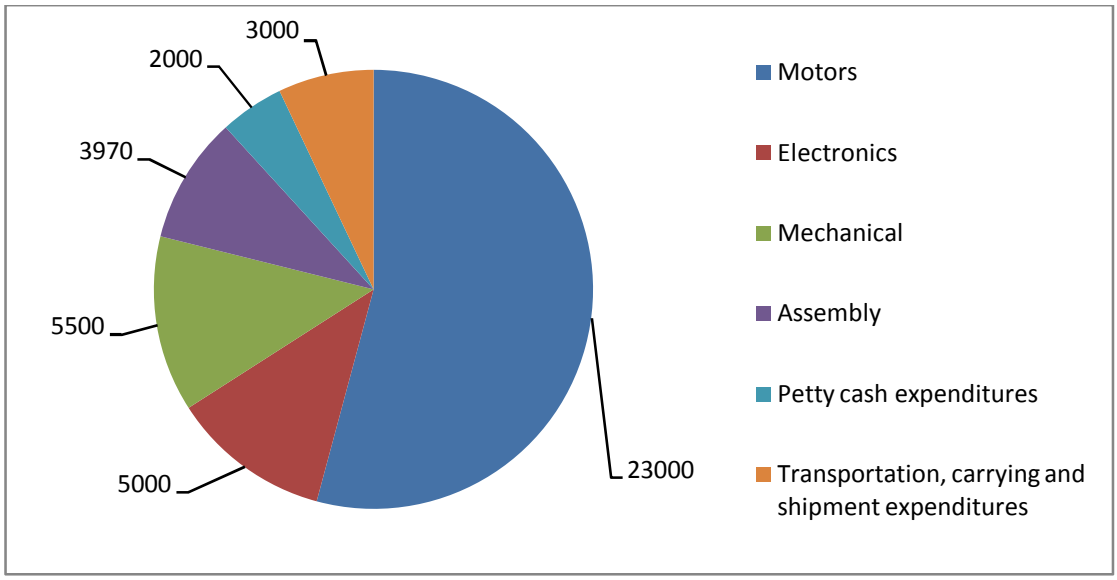


**Figure 3-28: Pie-chart distribution of the total cost in general**

| No. | Particulars | Cost (Rupees) |
|---|---|---|
| 1 | Motors | 23000 |
| 2 | Electronics | 5000 |
| 3 | Mechanical | 5500 |
| 4 | Assembly | 3970 |
| 5 | Petty cash expenditures | 2000 |
| 6 | Transportation, carrying and shipment expenditures | 3000 |
| | Total | 42470 |

**Table 3-6: Overview of cost in general terms**

Break costs are as follows along with the reference number of the particulars given above:

| Particulars | Amount (Rupees) | Ref number |
|---|---|---|
| Acrylic | 600 | 4 |
| Laser cutting | 500 | 4 |
| Cylinders and base | 2000 | 3 |
| Gears | 2000 | 3 |
| Bearings | 400 | 4 |
| LCD(16x2) | 250 | 2 |
| Shaft (5mm) | 120 | 4 |
| Nuts and bolts | 200 | 4 |
| Other material | 1000 | 4 |
| Casing/box | 1500 | 3 |
| Circuits | 1000 | 2 |
| Usb2serial and other electronics | 950 | 3 |
| Transformers/power supply | 1300 | 2 |
| PCB raw boards | 500 | 2 |
| Panel cutting | 300 | 4 |
| Panel acrylic | 400 | 4 |
| Acrylic cover | 450 | 4 |
| Atmega128 | 1000 | 2 |
| **Sub Total** | 14470 | |
| Petty cash disbursements | 2000 | 5 |
| Transportation, carrying and shipment expenditures | 3000 | 6 |
| Motors | 23000 | 1 |
| Total | 42470 | |

**Table 3-7: Break-up of cost of the project**

## 3.15 Gantt Chart

[Please see the next page for the Gantt chart]

## 3.16 Safety Precautions

***Safety Precautions***

- Do not stop/obstruct the movement of the robotic arm forcefully, otherwise motor's internal calibration will be disturbed and it may become completely inaccurate.
- Obstructing the movement of robotic arm may also cause motors' internal gear train to be broken off.
- Do not touch or grab the robotic arm or place your fingers or palm on the external coupling gears, it may injure your hand or cause serious wounds..
- Do not move robotic arm with hands unnecessarily/abruptly, while powered off/on.
- Only use appropriate power supplies. Power supply for servo motors should be separate. Power supply for the control circuitry should not be greater 5V and that for servo motors should not be greater than 6V.
- Keep out of radius of operation of the robotic arm.
- Do not turn off power or disconnect the cable unless the robotic is in its rest position. This is because the arm can move unprecedentedly and may fall down abruptly to receive any damage.
- Do not try to send in any further commands from the GUI while one command and operation is in progress.
- Maintenance and service must be done by qualified technicians or engineers. A layman must not open the box and disassemble the arm.
- Electric shock hazard inside the box, strictly follow proper servicing and handling procedures.

- Servicing and maintenance must be done in ESD (Electrostatic Discharge) protected environments otherwise it might burn out the SMD microcontroller chip.
- ALWAYS PLACE THE ROBOTIC ARM IN THE ARM REST POSITION after operation, by pressing the Arm Rest Position button on the GUI.

# 4.0 TEST RESULTS AND THEIR ANALYSIS

Robotic systems need to be precise and error free. Since we used servo motors in our project, it was essential for us to test their angle responses under the different ranges of duty cycle of PWM. This is because every motor has mechanical constraints and differ from other motors. We tested each motor manually over wide ranges of duty cycle. Following sections provide the detailed tabulation and graphs of our test results. With these test results, equations for each motor's position were formed; these equations are used in the code for obtaining the desired position according to the task easily.

