ROBOTIC ARM MANIPULATOR FOR INDUSTRIAL PURPOSE

Report of Thesis-1

Submitted to the Department of Electronics and Electrical Engineering

of

BRAC University

By

MD.Adnan Al Moshi

Student ID: 09221002

MD.Eftakhairul Islam

Student ID: 09221001

Salwa Salam Cynthia

Student ID: 09221211

BRAC University, Dhaka, Bangladesh

(No Soft Copy)

15/4/10

ABSTRACT

Automation in industries helps in achieving the target of quality standardization and process visibility.In overseas the industries are automated using robotic arm which works efficiently. But in a developing country like ours, we can not afford to import those arms from overseas. Again our country is not technically sound enough to produce robotic arms commercially in large scale for industrial usage. Considering these aspects we have prepared a robotic arm manipulator which is efficient as well as cost effective.The robotic arm is able to pick and place objects controlled by three bipolar stepper motors. The motor will be operated in a closed loop (with feedback) system to reduce error.The operation of the Stepper Motor is controlled using RT-Linux to minimize the real-time error (jitter).Two or more robotic arms can parrallely work by the algorithm.This made the project more efficient as from one computer we can operate all the manipulators.Stepper motors are used because it gives the steps so precisely.the motor is operated in closed loop to.The theoretical results are confirmed with practical application.

TABLE OF CONTENTS

CHAPTER I. INTRODUCTION

1.1 What is Robotics

Robotics is the engineering science and technology of robots, and their design, manufacture, application, and structural disposition. Robotics is related to electronics, mechanics, and software. To perform high-precision jobs such as welding and riveting, robots are now widely used in factories. They are also used in special situations that would be dangerous for humans -- for example in cleaning toxic wastes or defusing bombs. The field of **robotics** is more practically defined as the study, design and use of robot systems for manufacturing a top-level definition relying on the prior definition of robot.

1.2 Introduction to Real time linux:

RT-Linux is an operating system in which a small real-time kernel coexists with the Posix-like Linux kernel. The intention is to make use of the sophisticated services and highly optimized average case behavior of a standard time-shared computer system while still permitting real-time functions to operate in a predictable and low-latency environment. In RT-Linux, all interrupts are initially handled by the Real-Time kernel and are passed to the Linux task only when there are no real-time tasks to run. In practice, the RT-Linux approach has proven to be very successful. Many applications appear to benefit from a synergy between the real-time system and the average case optimized standard operating system. For example, data-acquisition applications are usually composed a simple polling or interrupt driven real-time task that pipes data through a queue to a Linux process that takes care of logging and display. In such cases, the I/O buffering and aggregation performed by Linux provides a high level of average case performance while the real-time task meets strict worst-case limited deadlines. The operating system allows for great flexibility in such things as the characteristics of real-time tasks, communication, and synchronization.

CHAPTER I I

2.1 Basics Of Robotic Arm

The robotic arm manipulator (*Figure 1.1*) will be able to work in any environment which is basically independent on t0069me, emotions, and place. The whole operation will be controlled by a controller in windows or Real-Time Linux operating system. It will be able to sense any object in the proposed area and will start its operation. That is, it will sense the object first, then it will pick the object, and will rotate to any commanded direction (say right). After that it will execute the soldering and will

