Example Servo Timing Calculation

When controlling servo motors, we generally adjust frequency (or period) and duty cycle to set the servo angle. Recall the following basic formulas for period and frequency

$$\text{period} = \frac{1}{\text{frequency}} \qquad \text{frequency} = \frac{1}{\text{period}}$$

Next we should look up the timing for our specific servo, in the video the Hitec HS-645MG is used, a little Googling will lead to the following page:

https://www.servocity.com/html/hs-645mg_ultra_torque.html#.VgVjyOSFO38

Scroll to the bottom and you will find to command a 0 deg angle requires a pulse high for 600 us (0.6 ms) and a 180 deg angle requires a 2400 us (2.4 ms) pulse. Therefore to achieve a spread of 180 deg movement we need a spread in pulse time of 2.4 ms - 0.6 ms = 1.8 ms, as follows:

$$\text{0 deg} \xrightarrow{\text{180 deg}} \text{180 deg} \qquad \text{0.6 ms} \xrightarrow{\text{1.8 ms}} \text{2.4 ms}$$

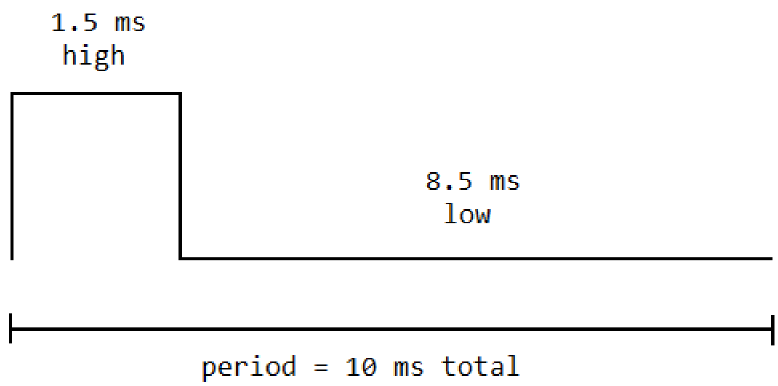
The spread continues proportionally (ideally linearly) throughout the 180 deg range of the servo, for example if we send the following pulse times we should achieve the following servo angles:

pulse	angle
0.6 ms	0 deg
1.05 ms	45 deg
1.5 ms	90 deg
1.95 ms	135 deg
2.4 ms	180 deg

With most servos if we send a pulse at least every 20 ms that is sufficient to maintain the servo position, to be on the safe side lets send a pulse every 10 ms. Therefore we require a period of 10 ms (0.01 sec), or a frequency of 100 Hz (see formula for frequency above):

$$\frac{1}{0.01\ \text{sec}} = 100\ \text{Hz}$$

Referring to the pulse & angle table above, a high pulse of 1.5 ms will set the servo to 90 deg, and we would like to use a period of 10 ms, so for example if we are commanding our servo to 90 deg our cycle sent to the servo should be as follows:



So now we have our desired frequency which we can set directly in the Raspberry Pi in code, but what about the duty cycle (the other number we can set in code in the RPi)? Since we have a total spread of 180 deg and 1.8 ms to cover, we can calculate the following:

$$\frac{1.8\ \text{ms}}{180\ \text{deg}} = 0.01\ \text{ms per deg}$$

Keeping in mind from above that our offset to get a 0 deg position is 0.6 ms and our period is 10 ms, we have the following:

$$\left(\left(\begin{matrix}\text{desired}\\\text{angle}\end{matrix} \times \begin{matrix}\text{ms per}\\\text{deg}\end{matrix}\right) + \begin{matrix}\text{offset in ms}\\\text{to get 0 deg}\end{matrix}\right) \times \begin{matrix}\text{period}\\\text{in ms}\end{matrix} = \begin{matrix}\text{duty cycle}\\\text{in \%}\end{matrix}$$

For the Hitec HS-645MG specifically we would have:

$$\left(\left(\begin{matrix}\text{desired}\\\text{angle}\end{matrix} \times \begin{matrix}\text{0.01 ms}\\\text{per deg}\end{matrix}\right) + 0.6\ \text{ms}\right) \times 10\ \text{ms} = \begin{matrix}\text{duty cycle}\\\text{in \%}\end{matrix}$$

In Python we could code this as:

```
dutyCycle = ((float(intAngle) * 0.01) + 0.6) * 10
```

This will give us a starting point. Due to various things not working out as ideally as calculated you will in most cases have to make minor adjustments. For example starting with the above formula produced close results but on the RPi I found the servo angle to move from -5 deg to 175 deg, so I experimented until I found changing the offset as follows produced 0 deg to 180 deg movement:

```
dutyCycle = ((float(intAngle) * 0.01) + 0.5) * 10        # note the offset is changed from 0.6 to 0.5
```

With most servos and most controllers you will find it necessary to make final experimental fine tuning changes like this at the end, however the above process should at least get you close.