## 4.1 Base Servo Motor

**Hitec HS-805BB+**

| Angle Response (degrees) | OCRnA |
|---|---|
| -71 | 850 |
| 0 | 1500 |
| 71 | 2150 |

Table 4-1: Base servo angle response and corresponding OCRnA register value

| Angle Response Slope | Intercept |
|---|---|
| 9.154929577 | 1500 |

Table 4-2: Base servo angle response slope and intercept

**Equation:**

$$OCR1C = 9.154929577 * \mathrm{q1} + 1500$$


**Figure 4-1: Angle response curve of the base motor w.r.t OCRnA**

## 4.2 Shoulder Servo Motor

**Hitec HS-805BB+**

| Angle Response (degrees) | OCRnB |
|---|---|
| 0 | 1900 |
| -90 | 760 |

Table 4-3: Shoulder servo angle response with the corresponding OCRnB register value

| Angle Response Slope | Intercept |
|---|---|
| 12.66666667 | 1900 |

Table 4-4: Shoulder servo angle response slope and intercept

**Equation:**

$$OCR1B = 12.6667 * \text{q2} + 1900$$

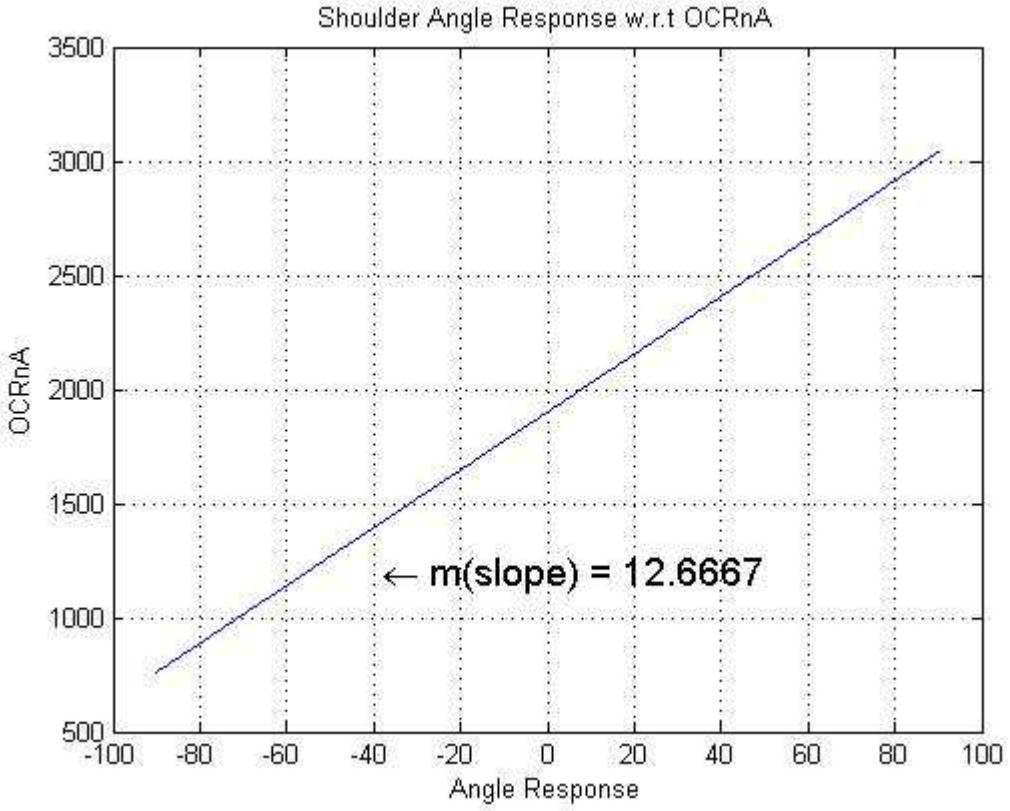


**Figure 4-2: Angle response curve of the shoulder motor w.r.t OCRnA**

## 4.3 Elbow Servo Motor

**Hitec HS-7955TG**

| Angle Response (degrees) | OCRnA |
|---|---|
| 0 | 980 |
| 90 | 1900 |

Table 4-5: Elbow servo angle response with the corresponding OCRnB register value

| Angle Response Slope | Intercept |
|---|---|
| 10.22222222 | 980 |

Table 4-6: Shoulder motor angle response slope with intercept

**Equation:**

$$OCR1A = 10.2222222 * \mathrm{q3} + 980$$

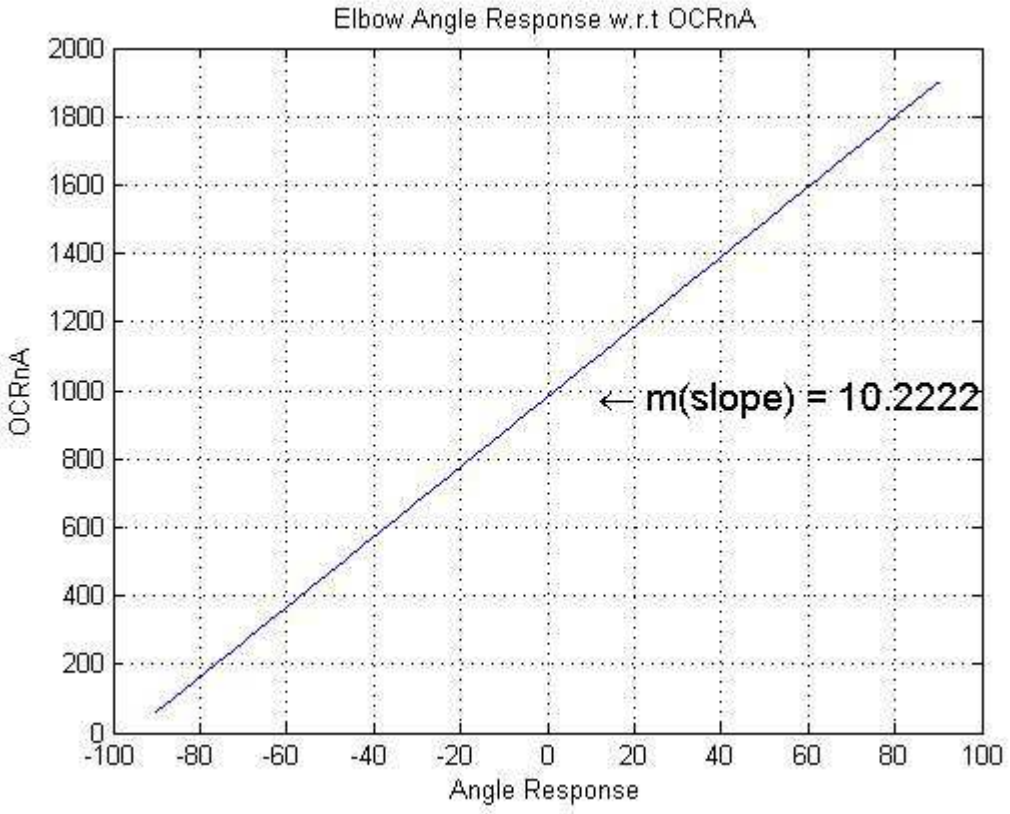


**Figure 4-3: Angle response curve of the elbow motor w.r.t OCRnA**

## 4.4 Wrist Pitch

**Hitec HS-5485HB**

| Angle Response (degrees) | OCRnC |
|---|---|
| 0 | 750 |
| -90 | 1610 |

Table 4-7: Wrist pitch servo angle response with the corresponding OCRnB register value

