

Axial Flux Permanent Magnet Designer

Generated by Doxygen 1.8.10

Tue Sep 22 2015 20:48:11

Contents

1	Namespace Index	1
1.1	Packages	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
5.1	AxialFluxGeneratorDesigner Namespace Reference	9
6	Class Documentation	11
6.1	AxialFluxGeneratorDesigner.Afpm Class Reference	11
6.1.1	Detailed Description	15
6.1.2	Member Function Documentation	15
6.1.2.1	CalculateBatteryVoltage(double rpmMin, double rpmMax, double minimal↵ PhaseVoltage)	15
6.1.2.2	CalculateCalculateGeneratorOuterRadius(double generatorInnerRadius, double magnetLength)	16
6.1.2.3	CalculateCoilAngle(int coilCount)	16
6.1.2.4	CalculateCoilCrossSectionalArea(double coilWidth, double statorThickness, int coilWindings)	16
6.1.2.5	CalculateCoilInductance(int windingCount, double coilDiameter, double coil↵ Thickness)	16
6.1.2.6	CalculateCoilLegWidth(double maxPhaseCurrent, int coilWindings, double axialThickness)	17
6.1.2.7	CalculateCoilResistance(double coilWireLength, double wireDiameter)	17
6.1.2.8	CalculateCoilWindings(double phaseVoltage, int magnets, double rpm, int coils↵ Phase, double poleFlux, double coilWindingFactor)	17
6.1.2.9	CalculateCoilWireDiameter(double crossSectionalArea)	18

6.1.2.10	CalculateCoilWireLength(int windingCount, double insideCircumference, double outsideCircumference)	18
6.1.2.11	CalculateDCVoltage(double phaseVoltage, double voltageDrop)	18
6.1.2.12	CalculateGeneratorInnerOuterRadiusRatio(double generatorInnerRadius, double generatorOuterRadius)	18
6.1.2.13	CalculateGeneratorInnerRadius(int totalCoils, double coilWidth, int polePairs, double magnetWidth)	19
6.1.2.14	CalculateGridRpm(double phaseVoltageMin, double phaseVoltageMax, int nominalRpm)	19
6.1.2.15	CalculateMagnetFluxDensity(double remanentFluxDensity, double coercive↔FieldStrength, double magnetThickness, double gap)	19
6.1.2.16	CalculateMagnetPoleArcPitch(double magnetWidth, double magnetsDistance)	20
6.1.2.17	CalculateMaximumCurrentDensity(double maxPhaseCurrent, double cross↔SectionalArea)	20
6.1.2.18	CalculateMaximumPhaseCurrent(double generatorNominalPower, double phaseVoltageCutin)	20
6.1.2.19	CalculateMaximumPoleFlux(double fluxDensity, double magnetWidth, double magnetLength)	21
6.1.2.20	CalculatePhaseVoltage(double dcVoltage, double voltageDrop)	22
6.1.2.21	CalculatePolePairs(int coilCount)	22
6.1.2.22	CalculateStatorThickness(double magnetThickness, double gap)	22
6.1.2.23	CalculateTorque(double power, int rpm)	22
6.1.2.24	CalculateTurbineOptimalRotationSpeed(double windSpeed, double tipSpeed↔Ratio, double turbineRotorRadius)	23
6.1.2.25	CalculateTurbineOptimalWindSpeed(double speedRpm, double turbineRotor↔Radius, double tipSpeedRatio)	23
6.1.2.26	CalculateTurbineRotorRadius(double generatorNominalPower, double airDensity, double maximumPowerCoefficient, double windSpeed)	23
6.1.2.27	CalculateWireResistance(double wireLength, double wireDiameter)	24
6.1.2.28	CoilInnerDimensions(double coilLegWidth, int coilCount, double coilGap, double betweenCoilGap, double magnetHeight)	24
6.1.2.29	CoilOuterDimensions(double coilLegWidth, int coilCount, double coilGap, double betweenCoilGap, double magnetHeight)	24
6.1.2.30	UpdateCalculations()	24
6.1.2.31	VoltageDrop(double wireLength, double wireDiameter, double wireCurrent, int phaseType)	25
6.1.3	Member Data Documentation	26
6.1.3.1	MagnetProperties	26
6.1.4	Property Documentation	26
6.1.4.1	CoilCount	26
6.1.4.2	CoilCrossSectionalArea	26
6.1.4.3	CoilFillFactor	26
6.1.4.4	CoilHeatCoefficient	26
6.1.4.5	CoilInductance	26

