Design of Pick and Place Robot Test Rig

Harshavardhan B P, Salamakalahalli Ramakrishnappa Bharat, Sushanth Kanbail Srinivas
1. Trainee Engineer, Zeco Aircon Ltd, Bangalore
2. Student, BE - Mechanical Engineering, AMC Engineering College, Bangalore
3. Graduate Trainee Officer-Production, Y K Creation, Bangalore

Abstract

The industry is moving from current state of automation to Robotization, to increase productivity and to deliver uniform quality. The industrial robots of today may not look the least bit like a human being although all the research is directed to provide more and more anthropomorphic and humanlike features and super-human capabilities in these.

One type of robot commonly used in industry is a robotic manipulator or simply a robotic arm. It is an open or closed kinematic chain of rigid links interconnected by movable joints. In some configurations, links can be considered to correspond to human anatomy as waist, upper arm and forearm with joint at shoulder and elbow. At end of arm a wrist joint connects an end effector which may be a tool and its fixture or a gripper or any other device to work.

1.Introduction

Robotics is the branch of Engineering Science & Technology related to robots, and their design, manufacture, application, and structural disposition. Robotics is related to electronics, mechanics, and software. Robotics research today is focused on developing systems that exhibit modularity, flexibility, redundancy, fault-tolerance, a general and extensible software environment and seamless connectivity to other machines, some researchers focus on completely automating a manufacturing process or a task, by providing sensor based intelligence to the robot arm, while others try to solidify the analytical foundations on which many of the basic concepts in robotics are built.

In this highly developing society time and man power are critical constrains for completion of task in large scales. The automation is playing important role to save human efforts in most of the regular and frequently carried works. One of the major and most commonly performed works is picking and placing of jobs from source to destination.

The pick and place robot is a microcontroller based mechatronic system that detects the object, picks that object from source location and places at desired location. For detection of object, infrared sensors are used which detect presence of object as the transmitter to receiver path for infrared sensor is interrupted by placed object.

1.2 Law of Robotics

Isaac Asimov conceived the robots as humanoids, devoid of feelings, and used them in a number of stories. His robots were well-designed, fail-safe machines, whose brains were programmed by human beings. Anticipating the dangers and havoc such a device could cause, he postulated rules for their ethical conduct.

Robots were required to perform according to three principles known as "Three laws of Robotics"' which are as valid for real robots as they were for Asimov's robots and they are:

- A robot should not injure a human being or, through inaction, allow a human to be harmed.
- A robot must obey orders given by humans except when that conflicts with the First Law.
- A robot must protect its own existence unless that conflicts with the First or Second law.

These are very general laws and apply even to other machines and appliances. They are always taken care of in any robot design.

2. Literature Review

The concept of creating machines that can operate autonomously dates back to classical times, but research into the functionality and potential uses of robots did not grow substantially until the 20th century. Throughout history, robotics has been often seen to mimic human behaviour, and often manage tasks in a similar fashion. Today, robotics is a rapidly growing field, as technological advances continue; research, design, and building new robots serve various practical

purposes, whether domestically, commercially, or militarily. Many robots do jobs that are hazardous to people such as defusing bombs, exploring shipwrecks, and mines. Their findings and suggestions are reviewed here.

Rosenblatt R (1982) "The Robot Revolution" states that there is three levels of the technologisation of work, Basic Machines (Simple and Engine), Complex Machines (Electrical, Electronic and Computing), and Sophisticated Machines (Anthropomorphic Robots). In each case technologisation has brought benefits as well as disaster. However, we are reaching a point where further technologisation can only exacerbate the global problems of mass unemployment, climate chaos and depletion of the world's resources and exceed the planet's capacity for recovery.

3. Design Procedure

The robot design is the most important part in the process of constructing the robot. Here we develop new ideas for the construction of the robot and express these ideas in the form of plans and drawings.
The robot arm design procedure involves:

- Material selection
- Design of mechanism
- Preliminary design
- Revision of design
- Final drawings

3.1 Requirements

- The robot must be Robust, Rigid and Accurate.
- The arm must be rigid enough to withstand forces generated due to
  - Own body weight
  - Weight of the object to be lifted
  - Inertia forces due to changes in velocity
  - Centrifugal forces due to changes in velocity
- The mechanism must be simple such that the manufacturing process is simplified.
- The cost of the producing the robot must be reduced.

