International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 10, October 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

Analysis of Low Cost Naturally Programmable Robotic ARM

K.Deepikavalli[1], S.Asvani[2], R.Puviarasi[3]

[1,2,3,]*Department of ECE, Saveetha School of Engineering, Saveetha University, Chennai (India)*

ABSTRACT

Most programming methods used today involve some degree of programming knowledge or involves creating an entire three dimensional workspace to virtually program the robot arm. To address these two limitations, this project aims at reducing the cost of the industrial arm by using multiple low cost embedded processors working in a coordinated and distributed fashion. This project also implements a master-slave based control system for real time control and programming of the robotic arm. This proposed master-slave control system is analogous to lead by nose programming method, but overcomes some of its limitations. In this paper it is implemented using CCS IDE.

Keywords: code composer studio, IDE, Microcontroller, Robot

I. INTRODUCTION

Robotic arm is being widely used in the industrial automation domain in recent years. They are designed to perform any desired task such as welding, gripping, spinning etc., depending on the application. Each unique application requires a different program to be loaded and executed by the arm. The programming of motions and sequences for an industrial robot is typically taught by linking the robot controller to a laptop, desktop computer or (internal or Internet) network. A robot and a collection of machines or peripherals is referred to as a work cell, or cell. The various machines are 'integrated' and controlled by a single computer or PLC. How the robot interacts with other machines in the cell must be programmed, both with regard to their positions in the cell and synchronizing with them. Though the programming effort is one time activity, the cost and the overhead involved is high. It discusses an improved form of dynamic systems motor primitive by which rhythmic tasks can be learned using a concerted approach of both imitation and reinforcement learning. [1].G. Hirzinger et al (2001) presents a mechatronic approach incorporating a tight collaboration between mechanics, dynamics, electronics and controller design. Details of the prototype arm developed are also discussed in detail. [2] Ikuo Yamano et al (2002) discuss a new type of master-slave control methodology which has the merits of both unilateral and bilateral ones. The proposed method is built on switching the unilateral feedback controls of position and force as required using switching and elastic elements [3]

The design and control of actuators for machines and robots physically interacting with humans. [4] The proposed method uses variable stiffness actuators for safe and controlled actuation. Discuss the development and characterization of a novel 5cm, 7g jumping robot using fast manipulators. [5] From the German Aerospace

Center Institute of Robotics and Mechatronics present a new design for the next generation robots. Their design is compact and has high performance actuation. [6] Presents design and performance of a novel joint based actuator for a robot run by Variable Stiffness Actuation, meant for systems physically interacting with humans. [7]. A general approach for learning robotic motor skills from human demonstration. To represent an observed movement, a non-linear differential equation is learned such that it reproduces this movement [8]. From the Princeton University provides a wealth of information on Robotics and intelligent systems design. Reference designs and advanced mechatronic and electronic concepts are discussed in the society of robots. [9]. the current programming methods employ some form of programming language to program the robot. This requires skilled programmers and has some set up time [10] associated with the programming time. Another factor discouraging the widespread use of robotic arm is the cost of the robotic arm itself. Thus this project aims to address these two concerns by employing some new approaches including master slave control based programming and using multiple low cost embedded controllers instead of a single powerful computer controlling the arm.

II. OVERVIEW OF ROBOT PROGRAMMING AND INTERFACES

The setup or programming of motions and sequences for an industrial robot is typically taught by linking the robot controller to a laptop, desktop computer or (internal or Internet) network.

A robot and a collection of machines or peripherals is referred to as a work cell, or cell. A typical cell might contain a parts feeder, a molding machine and a robot. The various machines are 'integrated' and controlled by a single computer or PLC. How the robot interacts with other machines in the 6 cell must be programmed, both with regard to their positions in the cell and synchronizing with them.

Software: The computer is installed with corresponding interface software. The use of a computer greatly simplifies the programming process. Specialized robot software is run either in the robot controller or in the computer or both depending on the system design.

There are two basic entities that need to be taught (or programmed): positional data and procedure. For example in a task to move a screw from a feeder to a hole the positions of the feeder and the hole must first be taught or programmed. Secondly the procedure to get the screw from the feeder to the hole must be programmed along with any I/O involved, for example a signal to indicate when the screw is in the feeder ready to be picked up. The purpose of the robot software is to facilitate both these programming tasks. Teaching the robot positions may be achieved a number of ways:

Positional commands: The robot can be directed to the required position using a GUI or text based commands in which the required X-Y-Z position may be specified and edited.