| Angle Response Slope | Intercept |
|---|---|
| -9.555555556 | 750 |

Table 4-8: Wrist pitch motor angle response slope with intercept

**Equation:**

$$OCR3C = -9.5555556 * \mathrm{q4} + 750$$

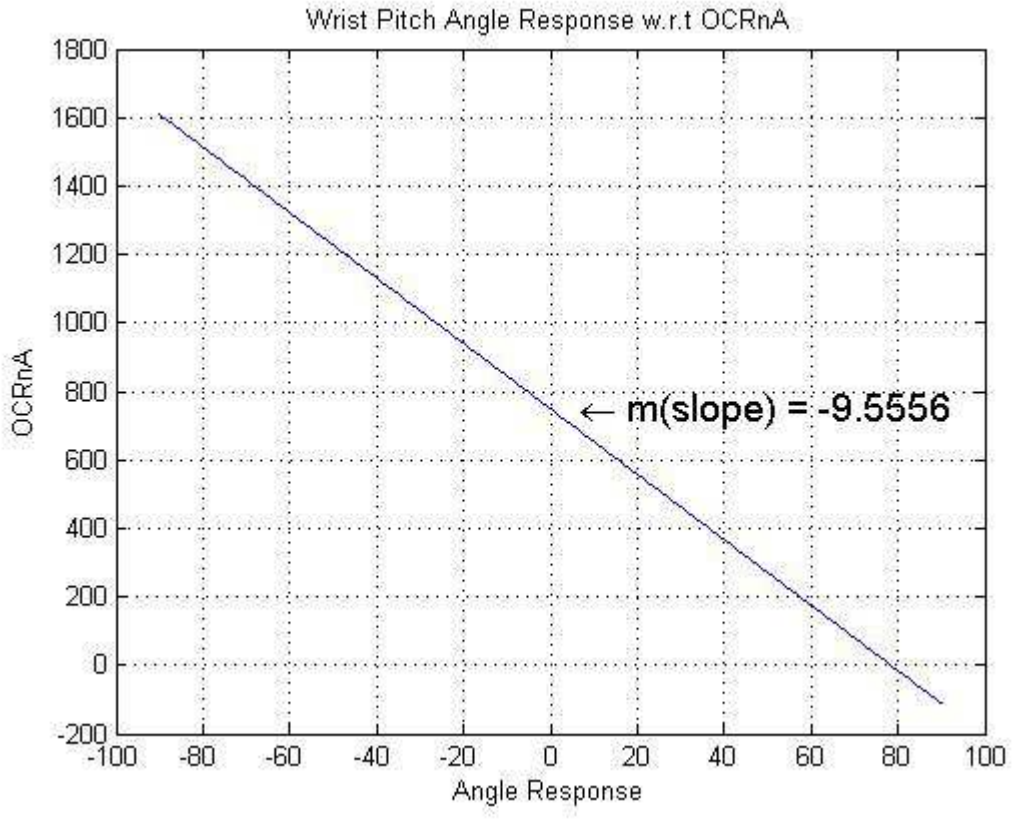


**Figure 4-4: Angle response curve of the wrist pitch motor w.r.t OCRnA**

## 4.5 Wrist Roll

**Hitec HS-5645MG**

| Angle Response (degrees) | OCRnB |
|---|---|
| -45 | 2000 |
| 0 | 1500 |
| 45 | 1000 |

Table 4-9: Wrist roll servo angle response with the corresponding OCRnB register value

| Angle Response Slope | Intercept |
|---|---|
| -11.11111111 | 1500 |

Table 4-10: Wrist roll motor angle response slope with intercept

**Equation:**

$$OCR3B = -11.11111111 * \mathrm{q5} + 1500$$

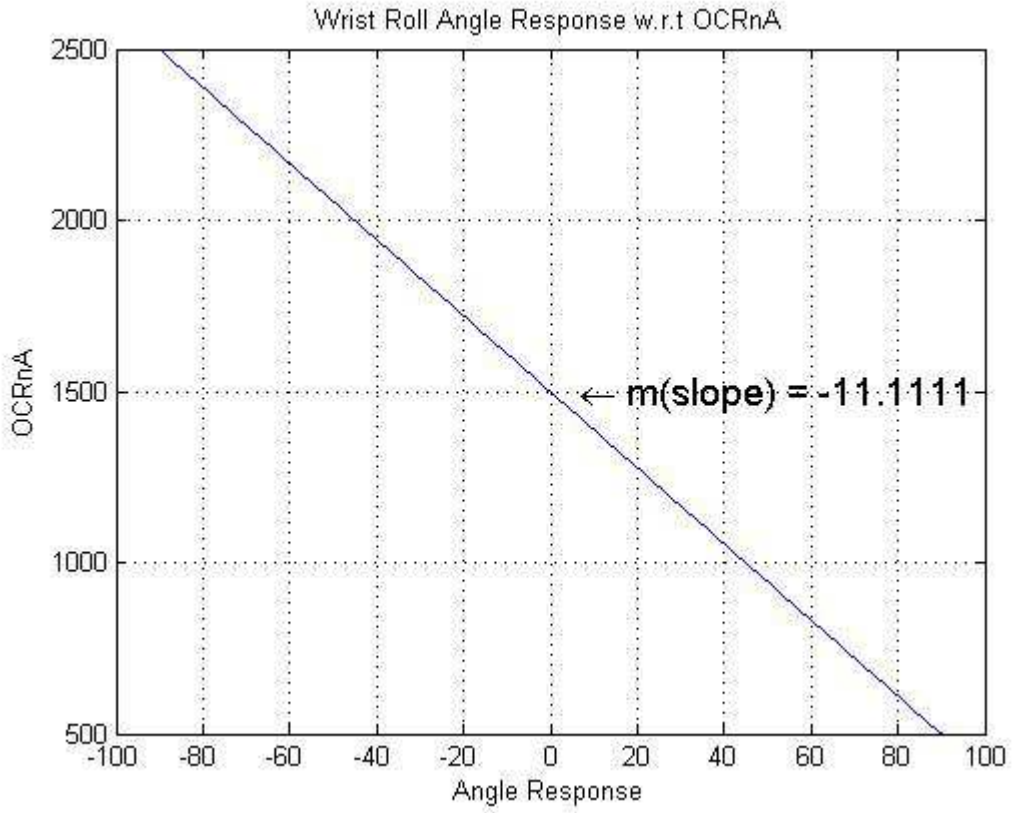


**Figure 4-5: Angle response curve of the wrist roll motor w.r.t OCRnA**

# 5.0 ECONOMIC ANALYSIS

As discussed in the introduction and the project objectives, this project was intended to be the low-cost alternative to its foreign counterparts. Though we have made it quite efficiently with inexpensive items and parts, still this thing is quite alarming that in Pakistan we have no cheap facilities to get engineering works done. Every major and good part is imported and taxes imposed elevate their cost.