3.2 Inventors Design on Model

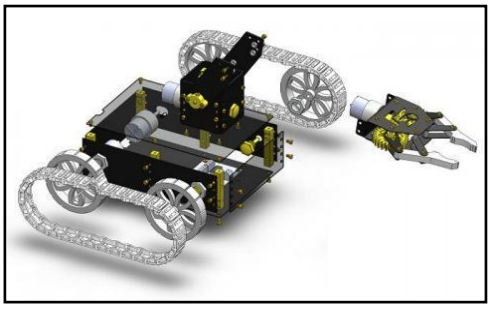

Fig: Inventors Design.

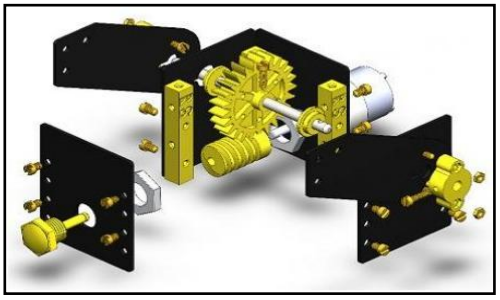

Fig: Body Design.

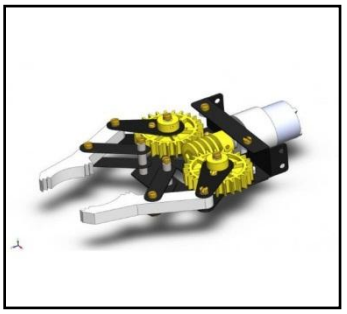

Fig: Gripper Design.

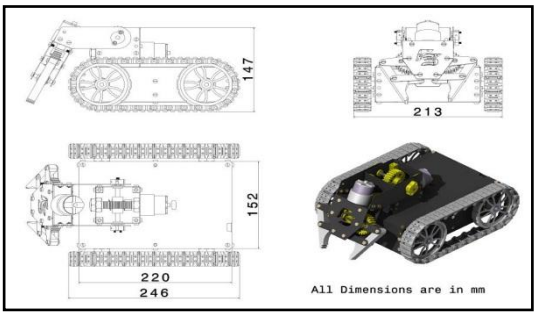

Fig: Full body Assembly.

3.3 Description of Hardware Components

3.3.1Introduction to AT89S52

Microprocessor has following instructions to perform:

- Reading instructions or data from program memory ROM.
- Interpreting the instruction and executing it.
- Microprocessor Program is a collection of instructions stored in a non-volatile memory.

- Read Data from I/O device
- Process the input read, as per the instructions read in program memory.
- Read or write data to Data memory.
- Write data to I/O device and output the result of processing to O/P device.

Microprocessors brought the concept of programmable devices and made many applications of intelligent equipment. Most applications, which do not need large amount of data and program memory, tended to be costly [1].

The microprocessor system had to satisfy the data and program requirements so; sufficient RAM and ROM are used to satisfy most applications .The peripheral control equipment also had to be satisfied. Therefore, almost all-peripheral chips were used in the design. Because of these additional peripherals cost will be comparatively high.

3.3.1aFeatures

- 8/16/32 CPU
- Instruction set rich in I/O & bit operations.
- One or more I/O ports.
- One or more timer/counters.
- One or more interrupt inputs and an interrupt controller.
- One or more serial communication ports.
- Analog to Digital /Digital to Analog converter.
- One or more PWM output.
- Network controlled interface.

3.3.1bWhy AT89S52?

The system requirements and control specifications clearly rule out the use of 16, 32 or 64 bit micro controllers or microprocessors. Systems using these may be earlier to implement due to large number of internal features. They are also faster and more reliable but, the above application is satisfactorily served by 8-bit micro controller. Using an inexpensive 8-bit Microcontroller will doom the 32-bit product failure in any competitive market place. Coming to the question of why to use 89S52 of all the 8-bit Microcontroller available in the market the main answer would be because it has 64 kB Flash and 1024 bytes of data RAM. . The Flash program memory supports both parallel programming and in Serial In-System Programming (ISP). The 89S52 is also In-Application Programmable (IAP), allowing the Flash program memory to be reconfigured even while the application is running.