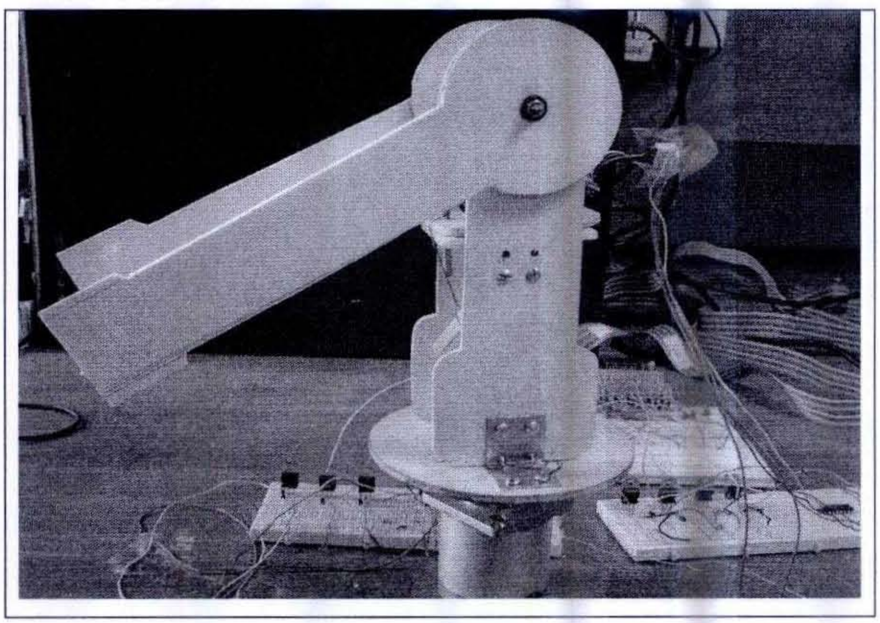


Figure:1 Robotic Arm

rotate in the direction of destination (say right) and finally, will place the object to its final destination. The robot motion will be controlled using stepper motors incorporated at the arm and the gripper respectively.

2.2 Industrial Purpose of a Robotic Arm:

Industrial Robots performs or assists to perform hazardous tasks in multiple industrial applications. Welding ,Soldering ,Material handling ,Thermal spraying Painting ,Drilling these are some tasks that can be done by a robot.

CHAPTER III Components Of Robotic Arm

3.1 PVC board

The PVC board has Less weight compared to iron or plastiv board that's why we choose PVC board to make the robotic arm. This is also inexpensive. For industrial use in our country we were looking for a material by which a robotic arm can be made easily.This board doesn't require any maintenance for rusting . It ensures inherent fire safety,Excellent durability and long-life expectancy .

3.2 Stepper Motor

A stepper motor (fig: 3.1) is an electromechanical device which converts electrical pulses into discrete mechanical movements. Steppers can be moved to any desired position reliably by sending them the proper number of step pulses.

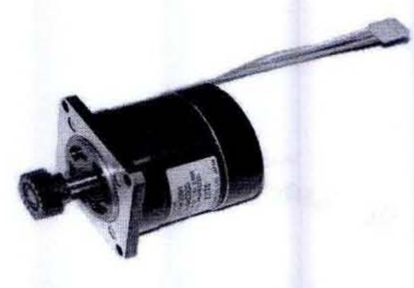
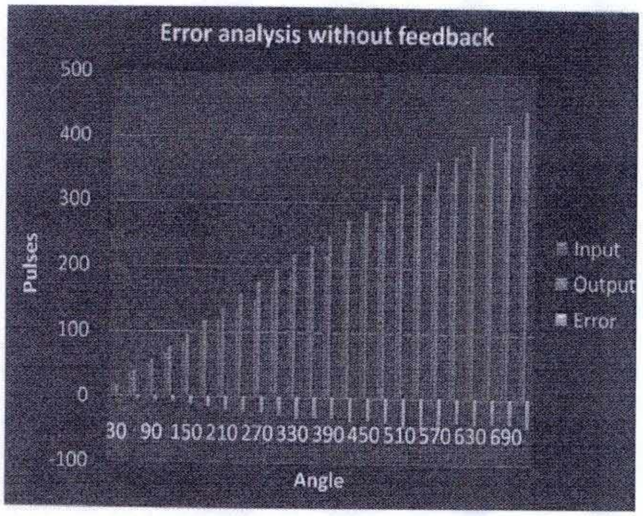


Fig 3.1: Stepper Motor

The motor will be operated in a closed loop (with feedback) system to reduce error. The following graphical representations show the error reduction of the closed loop mode when compared to the open loop mode. (Fig: 3.2 & Fig 3.3)