Even after having it done with low-cost material and parts, we faced some issues like quite a few things were costing too much since they are not produced in Pakistan and the excise duties increase the cost to a level that one starts thinking of importing the whole product instead of manufacturing it locally.

But since we had a determination, we have still managed it to keep the costs low and yet quality at par.

# 6.0 CONCLUSION

After 10 months of non-stop struggle, we have come up with the Edu.bot, which is a 5-axis articulated robotic arm trainer. We have achieved all of our objectives and in fact, we have achieved a success which we did not even think of. This is a quality work, since we used every part and got every service from quality workshops and experienced technicians. We used world's renowned Hitec servos.

So the conclusion is, our Edu.bot is cheapest Robotic Arm Trainer produced with high quality in Pakistan. It is 100% in working condition and is really beyond the expectations.

This is the version 1.0 of our educational robotic arm. There is a lot of room for improvement and expansion. More advanced labs experiments and post-graduate level research can also be extracted out of this robotic arm. We would be glad if someone

from our juniors or some other researchers from other universities take up this project and enhance it.

We have compiled lab manuals and lab software for undergraduate students of robotics courses so that they may get a clear idea of what they learn in theory classes with rather dry and boring lectures full of mathematics and matrices. These labs will not only help them understand the mathematical modeling of robotic arms and also understand the operations and specific behavior of them.

A brief User Guide is compiled for the safe operation of robotic arm so that students can operate it properly without damaging it. This will also help them getting it to work fast, resulting in better time management during the lab hours.

# 7.0 FUTURE RECOMMENDATIONS

Though we have taken the best care for quality and design, it still needs to be improved over time. Some suggestions and recommendations for future enhancements are as follows:

- Mechanical design needs more stability and mathematical calculations.
- DC motors with encoders should be used to increase precision and this would be the real engineering.
- Inverse kinematics should be added.
- Cost can be lowered, since this was a prototype it costs a bit more.

# 8.0 REFERENCES

[1] Niku, Saeed B. *Introduction to Robotics: Analysis, Systems & Applications, 1st Edition.* New Jersey: Prentice Hall International Inc., 2001.

[2] Intelitek Corp. *Scorbot –ER4u page*, Date of retrieval: 23th March, 2011. <http://www.intelitek.com/ProductDetails.asp?Product_ID=17&CategoryID=3&Industrial=&Education=yes >

[3] Intelitek Corp. *Scorbot-ER 9Pro page*.Date of retrieval: 13th March2011. <http://www.intelitek.com/ProductDetails.asp?Product_ID=18&CategoryID=3&Industrial=&Education=yes>

[4] MTAB India.*Aristo 6XT*. Date of retrieval: 13th March 2011. <http://www.mtabindia.com/aristo.htm>

[5]Sie Deen lau.*Experimental Study of Robotic Assembly and Force Control Tasks*.University of Washington Robotic Arm Project. 13th October 2009. Date of retrieval: 2nd May 2011.<http://ese.wustl.edu/ContentFiles/Research/UndergraduateResearch/CompletedProjects/WebPages/sl2/ESE499/roboticArm.html>

[6] *Work Envelope Definition* at Welding Robots site. Date of Retrieval:15th June 2011.<http://www.welding-robots.com/faq.php?question=robot+work+envelope>

# APPENDICES

# A-1 Source Code

### A-1-1 Atmega128 External Control Board:

```
#define F_CPU 8000000UL // 8MHz
//default fuses: LOW=E1; HIGH=99; EXTENDED=FD

#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include <stdlib.h>
#include <string.h>
#include "lcd_lib.h"

voidinitPWM(void);
voidinitUSART(void);
voidset_angles(uint16_t, uint16_t, uint16_t, uint16_t, uint16_t);

//other funcs________
voidsetanglefunc(void);
voidexecute_command(void);
voidLCDstatus(char *, int *);

//----------------------


//________GLOBAL VARIABLES___________

charrxdata[80];
char str1[]=" ";

int i=0;
//_____________________________________
int main (void)
{
initPWM();
initUSART();
LCDinit();
LCDclr();
sei();


OCR3A = 1500;          //Tool
OCR3B = 1500;          //Wrist Roll
OCR3C = 750;  //Wrist Pitch

OCR1A = 750;  //ELbow
OCR1B = 2310;          //Shoulder
OCR1C = 2100;          //Base
_delay_ms(4000);
```

```
while(1){;}

}

//_____________________________FUNCTIONS_____________________________________
____

voidinitPWM(void)
{

DDRB |= (1<<PB5) | (1<<PB6 )| (1<<PB7); // OC1A and OC1B OC1C as servo signal
outputs
DDRE |= (1<<PE3) | (1<<PE4 )| (1<<PE5); // OC3A and OC3B OC3C as servo signal
outputs

TCCR1A = (1<<COM1A1) | (1<<COM1B1) | (1<<COM1C1)| (1<<WGM11);    // Cntr1:
Mode14 Fast 14, clear on compare
TCCR1B = (1<<WGM13) | (1<<WGM12) | (1<<CS11); // Set prescaler to 8

TCCR3A = (1<<COM3A1) | (1<<COM3B1) | (1<<COM3C1)| (1<<WGM31);    // Cntr3:
Mode14 Fast 14, clear on compare
TCCR3B = (1<<WGM33) | (1<<WGM32) | (1<<CS31); // Set prescaler to 8

ICR1 = 20000; // Counter TOP value set to 50Hz frequency
ICR3 = 20000;

}


voidset_angles(uint16_t q1, uint16_t q2, uint16_t q3, uint16_t q4, uint16_t q5)
{
        uint8_t total=0, q1f=0, q2f=0, q3f=0, q4f=0, q5f=0;

        while(total!=5)
        {

                if(OCR1C<q1) OCR1C++;
                else if(OCR1C>q1) OCR1C--;
                else q1f=1;

                if(OCR1B<q2) OCR1B++;
                else if(OCR1B>q2) OCR1B--;
                else q2f=1;

                if(OCR1A<q3) OCR1A++;
                else if(OCR1A>q3) OCR1A--;
                else q3f=1;

                if(OCR3C<q4) OCR3C++;
                else if(OCR3C>q4) OCR3C--;
                else q4f=1;

                if(OCR3B<q5) OCR3B++;
                else if(OCR3B>q5) OCR3B--;
```

```
                else q5f=1;

                total = q1f + q2f + q3f + q4f + q5f;
                _delay_ms(3);

        }

}


voidinitUSART(void)
{
UBRR1L=51;

UCSR1B|=(1<<RXCIE1)|(1<<RXEN1)|(1<<TXEN1);

}


ISR(USART1_RX_vect)
{

cli();
while(!(UCSR1A&(1<<RXC)));
rxdata[i]=UDR1;

if(rxdata[i]==';')
{
i=0;
execute_command();}
else i++;

sei();
}


voidexecute_command(void)
{
        LCDstatus("wait...",7);
        switch(rxdata[0])
        {
        case 's': //setting the angles...
                setanglefunc();
                LCDstatus("Done...",7);

                break;

        case 'd': //delay x seconds
                break;

        case 'g': //gripper on/off
                break;
        case 'c': //conditional input
                break;
```

```
		case 'i': //wait for input
			break;

		default:;
		}
}

voidLCDstatus(char *charray, int *n)
{
		LCDclr();
		LCDGotoXY(0,0); _delay_ms(20);
		LCDstring(charray,n);
}


//==================SEGMENTATION COMMANDS
(FUNCTIONS)=================

voidsetanglefunc(void)
{

int n=0, c=0, l=0, q[6]={0,0,0,0,0,0};
char temp[]="__________";

		while(rxdata[l]!=';')
		{
			l++;
			if(rxdata[l]==','| rxdata[l]== ';')
			{
			q[n]=atoi(temp);
			/*sprintf(str1, "%5d", q[n]);

			_delay_ms(30);
			LCDstring(str1, 5);*/
			//_delay_ms(1000);

			n++;

			for(int d=0; d<=c; d++) temp[d]='_';
			c=0;
			}

			else
			{
			temp[c]=rxdata[l];
			c++;
			}

		}

		set_angles(q[0],q[1],q[2],q[3],q[4]);

}
```

