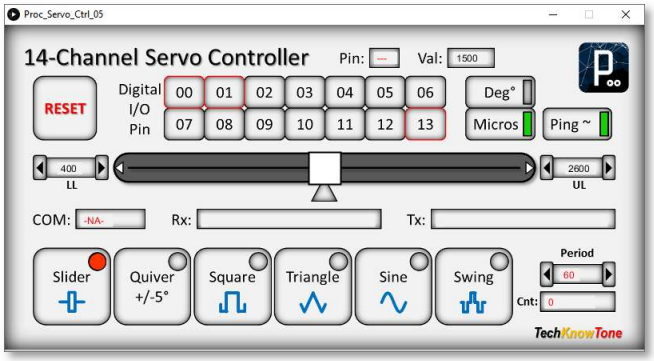
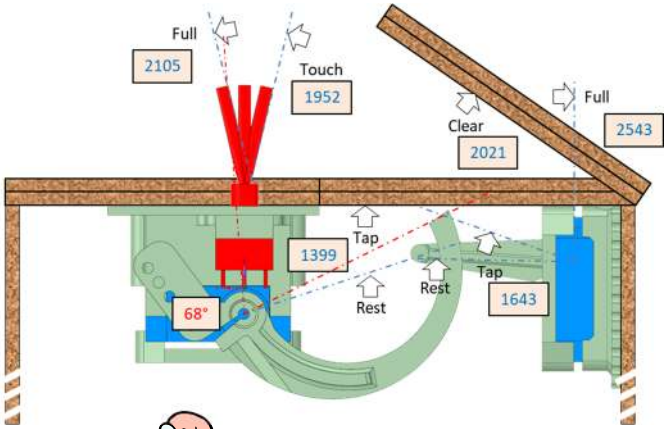
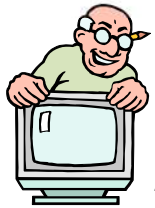
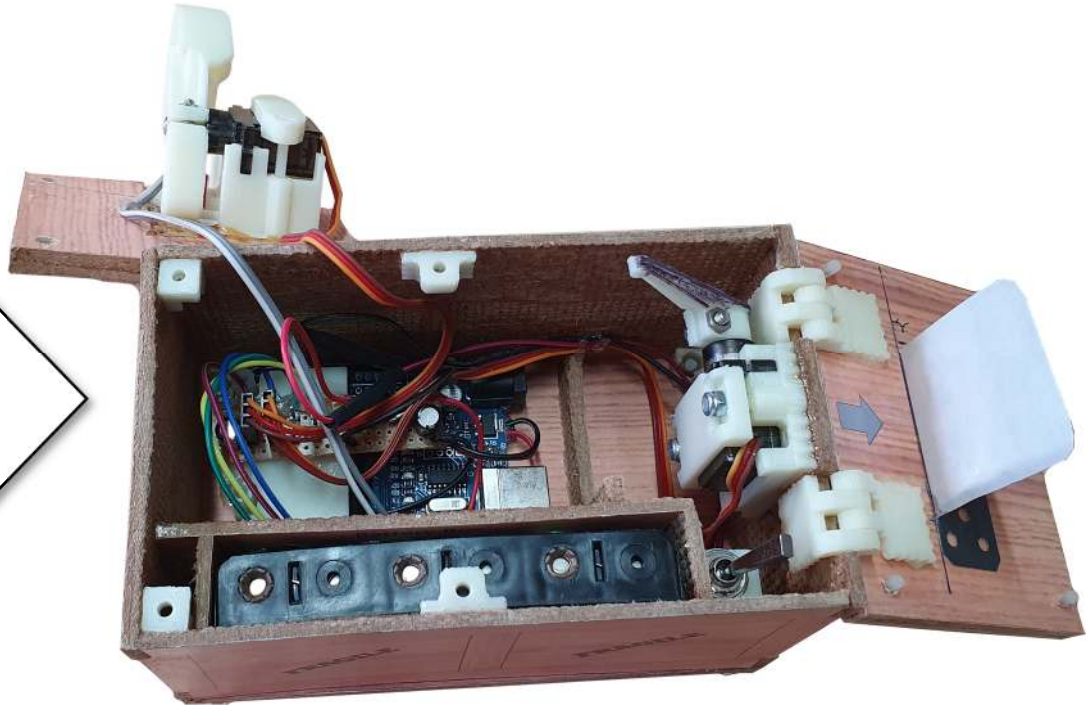
Useless Machine

Servo Calibration









An important process, that will govern your robots performance.