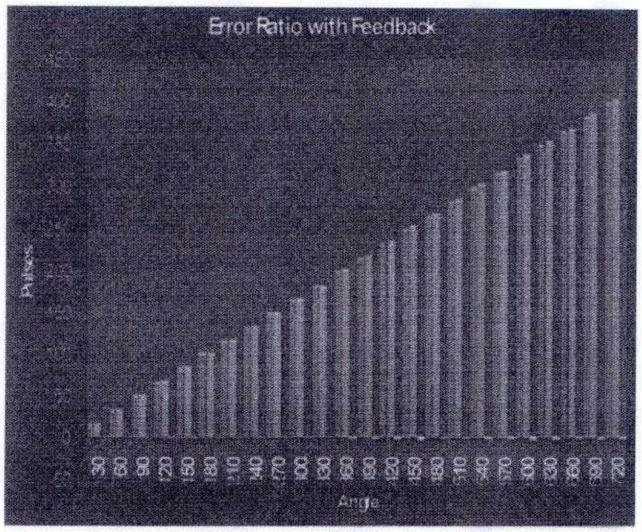


(a)

i/p angle	i/p pulses	Received pulses(avg.)
30	16	15.875
60	33	36.85
90	49	55.25
120	66	76.143
150	83	95.71
180	99	115.85

(b)

Fig3.2: (a) Output analysis of the open loop mode of the stepper motor; (b) The measurement of i/p & o/p pulses



(a)

i/p angle	i/p pulses	Received pulses(avg.)
30	16	16
60	33	33
90	50	50
120	66	66
150	83	83.33
180	100	99.5

(b)

Fig3.2: **(a) Output analysis of the closed loop mode of the stepper motor; (b) The measurement of i/p & o/p pulses**

3.3 The Driver Circuit

The driver circuit provides more current to rotate the motor.It consists of Darlington pair which amplifies current. It has 4 Bipolar Junction Transistors (TIP 122) and 4 diodes are used in each Darlington Pair.It provides 3 A more current than other stepper motor drivers.

