Embedded Design and Development of Pick and Place Robotic Arm

[Group: Robotic Arm Control 4]

Name	Student ID	Role
[redacted]		Hardware Lead & programmer
[redacted]		Software Lead & hardware assistance
Sepehr Rezvani	301291960	Software and Hardware assistance

MSE 450/750 – REAL TIME AND EMBEDDED CONTROL SYSTEMS
[Prepared for Dr. [redacted]
SIMON FRASER UNIVERSITY
April 6, 2020

Abstract

In this project, we aim to use a one degree of freedom robotic arm to solve a real-world industrial application problem of pick and place. In order to achieve our goal for the implementation of pick and place robot, these tasks need to be completed; (i) reading potentiometer values for locating end-effector's position in space, (ii) generating pulse width modulation signals for motor's speed and direction control, (iii) activating or deactivating magnetic gripper for the task of pick and place, and (iv) software integration to make all the components work together in order to achieve the main goal. The main design requirements of this project are that the system should be embedded, real-time, fast, and reliable. We are using Texas Instrument's (TI's) TM4C123GH6PM microcontroller for this purpose, which will control the robotic arm. For software implementation, we are using Code Composer Studio Integrated Development Environment that supports TI's microcontroller. This report presents the design processes and methodologies that will applied to complete the term project. The functionalities needed to successfully complete the project will be built and tested individually and then integrated together to achieve a complete 1 DoF robotic arm system that will be used for pick and place purposes.

Table of Contents

List of Figures

List of Tables

1.Introduction

Industrial robots are used for many different applications such as welding, additive manufacturing, pick and place, etc. In this project, we are going to make an embedded design and implement of one degree of freedom (DoF) pick and place robotic arm.

The requirement of this term project is to design and build an embedded system for a real-time control of a 1 DoF Robotic Arm. A Tiva LaunchPad TM4C123GH6PM microcontroller is being used as the main development board and Code Composer Studio (CCS) as the main integrated development environment.

The 1 DoF robotic arm is armed with a potentiometer, an encoder and two actuators, to elaborate an electromagnet is used as an end effector and a DC motor.

2. Design Goals

The mandatory requirement for this term project is to design, implement, and integrate embedded software and hardware with a 1 (DoF) Pick and Place robotic arm.

A key functionality of the robotic arm would be to take, object location (in radians or degree) as user input and rotate the arm to the specified location. The target object will be a ferrous object and electromagnetic gripper will be used to pick it up and place it to a user specified location.

In addition to pick and place, the feedback from the potentiometer will be converted to degrees/radians (showing real time location of arm) and displayed on a 7 segment Display/ LCD.

Objectives

In order to achieve the design goals stated above, the following objectives need to be fulfilled:

- Develop effective and efficient communication method between input, microcontroller and key peripheral devices such DC motor, potentiometer, electromagnet, and the 7-segment display.
- Read potentiometer readings using TM4C123gh6pm's built in Analog-to-Digital (ADC) module.
- Determine the scale factor for converting the voltage reading of the potentiometer to position of the shaft (in radians or degrees)
- Generate two PWM signals from appropriate ports of the microcontroller. Accurate signal generation is required for speed control. In addition, to make sure the robotic arm is bidirectional, one of the PWM signal is responsible for

clockwise rotation of the motor while the other signal takes care of the counter clockwise rotation.

- Activate/Deactivate the electromagnetic end effector for pick and place functionality.

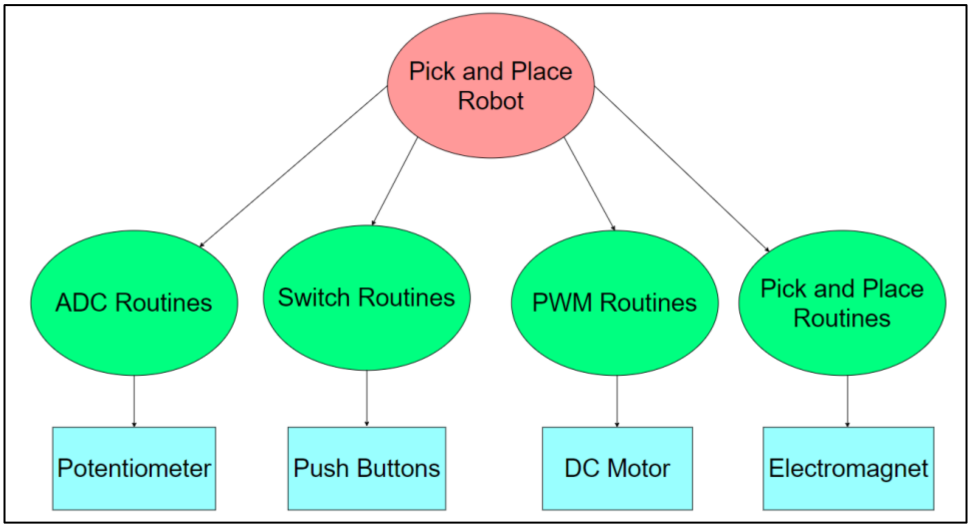

Figure 1: Call graph for robotic Arm

- 7 Segment Display/LCD:

  Seven Segment Display is a set of seven bar shaped LED's that could show numbers from 0-9 or some letters. What we are focused on for this project is to use three SSDs to show the distance between the robotic arm's reference line (0 degrees) and object's location in degrees. As discussed in the previous section, the arm's range of motion is half a circle (approximately 180 degrees or π), so three SSDs could show '0 0 0' (object on the reference line) or '1 8 0' (object positioned half circle away from the reference line).

Since the SSDs are common cathode, for each LED on the display to turn on, the anode side must be connected to a voltage source, and its cathode to the ground, otherwise the LED will be off. In this case, LED's cathode must be connected to port B (B0-B7).

3. Requirements

The following tables represent functional and non-functional requirements.

Functional Requirements

Table 1: Functional Requirements

Requirement #	Functional Requirements
1.	*It must pick a ferrous object from its current position and place it to a desired location, within the workspace of the robotic arm.*
2.	*Allow user to set the pick position of the object*
3.	*Allow user to set the place position for placing the object* The user has freedom to choose the location if it lies on robotic arm' workspace.
4.	*Allow user to stop the process in between and the arm goes back to the origin. (Emergency stop button)*
5.	*Allow user to set the time to complete the task.* User can change the time in milliseconds only in the software and there are not any options of doing this by hand.
6.	*Display the current location of the arm*

7.	*Indicate when the task is in process (LED indicators).*

Non-functional Requirements

Table 2: Non-functional Requirements

Requirement #	Non-functional Requirements
1.	*The system should be embedded*
2.	*The system should be real-time*
3.	*The system should be fast*
4.	*The system should be reliable* *To make sure our system is reliable; we use a potentiometer to learn the real time position of the robotic arm. Additionally, we use the emergency stop button in case of human error, for instance the user moves the robotic arm outside of its range* *(0-180 degrees)*

4. Methodology

The goal of a motor controller is to cause a motor to spin in a desired manner. We control position as in moving end-effector. It is important to understand the mechanics of how the motor interacts with its world and the behavior of the electronics interfaces. The motor controller uses sensors to measure the current position of the motor. The controller accepts input commands defining the desired operation. The microcontroller periodically senses the inputs (potentiometer value) and calculates the power needed (using PWM signals) to minimize the difference between measured a desired position. This needed power is output to the motor. Motor controllers are real-time systems, because performance depends greatly on when and how fast the controller software runs. Accuracy, stability, and time are important.

