IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE)
e-ISSN: 2278-1676,p-ISSN: 2320-3331, Volume 6, Issue 5 (Jul. - Aug. 2013), PP 80-84
www.iosrjournals.org

Design and Implementation of Robot Arm Control Using LabVIEW and ARM Controller

Mr. C. Chandra Mouli[1], Ms. P. Jyothi[1], Prof. K. Nagabhushan Raju[2], Prof. C. Nagaraja[2]

[1] *Research Scholar of Instrumentation Department, Sri Krishnadevaraya University, Anantapur, A.P., INDIA*
[2] *Department of Instrumentation, Sri Krishnadevaraya University, Anantapur, A.P., INDIA*

Abstract : *The design, analysis and implementation of a robot arm system, which is expressed towards its performance with an analytical model by using LabVIEW and embedded system tools was presented in this article. Mathematical modeling of kinematics plays an essential role in design and implementation of a robot arm control. The projected work was focused to control the end effector of the robot arm to achieve any accessible point in an amorphous region using LabVIEW, ARM (Advanced RISC (Reduced Instruction Set Computing) Machine) microcontroller and Dexter ER2 robotic arm. LabVIEW uses analytical method to design the inverse kinematic model of the robot arm. The inverse kinematic model and robot arm control was implemented using LabVIEW and ARM microcontroller. LabVIEW uses the parallel communication to send the joint angles of the robot arm to the ARM. ARM microcontroller uses five PWM (Pulse Width Modulation) signals in order to control the robot arm, which was geared up with servo motors. Robot arm was controlled manually through the LabVIEW GUI (Graphical User Interface) controls. The present paper discussed about the mechanical configuration, analytical modeling, software and hardware of the above said work.*
Keywords: *Robot Arm, LabVIEW, ARM microcontroller, Parallel Communication, Inverse Kinematics.*

I. INTRODUCTION

Since 50years robots are in use in variety of industries. The most important applications of industrial robots are material handling, welding, assembling, dispensing and processing where the robotic arm manipulator needs to perform pick and place operations incessantly. One of such industrial standard robots is a generic serial arm which consists of a base, a link or series of links connected at joints, and an end effector. Generally, end effector is a gripper which is at the end of the last link and the base is the first link of a serial arm [2].

Modeling, analysis and implementation of a serial robot arm implicates the study of its kinematic behavior [3]. Kinematics gives the motion of bodies without concern of the forces or moments that cause the motion. Analytical study of the motion of a robot manipulator is called robot kinematics. Modeling the right kinematics is important for analyzing the actions of a robot manipulator. Robot kinematics has been divided into forward kinematics and inverse kinematics [1]. The problem of solving the Cartesian position and orientation of a robot arm which gives the knowledge of the kinematic structure and joint coordinates is called forward kinematics.

The problem of computing the joint variables through the position and orientation of robot's end-effector is called inverse kinematics (IK) [3]. IK problem is more complex than forward kinematic problem in case of serial robotic arm [1]. Many researchers have evaluated and executed these problems in different scenarios using different tools and devices.

Some researchers frequently use geometric methods for the serial manipulators which are relatively simple geometry [4]. Past work on a 5-axes articulated robot arm kinematic model was designed with a homogenous 4 x 4 matrix calculation [5]. A low cost 4 DOF robot arm was designed using LabVIEW where joint angles are calculated and transmitted to the microcontroller through LabVIEW [6]. To optimize the inverse kinematics problem of robot arm, an optimization process was carried out using neural networks and LabVIEW for simulation [7]. In the present work to control the 5DOF robot arm, its modeling and analysis was designed by using analytical method, due to its versatility and acceptability to solve the kinematic model of any number of joints and links of a serial manipulator regardless of complexity [3].

The present work provides the evaluation of inverse kinematics using LabVIEW where the joint angles are carried out. The joint angles are transmitted to the ARM microcontroller. ARM controller receives the joint angles and controls the robot arm.

II. MECHANICAL CONFIGURATION

The mechanical configuration describes the physical components used to fabricate the robot arm in the present work. The robot arm used in the present work is Dexter ER-2 heavy duty robotic arm which is fully assembled using servo motors.

