

Energy360 – Startup Overview and Procedure

There is a lot of information on discord server here <https://discord.gg/2RUEuKFFvZ>

Dr ADHD – Together We build Motion Sims.

Document Version

This is version 1.1

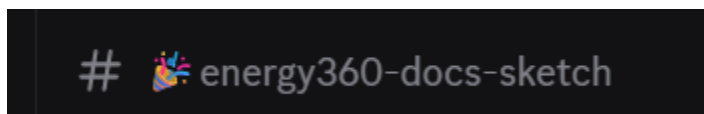
Purpose

The purpose of this document is to try to make the startup procedure of the motion sim as simple as possible, this is based on the new Energy360 boards and Energy360 Sketches.

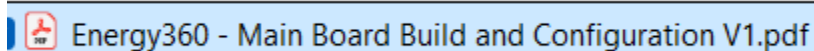
Assumptions

For this we will assume the following

- 1) You have completed the motion sim metal build as outline on the Dr ADHD video series
- 2) You have ordered the boards from www.energy360.ca the standard build includes
 - a. 2 x Main Boards [here](#)
 - b. 2 x Main Board mounting brackets [here](#)
 - c. 2 or 4 PSU LT1100A-S0 – Ebay is your friend
 - d. 2 mounting brackets for each of your PSU [here](#)
 - e. 4 x Position Sensor Boards [here](#)
 - f. 2 x Capacitor Banks [here](#)
 - g. 2 x Power Resistor Kits [here](#)
 - h. 2 x Capacitor Relay Kits [here](#)
- 3) You have followed the E360 Build procedure outlining how to build up the E360 Main Boards this documentation can be found on the discord server or energy360.ca



>>



- 4) You have mounted all the electronics and followed the wiring and fusing suggestions

 [E360 Fuse Recommendations .pdf](#)

- 5) You have read and understand the following documents

- a. Understanding SMC3Utils

[Energy360 - Demistify SMC3Utils V1.pdf](#)

- b. The various Compile time options for the E360 Sketch

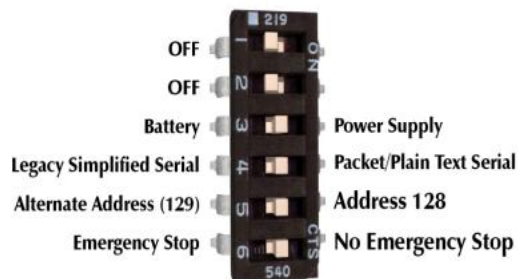
 [Energy360 - Energy360.ino Compile Time Option V1.1.pdf](#)

- c. The features of the E360 Sketch

 [Energy360 - Energy360.ino V1.pdf](#)

Step 1 - Sabertooth Hardware Switch settings

Dip Switches



Switch 1 & 2 OFF

Switch 3,4,5 ON

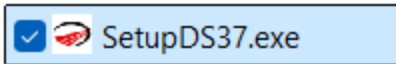
Switch 6 ON for NO Emergency STOP (start here if you like then come back)

Switch 6 OFF for Emergency Stop - Requires short circuit between Energy360 Main board V5 and MA1 and V5 and MA2, if this circuit is broken the motors will not function and stop

Step 2 – Programming the Sabertooth

Download the software and install it on the PC

<https://www.dimensionengineering.com/info/describe>



Start Software

Plug in Sabertooth you should get Red/Green Light flash every second

Select sabretooth 2x32

Press Connect and Download Setting – Note some micro usb cable will not work.

Click on Serial and USB

Serial and USB

Change Baud rate to 115200 and timeout 100mSec (0.1)

Baud Rate:	115200	▼
Plain Text Serial Addresses:	1 and 2	▼
Serial Timeout:	0.1	s

Then click Upload Setting to Device

This is now complete you can close Describe Software.

