Energy360.ino Sketch
Compile-Time Configuration Options

Energy360 uses a small set of compile-time constants to tailor the firmware to your hardware, driver configuration, and desired simulator behavior. These values are set before compilation and define how the control loop, safety systems, wind effects, and communications behave at runtime. Any changes in these values require a code compile and reload from the Arduino IDE Environment.

The table at the bottom of this document summarises these settings.

This document covers **only the Energy360.ino and Energy360 Main board**

Serial Communication

`USB_to_Arduino_Baudrate`

Purpose:
Defines the serial speed between the PC (SimTools / SMC3 / tuning software) and the Arduino.

Why it matters:
This link carries position commands and tuning data. Too low wastes bandwidth; too high can expose USB-serial stability issues on some systems.

Typical values:

- `500000` **`Default:`** `(setting for game mode, SMC3Tools and Simtools`
- `115200 (setting for debug and  POT tuning)`

Notes:

- Must match the baud rate configured in your PC software.
- This does **not** affect Sabertooth communication.

`Arduino_to_Sabertooth_Baudrate`

Purpose:
Defines the serial speed used to send motor commands to the Sabertooth motor driver.

Why it matters:
This directly impacts command latency and reliability. The Sabertooth can handle high baud rates, but the Arduino's ability to generate clean serial data under load is the real constraint.If this value is changed it must also be changed on the Sabertooth.

Typical values:

- `115200` **`Default:`** `(fastest response, highest loading)`
- `57600 (Balanced approach is there are issues in timing)`
- `38400 (maximum stability margin)`

Notes:

- If you see jitter, dropped motion, or erratic motor behavior, reduce this value first.

Control Loop / Sampling

AVERAGE_SAMPLE

Purpose:
Defines how many ADC samples are averaged when reading position feedback.

Why it matters:
Noisey values on the POT reading can accumulate in errors for the motion.

Behavior:

- Lower values → more noisy
- Higher values → More accuracy smoother feedback

Typical values:

- `3 (Default)`
- `4-5 (For noisier environment)`
- `Do NOT Exceed 5 (Effects Cycle Timing)`

Notes:
This is one of the most powerful "feel" tuning knobs after PID gains.

Capacitor Bank / Soft-Start

CAP_BANK_ENABLE

Purpose:
Enables or disables capacitor bank support and controlled charging.

Why it matters:
Large capacitor banks reduce voltage sag and absorb regen energy, but uncontrolled inrush current can damage power supplies or trip breakers on startup. The Energy360 does a control charge to eliminate this

Values:

- `0 → Default: (feature disabled)`
- `1 → (capacitor bank enabled)`

CAP_BANK_TIMEOUT_MS

Purpose:
Maximum time allowed for the capacitor bank to reach the target voltage. Higher Capacitor banks take longer to charge, CAP_BANK_ENABLE must be enabled to work.

Why it matters:
Acts as a safety cutoff if charging fails (bad wiring, dead PSU, faulty caps).

Typical values:

- `5000 Default (provides about 5 Seconds before timeout)`
- `2000–10000 ms`

Behavior:
If the timeout is exceeded, the bypass relay is not closed, this value should be set such that plenty of time is allocated to charging the caps. This will block to main code from running until charge is complete.

CAP_BANK_TARGET_VOLTS

Purpose:
Defines the voltage at which the capacitor bank is considered “ready.”, the Arduino constantly monitors the capacitor bank voltage to ensure it if fully charge before switching it into circuit. CAP_BANK_ENABLE must be enabled to work.

Why it matters:
Ensure the capacitor bank is at a safe level before it is switched into circuit

Typical values:

- `11.8 Default`
- `Slightly below nominal PSU voltage (e.g. 11.5 – 12.0V for a 12V system)`

Notes:
This is a **measured voltage**, from the Arduino ADC.

Wind System (4-Wire PWM Fans)

`WIND_ENABLED`

Purpose:
Globally enables or disables the wind simulation system.

Values:

- `1` → Default: wind control active
- `0` → wind system disabled entirely

Notes:
When disabled, all wind-related code is removed at compile time.

`WIND_START_SPEED`

Purpose:
Sets the fan speed applied immediately at power-up.

Why it matters:
Useful for:

- Keeping airflow during idle
- Used to just cool you without game wind
- Preventing fans from stalling at very low PWM values

Range:

- `300` **`Default:`** `(This represents about 30% fan speed)`
- `0–999 (mapped to PWM output)`

`WIND_PWM_FREQ_HZ`

Purpose:
Defines the PWM frequency used to drive 4-wire fans.

Why it matters:
PC-style PWM fans expect ~25 kHz. Lower frequencies can cause noise, whining, or poor control. If your fans are noisy or whining try adjusting this value

Recommended value:

- `25000` Default: (for typical 25 kHz fans)

Neutral Position & Safety Behavior

`MOTOR1_NEUTRAL_POS`

`MOTOR2_NEUTRAL_POS`

Purpose:
Defines the neutral (center) feedback position for each motor.

Why it matters:
This is the reference point used for:

- Startup alignment
- Return-to-neutral safety behavior
- Centering logic

Typical value:

- `512` **`Default:`** `(this is the midpoint of the ADC on the Arduino)`

Notes:
These should be calibrated to your actual mechanical center, not assumed. Use the POT tuning system to calculate this value and change it here.

`RETURN_NEUTRAL_THRESHOLD`

Purpose:
Defines how close the motor must be to neutral to be considered “at center.”

Why it matters:
Prevents endless micro-adjustments and oscillation around neutral.

Typical values:

- `10` **`Default`**
- `5–15 ADC counts range`

`RETURN_NEUTRAL_RAMP_MS`

Purpose:
Controls how gently the system ramps motors back to neutral.

Why it matters:
A slower ramp:

- Reduces mechanical shock
- Feels safer on large rigs

A faster ramp:

- Clears faults quicker
- Feels more aggressive

Typical values:

- `3000` **`Default:`** `( ~3second)`
- `200–4000 Range`

Calibration / Tuning Modes

`Pot_Tunning`

Purpose:
Enables potentiometer tuning / calibration mode. Each POT is calibrated independently so that means you need to reload the code for each time

Why it matters:
Used during setup to verify feedback direction, range, and stability. You need to be connected to Serial Monitor of the Ardunio IDE and the pot values are outputted in the window. SMC3Utils

and other game software needs to be shut down.
USB_to_Arduino_Baudrate must be set to 115200 and Arduino IDE Serial monitor set to 115200

Typical Values

- `0 **Default:** (- Game mode- Note USB_to_Arduino_Baudrate = 500000)`

`Below settings USB_to_Arduino_Baudrate = 115200`

- `1 (Position sensor #1 raw Values are outputted to IDE serial Monitor)`
- `2 (Position sensor #2 raw Values are outputted to IDE serial Monitor)`

Notes:

- This is a setup-only mode. Will not communicate with SMC while this is set
- Should normally be disabled for regular operation. Set to 0 and SB_to_Arduino_Baudrate set back to 500000

Closing Notes

These compile-time options define **behavioral boundaries**, not moment-to-moment motion.
PID tuning, limits, and live commands operate *within* the framework these settings establish.

Think of them as the simulator's laws of physics.
PID just decides how enthusiastically the system obeys them.

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_Tunning	0	0	0	1	2	Outputs POT values to Serial Monitor