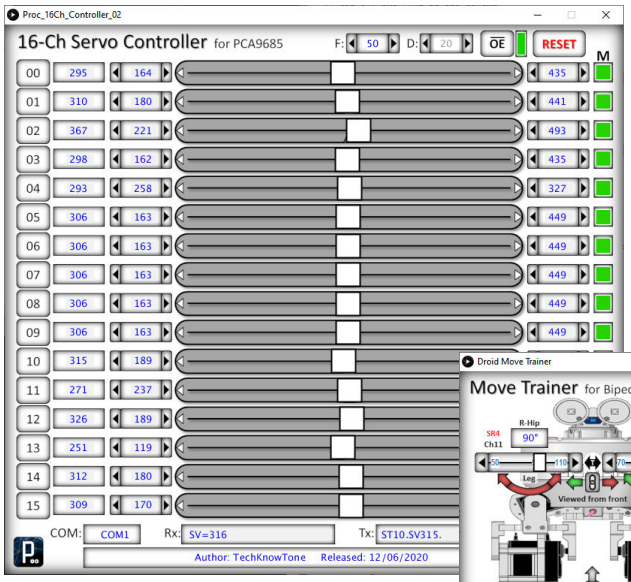
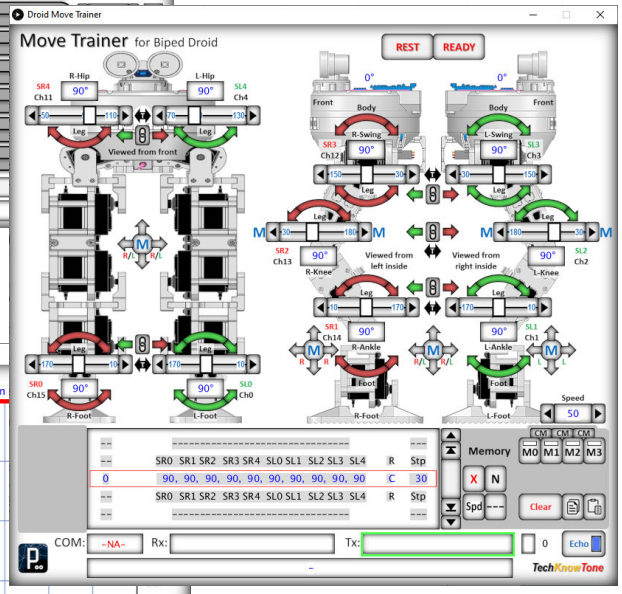
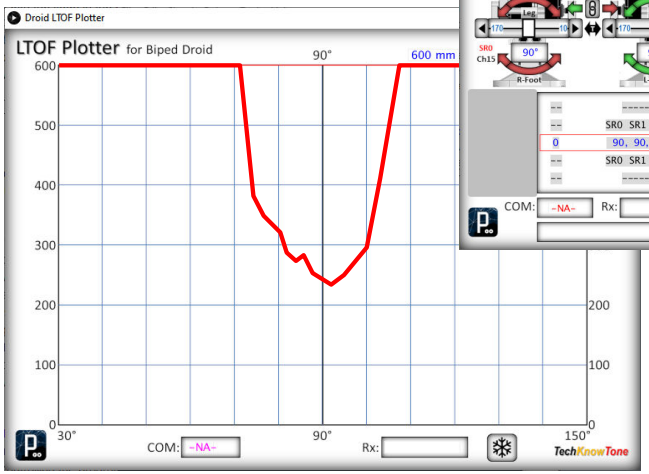
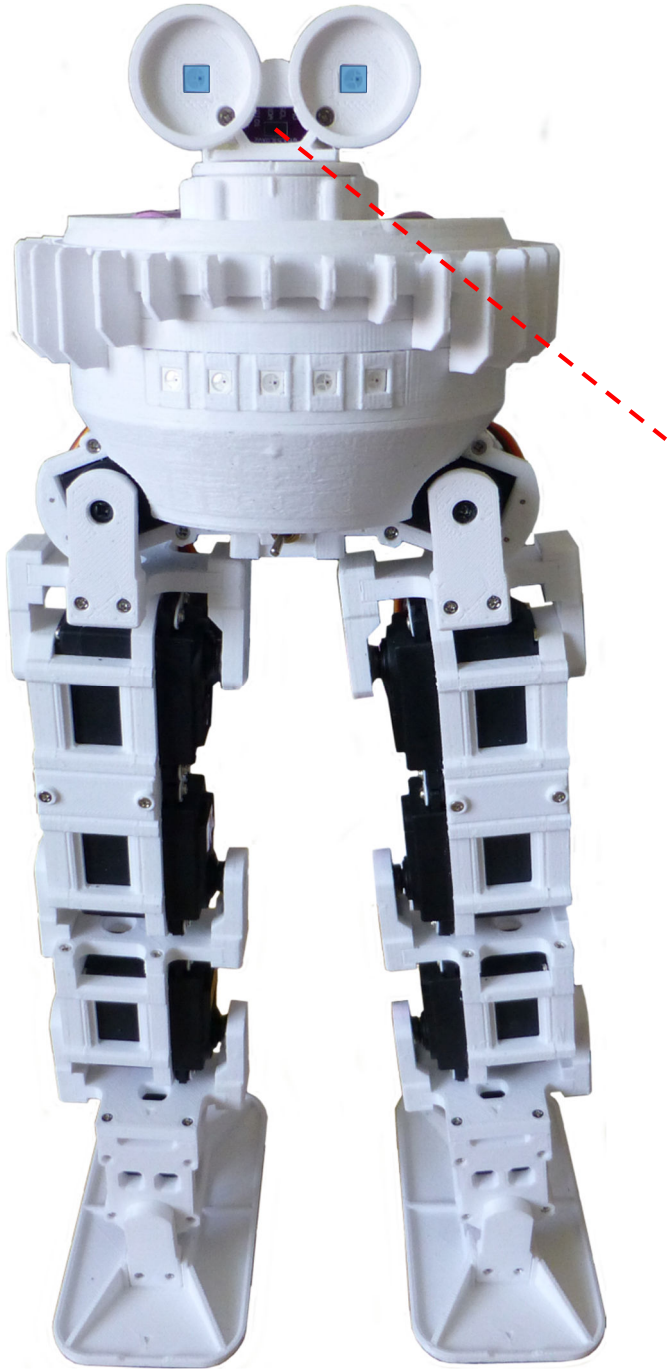
Biped Droid