3.3.1cArchitecture Overview

The 8051 architecture consists of these specific features:

- Eight –bit CPU with registers A (the accumulator) and B
- Sixteen-bit program counter (PC) and data pointer (DPTR)
- Eight- bit stack pointer (PSW)
- Eight-bit stack pointer (Sp)
- Internal ROM or EPROM (8751) of 0(8031) to 4K (8051)
- Internal RAM of 128 bytes:
  - Four register banks, each containing eight registers.
  - Sixteen bytes, which may be addressed at the bit level
  - Eighty bytes of general- purpose data memory
- Thirty –two input/output pins arranged as four 8-bit ports:p0-p3
- Two 16-bit timer/counters: T0 and T1
- Full duplex serial data receiver/transmitter: SBUF
- Control registers: TCON, TMOD, SCON, PCON, IP, and IE
- Two external and three internal interrupts sources.
- Oscillator and clock circuits.

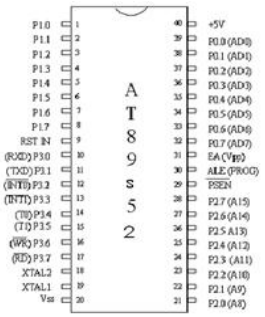

Fig: Architecture Overview of AT89S52.

I/O PORTS: One major feature of a microcontroller is the versatility built into the input/output (I/O) circuits that connect the 8052 to the outside world. The main constraint that limits numerous functions is the number of pins available in the 8051 circuit. The DIP had 40 pins and the success of the design depends on the flexibility incorporated into use of these pins. For this reason, 24 of the pins may each used for one of the two entirely different functions which depend, first, on what is physically connected to it and, then, on what software programs are used to “program” the pins.

PORT 0: Port 0 pins may serve as inputs, outputs, or, when used together, as a bi directional low-order address and data bus for external memory. To configure a pin as input, 1 must be written into the corresponding port 0 latch by the program. When used

for interfacing with the external memory, the lower byte of address is first sent via PORT0, latched using Address latch enable (ALE) pulse and then the bus is turned around to become the data bus for external memory.

PORT 1: Port 1 is exclusively used for input/output operations. PORTS 1 pin have no dual function. When a pin is to be configured as input, 1 is to be written into the corresponding Port 1 latch.

PORT 2: Port 2 may be used as an input/output port. It may also be used to supply a high –order address byte in conjunction with Port 0 low-order byte to address external memory. Port 2 pins are momentarily changed by the address control signals when supplying the high byte a 16-bit address. Port 2 latches remain stable when external memory is addressed, as they do not have to be turned around (set to 1) for data input as in the case for Port 0.

PORT 3: Port 3 may be used to input /output port. The input and output functions can be programmed under the control of the P3 latches or under the control of various special function registers. Unlike Port 0 and Port 2, which can have external addressing functions and change all eight-port b se, each pin of port 3 maybe individually programmed to be used as I/O or as one of the alternate functions.

Pin (SFR)	Alternate Use
P3.0-RXD (SBUF)	Serial data input
P3.1-TXD (SBUF)	Serial data output
P3.2-INTO 0 (TCON.1)	External interrupt 0
P3.3 - INTO 1 (TCON.3)	External interrupt 1
P3.4 - T0 (TMOD)	External Timer 0 input
P3.5 – T1 (TMOD)	External timer 1 input
P3.6 - $\overline{WR}$	External memory write pulse
P3.7 - $\overline{RD}$	External memory read pulse

Table: Alternate use of port.

3.3.2Power Supply

Introduction to Power Supply: There are many types of power supply. Most are designed to convert high voltage AC mains electricity to a suitable low voltage supply for electronic circuits and other devices. A power supply can by broken down into a series of blocks, each of which performs a particular function. For example a 5V regulated supply can be shown as below

3.3.3 Regulator

Regulator eliminates ripple by setting DC output to a fixed voltage. Voltage regulator ICs are available with fixed (typically 5V, 12V and 15V) or variable output voltages. Negative voltage regulators are also available. Many of the fixed voltage regulator ICs has 3 leads (input, output and high impedance). They include a hole for attaching a heat sink if necessary. Zener diode is an example of fixed regulator which is shown here.

