Micromega Corporation

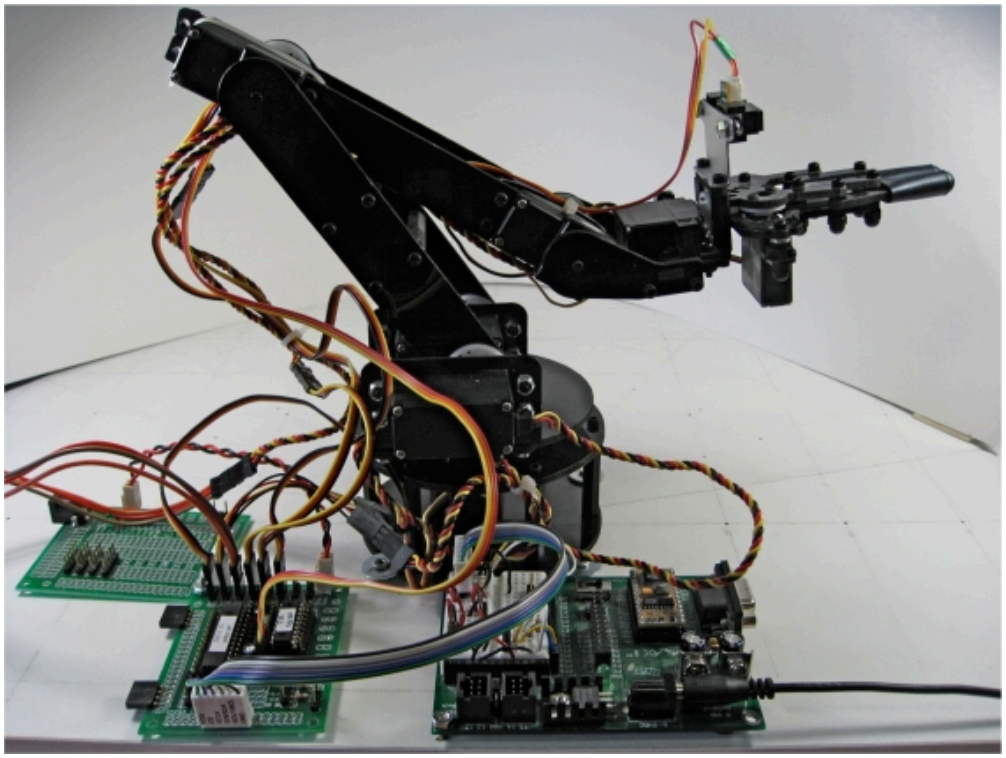
Application Note 44

Controlling a Lynx6 Robotic Arm

Introduction

This application note describes the control of a Lynx6 robotic arm (www.lynxmotion.com) using an embedded microcontroller and the uM-FPU V3 floating point chip. The Lynx6 has six separate servo motors that control the position of the arm and the gripper. The uM-FPU V3 chip is used to perform the necessary inverse kinematic calculations to determine the angle of each servo motor to position the gripper at a particular x, y, z coordinate.

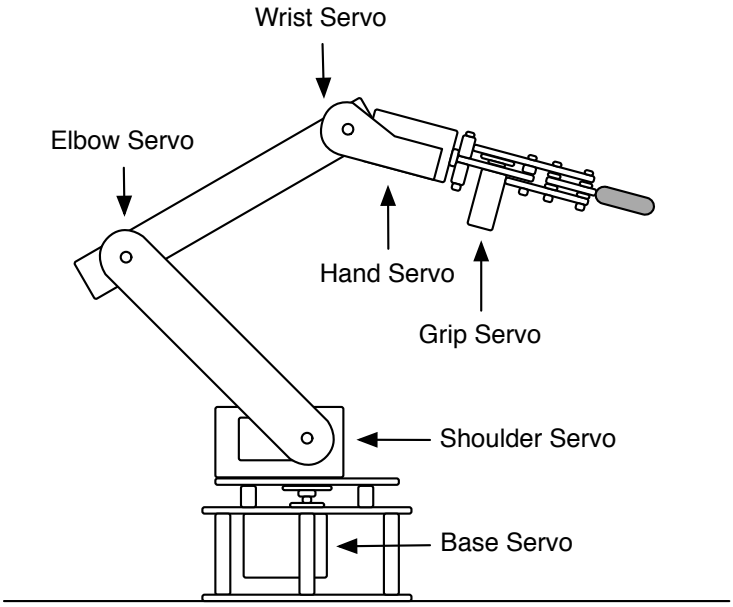
The following picture shows the Lynx6 robobotic arm with a GP2D120 distance sensor mounted on the top of the hand. This example uses a Basic Stamp microcontroller connected to a uM-PWM1 servo co-processor and uM-FPU V3 floating point coprocessor. Almost any microcontroller could be used, since the majority of the work is done by the uM-FPU and uM-PWM1 chips.