## A-1-2 MATLAB Code (GUI)

```
functionvarargout = FK_general(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @FK_general_OpeningFcn, ...
                   'gui_OutputFcn',  @FK_general_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
ifnargin&&ischar(varargin{1})
gui_State.gui_Callback = str2func(varargin{1});
end

ifnargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
gui_mainfcn(gui_State, varargin{:});
end

functionFK_general_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

clc;
% UIWAIT makes FK_general wait for user response (see UIRESUME)
% uiwait(handles.figure1);



% --- Outputs from this function are returned to the command line.
functionvarargout = FK_general_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;



functionw_roll_Callback(hObject, eventdata, handles)

value = str2num(get(handles.w_roll, 'string'));
set(handles.sendcommand, 'enable', 'off');
if value >45 || value <-45
set(handles.w_roll, 'string', '0');
set(handles.angle_status, 'string', strcat('Shoulder: Value out of range.', get(handles.angle_status,
'string')));
end

% --- Executes during object creation, after setting all properties.
functionw_roll_CreateFcn(hObject, eventdata, handles)

ifispc&&isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
```

```
functionw_pitch_Callback(hObject, eventdata, handles)
% hObject    handle to w_pitch (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
value = str2num(get(handles.w_pitch, 'string'));
set(handles.sendcommand, 'enable', 'off');
if value >0 || value <-135
set(handles.w_pitch, 'string', '0');
set(handles.angle_status, 'string', strcat('Shoulder: Value out of range.', get(handles.angle_status,
'string')));
end
% Hints: get(hObject,'String') returns contents of w_pitch as text
%        str2double(get(hObject,'String')) returns contents of w_pitch as a double


% --- Executes during object creation, after setting all properties.
functionw_pitch_CreateFcn(hObject, eventdata, handles)

ifispc&&isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end



functionelbow_Callback(hObject, eventdata, handles)

value = str2num(get(handles.elbow, 'string'));
set(handles.sendcommand, 'enable', 'off');
if value >135 || value <-30
set(handles.elbow, 'string', '0');
set(handles.angle_status, 'string', strcat('Elbow: Value out of range.', get(handles.angle_status,
'string')));
end

% --- Executes during object creation, after setting all properties.
functionelbow_CreateFcn(hObject, eventdata, handles)

ifispc&&isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end



functionshoulder_Callback(hObject, eventdata, handles)

value = str2num(get(handles.shoulder, 'string'));
set(handles.sendcommand, 'enable', 'off');
if value >30 || value <-90
set(handles.shoulder, 'string', '0');
set(handles.angle_status, 'string', strcat('Shoulder: Value out of range.', get(handles.angle_status,
'string')));
end

% --- Executes during object creation, after setting all properties.
```

```
functionshoulder_CreateFcn(hObject, eventdata, handles)

ifispc&&isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end



functionbase_Callback(hObject, eventdata, handles)

value = str2num(get(handles.base, 'string'));
set(handles.sendcommand, 'enable', 'off');
if value >71 || value <-71
set(handles.base, 'string', '0');
set(handles.angle_status, 'string', strcat('Base: Value out of range.', get(handles.angle_status, 'string')));
end


% --- Executes during object creation, after setting all properties.
functionbase_CreateFcn(hObject, eventdata, handles)

ifispc&&isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end



functionfinger_distance_Callback(hObject, eventdata, handles)

value = str2num(get(handles.finger_distance, 'string'));
set(handles.sendcommand, 'enable', 'off');
if value >60 || value <0
set(handles.toollength, 'string', '0');
set(handles.angle_status, 'string', strcat('Shoulder: Value out of range.', get(handles.angle_status,
'string')));
end

% --- Executes during object creation, after setting all properties.
functionfinger_distance_CreateFcn(hObject, eventdata, handles)

ifispc&&isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end



functionangle_status_Callback(hObject, eventdata, handles)

functionangle_status_CreateFcn(hObject, eventdata, handles)

ifispc&&isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton1.
```

```
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in test_trans.
functiontest_trans_Callback(hObject, eventdata, handles)

globalstrarray;
globalhome_angles;

d=[117, 0, 0, 0, 76 + str2double(get(handles.toollength, 'string'))];
a=[0 196 148.9 2.5 0];
alpha=[-90 0 0 -90 0];
home_angles=[0 -90 90 0 -45];

m=[9.1549 12.6667 10.2222 -9.5556 -11.1111];
c=[1500, 1900, 980,750, 1500];

q(1)=str2double(get(handles.base, 'string'));
q(2)=str2double(get(handles.shoulder, 'string'));
q(3)=str2double(get(handles.elbow, 'string'));
q(4)=str2double(get(handles.w_pitch, 'string'));
q(5)=str2double(get(handles.w_roll, 'string'));


for k=1:5
M1=[cosd(q(k)), -cosd(alpha(k)).*sind(q(k)), sind(alpha(k)).*sind(q(k)), a(k).*cosd(q(k))];
M2=[sind(q(k)), cosd(alpha(k)).*cosd(q(k)),  -sind(alpha(k)).*cosd(q(k)), a(k).*sind(q(k)) ];
M3=[0 sind(alpha(k)) cosd(alpha(k)) d(k)];
M4=[0 0 0 1];

switch k
case 1
      T_1_0=[M1; M2; M3; M4];
case 2
      T_2_1=[M1; M2; M3; M4];
case 3
      T_3_2=[M1; M2; M3; M4];
case 4
      T_4_3=[M1; M2; M3; M4];
case 5
      T_5_4=[M1; M2; M3; M4];
otherwise ;
end

end




T_sh__base=T_1_0;
T_elbow_base=T_1_0*T_2_1;
T_pitch_base=T_1_0*T_2_1*T_3_2;
T_roll_base=T_1_0*T_2_1*T_3_2*T_4_3;
T_tool_base=T_1_0*T_2_1*T_3_2*T_4_3*T_5_4;
```

```
T=T_tool_base;

p=m.*q+c; %angle conversion
p=round(p);

strarray = strcat('s',num2str(p(1)),',',num2str(p(2)),',',num2str(p(3)),',',num2str(p(4)),',',num2str(p(5)),';')

set(handles.sendcommand, 'enable', 'on');

%##########################################################################

% --- Executes on button press in sendcommand.
functionsendcommand_Callback(hObject, eventdata, handles)

global s;
globalstrarray;
disp('Key was pressed!')
fwrite(s, strarray, 'char');

% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)

function popupmenu2_CreateFcn(hObject, eventdata, handles)

ifispc&&isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in initserport.
functioninitserport_Callback(hObject, eventdata, handles)

global s;
set(handles.initserport, 'Enable', 'off');
set(handles.closeserport, 'Enable', 'on');
s=serial('COM6');
s.baudrate=9600;
fopen(s);

% --- Executes on button press in closeserport.
functioncloseserport_Callback(hObject, eventdata, handles)

global s;
set(handles.initserport, 'Enable', 'on');
set(handles.closeserport, 'Enable', 'off');
fclose(s);
delete(s);
clear s;

% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over initserport.
functioninitserport_ButtonDownFcn(hObject, eventdata, handles)

function figure1_CloseRequestFcn(hObject, eventdata, handles)
```

```
delete(hObject);



functiontoollength_Callback(hObject, eventdata, handles)

value = str2num(get(handles.toollength, 'string'));
set(handles.sendcommand, 'enable', 'off');
if value >70 || value <0
set(handles.toollength, 'string', '0');
set(handles.angle_status, 'string', strvcat('Shoulder: Value out of range.', get(handles.angle_status,
'string')));
else set(handles.tot_toollength, 'string', strcat('Total tool length: ',num2str(76 +
str2double(get(handles.toollength, 'string')))));
end



% --- Executes during object creation, after setting all properties.
functiontoollength_CreateFcn(hObject, eventdata, handles)

ifispc&&isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in clear.
functionclear_Callback(hObject, eventdata, handles)

functionhome_btn_Callback(hObject, eventdata, handles)

globalhome_angles;
global s;

m=[9.1549 12.6667 10.2222 -9.5556 -11.1111];
c=[1500, 1900, 980, 750, 1500];

q=home_angles;
p=m.*q+c; %angle conversion
p=round(p);

strarray =
strcat('s',num2str(p(1)),',',num2str(p(2)),',',num2str(p(3)),',',num2str(p(4)),',',num2str(p(5)),';');
fwrite(s, strarray, 'char');
% --- Executes on button press in rest_btn.
functionrest_btn_Callback(hObject, eventdata, handles)

global s;

rest=[-65 30 -30 -90 -45];
m=[9.1549 12.6667 10.2222 -9.5556 -11.1111];
c=[1500, 1900, 980,750, 1500];

q=rest;
p=m.*q+c; %angle conversion
p=round(p);
```

```
strarray =
strcat('s',num2str(p(1)),',',num2str(p(2)),',',num2str(p(3)),',',num2str(p(4)),',',num2str(p(5)),';');
fwrite(s, strarray, 'char');


% --- Executes on selection change in select_graph.
functionselect_graph_Callback(hObject, eventdata, handles)

functionselect_graph_CreateFcn(hObject, eventdata, handles)

ifispc&&isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end
```