Teach pendant: Robot positions can be taught via a teach pendant. This is a handheld control and programming unit. The common features of such units are the ability to manually send the robot to a desired position, or "inch" or "jog" to adjust a position. They also have a means to change the speed since a low speed is usually required for careful positioning, or while test-running through a new or modified routine. A large emergency stop button is usually included as well. Typically once the robot has been programmed there is no more use for the teach pendant. Lead-by-the-nose is a technique offered by many robot manufacturers. In this method, one

user holds the robot's manipulator, while another person enters a command which de-energizes the robot causing it to go limp. The user then moves the robot by hand to the required positions and/or along a required path while the software logs these positions into memory. The program can 7 later run the robot to these positions or along the taught path. This technique is popular for tasks such as paint spraying.

Offline programming is where the entire cell, the robot and all the machines or instruments in the workspace are mapped graphically. The robot can then be moved on screen and the process simulated. The technique has limited value because it relies on accurate measurement of the positions of the associated equipment and also relies on the positional accuracy the robot which may or may not conform to what is programmed.

Others In addition, machine operators often use user interface devices, typically touch screen units, which serve as the operator control panel. The operator can switch from program to program, make adjustments within a program and also operate a host of peripheral devices that may be integrated within the same robotic system. These include end effectors, feeders that supply components to the robot, conveyor belts, emergency stop controls, machine vision systems, safety interlock systems, bar code printers and an almost infinite array of other industrial devices which are accessed and controlled via the operator control panel.

The teach pendant or PC is usually disconnected after programming and the robot then runs on the program that has been installed in its controller. However a computer is often used to 'supervise' the robot and any peripherals, or to provide additional storage for access to numerous complex paths and routines.MSP430 microcontrollers have a von Neumann architecture based CPU as shown in the simplified architecture diagram shown in fig2.3. Some of the general peripherals present in the MSP 430 family are also seen in figure 1.1. MSP430 devices offer high-performance peripherals including USB, RF, LCD controllers and Sigma-Delta ADCs. This allows designers to find the appropriate MSP430 device for many applications. This integration enables solutions with smaller physical footprints and reduced bill of materials costs.

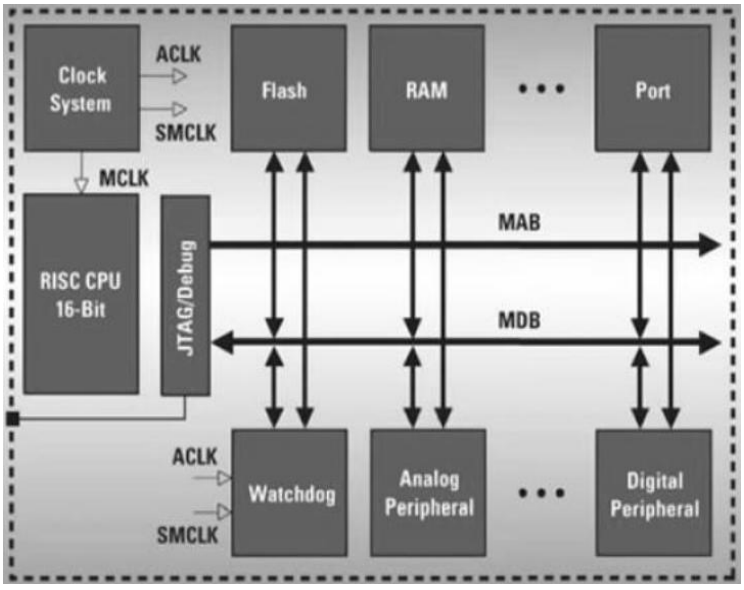
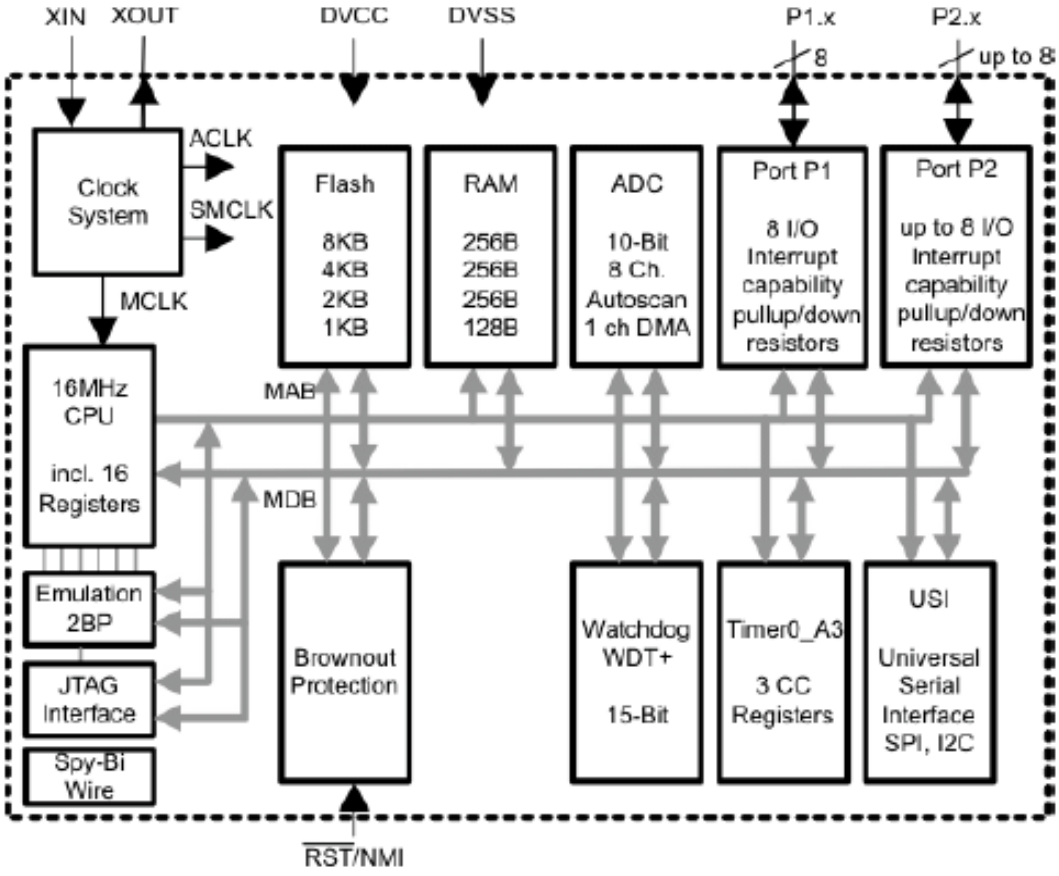