Unplug the Sabertooth, its ready to be mounted to the Energy360 Board

Step 3 Install the Auduino IDE

Go to the Arduino website and install the latest version [here](#)



Arduino IDE 2.3.8

Release notes

The new major release of the Arduino IDE is

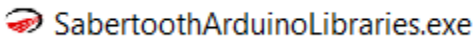
Step 4 install the Sabertooth Arduino Libraries

Go to Dimension Engineering website [here](#)

Library	Download
Arduino Libraries for SyRen/Sabertooth Serial (includes examples) v1.5 by Dimension Engineering	Installer (for Windows) or .zip File

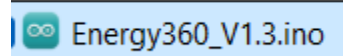
Download the Sabertooth Arduino Libraries

In downloads double click and install it, this will install the features into Arduino IDE

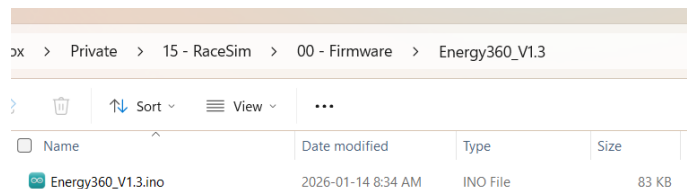


Step 5 Test Compile the Energy360 Sketch

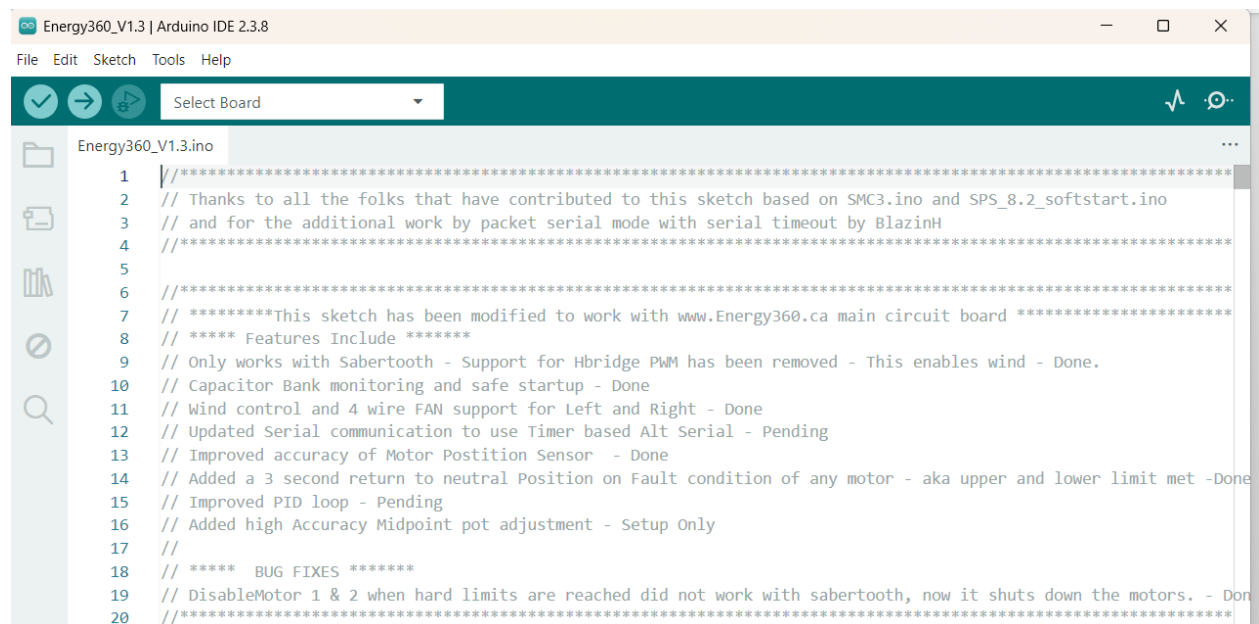
Download the Energy360 Sketch from discord or energy360.ca



Create a folder with the name Energy360_V1.3.ino and place this file inside this folder



Double click the file and the IDE should open



Remove the Arduino from the E360 Board if you have it installed

Plug it into the USB port and the Arduino IDE should recognize the board in the bottom right hand corner of the IDE.