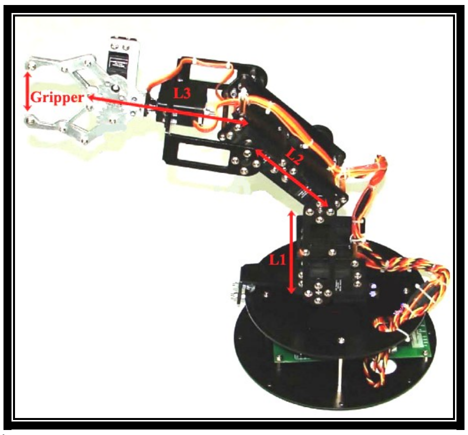

Fig. 1. **Robot Arm with Link Measurements**

The robotic arm is a 5 axis robotic arm plus servo gripper. It uses seven metal gear NRS-995 6VDC servo motors with 15Kg/cm torque and two NRS-585 6VDC servo motors with 4kg/cm torque. Robot arm has 5 degrees of freedom which includes base rotation, shoulder rotation, elbow rotation, wrist pitch and roll. Out of which shoulder rotation, elbow rotation, wrist pitch has two 15Kg/cm torque servo motors in parallel for enhancing the performance of the robotic arm.

NRS-995 is a high torque metal gear servo with dual ball bearings and NRS-585 is a low cost, high performance servo motor. NRS-995 produces 15.5kg/cm torque at 4.8V and NRS-585 produces 4kg/cm torque at 6V which has built in motor, gearbox, position feedback mechanism and motor controller. The servo motor can be controlled to move any position just by using simple pulse controlling with three wire interface for control and power supply.

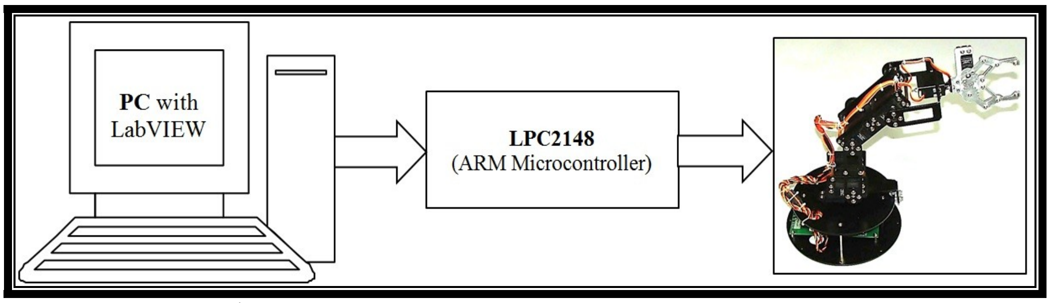

Fig. 2. **Block Diagram of the 5DOF robot arm control**

The mechanical structure of the robot arm is vertical articulated with 5 axes and one servo gripper. As shown in figure 1, the robot arm consists of six rotational joints from base to griper and the actuators are denoted from 1 to 6 respectively. It has four links L1, L2 and L3 with lengths 9cm, 8cm and 13.8cm respectively. The gripper can open its tooth up to 5.5cm.

III. HARDWARE DESCRIPTION

Figure 2 shows the block diagram of a 5DOF robot arm control using LabVIEW and ARM controller. The hardware setup contains a PC (Personal Computer), Arm controller (LPC2148) and Dexter ER-2 a heavy 5DOF robotic arm. PC used in the present work is DELL Optiplex380 which is capable of interfacing a device through parallel communication and serial communication. LPC2148 is an ARM7TDMI-S based high-performance 32-bit RISC Microcontroller with 6 PWM channels. The description of robot arm was discussed in the mechanical configuration section. Dell Optiplex 380 consists of one parallel printer port with the base addresses of 0x338 and 0x778. A device can be interfaced to the PC using DB25 parallel printer port connector at the rear end of the computer.