Fig 1. Simplified block diagram of MSP430 family microcontroller

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 10, October 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

The peripherals have been designed to give you maximum functionality and provide system-level interrupts resets and bus arbitration at the lowest power.



Note: Available memory, peripherals, and ports may vary, depending on the device.

Fig 2. Detailed block diagram of the MSP430 microcontroller

Many peripherals may function autonomously, thereby minimizing CPU time spent in active mode. This means that MSP430 MCUs offer more performance with less power.

III. CODE COMPOSER STUDIO

Code Composer Studio is an integrated development environment (IDE) that supports TI's Microcontroller and Embedded Processors portfolio. Code Composer Studio comprises a suite of tools used to develop and debug embedded applications. It includes an optimizing C/C++ compiler, source code editor, project build environment, debugger, profiler, and many other features. The intuitive IDE provides a single user interface taking through each step of the application development flow. Familiar tools and interfaces allow users to get started faster than ever before. Code Composer Studio combines the advantages of the Eclipse software framework with advanced embedded debug capabilities from TI resulting in a compelling feature-rich development environment for embedded developers.

International Journal of Advance Research in Science and Engineering
Volume No.06, Issue No. 10, October 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

Code Composer Studio being an eclipse based tool provides the user with multiple perspectives to suit different stages of development. The edit perspective is shown in the figure 1.4. The edit perspective provides all the options and navigation features required during programming.

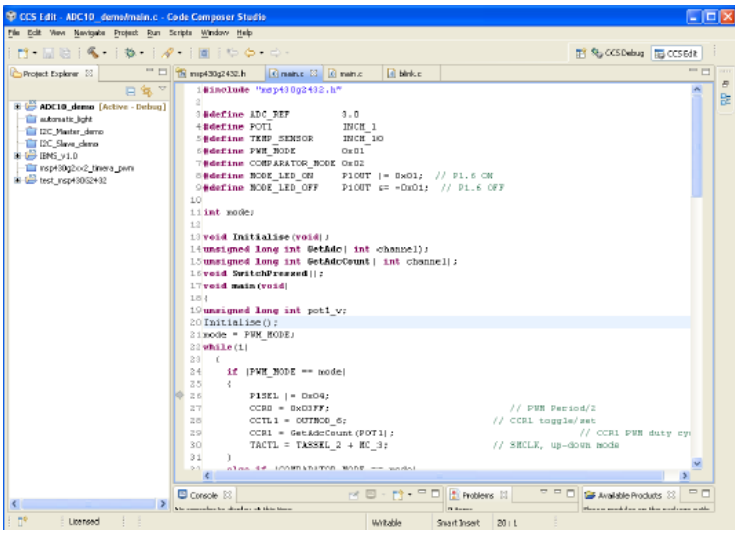


Fig 3. The CCS IDE showing the edit view

The Code Composer Studio also features a debug perspective which provides the user with debug tools and expression and memory watch windows which would be required to the developer during debugging of the software. A view of the debug perspective is shown in figure 1.4.