## A-2 User Guide And General Information

**TURNING THE MACHINE ON/OFF:**

There is a button on the left upper corner of the robotic arm box's panel. Along with the button, there is a three-pin 220VAC socket. It must be plugged on first. The machine can be turned **ON** by the illuminated switch model KCD-4. If the switch is illuminated it will show that the machine has been turned **ON**. Always connect serial port with PC before turning on the robotic arm. Serial port is also on the same panel where ON/OFF switch and AC socket are attached.

The following figure shows the arrangement of the three components:

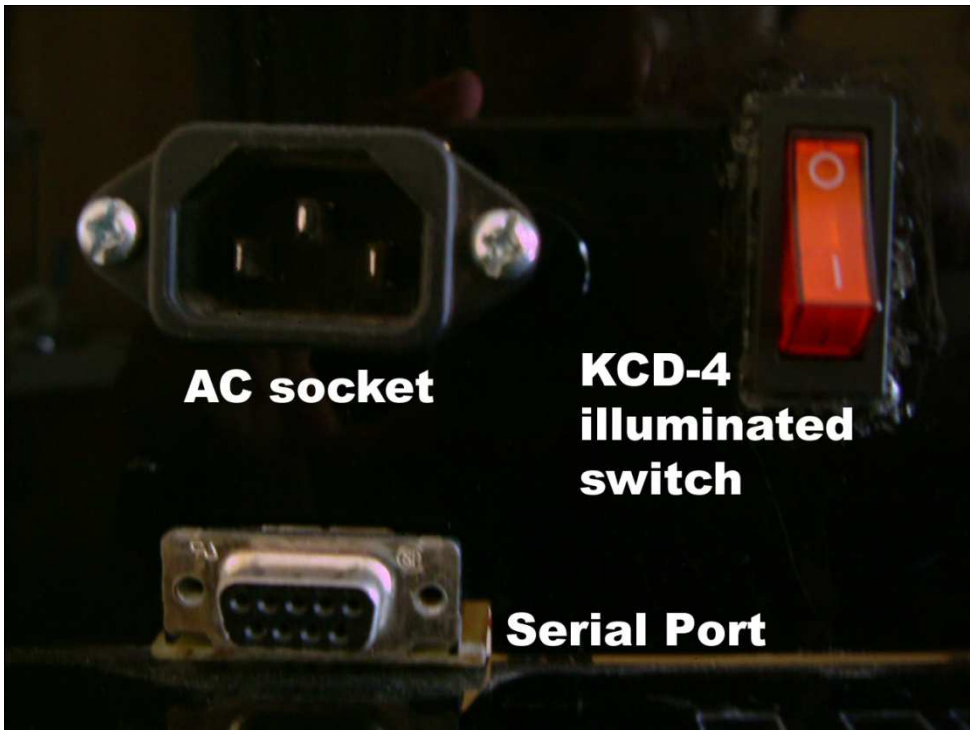


**Figure 6: Front panel of the trainer**

**THE ROBOT'S WORKSPACE AREA:**

The raised platform on the box is the robot's workspace area. All the lab procedures and experiments with self-made OR built-in modules can be performed over this area. The Dimensions of the workspace area is