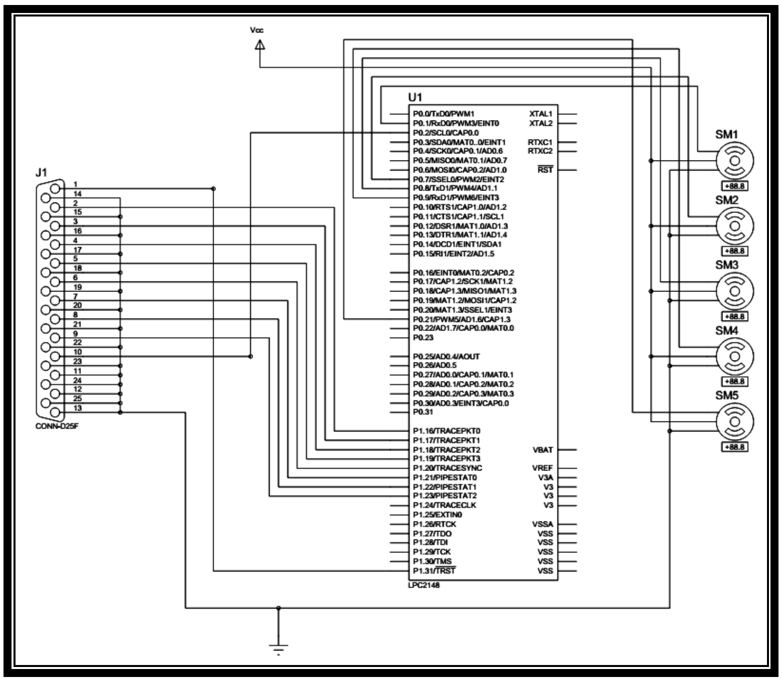

Fig. 2. **Schematic of Robot Arm Control Using LabVIEW and ARM Controller**

Fig. 3 shows the schematic of the present work. As shown in the schematic conventional printer port comes with 25 pins of which 8 data lines D0-D7, 9 control lines and rest of the pins are connected to ground. Eight data pins D0 to D7 are used to transmit the data from PC to ARM microcontroller. Pins D0 to D7 was connected to the port pins P1.16 to P1.22 of ARM microcontroller respectively. Strobe line (nStrobe) is connected to the pin P1.31 of ARM microcontroller as one of the control pin from PC. ARM microcontroller pin P0.2 is connected to the Acknowledge pin (ACK) of PC as one of the control pin from controller. Hence nStrobe, ACK, P1.31 and P0.2 acts as control lines for data synchronization between PC and ARM microcontroller. The remaining lines are connected to the ground. Port pins P0.1, P0.7, P0.8, P0.9 and P0.21, which generates PWM signals were connected to robotic arm servo motors SM1, SM2, SM3, SM4, SM5 respectively. Here these pins are assigned as PWM generating signal through LabVIEW software. Each servo motor consists of +5V, GND and control pins. The port pins from ARM microcontroller were connected to the control pin of servo motors.

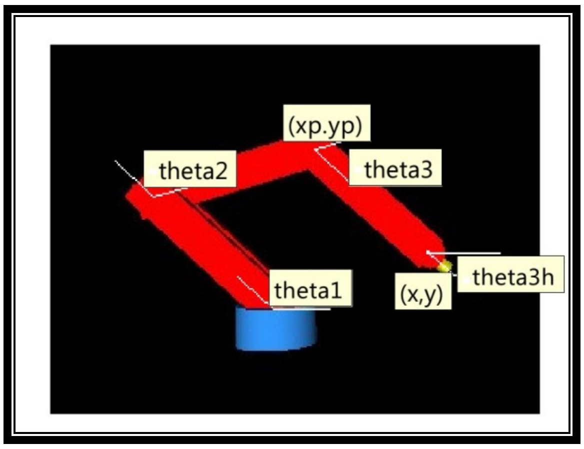

Fig. 4. **Robot Arm with Joint Angle Variables**

IV. SOFTWARE DESCRIPTION

The software of the present work includes robot arm modeling and programming. Robot arm modeling was done using LabVIEW and its programming was done using embedded C language. IK program for the present work was established using the example program of robotics in LabVIEW. The output of the IK program gives three joint angles of robot arm excluding robot arm base and gripper. The joint angles are transmitted by using parallel printer port and the program for parallel communication was written and integrated into the inverse kinematics program.

IK model calculates the joint angles to the desired position and orientation. Analytical method was followed to develop IK model. This method determines the correct joint angles for any object in the workspace of the robot arm. The three joint angles i.e. waist (*theta1*), shoulder (*theta2*), elbow (*theta3*) is calculated using this method while the base (*thetab*) and end effector (*thetag*) is straight away given for desire position.

Figure 4 shows the robot arm with joint angle variables. The required coordinates for robot arm end effector position is in the form (*x, y, thetab, thetag*). Where x, y gives the position of robot arm end effector with respect to the base coordinates. The variable *thetab* is used as input variable to rotate the base and *thetag* is used to open or close the robot arm gripper. The VI calculates values for theta1, theta2, theta3 that will put the end effector at the desired coordinates (*x, y*) and transmits the values of *thetab, theta1, theta2, theta3* and *thetag* simultaneously one by one with 40ms delay using parallel printer port. Formulas used in the sub VI *IK* to evaluate the desired joint angles *theta1, theta2* and *theta3* are shown below.