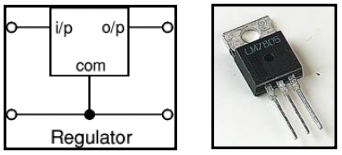

Fig: Regulator.

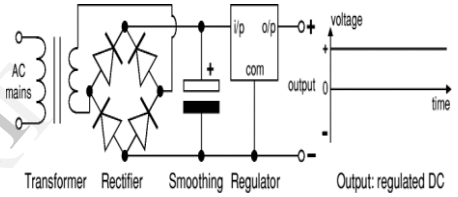

Fig: Transformer, Rectifier, Smoothing, Regulator.

3.3.4DC Motor

DC motors are configured in many types and sizes, including brush less, servo, and gear motor types. A motor consists of a rotor and a permanent magnetic field stator. The magnetic field is maintained using either permanent magnets or electromagnetic windings. DC motors are most commonly used in variable speed and torque.

Motion and controls cover a wide range of components that in some way are used to generate and/or control motion. Areas within this category include bearings and bushings, clutches and brakes, controls and drives, drive components, encoders and resolves, Integrated motion control, limit switches, linear actuators, linear and rotary motion components, linear position sensing, motors (both AC and DC motors), orientation position sensing, pneumatics and pneumatic components, positioning stages, slides and guides, power transmission (mechanical), seals, slip rings, solenoids, springs.
Motors are the devices that provide the actual speed and torque in a drive system. This family includes AC

motor types (single and multiphase motors, universal, servo motors, induction, synchronous, and gear motor) and DC motors (brush less, servo motor, and gear motor) as well as linear, stepper and air motors, and motor contactors and starters.

In any electric motor, operation is based on simple electromagnetism. A current-carrying conductor generates a magnetic field; when this is then placed in an external magnetic field, it will experience a force proportional to the current in the conductor, and to the strength of the external magnetic field. As you are well aware of from playing with magnets as a kid, opposite (North and South) polarities attract, while like polarities (North and North, South and South) repel. The internal configuration of a DC motor is designed to harness the magnetic interaction between a current-carrying conductor and an external magnetic field to generate rotational motion.

3.3.5 Single Sided Board

The term single sided means the insulating substances contains circuit pattern on only one side, which is called as solder side & the other side is component side these kinds of P.C.B are used in professional & nonprofessional grades & are manufactured by print & etch method. In this method, single sided board is manufactured with limited copper on the surface. It is used in professional & non- professional grade of P.C.B's the precision of the circuit boards depends on the image transfer on the substrate. This process is used for low cost commercial mass production. The entire multi-layer, inner layers are prepared in this method.

The mechanical machining operations involved in P.C.B manufacturing are shearing, sawing, punching, blanking, routing and darling, welding.

3.3.6 Diodes

The diodes re junctions, which allows the current to flow in one direction only. Its main function is to avoid back e.m.f although the reverse break down occurs at very high voltage, beyond which he current increase steeply for every increase in applied voltage. Here we have used 12 numbers of in 4007 rectified diodes.

3.3.7 Capacitors

The capacitor is a device, which stores electrical charges, it consist of two conductors separated by dialectical medium. The dielectric medium is a substance of high resistance (mica, paper, ceramic, air, etc) for the flow of electric current. Here we have used ceramic capacitors.

3.3.8 Transistors

The transistor can be considered as two semiconductor diodes connected back-to-back in a semiconductor crystal, it consist of three regions known as emitter, base & collector.

3.3.9 Relays

A relay is the simplest form can be defined as an electrically operated switch. This description implies that the Relay consist of mechanism (electromagnet or thermal or electronic relay) to make or break connection in electric circuit here we have used 12v, 279Ω relay.

3.3.10 Resistors

Resistor is a component which offers a specific amount of resistance to the flow of current. These are made up of film of carbon or a coil of nichrome wire.