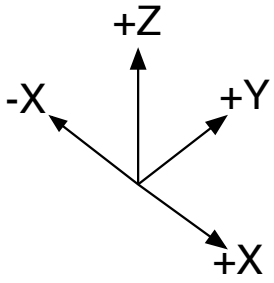
The Lynx6 Robotic Arm

The Lynx6 robotic arm has six servo motors that control the movement of the robotic arm: base, shoulder, elbow, wrist, hand (wrist rotation), and grip. The *x*, *y*, *z* coordinate space for the arm is defined as follows:

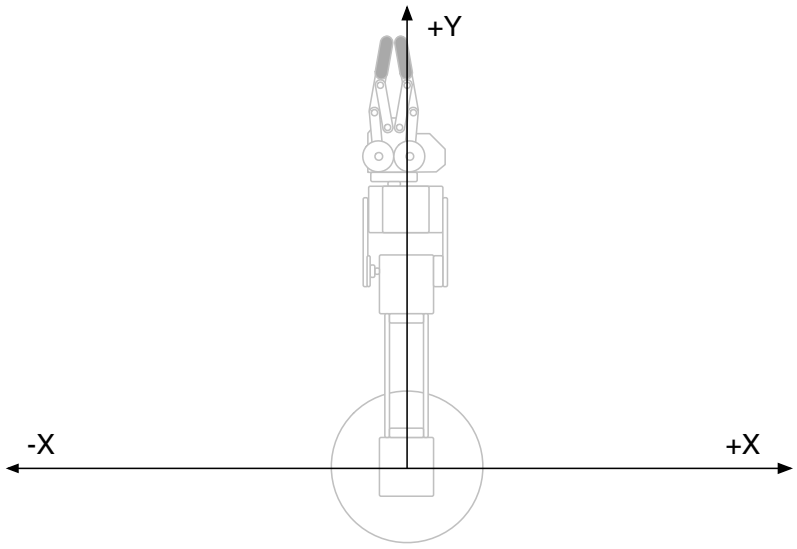
- the base of the robotic arm rests on the horizontal *x*, *y* plane
- the vertical z-axis is perpendicular to the *x, y* plane
- the center of the base is defined as *x, y, z* = 0, 0, 0



Side View



Coordinate Space



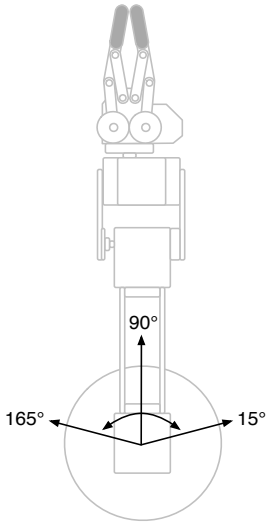
Top View

Range of Motion

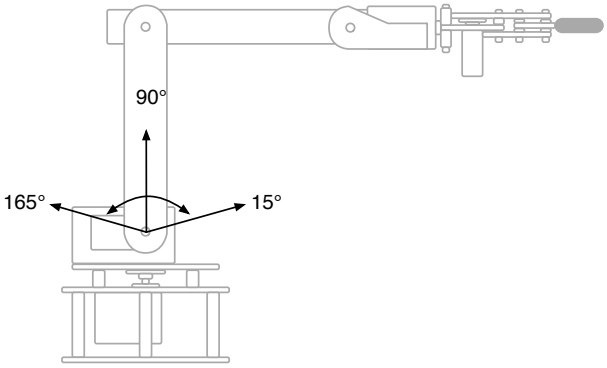
Servo motors are controlled by pulse-width modulated signals that control the position of the servo actuator. Servo pulse widths vary from 500 to 2500 microseconds, which corresponds to a servo rotation angle of approximately 0° to 180°. A pulse width of 1500 microseconds will set the servo at the neutral position, or 90°. The range of motion and direction of travel for the servos used in the Lynx6 robotic arm are summarized below. The base servo controls the rotation of the robotic arm in the horizontal *x*, *y* plane. The shoulder, elbow and wrist servos position the robotic arm in the vertical *z* plane.