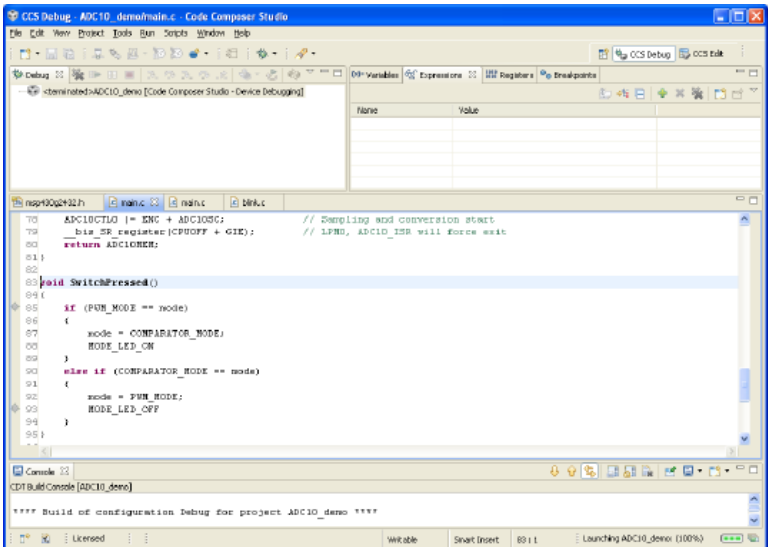


Fig .4. The CCS IDE showing the Debug view with a built project

IV. CONCLUSION

In this project, a master-slave based method of programming and control has been designed using multiple embedded controllers and the performance is evaluated. Literature survey on various concepts related to

industrial robotic arm design. Analyzed the various types of robotic arm designs used in industries. Learnt MSP430 microcontroller peripherals and using the launch pad with CCS IDE for flashing and real time debugging. Created a sample demo application demonstrating the various MSP430 peripheral features leaned.

REFERENCES

[1] Jens Kober, Jan Peters "Learning Motor Primitives for Robotics", 2009 IEEE International Conference on Robotics and Automation

[2] G. Hirzinger, A. Albu-Schaffer, M. Hiihnle, I. Schaefer, N. Sporer "On a New Generation of Torque Controlled Light-Weight Robots". Proceedings of the 2001 IEEE International Conference on Robotics & Automation Seoul, Korea . May 21-26, 2001

[3] Ikuo Yamano, Kenjiro Takemura, Ken Endo, Takashi Maeno," Method for Controlling Master-Slave Robots using Switching and Elastic Elements", Proceedings of the 2002 IEEE lntemational Conference on Robotics 8 Automation Washington, DC May 2002

[4] Giovanni Tonietti, Riccardo Schiavi and Antonio Bicchi "Design and Control of a Variable Stiffness Actuator for Safe and Fast Physical Human/Robot Interaction" Proceedings of the 2005 IEEE Int1ernational Conference on Robotics and AutomationIan Downard, "SIMULATING SENSOR NETWORKS IN NS-2"

[5] Mirko Kovaˇc, Martin Fuchs, Andr´e Guignard, Jean-Christophe Zufferey, Dario Floreano , "A miniature 7g jumping robot", 2008 IEEE International Conference on Robotics and Automation Pasadena, CA, USA, May 19-23, 2008

[6] Sebastian Wolf and Gerd Hirzinger DLR - German Aerospace Center Institute of Robotics and Mechatronics "A New Variable Stiffness Design: Matching Requirements of the Next Robot Generation", 2008 IEEE International Conference on Robotics and Automation Pasadena, CA, USA, May 19-23, 2008

[7] R. Schiavi, G. Grioli, S. Sen, A. Bicchi ,"VSA-II: a Novel Prototype of Variable Stiffness Actuator for Safe and Performing Robots Interacting with Humans", 2008 IEEE International Conference on Robotics and Automation Pasadena, CA, USA, May 19-23, 2008.

[8] Peter Pastor, Heiko Hoffmann, Tamim Asfour, and Stefan Schaal, "Learning and Generalization of Motor Skills by Learning from Demonstration", 2009 IEEE International Conference on Robotics and Automation Kobe International Conference Center Kobe, Japan,

[9] "Robotics and Intelligent Systems Virtual Reference Book" by Robert F. Stengel, Princeton University Princeton, NJ August 29, 2013

[10] Reference designs and mechatronic designs "http://www.societyofrobots.com/ "