$$\begin{aligned}
&\mathrm{thetai} = \mathrm{atan\,2(y, x)}\\
&\mathrm{theta\,3\,h} = \mathrm{thetai} + \mathrm{pi}/\,2 * ((\mathrm{x}**2 + \mathrm{y}**2)\,/\,(\mathrm{L1} + \mathrm{L\,2} + L3)**2)**(0.5) - \mathrm{pi}/\,2\\
&\mathrm{xp} = \mathrm{x} - \mathrm{L\,3} * \cos(\mathrm{theta\,3\,h})\\
&\mathrm{yp} = \mathrm{y} - \mathrm{L\,3} * \sin(\mathrm{theta\,3\,h})\\
&\mathrm{c} = (\mathrm{xp}**2 + \mathrm{yp}**2 - \mathrm{L1}**2 - \mathrm{L\,2}**2)\,/\,(2*\mathrm{L1}*\mathrm{L\,2})\\
&\mathrm{alpha} = \mathrm{acos}(-\mathrm{c})\\
&\mathrm{theta\,2} = \mathrm{alpha} + \mathrm{pi}\\
&\mathrm{theta\,1} = \mathrm{atan\,2(yp, xp)} + \mathrm{asin}(\mathrm{L\,2} * \sin(\mathrm{alpha})\,/\,\mathrm{xp}**2 + \mathrm{yp}**2)**(0.5))\\
&\mathrm{theta\,3} = \mathrm{theta\,3\,h} - \mathrm{theta\,1} - \mathrm{theta\,2}
\end{aligned}$$

The sub VI *IK* was developed from the formula node that gives the detailed evaluation description of inverse kinematics for robot arm. Two input constraints *x* and *y* are used to determine three angles. *thetai* is the angle from the base to the end effector, *theta3h* is the position of the arm based on the distance from the base, *xp* and *yp* are the proximal joint of the last link and alpha is just a temporary variable. *theta1* and *theta2* is solved using the cosine rule and *theta3* solved from *theta1, theta2* and *theta3h*.

The most important phase in the servo motor position control is the software coding of the ARM controller. The PWM signal was generated from the pin P0.7 which is the output of PWM1 of LPC2148 ARM controller. After a deep investigation of servo motors, 0^{o} angular position can be achieved with 600us on-time in total time period of 20ms and 180^{o} with 2.2ms on-time in total time period of 20ms.

PWM signals were generated on the corresponding pins of LPC2148 by using the registers PINSEL, PWMPR, PWMPCR, PWMMCR, PWMMR0, PWMMR1, PWMMR2, PWMMR3, PWMMR4, PWMMR5, PWMLER and PWMTCR. The present work five PWM signals are generated. The PWM function for IO pins were selected by programming the control word to PINSEL registers. PWMPR is a PWM Prescale Register in which the value is incremented by PWMPR+1 for every phase locked loop clock cycle. PWMPCR is a PWM control register which enables the PWM output and controls the PWM channel type as either single edge or double edge. In the present work the PWM channel is controlled by using single edge. PWMMCR is a PWM Match Control Register and controls the PWM signal and if the PWMTC is reset when a Match occurs. PWMLER is PWM Latch Enable Register which enables the use of new PWM match values. PWMTCR is PWM Timer Control Register which is used to control the timer counter functions and timer counter can be disabled or reset. PWMMR x (where x is 0/1/2/3/4/5) are the match control registers which controls the total time period of the PWM and on-time of the pulse.

The total time period and on-time of one pulse depends on the values programmed to PWMMR0 and PWMMRx (where x is 1/2/3/4/5) respectively. In the present work 12MHz crystal is used for the controller crystal frequency. The crystal frequency is multiplied by a constant value 5 to get the processor frequency of 60MHz with the help of phased lock loop in the controller. The value in the registers is modified using this processor frequency of 60MHz.

$$PWMMR0 = Fcclk\,/\,83 \tag{1}$$

$$PWMMR1 = ((\mathrm{Fcclk}/\,83)\,\mathrm{X}(100 - \mathrm{count}))\,/\,100 \tag{2}$$

Equation (1) shows the formula to calculate the value to be programmed in PWMMR0 register to get total time period of one pulse. Here *FFclk* is user defined macro which is used to represent the frequency of

60MHz. To get the 1Hz frequency pulse the value of the denominator should be 1.66 constant values. The denominator value of (1) is calibrated to 83 to get the 50Hz frequency pulse i.e. 20ms pulse. Equation (2) shows the formula to calculate the value to be stored in PWMMR1 register to get the on-time of the pulse. Here the value of PWMMR0 i.e. *Fcclk/83* is multiplied with *100-count* value, where *count* is the percentage of on-time with respect to the duty cycle of the pulse. The product of these values is going to be divided with 100 to get the value into PWMMR1. The on-time of the pulse is decided by the variable, *count* and the formula is shown in equation (3).

$$count = (initialtime / totaltimeperiod) \times 100 \tag{3}$$

Equation (3) shows the formula to calculate the percentage of on-time with respect to the duty cycle. To get 0^o angular position of a servo motor a 600µs on-time is required. According to equation (3) the *count* value for 600µs on-time is 3. Then the *count* value is incremented by 0.044 with an on-time of 8.8µs and keeping the total time period as 20ms to get 1^o angle resolution.