The base servo has a range of motion from 15° to 165°. The robotic arm is aligned with the positive *y*-axis when the angle of the base servo is at 90°.

Base Servo

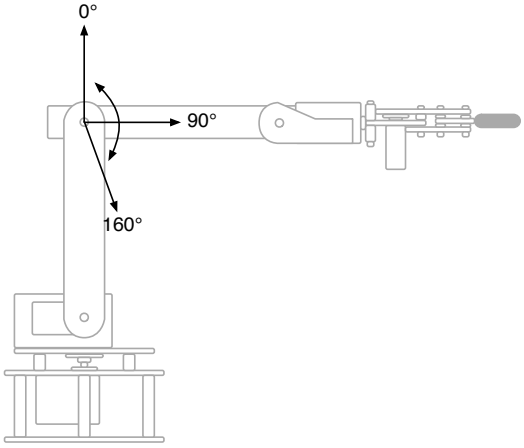


Shoulder Servo

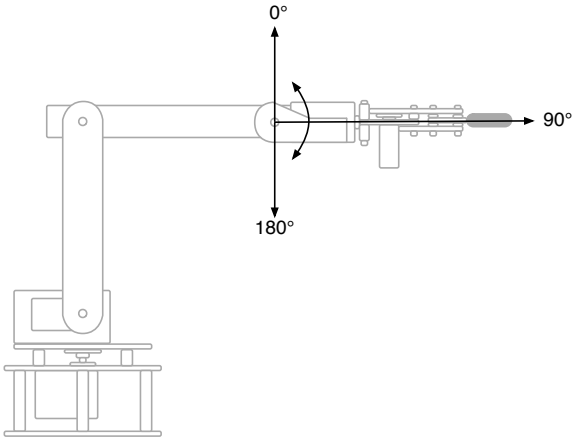


The range of motion for the shoulder servo is from 15° to 165°. The upper arm (from shoulder to elbow) is aligned with the positive *z*-axis when the angle of the shoulder servo is at 90°.

Elbow Servo



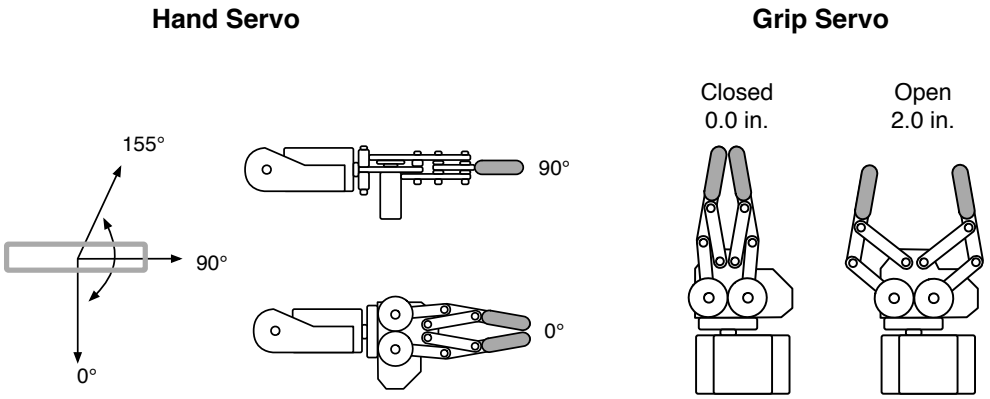
Wrist Servo



The range of motion for the elbow servo is from 0° to 160°. The lower arm (elbow to the wrist) is at right angles to

the upper arm when the angle of the elbow servo is at 90°. If the shoulder and elbow servos are both at 90°, the upper arm will be parallel with the horizontal *x*, *y* plane.

The range of motion for the wrist servo is from 0° to 180°. The hand and gripper are colinear with the lower arm when the angle of the wrist servo is at 90°.



The hand servo is used to rotate the hand and gripper mechanism. The range of motion for the wrist servo is from 0° to 155°. If the lower arm is parallel to the horizontal *x*, *y* plane, the hand would be horizontal when its angle is 90°, and vertical when its angle is 0°. When viewed from the wrist, the rotation of the hand is clockwise as the hand angle decreases, and counter-clockwise as the hand angle increases.