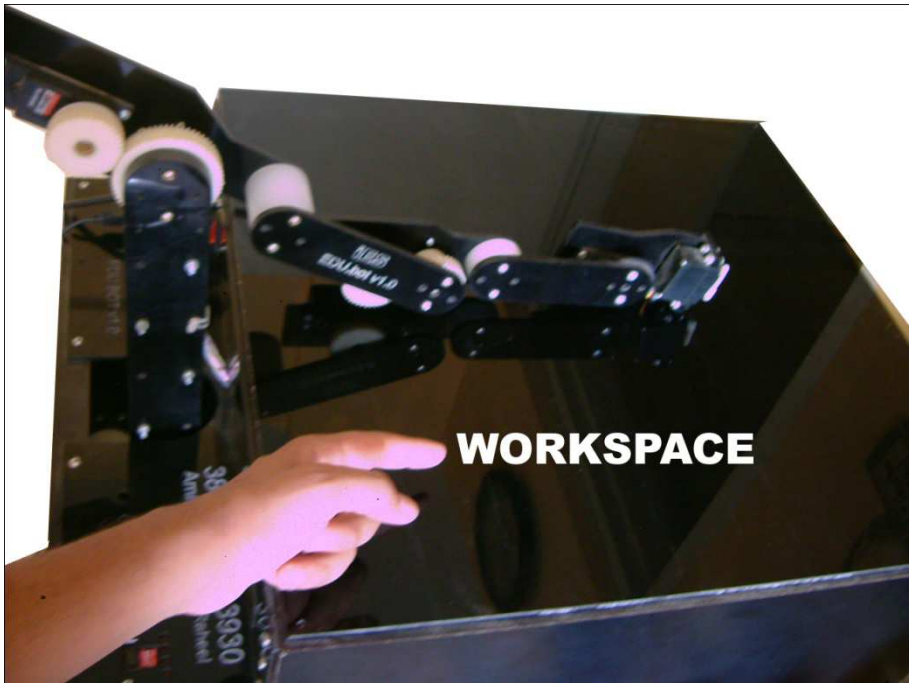


**Figure 7: Workspace of the trainer**

## EXTERNAL INPUTS AND OUTPUTS

4-Bit external inputs and outputs have been provided to interface the robotic arm's controller and controller software with the external world. For example a sensor assembly placed on the workspace of the robotic arm can interchange bits with the robotic arm's control-board for the purpose defined by the user.

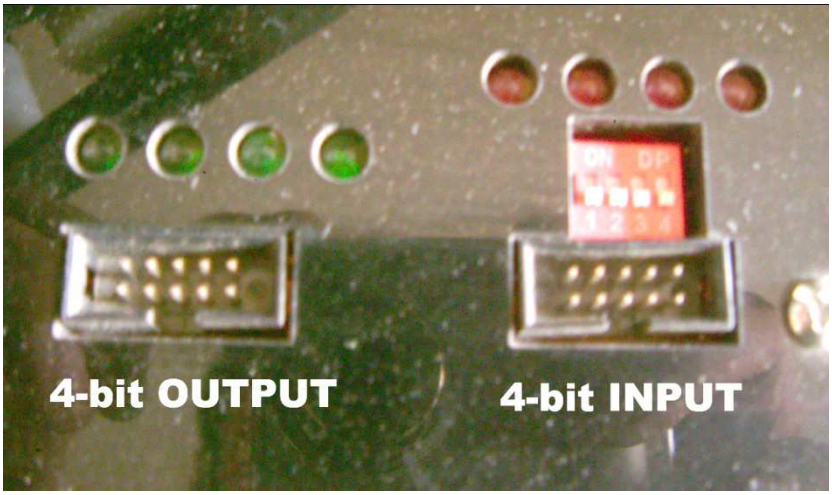


**Figure 8: Input and output panel**

Moreover, the DIP switches on inputs are also provided for manual testing of user program for the IES Edu.bot v1.0. LEDs for each of the 4-bit input and 4-bit output headers are provided to clearly show the values on the given External I/O port. Red LEDs represent 'Inputs', while green LEDs represent 'Outputs'.

## BREADBOARD

Bread board on the workspace is also provided for helping the student in making custom sensor assemblies to interface with the IES Edu.bot v1.0. The breadboard is provided in a smaller sized model.

**Figure 9: Mini breadboard for optional working**

## MOTORS

The motors used for actuating the robotic arm are RC servo motors made by HITEC, a leading company in the manufacturing of RC SERVO motors. All motors run at voltages under 6V.

The motors used are of following models and should only be replaced with the stated models below:

Base : HS-805BB or HS-805MG

Shoulder : HS-805BB or HS-805MG

Elbow : HS-7955TG OR HS-7955MG

Wrist pitch : HS-5645MG

Wrist Roll : HS-5485HB

## CONTROL CIRCUIT AND POWER SUPPLIES

The control circuit is based on ATMEGA128 microcontroller. It is located inside the box. Moreover an RS-232 level converter circuit is also installed inside. Power supplies are separate for motors and the control board. For motors; power-supply not more than 6V should be used as replacement. For control board; power-supply not more than 5V should be used as replacement.

## SOFTWARE FOR CONTROL AND OPERATION BASED ON FORWARD KINEMATICS

The basic software for Robot operations and movements based on Forward Kinematics is fairly easy to use. All the Text Boxes are provided for robotic arm axis. Angles can be put in those text boxes and the transformation matrices for all joints can be tested. The software also provides limiting angle values for each axis and does not accept an angle out of that range.