3 Windows Apps









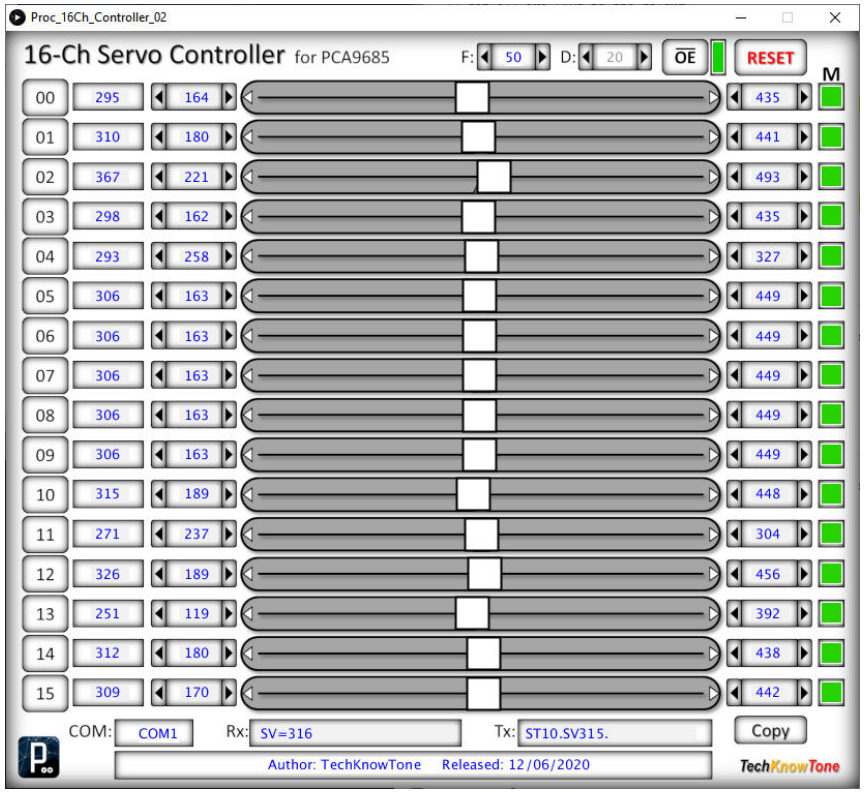
ARDUINO OPEN-SOURCE COMMUNITY

Issue: 1.0 Released: 20/06/2020 TechKnowTone

16-Channel Servo Controller

This app is designed to work in conjunction with the Arduino code in the file 16-Ch_Controller_00.ino. Its purpose is enable you to determine the PWM values needed from the PCA9685 driver board, to drive given servos to their respective end points. Once you have determined these values you simply copy and paste them directly into the Arduino IDE. When the app starts it looks for an active COM port and connected to it. This will reset the Arduino, which in turn will transmit its default values over the serial port. On receipt of these values the app will use this data to set limits and sliders for each channel. You can then toggle the OE pin to enable the output of the driver board and select a channel number on the left to enable that slider value to be sent to the serial port. The Arduino receives this and sets the corresponding servo PWM to suit and move it. The channel lamp will go from green to yellow, indicating that a change has occurred. Click on the lamp to restore the value, or click and hold the lamp to change it to blue and record the new setting. When the COPY button is pressed it is only the green and blue settings that are sent to the Window clipboard.