The grip servo is used to open and close the gripper mechanism. The range of motion for the gripper is from closed at 0.00 inches, to fully open at 2.0 inches.

Positioning the Robotic Arm in 3-D Space

To position the robotic arm in 3-D space, the angle of each joint must be set. If the physical dimensions of the robotic arm and the angle of all joints is known, the position of any point in the robotic arm assembly can be calculated by starting from the base and calculating the position of each joint successively, until the *x, y, z* coordinates of the point of interest are determined. This is called forward kinematics.

The opposite calculation, calculating the required angle for each joint that results in the point of interest being located at a specific *x, y, z* coordinates, is called inverse kinematics.

The sample program described in this application note uses inverse kinematic calculations. The *x, y, z* coordinate being solved for is the grip point, which is defined as the tip of the gripper when the gripper is fully closed, or the center of the grip mechanism when the gripper is open. The reason for the difference is based on the assumption of how the gripper is being used. If the gripper is open, it's being used to pick up an object, so the point of interest is the center of the grip mechanism. If the gripper is closed, it's being used to touch objects (i.e. press a button, activate a lever), so the point of interest is the tip of the gripper.

Calibration

The positioning angle of the servos in the robotic arm are not completely linear with respect to the pulse widths provided to them. This is due to non-linearities in the servo motors themselves, and the various effects, or constraints, due to the mechanical connections of the robotic arm. In order to achieve proper positioning for this application, each servo is calibrated at the start and end point of its range of motion, and at 30° intervals. The pulse

width for each of these calibration points is stored in a table. The pulse width for a servo is then determined by interpolating between these calibration points. The *SetServo* and *TableLookup* functions perform the interpolation.

Inverse Kinematic Calculations

The inverse kinematic calculations are performed on the uM-FPU V3 floating point chip. The FPU functions that perform the calculations are summarized later in this document. The Basic Stamp sample program and the source code for the FPU functions can be downloaded from the Micromega website. In this application note, all coordinates are specified in inches.

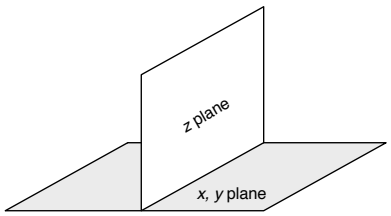
The following inputs are specified:

- The *x, y, z* coordinate for the grip point.
- The angle of the grip from horizontal.
- The width of the grip.

The following values are known:

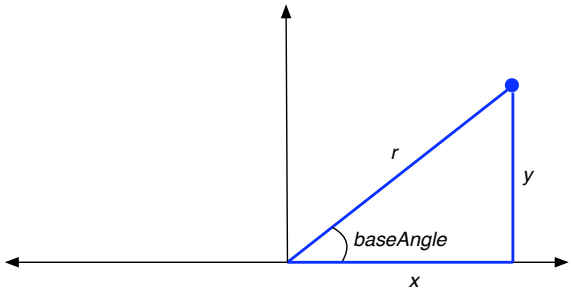
- The base is located at coordinate 0, 0, 0.
- The height of the shoulder joint is 3.0 inches above coordinate 0, 0, 0.
- The length of the upper arm (shoulder joint to elbow joint) is 4.75 inches.
- The length of the lower arm (elbow joint to wrist joint) is 4.75 inches.
- The hand is held at 90° (i.e. the hand is horizontal when the lower arm is parallel to the *x, y* plane).
- The grip length (wrist joint to grip point) varies from 5.2 inches to 5.9 inches, depending on the grip width.

The 3-dimensional inverse kinematic calculation can be reduced to 2-dimensional calculations, by viewing the robotic arm as being located in a *z* plane that is perpendicular to the *x, y* plane. The *x, y* plane is the horizontal surface that the base of the robotic arm rests on. The *z* plane is the vertical plane that the center line of all parts of the robotic arm mechanism resides in. If you look down at the robotic arm, you are looking at the *x, y* plane, and the robotic arm components will all be in a straight line. If you look at the robotic arm from the side, you are looking at the *z* plane.



The *x, y* Plane

Calculations in the *x, y* plane use coordinates *x, y*.