3.3.11 Remote

This is a device to control the robot in according to the convenience. This consists of IR transmitter, which send signal through IR waves. Here we have used general T V remote to operate. The Remote used here controls the two stepper motor for two degree of freedom.

4. Sequence of operation

4.1 Operation of Robot

- Initially we will assume the rest position of entire system, i.e. state when no object is placed.
- For understanding operation, let us rename the two motors used here. Let the name of motor be M1 and motor is M2.
- As the microcontroller detects:
  - For forward direction, when the motor M1 & motor M2 be ON stage then the both the motors move in forward direction [2].
  - For reverse direction, when the motor M1 &motor M2 be OFF stage then the both the motors move in reverse direction.
  - For left side, when the motor M1 is OFF stage &M2 be ON stage then the motor M1 move in reverse direction and motor M2 move in forward direction.
  - For Right side, when the motor M1 is OFF stage &M2 be ON stage then the motor M1

move in reverse direction and motor M2 move in forward direction.

- For Arm movement: Let us rename the two motors arm used here. Let the name of motor be M3 and motor is M4.
- Now as microcontroller detects andif the motor M3 is in ON stage it moves in clockwise direction for a fixed time due to which whole arm moves towards the direction and the motor M4 is in OFF stage.
- As it reaches there, M3 stops and now motor M4 is started in say clockwise direction to hold the object by closing jaw. This motor also, is on for particular fixed time instant.
- As M4 gets OFF, motor M3 is moved again in anticlockwise direction till the time it reaches the placing platform.
- As it reaches placing platform, the motor M3 stops and M4 is switched ON in anticlockwise direction till it releases object properly on desired place.
- These motors are driven through the motor driver called L293D.

4.2 Operating the L293d Motor Driver

Using the L293D motor driver, makes controlling a motor as simple as operating a buffer gate IC. It totally isolates the TTL logic inputs from the high current outputs. Putting logic 1 on the pin In1 will make Out1 pin goes to Vpower (36 Volts MAX.); while a logic 0 will make it go to 0V. Each couple of channels can be enabled and disabled using E1 and E2 pins. When disabled a channel provide very high impedance (resistance) to the motor, exactly as if the motor wasn't connected to the driver IC at all, which makes this feature very useful for PWM speed control [4].

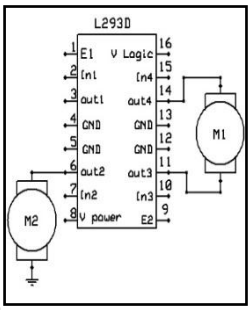

Fig: shows different ways to connect a motor to the IC.

One way is to use 2 channels to build a bi-directional motor driver; another way is to use 1 channel per motor, building a unidirectional driver. In this project, we will be using the 4 channels to drive the 2 motors in both directions.

4.3 Methodology

In completing this project there are several steps or project approaches that have been executed. Every single step of selection and action that has been done while doing this project will be explained through this section. It is important to make sure project that consists of hardware and software development can be executed successfully. Here, the flow of the methodology of this project have been designed to be clear and logical, in order to ease the understanding of what approach that have been taken in completing this type of project.

This project involved the following tasks and organized as follows

- Research by Reading overview:
  Choose the optimum material and combination of the motors taking into consideration the size and weight constraint. Get the literature review about the whole system from books, internet, lecture's note and journal.
- Research on Hardware development:
  - Design the model for the robotic arm according to the size and weight constraint and the tasks that the manipulator is required to perform [3].
  - Develop the (microcontroller, motor controller, power supply etc) to control the robotic arm.
  - Identify the best selection of motor to be used for this project whether stepper motor, servo motor or others to move the robotic arm.
- Research on Software development:
  - Study and develop command language for the PIC microcontroller in order to interface the robotic arm with the PIC microcontroller program.
  - Identify the software to be used such as MPLAB, Proteus or others to build up the controller command system.

5. Software Development

In this chapter the software used and the language in which the program code is defined is mentioned and the program code dumping tools are explained. The chapter also documents the development of the program for the application. This program has been termed as "Source code". Before we look at the source code we define the two header files that we have used in the code.

5.1 Tools Used

Keil development tools for the 8051 Microcontroller Architecture support every level of software developer from the professional applications.