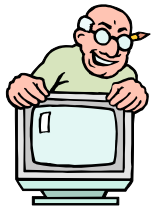
Why do we need to calibrate the servos?

- No two servos are the same.
- Servos can be damaged if not setup correctly.
- Course calibration must be performed during the assembly process.
- This sets approximate positions for the leaver arms.
- A servo drive shaft with 20 splined teeth == 18° (best fit +/-9°)
- Course calibration ensures servos are within mechanical range/limits.
- Fine calibration determines min/max robot physical limits.
- The C++ code needs limit values in order to work accurately.
- Hence, all servos have a unique set of calibrated PWM code values.
- No two Useless Machines would ever be exactly the same.
- Code limits and default values would be different for each one.
- Once calibrated we can use angles, as common values, not PWM.

Servo calibration is performed in two/three stages:

- Course ensures mechanical parts are assembled correctly.
- Fine calibration, performed during testing, for accurate movement.

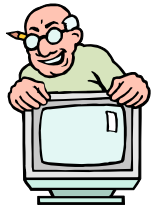
- Repeat this process for a given servo if it is ever replaced.



Only use genuine Tower Pro servos, for best performance.

Issue: 1.2 Released: 18/02/2024 TechKnowTone

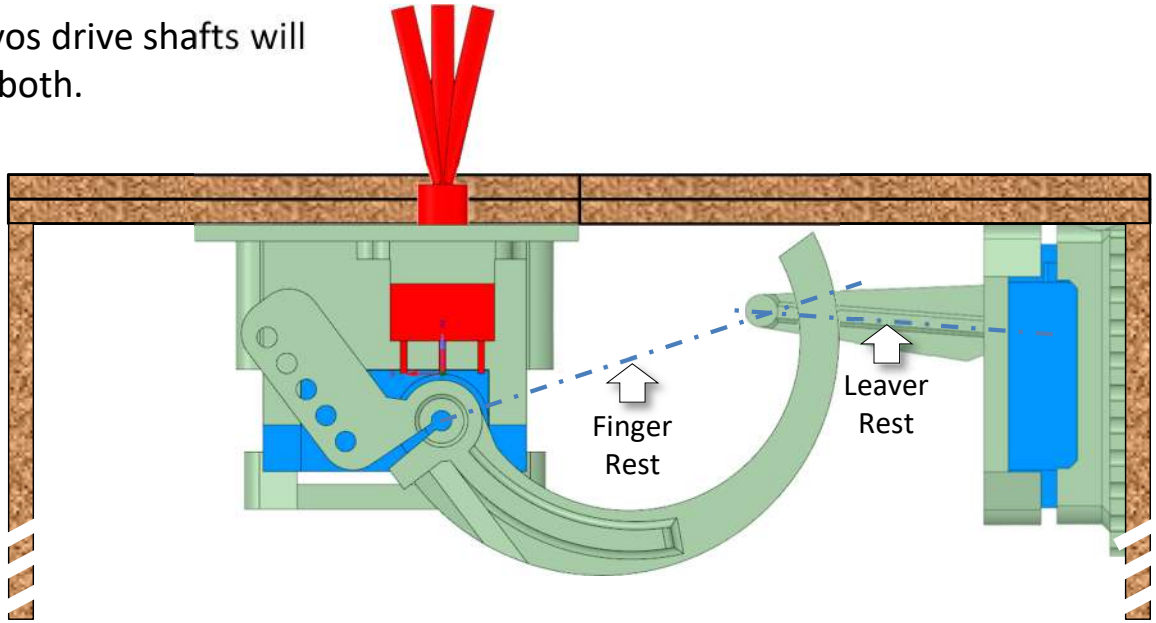
Course Calibration



This process is performed before connecting the lid leaver, and finger to their respective servos. It ensures that the two parts are fitted in their approximate default positions. Transfer the code supplied to the Arduino Uno, using the IDE and a suitable USB cable. Power up your project with batteries, or a suitable power source, set to the correct voltage. On power up, or following a RESET, the code will output PWM values which represent the default 'REST' positions.