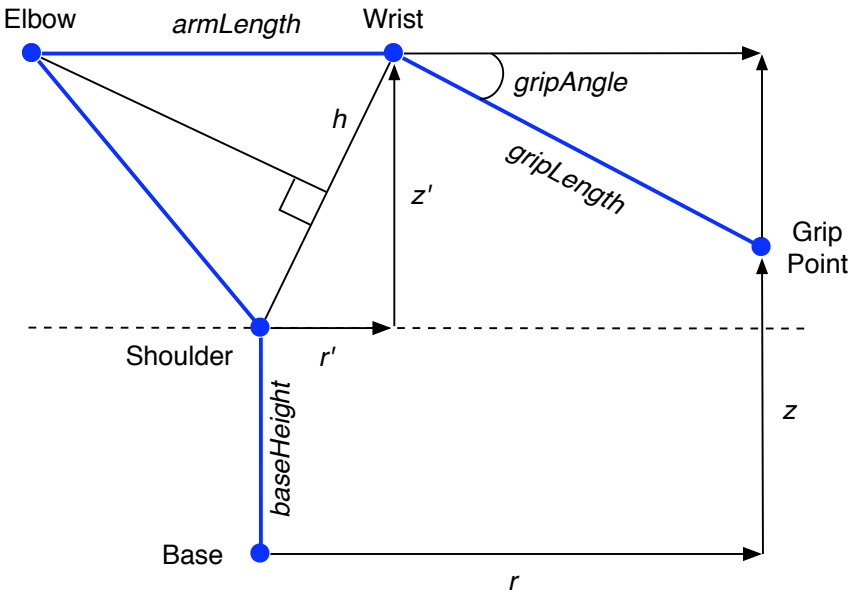
The *z* plane is positioned by the rotation of the base servo and passes through the points *0, 0, 0* and *x, y, 0*. The distance from the point *0, 0, 0* to *x, y, 0* is the radial distance, *r*.

The base angle and the radial distance *r* is calculated in the *SolveXYZ* function as follows:

```
baseAngle = atan(y/x)
r = sqrt(x² + y²)
```

The *z* Plane

Calculations in the *z* plane use coordinates *r, z*.



Grip Length

The grip mechanism uses a scissor action to open and close the gripper, so the grip length changes depending on the grip width (the distance between the jaws of the grip mechanism). To determine the relationship of length to width, the grip length was measured at various grip widths and the results were plotted. The trend line was not a straight line. It was determined to be a second order polynomial. The uM-FPU V3 chip has a POLY instruction that solves for Nth order polynomial equations. The *setGripLength* function uses the POLY instruction to calculate the length of the grip at a specified width. If the width is zero, the length is adjusted to extend to the tip of the gripper.

Grip Angle

There are many different solutions for the joint angles that can position the grip point at a desired *x, y, z* coordinate. For this application, the grip angle (the angle of the gripper from horizontal) is specified, to constrain the solution to a single result. This has the desired effect of knowing from what angle an object will be gripped. The calculations are performed in the *SolveRZ* function. Using the grip angle and grip length, the location of the wrist joint is determined, and the r′, z′ values are calculated.

```
r′ = r - (sin(gripAngle) * gripLength)
z′ = z - baseHeight + (cos(gripAngle) * gripLength)
```

The elbow angle can now be determined as follows:

```
h = sqrt(z′² * r′²) / 2
elbowAngle = asin(h / armLength) * 2
```

Knowing the elbow angle, the shoulder angle can be calculated.

```
shoulderAngle = atan2(z' / r') + ((PI - elbowAngle) / 2)
```

The wrist angle is then determined by the summing the other joint angles.

```
wristAngle = PI + gripAngle - shoulderAngle - elbowAngle
```

Sample Application

The sample application shows two scenerios:

- picking up an object at a specified location and moving it to a central container
- searching for an object, then picking up the object and moving it to a central container