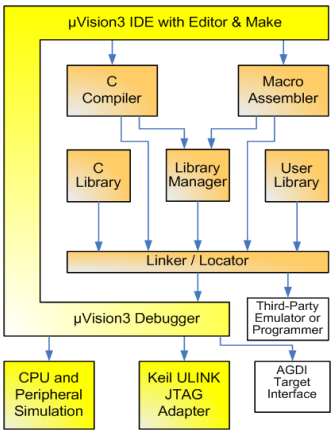

Fig: Keil Software- Internal Stages.

5.2 C51 Compiler & A51 Macro Assembler

Source files are created by the µVision IDE and are passed to the C51 Compiler or A51 Macro Assembler. The compiler and assembler process source files and create replaceable object files.

The Keil C51 Compiler is a full ANSI implementation of the C programming language that supports all standard features of the C language. In addition, numerous features for direct support of the 8051 architecture have been added [5].

5.3 µVISION

µVISION3 is an IDE (Integrated Development Environment) that helps you write, compile, and debug embedded programs. It encapsulates the following components:

- A project manager.
- A make facility.
- Tool configuration.
- Editor.
- A powerful debugger.

To help you get started, several example programs (located in the **\C51\Examples, \C251\Examples, \C166\Examples, and \ARM\...\Examples**) are provided.

HELLO is a simple program that prints the string "Hello World" using the Serial Interface.

MEASURE is a data acquisition system for analog and digital systems.

TRAFFIC is a traffic light controller with the RTX Tiny operating system.

SIEVE is the SIEVE Benchmark.

DHRY is the Dhrystone Benchmark.

WHETS are the Single-Precision Whetstone Benchmark.

5.4 Building an Application inµVISION

To build (compile, assemble, and link) an application in µVision2, you must:

- Select Project (for example, **166\EXAMPLES\HELLO\HELLO.UV2**).
- Select Project - Rebuild all target files or Build target.
- µVision2 compiles, assembles, and links the files in your project.
- Creating Your Own Application in µVision2

5.4 .1To create a new project in µVision2

- Select Project - New Project.
- Select a directory and enter the name of the project file.
- Select Project - Select Device and select an 8051, 251, or C16x/ST10 device from the Device Database™.
- Create source files to add to the project.
- Select Project - Targets, Groups, and Files. Add/Files, select Source Group1, and add the source files to the project.
- Select Project - Options and set the tool options. Note when you select the target device from the Device Database all special options are set automatically. You typically only need to configure the memory map of your target hardware. Default memory model settings are optimal for most applications.
- Select Project - Rebuild all target files or Build target.

5.4 .2 Debugging an Application in µVision2

To debug an application created using µVision2, you must:

- Select Debug - Start/Stop Debug Session.
- Use the Step toolbar buttons to single-step through your program. You may enter G, main in the Output Window to execute to the main C function.
- Open the Serial Window using the Serial #1 button on the toolbar.
- Debug your program using standard options like Step, Go, Break, and so on.

5.4 .3 Starting µVision2 and Creating a Project

µVision2 is a standard Windows application and started by clicking on the program icon. To create a new project file select from the µVision2 menu.

Project – New Project. This opens a standard Windows dialog that asks you for the new project file name.

You can simply use the icon Create New Folder in this dialog to get a new empty folder. Then select this folder and enter the file name for the new project, i.e. Project1.

µVision2 creates a new project file with the name PROJECT1.UV2 which contains a default target and file group name. You can see these names in the Project

Window – Files: Now use from the menu Project – Select Device for Target and select a CPU for your project. The Select Device dialog box shows the µVision2 device database. Just select the microcontroller you use. We are using for our examples the Philips 80C51RD+ CPU. This selection sets necessary tool options for the 80C51RD+ device and simplifies in this way the tool Configuration

5.4 .4 Building Projects and Creating a HEX Files

Typical, the tool settings under Options – Target are all you need to start a new application. You may translate all source files and line the application with a click on the Build Target toolbar icon. When you build an application with syntax errors, µVision2 will display errors and warning messages in the Output Window – Build page. A double click on a message line opens the source file on the correct location in a µVision2 editor window. Once you have successfully generated your application you can start debugging. After you have tested your application, it is required to create an Intel HEX file to download the software into an EPROM programmer or simulator. µVision2 creates HEX files with each build process when Create HEX files under Options for Target – Output is enabled. You may start your PROM programming utility after the make process when you specify the program under the option Run User Program #1.