The Help field at the bottom of the form changes as the mouse pointer is moved over items on the form. Use this feature to learn more about the various buttons and fields on the form. To release the COM port for Arduino coding, etc, simply right-click on the COM field. Then use a left-click to instruct the app to connect to the port again. I hope you find this useful in this project and any others that use the PCA9685 driver board.

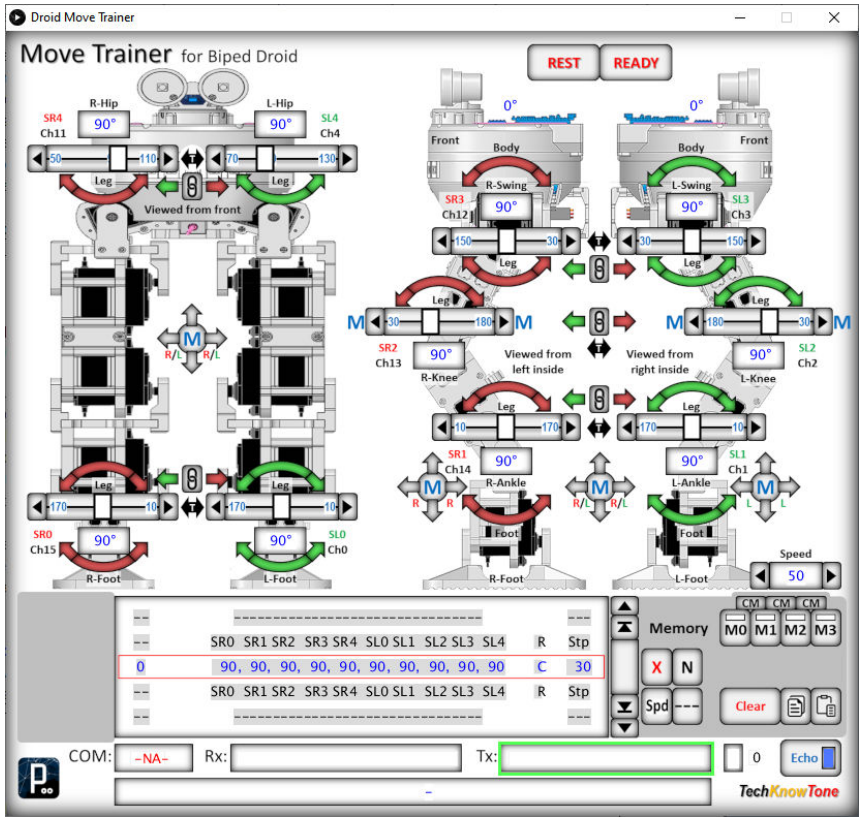



```
// Define PWM freq and servo calibration arrays
#define Freq 50  // MG996R servos run at ~50 Hz updates
// Servo:   { 0   1   2   3   4   5   6   7   8   9   A   B   C   D   E   F }
int SLL[] = {164,180,221,162,258,163,163,163,163,163,189,237,189,118,177,176};
int SRV[] = {295,310,367,298,295,306,306,306,306,306,315,271,326,248,310,309};
int SUL[] = {435,442,491,435,331,449,449,449,449,449,448,304,456,388,451,442};
```

Move Trainer

This app is designed to work in conjunction with the droids Arduino code. Its purpose is enable you to create servo movement sequences that can be performed directly by the droid. Once calibrated, each servo has an angle and a set of upper and lower limits. The background picture in the app shows the servo assignments and provides sliders for their adjustment. For example SR0 is the right-hand foot servo. The app lists up to 100 movement states for all 10 servos. By default all servos are at 90° and this is the first line shown in the list. When you move a servo slider the number in the associate field will change, and also the number in the list. You simply add more lines to build up a sequence of movements. The end figure is the number of discrete angle steps to be used to move from the previous line to the current line. So the actual servo speed is a function of the speed setting and the number of steps taken. You will see these lists in the Move_Engine code. You can copy and paste them into the app, and then copy them out again with changes made. You can single step through the move using keyboard arrow keys and/or the mouse wheel.