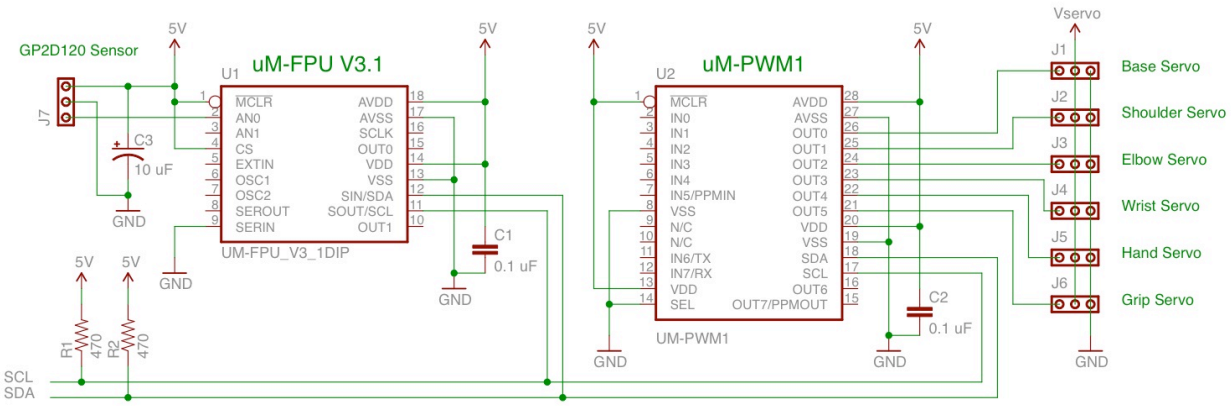
At the end of the program the robotic arm is returned to the rest position, and the uM-PWM1 chip disables the control signal for all servos.

A Youtube video has been posted that shows the sample application.

YouTube video: *Lynx6 Robotic Arm, uM-FPU V3, uM-PWM1*
www.youtube.com/watch?v=IolvRa4uZoA

Schematic

The schematic diagram for the circuit used in the sample application is show below.



Microcontroller Interface

The Basic Stamp microcontroller used in the sample application is an 8-bit device, so all of the *x, y, z* coordinates and angles are stored as 8-bit variables. The *x, y, z* coordinates are specified as signed 8-bit values in 1/10 inch units. This gives coordinates a range of -12.8 inches to 12.7 inches, which nicely covers the working space of the robotic arm. The angles are specified as unsigned 8-bit values in degrees. This covers the maximum range of motion of the servos from 0° to 180°.

On the uM-FPU V3 floating point chip the angles are converted to radians, and all calculations are performed using 32-bit floating point. The results are converted to integer values before being read back to the microcontroller.

uM-FPU V3 Floating Point Coprocessor

The uM-FPU V3 floating point coprocessor makes it possible for a microcontroller to control the Lynx6 robotic arm, by offloading the inverse kinematic calculations. The microcontroller communicates with the FPU using integer values, but calculations on the FPU are performed using 32-bit floating point. The FPU is also used to read the GD2D120 distance sensor through an analog input.

The inverse kinematic calculations are stored as user-defined functions on the FPU, which can be easily called from the microcontroller with very little overhead. The FPU functions are summarized in the section entitled *LynxArm.fpu Functions*.

For further details on this device see the *uM-FPU V3.1 datasheet*.

uM-PWM1 Servo Controller

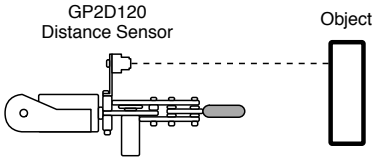
The uM-PWM1 provides an I²C interface for controlling up to eight servos. The uM-PWM1 servo controller is used to control the six servo motors in the robotic arm. Several features of this chip make it well suited to this application. Two of the most important features are speed control, and the coordinated movement of a group of servos. To move from one point in 3-dimensional space to another, requires different changes in angle for each servo. If these servos are instructed to move individually, the resulting move is irregular and jerky. The uM-PWM1 chip provides the ability to move smoothly from one point to another at a constant rate, by specifying that a group of servos should move simultaneously. The uM-PWM1 chip coordinates the individual servo movements so that all servos arrive at the new position simultaneously. This provides very smooth movement of the robotic arm.

Another concern when dealing with a robotic arm is the speed of movement. If a joint angle is changed too rapidly, mechanical damage could occur. This is particularly true for joints that are supporting a large part of the weight of the arm. The uM-PWM1 provides the capability to set the maximum output speed for each servo channel. These values can be stored in Flash memory on the uM-PWM1 so that the maximum output speed is set automatically at power up.

For further details on this device see the *uM-PWM1 datasheet*.

GP2D120 Distance Sensor

The Lynx6 robotic arm was enhanceed to include a GP2D120 infrared distance sensor mounted on a bracket at the rear of the gripper.