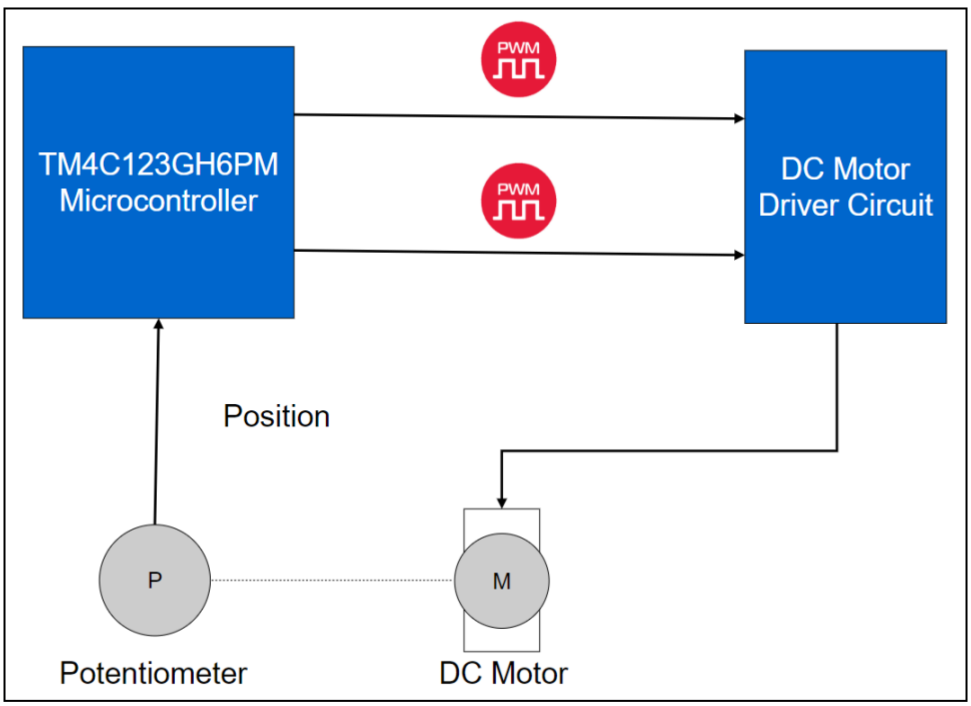

Figure 2: Robotic Arm's DC motor diagram

Figure 3 illustrates a simplified diagram for the 1 DoF robotic arm’s dc motor interfacing with the TM4C123GH6PM microcontroller. By mapping the potentiometer’s variable voltage values, we can get an accurate position of the end effector’s location in space.

Using the mapped location, we can control the direction and speed of the motor using the interface shown in figure 3, in order to reach the desired positions in the robot's workspace. This will help in achieving the main goal of pick and place.

One of the challenges that we envision in developing this project will be coming up with a good software design including an efficient algorithm. A good design early on will make the development process easier by allowing team members to be assigned well defined areas of responsibility. We decided to follow a modular programming approach for this project. Reduced coupling in the design will also allow for better software verification and improve flexibility in case implementation must be modified at a later stage.

Interrupts

There are many reasons to consider interrupt synchronization. The first consideration is that the software in a real-time system must respond to hardware events within a prescribed time. Given a change in input, it is not only necessary to get the correct response, but it will be necessary to get the correct response at the correct time.

An interrupt uses hardware to cause special software execution. With an input device, the hardware will request an interrupt when input device has new data. The software interrupt service will read from the input device and save in a global structure. With an output device, the hardware will request an interrupt when the output device is idle. The software interrupt service will get data from a global structure, and then write to the device.

Sometimes we configure the hardware timer to request interrupts on a periodic basis. The software interrupt service will perform a special function. A data acquisition system needs to read the ADC at a regular rate.

H-Bridge

Function

IN1	IN2	OUT1	OUT2	Mode
1	1	L	L	Brake
0	1	L	H	CW/CCW
1	0	H	L	CCW/CW
0	0	High impedance		Stop

MAKE A TABLE

5. System Description

- Picture of Robotic Arm embedded system

The component of the embedded robotic arm system is described next.

The Developed of the one DoF robot arm system has embedded platform comprises TM4C123GH6PM an ARM architecture-based development board. Developed robotic arm is depicted in figure 1. All modules of robotic system are described below.

5.1 Controller Module

TM4C123GH6PM Microcontroller

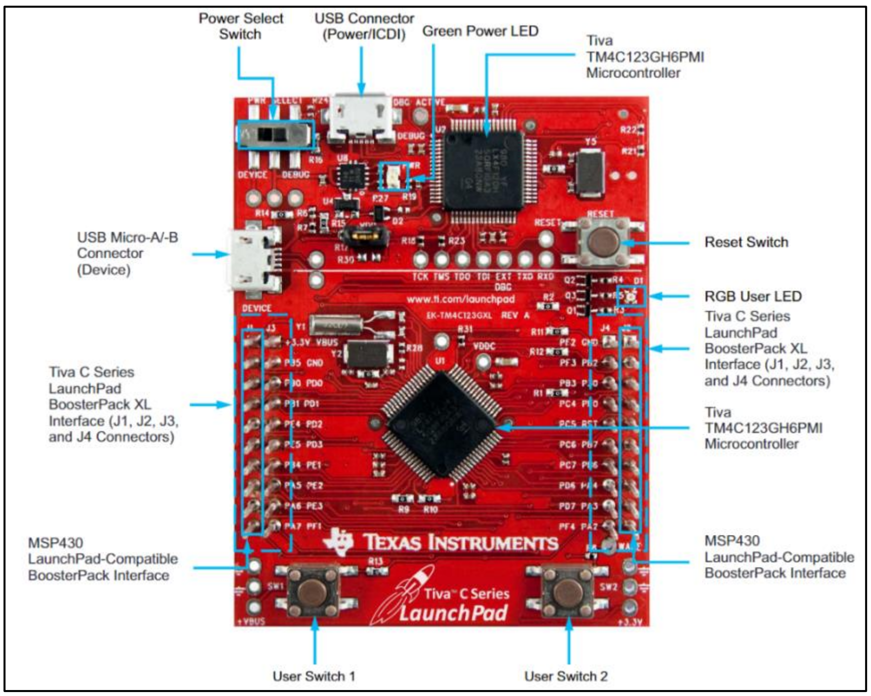

Figure 3: Tiva C Series TM4C123G LaunchPad Evaluation Board [1]

The TM4C123G LaunchPad Evaluation Kit is a low-cost evaluation platform for ARM Cortex-M4F based microcontrollers from Texas Instruments

5.2 Actuator Module

DC Motor

Electromagnet

5.3 Sensor Module

Potentiometer

We use a potentiometer to get the position of the robotic arm. Two terminals must be connected to the power supply and the third one is the variable that would give us the position of arm from the reference line. In order to find the exact position of the robotic arm, we must program ADC registers on the microcontroller because potentiometer terminals are connected to the microcontroller's registers.