The Help field at the bottom of the form changes as the mouse pointer is moved over items on the form. Use this feature to learn more about the various buttons and fields on the form. To release the COM port for Arduino coding, etc, simply right-click on the COM field. Then use a left-click to instruct the app to connect to the port again. I hope you find this useful in this project and any others that use the PCA9685 driver board.



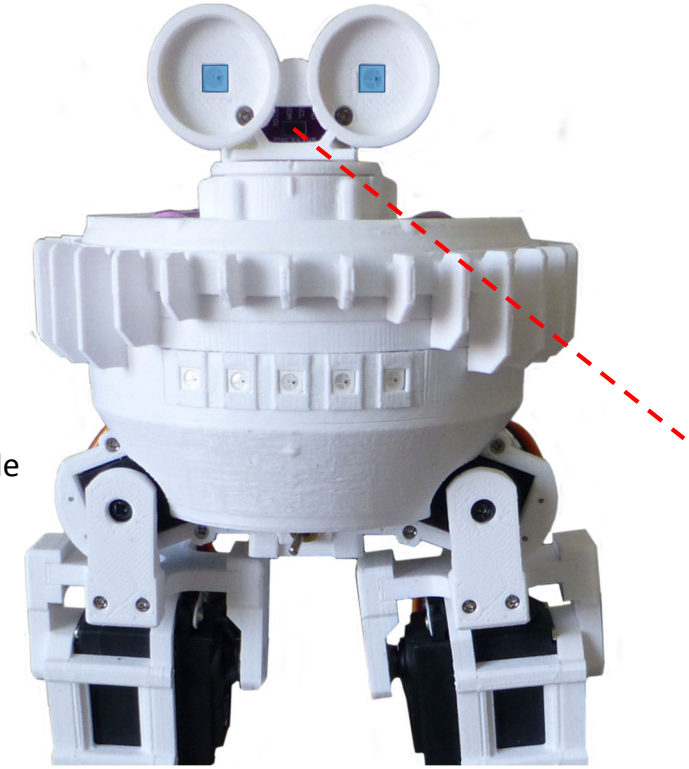

```
//      SR0,SR1,SR2,SR3,SR4,SL0,SL1,SL2,SL3,SL4,Stp
MemSet( 90, 70,130,110, 90, 90, 70,130,110, 90,420);
MemSet( 90, 30,180,126, 90, 90, 30,180,126, 90,230);
MemSet( 90, 30,180,126, 90, 90, 30,180,126, 90, 30);
MemSet( 90, 30,180,150, 90, 90, 30,180,150, 90,415);
MemSet( 90, 30,180,150, 90, 90, 30,180,150, 90, 30);
MemSet( 90, 30,180,115, 90, 90, 30,180,115, 90,430);
MemSet( 90, 30,180,115, 90, 90, 30,180,115, 90, 30);
MemSet( 90, 70,130,110, 90, 90, 70,130,110, 90, 30);
MemSet( 90, 90, 90, 90, 90, 90, 90, 90, 90, 90,430);
MemSet( 90, 90, 90, 90, 90, 90, 90, 90, 90, 90, 30);
MemSet( 90,137, 30,127, 90, 90,137, 30,127, 90,425);
MemSet( 90,137, 30,127, 90, 90,137, 30,127, 90,430);
MemSet( 90,137, 30, 79, 90, 90,140, 30, 79, 90,430);
MemSet( 90,137, 30,115, 90, 90,142, 30,115, 90,420);
MemSet( 90, 90, 90, 90, 90, 90, 90, 90, 90, 90,230);
MemSet( 90, 70,130,110, 90, 90, 70,130,110, 90,425);
```

LTOF Plotter

This app is designed to work in conjunction with the droid Arduino code. Its purpose is to receive data from the Arduino over the serial COM port link and display it in real time as a graph. When calibrated the droids head moves in angular terms from 30° (right) to 90° (centre) to 150° (left). Placing the droid in range plotting mode (pressing the left-hand button 4 time) causes the code to move the head in a scanning right-left fashion and to send range and angle data over the serial link as an 8 character packet RArrraaa, where rrr is the range from 000 – 600 mm, and aaa is the head angle from 030 – 150°. Using a fixed length packet makes it easier for the app to decode the values. So this helps you see what the droid can see with its VL53L0X laser range finder. The code in the droid locks onto any target which is closer than 500 mm, which you can see in the short video on my web site.

To release the COM port for Arduino coding, etc, simply right-click on the COM field. Then use a left-click to instruct the app to connect to the port again. I hope you find this useful in this project and any others that use the VL53L0X sensor.

Range + Angle



Issue: 1.0 Released: 20/06/2020 TechKnowTone