5.4 .5 CPU Simulation

µVision2 simulates up to 16 Mbytes of memory from which areas can be mapped for read, write, or code execution access. The µVision2 simulator traps and reports illegal memory accesses.

In addition to memory mapping, the simulator also provides support for the integrated peripherals of the various 8051 derivatives. The on-chip peripherals of the CPU you have selected are configured from the device.

5.4 .6 Start Debugging

You start the debug mode of µVision2 with the Debug – Start/Stop Debug Session command. Depending on the Options for Target – Debug Configuration, µVision2 will load the application program and run the start-up code µVision2 saves the editor screen layout and restores the screen layout of the last debug session. If the program execution stops, µVision2 opens an editor window with the source text or shows CPU instructions in the disassembly window. The next executable statement is marked with a yellow arrow. During debugging, most editor features are still available.

For example, you can use the find command or correct program errors. Program source text of your application is shown in the same windows. The µVision2 debug mode differs from the edit mode in the following aspects: The “Debug Menu and Debug Commands” described below are available. The additional debug windows are discussed in the following. The project structure or tool parameters cannot be modified. All build Commands are disabled.

5.4 .7 Disassembly Window

The Disassembly window shows your target program as mixed source and assembly program or just assembly code. A trace history of previously executed instructions may be displayed with Debug – View Trace Records. To enable the trace history, set Debug – Enable/Disable Trace Recording.

If you select the Disassembly Window as the active window all program step commands work on CPU instruction level rather than program source lines. You can select a text line and set or modify code breakpoints using toolbar buttons or the context menu commands.

You may use the dialog Debug – Inline Assembly… to modify the CPU instructions. That allows you to correct mistakes or to make temporary changes to the target program you are debugging.

5.4.8 Program for Controlling the Rotation of the Motors

```
#include"main.h"
//MAIN FUNCTION BEGINS
void main(void)
{
MainSystemInitialize();
while(1)
{
WaitForValidSignal;
switch(RF_DATA_PORT)
{
case ONE:       RotateMotor1Clockwise;
WaitUntilSignalPresent;
StopMotor1;
break;
case TWO:       RotateMotor1AntiClockwise;
WaitUntilSignalPresent;
StopMotor1;
break;
case THREE:     RotateMotor2Clockwise;
WaitUntilSignalPresent;
StopMotor2;
break;
case FOUR:      RotateMotor2AntiClockwise;
WaitUntilSignalPresent;
StopMotor2;
break;
case FIVE:      RotateMotor3Clockwise;
WaitUntilSignalPresent;
StopMotor3;
break;
case SIX:       RotateMotor3AntiClockwise;
WaitUntilSignalPresent;
StopMotor3;
break;
case SEVEN:     RotateMotor4Clockwise;
WaitUntilSignalPresent;
StopMotor4;
break;
case EIGHT:     RotateMotor4AntiClockwise;
WaitUntilSignalPresent;
StopMotor4;
break;
case NINE:      RotateMotor5Clockwise;
WaitUntilSignalPresent;
StopMotor5;
break;
case TEN:       RotateMotor5AntiClockwise;
WaitUntilSignalPresent;
StopMotor5;
break;
case ELEVEN:    RotateMotor6Clockwise;
WaitUntilSignalPresent;
StopMotor6;
break;
case TWLEVE:    RotateMotor6AntiClockwise;
WaitUntilSignalPresent;
StopMotor6;
break;
case THRTEEN:  RotateMotor7Clockwise;
WaitUntilSignalPresent;
StopMotor7;
break;
case FOURTEEN: RotateMotor7AntiClockwise;
WaitUntilSignalPresent;
StopMotor7;
break;
default:
break;
}
}
}
static void MainSystemInitialize(void)
{
RF_DATA_PORT        = INPUT_PORT;
VALID_SIGNAL_PIN = INPUT_PIN;
MOTOR1ANODE         = OUTPUT_PIN;
MOTOR1CATHODE       = OUTPUT_PIN;
MOTOR2ANODE         = OUTPUT_PIN;
MOTOR2CATHODE       = OUTPUT_PIN;
}
```