Figure 4: Potentiometer

The rotary potentiometer that is connected to the motor shaft will remain unchanged if the arm is not moving. Therefore, it can be implemented in the code that if this value does not change for a certain period, it indicated process has ended.

5.4 Driver Module

For controlling the DC's motor direction of rotation, we need to reverse the polarity to the voltage. For doing so, we would need to use the H-bridge in our system so we can control the motion of robotic arm. The module is shown below:

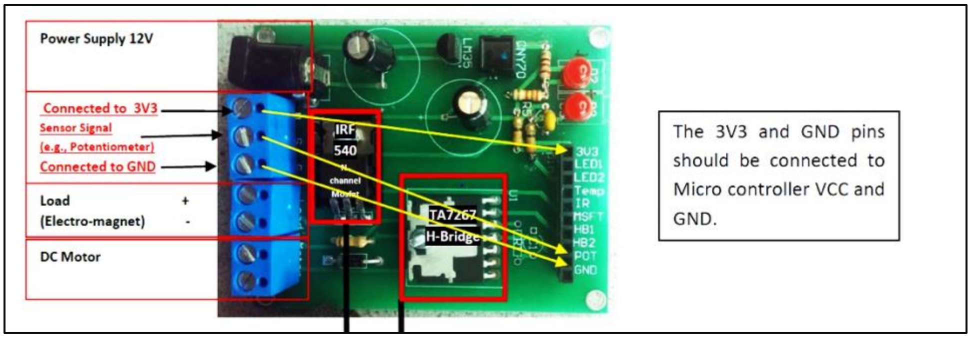

Figure 5: Motor Driver

5.5 PWM Module

5.6 ADC Module

5.7 User Interface Module

Push Buttons

Seven Segments

LED Indicators

6. Implementation

- **Elaborate on your implementation of the design, as well as observed/measured results in this section.**
- **Provide plots and figures from your experimentation to help explain your design and results.**

Explain all implementation steps in one paragraph.

6.1 DC Motor Model

- Schematic of DC motor
- DC motor equations derivation
- Transfer function
- Controller design

6.2 SOFTWARE DEVELOPMENT AND IMPLEMENTATION

Flowcharts

- Motor Control Flowchart
- Program Flowchart

7. Results and Discussions

Robot Pictures

- Start Picture
- Pick Picture
- Place Picture
- Reset Picture

Performance measures to evaluate the effectiveness of our system

- Latency
    - Latency is the time between when the I/O device indicated service is required and when service is initiated.
    - Software latency in this case is the time between when the ADC is supposed to be started and when it is started.
    - The microcontroller-based control system also employs periodic software processing.
    - Areal-time system is one that can guarantee a worst-case latency.

- Bandwidth

  Throughput or bandwidth is the maximum data flow (bytes per second) that can be processed by the system. Sometimes the bandwidth is limited by the I/O device, while other times it is limited by computer software.

- Priority

  Priority determines the order of service when two or more requests are made simultaneously. Priority also determines if a high-priority request should be allowed to suspend a low-priority request that is currently being processed. We may also wish to implement equal priority so that no one device can monopolize the computer. In some computer literature, the term soft real-time is used to describe a system that supports priority

8. Conclusions

- An embedded design for a classical pick and place robotic arm application has been developed and closed loop approach is adapted to control the speed of the DC motor using embedded system.
- Character device driver program is developed for DC motor control strategy.
- This design is implemented on ARM9 based TM4C123GH6PM microcontroller. Proportional, integral and derivative parameters are obtained for various load conditions to set the speed.
- The design goal of this project has been addressed which consists of picking an object from its current position and placing it at the desired location, within the workspace of the robotic arm. In addition, displaying the current position of the robotic arm on a 7-segment display is a part of design goals. In order to achieve a complete reliable system, objectives that need to be fulfilled has been discussed in the report.

References

[1] Tiva™ C Series TM4C123G LaunchPad Evaluation Board. 2020.

[2] J. Valvano, Embedded Systems Design. 2020.

[3] Interface board pdf, Jason Wang

[4] 1 DoF Robotic Arm Description, Jason Wang

[5] Lecture Notes, Jason Wang

Appendix

CCS Project Code