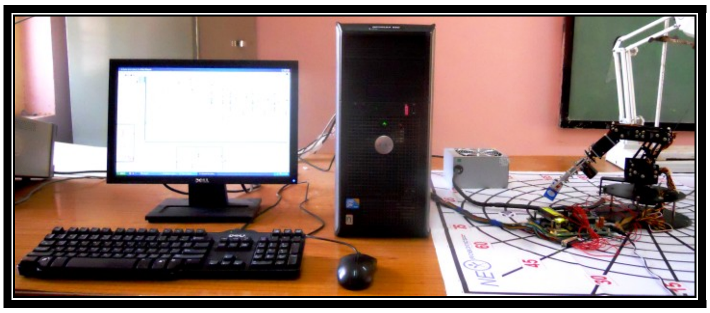
Fig. 5. **Photograph of Robot Arm Control Using LabVIEW and ARM Controller**

The communication between PC and LabVIEW was established and synchronized by using nStrobe and ACK lines of parallel printer port and P1.31 and P0.2 of ARM microcontroller. PC transmits the data of joint angles by setting the nStrobe pin high, which means that the data is ready to transmit on data lines of parallel printer port. ARM microcontroller receives the data of five joint angles one by one with a delay of 40ms for each joint angle. LabVIEW uses *Flat Sequence Structure* to send the data with a delay of 40ms. After receiving the data ARM microcontroller acknowledges PC by resetting the pin P0.2, which is connected to the ACK pin of printer port of PC.

V. RESULTS AND CONCLUSIONS

Dexter ER-2 5DOF robotic arm, inverse kinematic model and control was developed using LabVIEW and ARM microcontroller. LabVIEW was used to calculate the IK model to evaluate the joint angles of the robot arm. The correct position of the end effector with respect to the base was achieved using the joint angles. The performance of the robot arm has been compared with the application of picking a general purpose Hi-Watt 9V battery and placing it in the predefined space which was shown in the Fig. 5. The following observations was done while teaching the robot arm.

- The value of *theta3h* was decided based on the distance from the base
- *theta3= thetai* gives the full extension of robot arm
- *theta3* approaches right angle when the robot arm is nearby base.

REFERENCES

[1] Mark S., Seth H. and Vidyasagar M., Robot modeling and control (John Wiley & Sons, 2006).

[2] http://zone.ni.com/reference/en-XX/help/372983C-01/lvrobogsm/robo_arm_definition/

[3] Jamshed Iqbal, Raza ul Islam, and Hamza Khan, Modeling and Analysis of a 6 DOF Robotic Arm Manipulator, Canadian Journal on Electrical and Electronics Engineering , 3 (6) , July 2012, 300–306.

[4] Tsai L.W., Robot Analysis: The mechanics of serial and parallel manipulators, (John Wiley & Sons, 1999).

[5] Sanjay Lakshminarayan, Shweta Patil, Position Control of Pick and Place Robotic Arm, International Conference On Engineering Innovation and Technology, ISBN : 978-93-81693-77-3, Nagpur, 1st, July, 2012.

[6] Ashraf Elfasakhany, Eduardo Yanez, Karen Baylon, Ricardo Salgado, Design and Development of a Competitive Low-Cost Robot Arm with Four Degrees of Freedom, Modern Mechanical Engineering, 1, November 2011, 47-55.

[7] Adrian Olaru and Serban Olaru, Optimization of the robots inverse kinematics results by using the Neural Network and LabVIEW simulation, 2011 International Conference on Future Information Technology, IPCSIT vol.13 2011, IACSIT Press, Singapore, 40-45.

[8] Serdar Kucuk and Zafer Bingul, Industrial Robotics: Theory, Modelling and Control, (ISBN 3-86611- 285-8, Germany, December 2006, Edited by: Sam Cubero).

[9] R. S. Hartenberg and J. Denavit, A kinematic notation for lower pair mechanisms based on matrices, Journal of Applied Mechanics, vol. 77, pp. 215–221, June 1955.

[10] Paul, Richard (1981). Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators.(Cambridge, MA: MIT Press. ISBN 978-0-262-16082-7).