The orange light on the board should be flashing once per second.

If it does not automatically see the board you may have to play with the USB Com Port settings refer to the Arduino website on how to check this.

Click the compile tick mark



You should see it compile without any errors. In the Output window, the black window at the bottom.

Now lets try a load of the firmware

Click on the upload button



You should see the firmware compile and then load to the Arduino.

You can now reinstall the Arduino UNO into the E360 Boards. Keep the board plugged into the USB port of the PC for the next section.

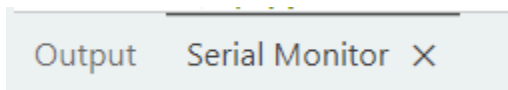
Step 6 Adjusting the POTS

This next step ensures that the Position Sensor are in their center position in relation to the neutral position of your rig. While this can be done in SMC3Utils, this procedure is way more accurate. It require the sketch to be modified for each POT and then downloaded. The position sensor raw sample rates will be outputted into the Arduino Serial Monitor screen

Click on the Serial Monitor Button



This will appear



And the white screen below.

Board 1 - POT 1

```
#define USB_to_Arduino_Baudrate 500000 // 500000 for SMC3 or 115200 for debug  
Arduino IDE serial monitor
```

Change the above to 115200 for this procedure – at the end remember to change it back to 500000

```
#define Pot_Tunning 0
```

Change the above to 1 for POT 1 (2 for POT2)

```
#define ENABLE_DEBUG_PRINTS 0 // Set to 0 for Gaming and SMC3 - keeps  
serial port quite
```

Change the above to 1 during POT calibration but remember to set it back or simtools will not communicate.

Now save the sketch, compile and load it into the Arduino UNO

Assuming you have your pots wired correctly to the Energy360 Board and the UNO board plugged into the USB port

On reboot the Serial Monitor screen should be filling with the raw samples from the POT1

Now physically adjust the POT 1 until it reads 512 on the screen then lock the POT in position in relation to the motor lever – This is the neutral position.

Now adjust the code again for POT 2

```
#define Pot_Tunning 0
```

Change the above to 2 for POT 2

Compile, reload and reboot and complete the position setting to 512 and lock that into the motor position.

Once complete make sure you set all of these setting back to normal, compile and reload

```
#define Pot_Tunning 0  
#define USB_to_Arduino_Baudrate 500000  
#define ENABLE_DEBUG_PRINTS 0
```

You will have to repeat this procedure for the second board for the other two motors

STEP 7 Setting the Compile time options for your RIG

Based on the features of your RIG you can configure the compile time options for the sketch.

Compile Time Options	GAME MODE	Wind Enabled	Cap Bank Enabled	POT #1 Tuning	POT #2 Tuning	Notes
USB_to_Arduino_Baudrate	500000	500000	500000	115200	115200	Baud rate is changed with POT Tuning
Arduino_to_Sabertooth_Baudrate	115200	115200	115200	115200	115200	
AVERAGE_SAMPLE	3	3	3	3	3	Higher accuracy POT reads
CAP_BANK_ENABLE	0	0	1	0	0	Capacitor Bank if present
CAP_BANK_TIMEOUT_MS	5000	5000	5000	5000	5000	Timeout for CapBank to reach charge
CAP_BANK_TARGET_VOLTS	11.8	11.8	11.8	11.8	11.8	Voltage of charge status
WIND_ENABLED	0	1	0	0	0	Wind Function if present
WIND_START_SPEED	3000	3000	3000	3000	3000	Default startup Winds speed 30%
WIND_PWM_FREQ_HZ	25000	25000	25000	25000	25000	For 25kHz 4 wire Fans
MOTOR1_NEUTRAL_POS	512	512	512	512	512	Adjust this after POT Trimming
MOTOR2_NEUTRAL_POS	512	512	512	512	512	Adjust this after POT Trimming
RETURN_NEUTRAL_THRESHOLD	10	10	10	10	10	
RETURN_NEUTRAL_RAMP_MS	3000	3000	3000	3000	3000	
Pot_Tuning	0	0	0	1	2	Outputs POT values to Serial Monitor

Step 8 – SMC3Utils Overview

At this point you have the neutral points set for your position sensors and you have loaded the final sketch based on the features of your Rig. Now its time to program the software limits and PID setting for motion. These setting are saved on the UNO for use during game play.