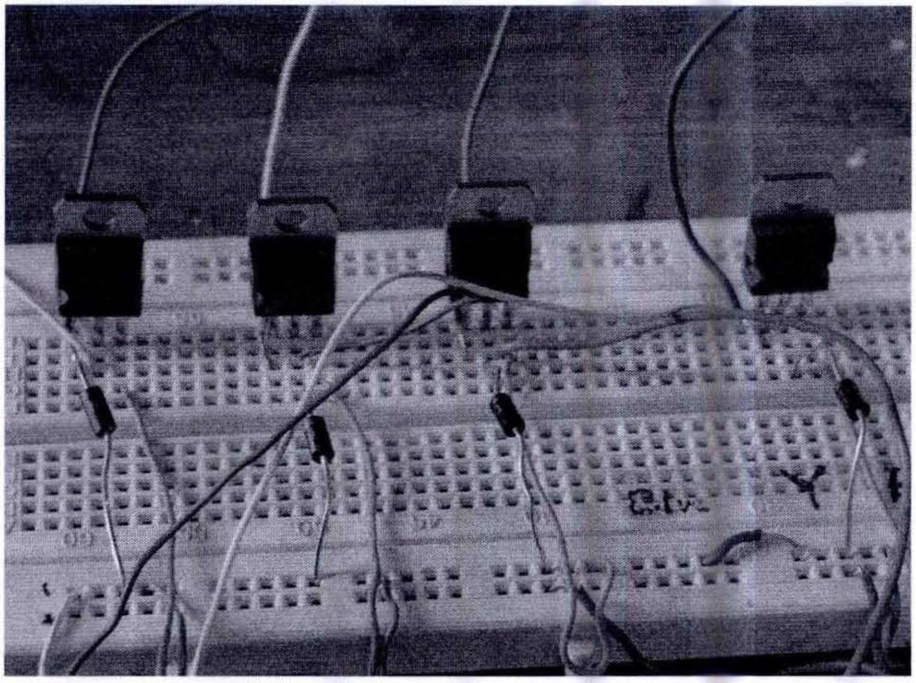


Fig:3.3 Driver Circuit

3.3 Data AcQuisition Card

A Data AcQuisition card (DAQ, PCL-812PG) will be used as a hardware interface and an incremental optical encoder (E30S4) as the feedback component. Data acquisition is the sampling of the real world to generate data that can be manipulated by a computer. DAQ is what usually interfaces between the signal and a PC. Driver software for DAQ card (PCL-812PG) will be used to register data coming from hardware.

3.4 Real-Time Linux Operating System

For this experiment we will control the operation of unipolar stepper motors with RT-Linux. RT-Linux is used to minimize the real-time error (jitter).

CHAPTER IV.

WORKING PRINCIPLE

The robotic arm will be able to rotate to multiple points. The first point of rotation will occur at the base of the device. This base will contain a stepper motor (M3). The base will mount to the end of the arm. (Figure: 1.2) Temperature also plays a critical role in how any mechanical device operates. If the manipulator works in two (Hot & Cold) extreme climates the device should still function without any discrepancies. The device should be able to operate between -20° and 60° Celsius. At the base, the arm

will be capable of rotating 360 degrees about the z-axis.

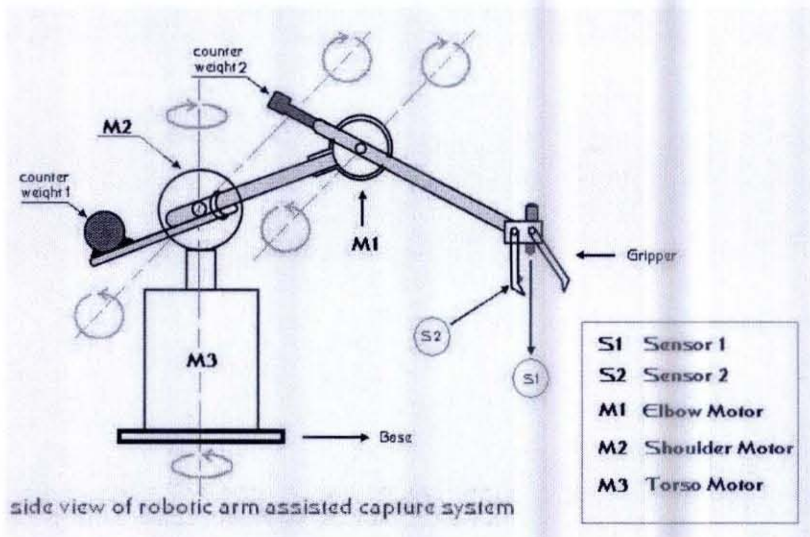


Fig.1.2: The working principle of Robotic Arm.

Once the joint is rotated to the desired position about the z-axis, the attachment between the base and upper arm will have the capability of rotating about either the X or Y axes depending upon the orientation of the base. The shoulder joint connection and will mimic the motion of a hinge joint. The upper arm and lower arm will be connected by the wrist joint. This joint will be capable of rotating about the z-axis at a full range of 360 degrees. This joint will contain another Stepper Motor (M2).The connection between the gripper and the lower arm of the device is stationary.

This connection is for stability purposes only. Finally, the gripper will be capable of opposition, the grasping motion, and reposition, the releasing motion of an object.

