Chapter 31   Simple Robot

Lab 13.4   Simple Robotic Arm

Using the program from Ch. 8 for the servo arm, design a robotic arm that has four degrees of motion. The arm is to move in a set number of motions with a time delay between motions so as to pick up a coin and lay it down in another spot.

The four axes control the four servo motors in the robot with PWM output control. The processor is the S7-1215 DCDCDC model. The axes are set up similar to the code below. The program can be written to move between various settings for the sp_x variables for the four axes.

Due to the weight of the end effector, the 5th and 6th servos are removed. The end of arm tool is replaced with the suction device shown below. This is attached to servo 4 shown at right.
Specifications for the MG996R Servo follow. The high current requires a significant 5V power supply. The Lambda 5V-5A supply shown lower right is sufficient for the load.

**Specifications of Servo**

- Weight: 55 g
- Dimension: 40.7 x 19.7 x 42.9 mm approx.
- Stall torque: 9.4 kgf-cm (4.8 V), 11 kgf-cm (6 V)
- Operating speed: 0.17 s/60° (4.8 V), 0.14 s/60° (6 V)
- Operating voltage: 4.8 V to 7.2 V
- Running Current 500 mA – 900 mA (6V)
- Stall Current 2.5 A (6V)
- Dead band width: 5 μs
- Stable and shock proof double ball bearing design
- Temperature range: 0 °C – 55 °C
We have four PWM outputs from the PLC but we need a fifth servo controller to control the lynxmotion vacuum end of arm tool. The best way to control it is to use an Arduino with a digital output connected from the PLC to the Arduino and the Arduino directly connected to the servo controlling the vacuum tube. The picture for the connection is similar to the button connection shown below:

![Wiring Diagram](image)

Code for the Arduino is as follows:

```c
#include <Servo.h>

// constants won’t change
const int BUTTON_PIN = 7; // Arduino pin connected to button's pin
const int SERVO_PIN = 9; // Arduino pin connected to servo motor's pin

Servo servo; // create servo object to control a servo

// variables will change:
int angle = 0; // the current angle of servo motor
int currentButtonState; // the current state of button

void setup() {
    pinMode(BUTTON_PIN, INPUT_PULLUP); // set arduino pin to input pull-up mode
    servo.attach(SERVO_PIN); // attaches the servo on pin 9 to the servo object

    servo.write(angle);
}

void loop() {
    currentButtonState = digitalRead(BUTTON_PIN); // read new state

    if(currentButtonState == LOW)
```

// change angle of servo motor
angle = 90;
else
angle = 0;

// control servo motor according to the angle
servo.write(angle);

Wiring for the four servo signals and the suction device is as follows:

![Wiring Diagram]

Configuration of the S7-1215 follows:

![Catalog Information]

Short designation: CPU 1215C DC/DC/DC
Description: Work memory 125 KB; 24VDC power supply with DI14 x 24VDC SINK/SOURCE, DQ10 x 24VDC and AI2 and AO2 on board; 6 high-speed counters and 4 pulse outputs on-board; signal board expands on-board I/O; up to 3 communication modules for serial communication; up to 8 signal modules for I/O expansion; PROFINETIO
Article number: 6ES7 215-1AG40-0XB0
Firmware version: V4.4
The PWM outputs are controlled through configuring PWM1, PWM2, PWM3 and PWM4 below:

Variables that control movement are sp_1, 2, 3, and 4 (setpoints). The program may be tested by setting ena to 1 and test_1, 2, 3, and 4 to 1. The setpoint variables are the position points.

Configuration of the PWM axes follow. They all resemble PWM1 shown below:
The other outputs are wired as follows:

- Axis 2 – Q0.1
- Axis 3 – Q0.3
- Axis 4 – Q0.4

(These values could have been Q0.0, Q0.1, Q0.2 and Q0.3 but it was found that the second output had some problems with a RC time constant and the waveform was not squared as with the other points. Use of an oscilloscope is extremely valuable in troubleshooting the waves.)
Output variables to be written to are:

- Axis 1 - QW1008
- Axis 2 – QW2
- Axis 3 - QW1014
- Axis 4 - QW8

Each variable is 16 bit (Word length).

Each axis must contain the following logic:

This instruction must be inserted to set up the PWM motion.

This lab may be programmed several different ways. The best would include arrays that have positional data for the various axes. As the robot moves through various positions, the robot program moves down through the list of position data at each position. There should also be a rate of movement for each of the axes (or perhaps just one common rate as programmed here). The rate of movement is important as well as sequence of the positional data.

At present, only one of these robotic arms is available. The 5 V separate power supply is extremely important as the current draw by the various axes is large.
The following logic is a timer that either increases or decreases the value to be output to the PWM. The setpoint determines whether the value is to grow or decrease. This code is repeated for each of the four axes. To save movements, begin to think about construction of an array or UDT. Either will allow a sequence of steps that can be programmed to control the movement.
For testing purposes, construct a watch table to change various axes’ setpoints.
The following pictures show the construction of the robot with a breadboard for resistor values.
This lab works! It allows the construction of a moving robot from the PLC. It also encourages the construction and use of arrays for recipe storage. Additional labs may be made to set up the movements and program the robot as well as just control the movements via set program statements.

Note that servo only attached on left side

End Tool – Modified from original
Ch 31  Simple Robot  13
Network 4: .....
Ch 31  Simple Robot
Ch 31  Simple Robot
The Purpose of this Senior Design Project was to integrate a Universal Robot (shown on the right side) with a Cognex Vision Camera and Siemens S7 1215C PLC. The demonstration can be viewed on YouTube at https://youtu.be/x7R7q14b8g4.

This work is licensed under a Creative Commons Attribution 4.0 International License.