At this point you should re-read this

[Energy360 - Demistify SMC3Utils V1.pdf](#)

It explains all the initial settings and what you should be looking for.

Lets start with setting the travel of each motor and the safety margins for the travel limits. There are 3 safety stops for each motor and each direction.

- 1) Physical hard limit on your SIM, that piece of metal that you added to stop the motor dead in its tracks at the end of its travel.
- 2) Software Hard Limit “Max Limits” – This is the RED line in SMC3Utils, if the motor hits this limit the firmware shuts down and requires a reboot.
- 3) Software Soft Limit “Clip Limit” This is the orange line and when the motor hits this limit the PID slows it down and pushes it back to ensure the Max Limits are not met. Remember you have to think about momentum of your rig and it will take some tweaking to get it right.

Step 9 – Install SMC3Utils

Download and install SMC3Utils

In the installation directory you will see a file called SMC3Utils.ini open that file and make sure the com port matches the com port of your Arduino Board, if not change it and save.

In the below example my Arduino is on Com Port 8.

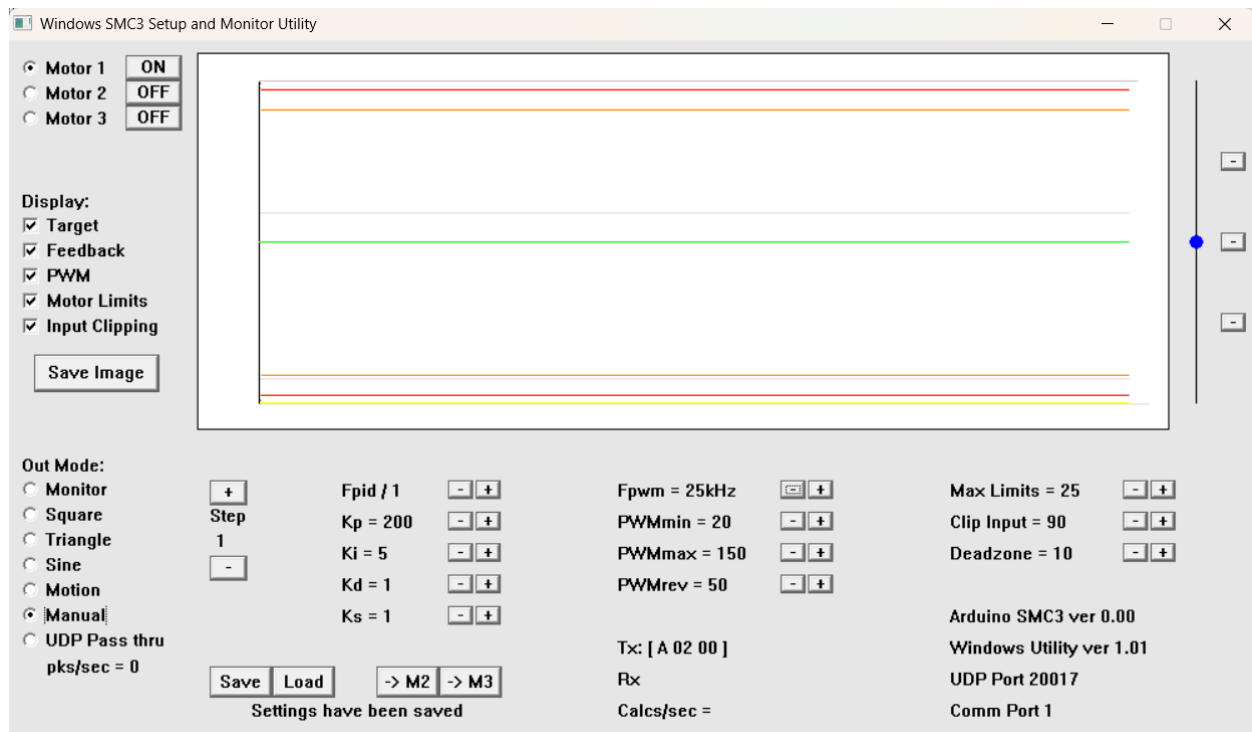
```
# SMC3Utils.ini  
  
UDP_PORT=20017  
COMM_PORT=8  
IMG_PATH=c:\temp
```