The GP2D120 sensor produces an analog voltage that is proportional to the distance to the closest object in its field of view. The analog output is connected to the uM_FPU V3 analog input for processing. The algorithm used is to scan the working field in an arc from 30° to 120° in 5° increments looking for a local maximum. If one is found, a reverse scan is performed at 1° increments to try and pinpoint the local maximum. The gripper is then aligned at the correct angle and the distance to the object is calculated. The robotic arm is then instructed to pick up the object.

LynxArm.fpu Functions

getID

Returns an ID number that the main program can use to determine if the correct set of user-defined functions have been programmed on the uM-FPU V3 chip. If the correct `getID` function is programmed, a value of 44 is returned.

Input:

`none`	

Output:

`register 0`	44 (long integer)

setGripWidth

Calculates the grip length. Since the gripper uses a scissor action to open and close the gripper, the length of the gripper changes depending on the width. This is not a linear relationship. A second order polynomial is used to calculate the length of the grip at the specified width. If the width is zero, the length is calculated to the end of the grip. If the width is not zero, the length is calculated to the center of the jaws of the grip.

Input:

`gripWidth`	grip width (1/100 inch units)

Output:

`gripLength`	grip length (1/10 inch units)
`gripPulse`	grip pulse width (microseconds)

SolveXYZ

Performs the inverse kinematic calculations to determine the shoulder, elbow, and wrist angles to position the grip point at the specified *x, y, z* coordinate. The grip angle is used to first determine the location of the wrist joint along the

Input:

`xval, yval, zval`	coordinates of grip point
`gripAngle`	angle of grip (degrees from horizontal)
`gripLength`	grip length (1/10 inch units)

Output:

`angleArray`	base, shoulder, elbow, wrist servo angles (radians)

SolveXYRZ

Performs the inverse kinematic calculations to determine the shoulder, elbow, and wrist angles to position the grip point at radius *r,* and height *z*, along the line extending from *0, 0* to the specified *x, y* coordinate.

Input:

`xval, yval`	*x, y* coordinate of grip point
`rval`	radius
`zval`	height, *z*
`gripAngle`	angle of grip (degrees from horizontal)
`gripLength`	grip length (1/10 inch units)

Output:

`angleArray`	base, shoulder, elbow, wrist servo angles (radians)

SolveRZ

Performs the inverse kinematic calculations to determine the shoulder, elbow, and wrist angles to position the grip point at radius *r,* and height *z*. This positions the robotic arm in the *z*-plane.

Input:

`rval`	radius, *r*
`zval`	height, *z*
`gripAngle`	angle of grip (degrees from horizontal)

	`gripLength`	grip length (1/10 inch units)
Output:		
	`angleArray`	shoulder, elbow, wrist servo angles (radians)

SetAllServos

Check all servo angles for minimum and maximum values, and calculate all pulse widths.

Input:		
	`angleArray`	servo angles
Output:		
	`angleArray`	servo angles (radians)
	`pulseArray`	servo pulse width (microseconds)

SetServo

Calculates the pulse width for the specified servo angle. If the servo angle is less than the minimum angle for that servo, the angle is changed to the minimum. If the servo angle is greater than the maximum angle for that servo, the angle is changed to the maximum. The pulse width is determined by using the calibration values stored in a lookup table. For angles between the calibration values the result is interpolated.

Input:		
	`unit`	servo unit (0-base, 1-shoulder, 2-elbow, 3-wrist, 4-hand, 5-grip)
	`angleArray`	servo angles (radians)
Output:		
	`angleArray`	servo angles (radians)
	`pulseArray`	servo pulse width (microseconds)

TableLookup

Returns a value from the calibration table. The calibration table contains calibration values for all servos. It specifies the minimum angle, maximum angle, and pulse width values for the minimum angle, maximum angle and at every 30° angle in between.

Input:		
	`register 0`	table index
Output:		
	`register A`	table value

Further Information

See the Micromega website (http://www.micromegacorp.com) for additional information regarding the uM-FPU V3.1 floating point coprocessor, and uM-PWM1 servo coprocessor, including:

uM-FPU V3.1 Datasheet
uM-FPU V3.1 Instruction Set
uM-PWM1 Datasheet
Using the uM-FPU V3 Integrated Development Environment (IDE)
YouTube video: Lynx6 Robotic Arm, uM-FPU V3, uM-PWM1
www.youtube.com/watch?v=IolvRa4uZoA