Then attach both leaver and finger, such that they are approximately 6-9mm below the inner surface of the lid. Note – in this diagram the finger looks closer to the lid than this.

The dimension is approximate, as the splines on the servos drive shafts will probably prevent you from achieving an exact figure for both.



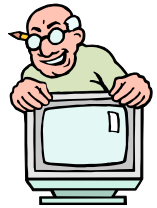
Box viewed from the side in cross section.

Now, press the RESET button on the UNO to cause it to run the runPOST() code, in which it will take the leavers to their REST positions, then twitch them towards their 'TAP' positions, before returning them to their REST positions, and detaching them.

If both leavers return to their REST positions, then you have completed the course calibration process, and can now move onto the fine calibration process.

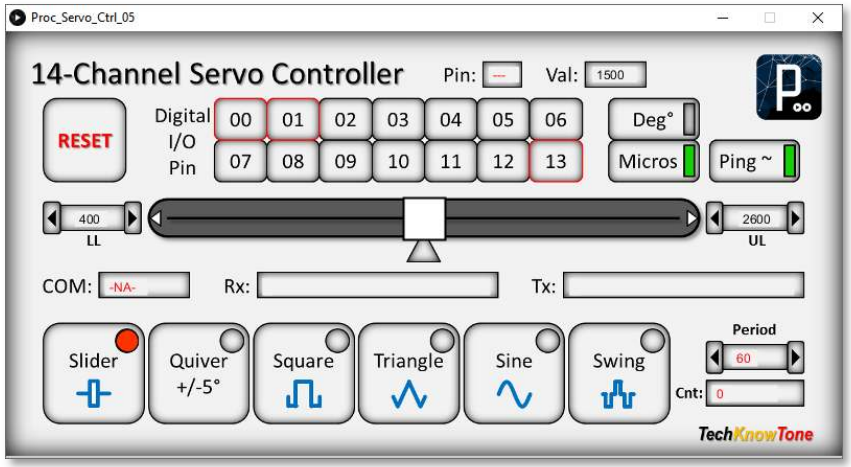
Note that the base of the box is removeable, to charge batteries, but also to gain access to the UNO with a USB lead.

Fine Calibration



This process is performed after completing the previous course calibration process, which has attached your lid leaver and finger to their respective servos, at their default positions.

Fine calibration is about determining the PWM values needed in the code, to drive your servos to specific angles, and there by lift the lid or operate the switch for example.



This is best done using PWM signals generated by the UNO itself, and to achieve this I have provided a Windows app, written specifically for this task. The app communicates with the code in the UNO over the serial port, at 19200 Baud. It connects automatically, provided the serial port is not already held by another application, or the Serial Monitor in the IDE. When it is connected the COM: field will indicate which COM port is being used.

Once connected, you simply click on a button to select a UNO output pin, to control (in our case it will be 04 or 05), then move the horizontal slide, left or right to change the PWM signal width, being generated. The variables shown opposite, will be set by you to the PWM values you determine, and relate to the angles shown in the diagram on the following page.

```
62  long servoFgrMax = 2105; //Full finger servo value
63  long servoFgrTch = 1952; //Touch finger servo value
64  long servoFgrTap = 1399; //Tap finger servo value
65  int servoLL = 544; // servo min time setting
66  long servoLidMax = 2543; //minimum lid servo value
67  long servoLidClr = 2021; //Clear lif servo value
68  long servoLidTap = 1643; //Tap lid servo value
```

Because of differences in servos and setups, it is very unlikely that my values will work in your project; but they may be close.

IDE - If this app does not work on your PC, you can still set servo PWM values using the IDE serial monitor, by typing in and sending values this app would normally generate. For example to select output pin 5 send **SP5.** or for output pint 4 send **SP4.**. To set a PWM value for a pin, say 1399, you would send **SM1399.** after the pin assignment. You only need to send the pin reference, when changing the selection, or for the first time. After that you can simply send PWM values. All commands must end in a '**.**' to be recognised by the code in the UNO.

Servo calibration (Example):

Here I show the PWM values I determined from my two servos, for given angles of the finger and lid lifting leaver.

