

## Optimising the response of your Throttle

### A. Checking throttle response

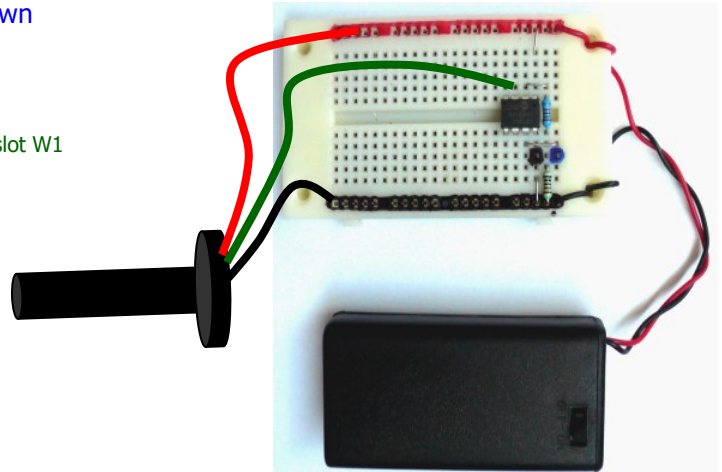
- Connect the throttle to a Picaxe prototyping board as shown
- Program the Picaxe with

```

Main:
  Readadc C.4,W1    'start of program
  Debug W1          'store the incoming ADC value in memory slot W1
  Pause 50          'show the incoming W1 values
  Goto Main         'Delay for 50/1000ths of a second
                   'Return to start
  
```

Watch the debug numbers from throttle off to throttle full.

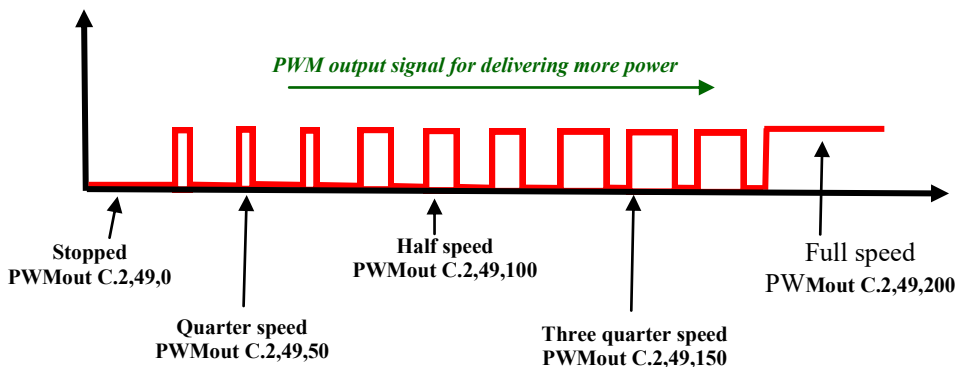
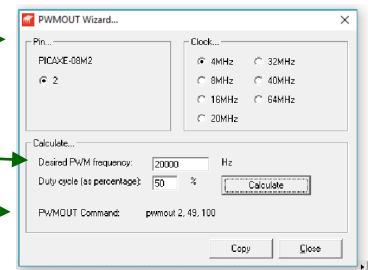
	W1 debug values
Throttle off	
Throttle full	



### B. Setting your PWM output values

Use the **PWM Wizard** to calculate the PWMout command

- Click on PICAXE—Wizards—PWMout
- You will come to this box
- Set frequency to **2000Hz** instead of the default 10000Hz as the motor responds best to this higher frequency
- This tool will calculate the PWM values for various speeds. Here it has calculated the values in the command for half speed.



The PWMout command is in the form **PWMout Pin, Period, Duty** where the duty is the on time for each cycle

As you can see only the Duty changes from Off 0 to On=200

### C. Relating Throttle debug number to motor speed on PWMout command (Duty)

	W1 debug values	Duty values (W2)
Throttle off	Your throttle value (mine=60)	0
Throttle full	Your throttle value	200

Graph is of type  $y = mx + c$

So  $W2 = mW1 + C$

To show this calculation I have put some numbers in from my throttle. Repeat this with your own throttle values.

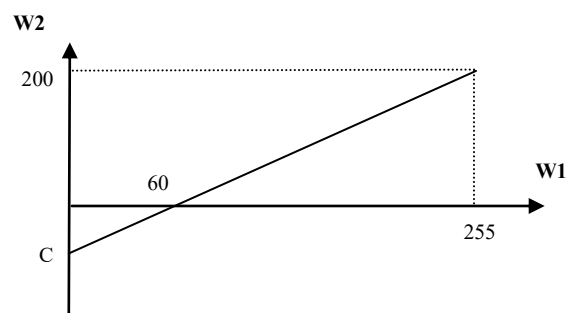
Where Slope  $m = \frac{200}{195}$

Intercept by similar triangles  $\frac{C}{200} = \frac{60}{195}$

$m = 1.03$

$C = 61.54$

Therefore equation for my throttle is  **$W2 = 1.03 * W1 + 61.54$**



## D. Using the equation will get you the most out of your controller

$$W2 = 1.03W1 + 61.54$$

Because the Picaxe can't handle decimal points we will multiply everything by 100 and then at the end divide it by 100

This becomes the program line

$$W2 = 103 * W1 + 6154$$

Note also the Word variables are memory locations that handle these bigger numbers.

## E. Programme

Adjust this to suit your equation.

Main:

Readadc C.4,W1

If W1<62 then stopped

W2=103\*W1— 6154/100

'debug W2

PWMout C.2,49,W2

pause 100

goto main

*'start of main program*

*'Takes throttle voltage and stores the number in memory W1*

*'conditional statement for stopped (62 is throttle voltage when throttle turned off)*

*'This calculates the PWMout number from the incoming throttle voltage*

*'TAKE THIS OUT BEFORE RUNNING PROGRAM - shows W values*

*'Gives the right PWMout power for the throttle position*

*'Delay allowing the motor to respond to the PWMout command*

*'returns program to start*

Stopped:

PWMout C.2,49,0

pause 100

goto main

*'Subroutine that defines stopped*

*'PWMout command to stop motor*

*'Allows the motor to respond to PWM command*

*'Returns to start of Main program*

*'the debug command slows the program execution and SHOULD NOT Be used with the motor connected..*

*'Including it will result in peculiar motor response*

*'Writing in green doesn't get downloaded*

*'Without the motor connected, remove the apostrophe in front of debug and it will change colour*

*'This allows it to be downloaded but will let you see the throttle response numbers.*

*'Hopefully the PWMout numbers (W2 values) will go from 0 (stopped) to 200 (Full throttle)*