Algorithm:

```
#include <linux/module.h>
#include <linux/kernel.h>
#include <rtl_time.h>
#include <rtl_sched.h>
#include <asm/io.h>
#include <time.h>


#define base 0x220

//Lower 8 bit (DOL)
#define   DOL       base+13
//Upper 8 bit (DOH)
#define   DOH       base+14


int i = 0;
int j = 0;
int k = 0;
int l = 0;
int m =0;
int n =0;
```

```
// time delay or speed
int period=100000000;
int periodic_mode=0;


pthread_t thread;

// main function module  or my custom module
void * bit_toggle(void *t)
{
        pthread_make_periodic_np(thread, gethrtime(), period);

        while(j<15)
        {
        motorTwo();
        j++;
        }
        outb(0x00,DOL);



        while(i<10)
       {
        motorOne();
        i++;
```

```
 }
 outb(0x00,DOL);


while(k<15)
 {
motorTwoReverse();
 k++;
 }
 outb(0x00,DOL);

 while(m<15)
 {
 motorTwo();
 m++;
 }

 while(l<10)
 {
 motorOneReverse();
 l++;
 }
 outb(0x00,DOL);

 while(n<15)
 {
```

```
        motorTwoReverse();
        n++;
        }

}


//intialization module
int init_module(void)
{
        pthread_attr_t attr;
        struct sched_param sched_param;

        pthread_attr_init (&attr);
        sched_param.sched_priority = 1;
        pthread_attr_setschedparam (&attr, &sched_param);
        pthread_create (&thread,  &attr, bit_toggle, (void *)0);

        return 0;
}

//cleanup module
void cleanup_module(void)
{
        pthread_delete_np (thread);
}
```

```
//Motor 1
void motorOne()
{
        outb(0x01,DOL);
        pthread_wait_np();

        outb(0x02,DOL);
        pthread_wait_np();

       outb(0x04,DOL);
        pthread_wait_np();

        outb(0x08,DOL);
        pthread_wait_np();
}

//Motor 2
void motorTwo()
{
        outb(0x10,DOL);
        pthread_wait_np();
```

```
        outb(0x20,DOL);
        pthread_wait_np();

        outb(0x40,DOL);
        pthread_wait_np();

        outb(0x80,DOL);
        pthread_wait_np();
}

//Motor 3
void motorThree()
{
        outb(0x01,DOH);
        pthread_wait_np();

        outb(0x02,DOH);
        pthread_wait_np();

        outb(0x04,DOH);
        pthread_wait_np();

        outb(0x08,DOH);
        pthread_wait_np();
}
```

```
//Motor 1 for reverse
void motorOneReverse()
{
	outb(0x08,DOL);
	pthread_wait_np();

	outb(0x04,DOL);
	pthread_wait_np();



	outb(0x02,DOL);
	pthread_wait_np();

	outb(0x01,DOL);
	pthread_wait_np();

}

// reverse motor two
void motorTwoReverse()
{

	outb(0x80,DOL);
	pthread_wait_np();
	outb(0x40,DOL);
	pthread_wait_np();
```

```
        outb(0x20,DOL);
        pthread_wait_np();
        outb(0x10,DOL);
        pthread_wait_np();
}
```

5: Conclusion:

The Robotic Arm will be designed to fulfill the tasks without facing any hindrances due to the lack of workers inability. While there are numerous products available on the market that work as an active device, the Robotic Arm designed is unique to industrial condition.

Reference:

1. http://sachin-robotics.blogspot.com/2008/01/how-to-build-microcontroller-based.html
2. csited.org/2007/34OlawCSITEd.pdf
3. eia.udg.es/~forest/VLSI/lect.10.pdf
4. www.thomasnet.com/catalognavigator.html
5. Flexible, Low-mass Robotic Arm Actuated by Electroactive Polymers and Operated Equivalently to Human Arm and Hand by Y. Bar-Cohen1, T. Xue, M. Shahinpoor, J. Simpson, and J. Smith
6. Exploiting A Real-Time Linux platform in controlling robotic manipulators by C.Bellini, F. Panepinto, S.Panzieri ,G.Ulivi, 15th Triennial World Congress, Barcelona, Spain
7. RePLiCS: An Environment for Open Real-Time Control of a Dual-Arm Industrial Robotic Cell Based on RTAI-Linux, Fabrizio Caccavale, Vincenzo Lippiello, Bruno Siciliano, Luigi Villani
8. www.robot.lth.se
9. www.rtlinux.org