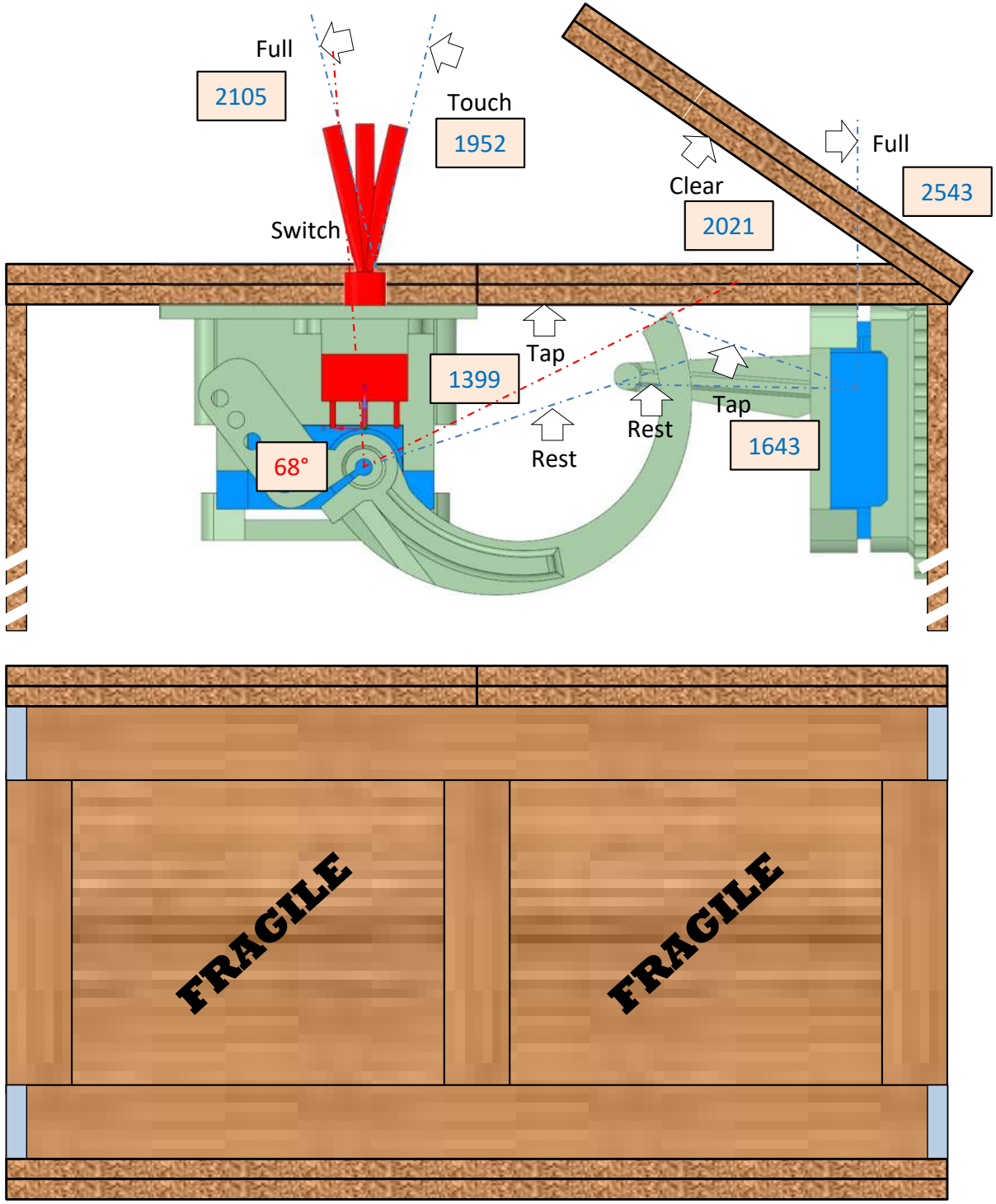
Record PWM µs values for:

- Finger Full (0°) – switch open
- Touch – finger touches switch
- Finger Tap (68°) – on lid
- Lid Full (0°) – see those eyes
- Clear – finger can pass
- Lid Tap (71°) – lid noise

Rest positions are set during course calibration.

The plastic finger shown is an early version, of the more natural looking one in the final build.

It is important to use a long levered switch, so that the servo has sufficient torque to operate it. Short stubby switches may not work.



Issue: 1.2 Released: 18/02/2024 TechKnowTone