The following needs to be done for each motor

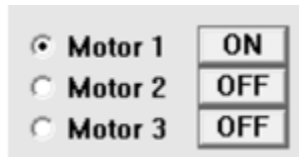
Step 10 – SMC3Utils Setting the Max Limit

Open up SMC3Utils, ensure it is communicating with the Arduino best way is to look at the Arduino SMC3 ver is should say 3.60 (Energy360 code)

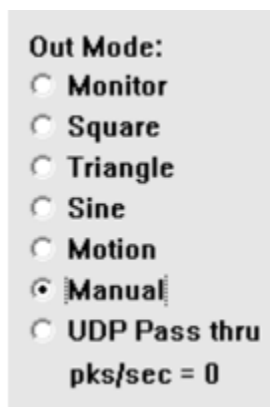
Here are my initial settings



Select Motor 1, turn it ON



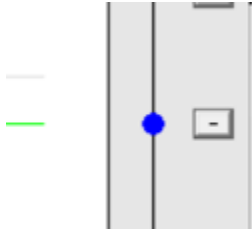
Select Output Mode > Manual



Adjust all the setting to match

Fpid / 1	- +	Fpwm = 25kHz	- +	Max Limits = 25	- +
Kp = 200	- +	PWMmin = 20	- +	Clip Input = 90	- +
Ki = 5	- +	PWMmax = 150	- +	Deadzone = 10	- +
Kd = 1	- +	PWMrev = 50	- +	Arduino SMC3 ver 0.00	
Ks = 1	- +				

Now grab the Blue ball (not your blue balls)



Moving it up will move the motor forward and down reverse. The Motor should be moving

Lets start with the forward direction, move the blue ball up until you get to the maximum travel on your RIG, aka the point where you want it to shut down, this should be just before your physical metal stop. Remember it is symmetrical, so if your forward movement is greater than your reverse movement you need to set it to the smaller one.

Now adjust the “Max Limits” to just above this point, as you adjust this the redline will move. If the redline is hit the motors will shut down and you need to reset them.

Max Limits = 25	- +
-----------------	---

Not the Max Limits will not be able to get bigger than the Clip Limit, In the screen shot above the Clip Limit is 90, so Max Limits will not go above 90. If you need more on the Max Limits first increase the Clip Limit value.

Step 11 – SMC3Utils Setting the Clip Limit

This is the limit where you run into where the PID start to slow down and push back against the motion this is like a protection limit so you don't hit the Max Limit. You will have to play around with this to get what feel you want for extreme limits. Normal game play should be within the two Orange bars inside the clip limits, but extreme movements like crashes may move you up to this limit or ever exceed it.

The gap between the orange line and the red line starts the push back force, the amount of force to slow down the rig is set by

PWMrev = 50	- +
-------------	---

If you want faster pushback increase this number slower pushback lower this number

Setting the max limit and clip limits need to be set for each Motor 1 and Motor 2 – Ignore Motor 3 and then repeated on the other E360 Board.

Step 12 – Playing around

You are now complete with the hardware settings and are all setup now, play around with the movements – You can select various output modes. Follow the explanations in the

[Energy360 - Demistify SMC3Utils V1.pdf](#)

To get a feel for what setting does what.

You can now jump into SIMTOOLS and start with the gameplay setups.

Note any changes to any setting will automatically be saved straight away into the Arduino UNO and will be restored on power up. You can also build various profiles and click Save Image and save them on your computer, this was if you screw around you can always load those setting and start again.

As always, send me any clarification or improvements to this document when you go through it.