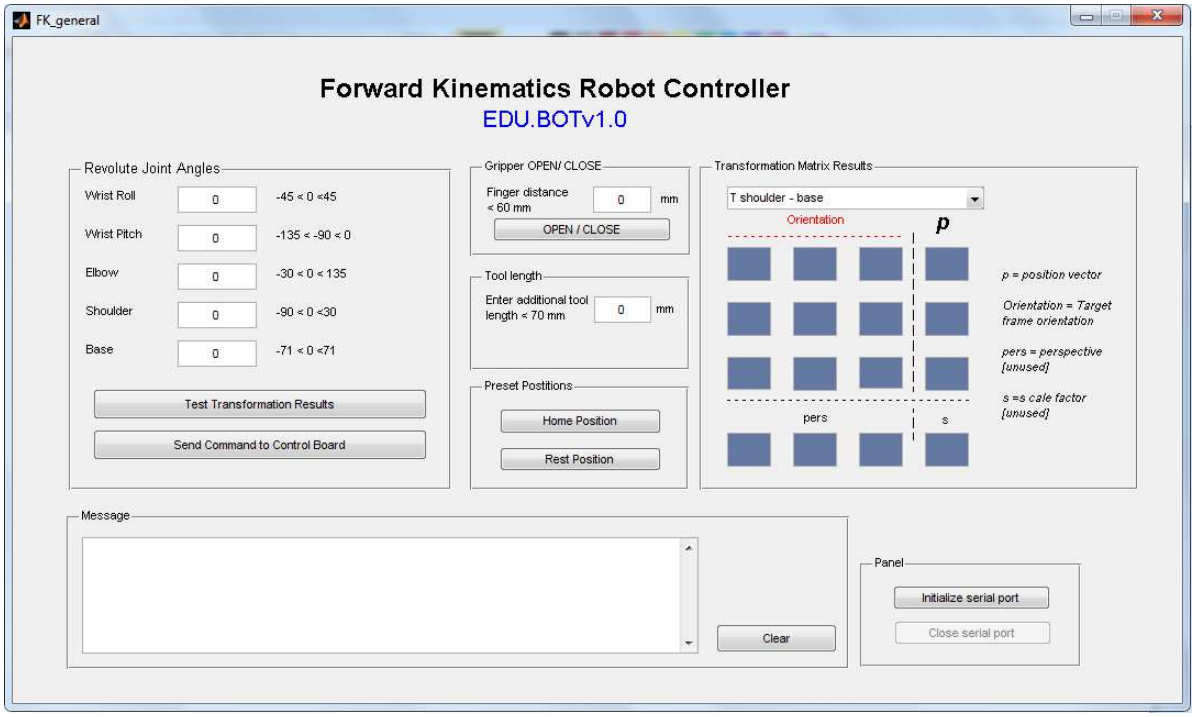


**Figure 10: Forward kinematics software snapshot**

The Pop-up-menu contains all transformation matrices, the user can choose. Those matrices are analysed by the software and warnings are shown if the robot is going to hit the work surface. If any angle is also out of range as stated above; the software displays a message in the MESSAGEBOX given on the GUI.

If the user has connected any tool e.g. gripper the user can enter the length of the tool in the text box dedicated for tool length, so that the software can adjust its calculation in the Transformation matrices.

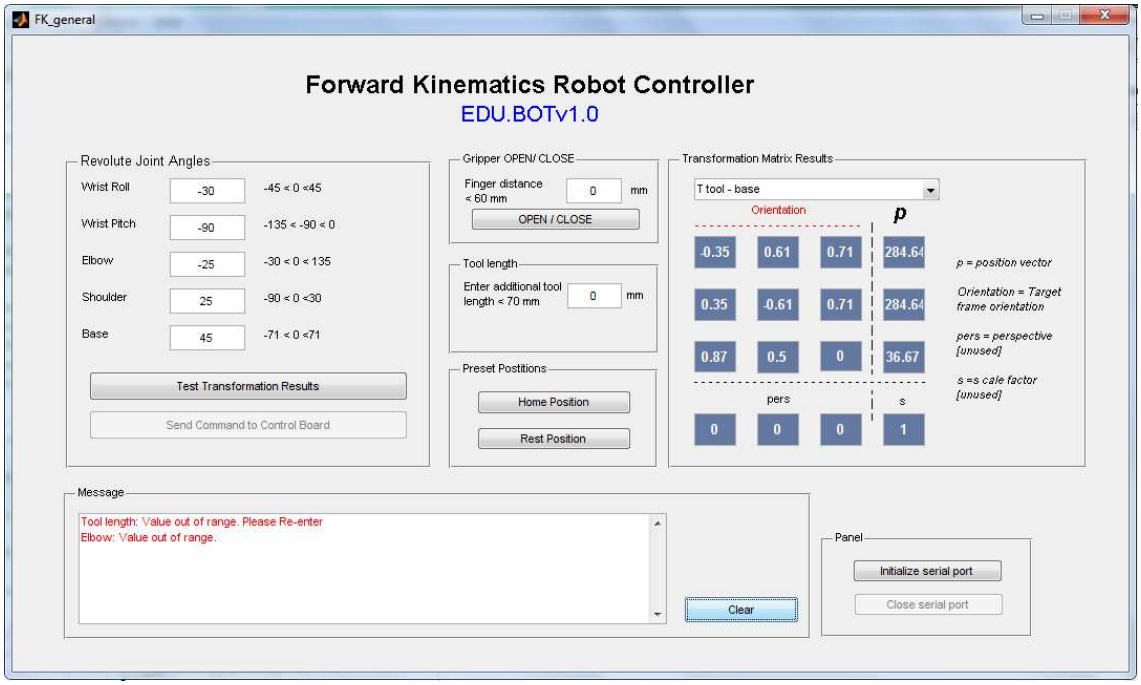


**Figure 11: Forward kinematics software showing the error**

The image shown above displays the result of a transformation test done by the user. The tool length was greater or less than allowed, therefore the software showed an error message in the message box. Similarly for elbow, the message is shown as “value out range” and the entered value is reset to ‘0’.

## *Safety Precautions*

- Do not stop/obstruct the movement of the robotic arm forcefully, otherwise motor's internal calibration will be disturbed and it may become completely inaccurate.
- Obstructing the movement of robotic arm may also cause motors' internal gear train to be broken off.
- Do not touch or grab the robotic arm or place your fingers or palm on the external coupling gears, it may injure your hand or cause serious wounds..
- Do not move robotic arm with hands unnecessarily/abruptly, while powered off/on.
- Only use appropriate power supplies. Power supply for servo motors should be separate. Power supply for the control circuitry should not be greater 5V and that for servo motors should not be greater than 6V.
- Keep out of radius of operation of the robotic arm.
- Do not turn off power or disconnect the cable unless the robotic is in its rest position. This is because the arm can move unprecedentedly and may fall down abruptly to receive any damage.
- Do not try to send in any further commands from the GUI while one command and operation is in progress.
- Maintenance and service must be done by qualified technicians or engineers. A layman must not open the box and disassemble the arm.
- Electric shock hazard inside the box, strictly follow proper servicing and handling procedures.
- Servicing and maintenance must be done in ESD (Electrostatic Discharge) protected environments otherwise it might burn out the SMD microcontroller chip.
- ALWAYS PLACE THE ROBOTIC ARM IN THE ARM REST POSITION after operation, by pressing the Arm Rest Position button on the GUI.

## A-3 Datasheets

**Atmega128**

[See the next page for the datasheet]

**Max232**

[See the next page for the datasheet]