6. Advantages and Applications

6.1 Advantages

- Accuracy and Pick and Place Robots:Robots are outfitted with wide reaches and slim arms, steady repeatability and precise tooling – all of which allows them to be extremely accurate. This high precision capability makes them a good match for pick and place applications [6].
- Flexible Pick and Place:One of the main advantages of robotics is flexibility. Pick and place robots are easily programmable. They are able to accommodate multiple changes in product shape and type. In addition, robots provide a high level of movement flexibility.
- Increase Consistency with Pick and Place:Pick and place robot systems have the ability to improve product quality and cycle time. Robotic movements are regulated, so the results are always the same. Quality is improved because of this regularity. Furthermore, this consistency allows the processes to take place.
- Robots are Space-Efficient:Because they are designed with compact bases, pick and place robots are ideal if you are looking to conserve floor space. Robots can be programmed to move within strict work envelope limits – leading to even better use of space.

- Robots Maximize Safety:Pick and place applications can be physically demanding. They are labour-intensive, repetitive, and monotonous. Depending on the weight and size of a part, moving it from one place to another can be very demanding work. Pick and place robots are unaffected by the stresses of the application. They are able to work without taking breaks or making mistakes.
- Save with Pick and Place Robots:Incorporating pick and place robots can effectively cut your costs. Robotic precision and reliability allow for less wasted material and more efficient use of time. Plus, the initial investment in robots is quickly recouped – making pick and place robots an extremely cost-effective solution.
- Cost-Effectiveness: The afore-mentioned features combine to lend a high degree of cost-effectiveness to such systems. Cost effectiveness also accrues from the fact that pick and place systems empower businesses to take up orders in bulk and thus aid business expansion and also reap the benefits of large-scale production.

6.2 Applications

- Welding:This is the process of joining two work—pieces together by applying molten weld metal.
- Cutting:This is the process of applying thermal or mechanical energy to cut a work piece into a specific shape.
- Assembly:This is the process of either adding components to form a single entity, or affixing components to a base unit (e.g. to place components on a printed circuit board).
- Material handling:This is the process of either packaging parts into a compartment (box) or loading, unloading parts from another station.

7. Conclusion

After a period of hardworking we have “PICK AND PLACE ROBOT” as a real working system in our hand. It will be really a good friend of users. It is very easy to use and user interactive. As we know robotic arm is implemented in the material handling operation. Still it has much functionality to be implemented and we will surly give our time to complete its remaining functionalities.

During this period we have been exposed to user-friendly environment and real time industrial application development. Because of it we learned skills like team work, working with time bound and dealing with professional persons in Industry.

The final objective is to develop a Robotic Arm which will be able to lift a sensitive object safely. The arm will employ a closed loop control system using the force sensor as the input sensor and the computer or microcontroller as the controller and the controller will control the stepper motor. We used C to write the program which will control the drive circuit. The data acquisition for the force sensor was done by a data acquisition card (DAQ). The arm once completed will greatly assist in the automation of the land mine clearing process. Hopefully this will result in improving the speed with which displaced people can be relocated.

Bibliography

The papers and books listed cover the various aspects of Robotics and Mechatronics:

[1] Roy PK (2001) Microprocessor Data Hand Book, BPB Publications, B-14, Connaught Place, New Delhi-110001, pp-101-120.

[2] Ayres RU, Miller SM (1981) the Impact of Robotics on the Workforce and Workplace. Carnegie Mellon University Report, Australia pp 60-66.

[3] Karel C (1923) Rossum`s Universal Robot, English version by P. Selver and N. Playfaff, New York: Doubleday, Page and Company, pp-52-60.

[4] Mechatronics systems Fundamentals by Rolf Isermann – Springer.

[5] Leinecker RC, Archer (2003) “Visual C++ Programming”, John Wiley & Sons Inc., U.S.A. pp. 60-66.

[6] Rosenblatt R (1982) “The Robot Revolution.” Editorial Research Reports, pp. 347364.