6.1.4.6	CoilLegWidth	27
6.1.4.7	CoilResistance	27
6.1.4.8	CoilsPerPhase	27
6.1.4.9	CoilThickness	27
6.1.4.10	CoilTurns	27
6.1.4.11	CoilWindingCoefficient	27
6.1.4.12	CoilWireDiameter	27
6.1.4.13	CoilWireLength	27
6.1.4.14	DcVoltageMaxBattery	27
6.1.4.15	DcVoltageMaxGrid	27
6.1.4.16	DcVoltageMinBattery	27
6.1.4.17	DcVoltageMinGrid	28
6.1.4.18	FrontEndTorque	28
6.1.4.19	GeneratorEfficiency	28
6.1.4.20	GeneratorEnergyStorageConnection	28
6.1.4.21	GeneratorFrontEnd	28
6.1.4.22	GeneratorPower	28
6.1.4.23	MagnetCoerciveFieldStrength	28
6.1.4.24	MagnetCount	28
6.1.4.25	MagnetDistance	28
6.1.4.26	MagnetFluxDensity	28
6.1.4.27	MagnetLength	29
6.1.4.28	MagnetPoleArcPitch	29
6.1.4.29	MagnetPoleFlux	29
6.1.4.30	MagnetRemanentFluxDensity	29
6.1.4.31	MagnetThickness	29
6.1.4.32	MagnetWidth	29
6.1.4.33	MaxCurrentDensity	29
6.1.4.34	MaxPhaseCurrent	29
6.1.4.35	MechanicalGap	29
6.1.4.36	OtherRpmMax	29
6.1.4.37	OtherRpmMin	29
6.1.4.38	PhaseCount	29
6.1.4.39	PhaseCurrent	30
6.1.4.40	PhaseVoltageMax	30
6.1.4.41	PhaseVoltageMin	30
6.1.4.42	PhaseWireDiameter	30
6.1.4.43	PhaseWireLength	30
6.1.4.44	PhaseWireResistance	30
6.1.4.45	PhaseWireVoltageDrop	30

6.1.4.46	RectifierWireDiameter	30
6.1.4.47	RectifierWireLength	30
6.1.4.48	RectifierWireResistance	30
6.1.4.49	RectifierWireVoltageDrop	30
6.1.4.50	RotorInnerOuterRadiusRatio	31
6.1.4.51	RotorInnerRadius	31
6.1.4.52	RotorOuterRadius	31
6.1.4.53	RotorThickness	31
6.1.4.54	TurbineAirDensity	31
6.1.4.55	TurbineMaximumPowerCoefficient	31
6.1.4.56	TurbineRotorRadius	31
6.1.4.57	TurbineRpmMax	31
6.1.4.58	TurbineRpmMin	31
6.1.4.59	TurbineSpeedTipRatioMax	31
6.1.4.60	TurbineSpeedTipRatioMin	31
6.1.4.61	TurbineWindspeedMax	32
6.1.4.62	TurbineWindspeedMin	32
6.2	AxialFluxGeneratorDesigner.FormAfpmDesigner Class Reference	32
6.2.1	Detailed Description	32
6.2.2	Constructor & Destructor Documentation	32
6.2.2.1	FormAfpmDesigner()	32
6.2.3	Member Function Documentation	32
6.2.3.1	Dispose(bool disposing)	32
6.2.4	Member Data Documentation	32
6.2.4.1	IsInitialized	32
7	File Documentation	35
7.1	AxialFluxGeneratorDesigner/AfpmCalculations.cs File Reference	35
7.2	AxialFluxGeneratorDesigner/FormAfpmDesigner.cs File Reference	35
7.3	AxialFluxGeneratorDesigner/FormAfpmDesigner.Designer.cs File Reference	35
Index		37

Chapter 1

Namespace Index

1.1 Packages

Here are the packages with brief descriptions (if available):

AxialFluxGeneratorDesigner	9
--	---

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

AxialFluxGeneratorDesigner.Afpm	11
Form	
AxialFluxGeneratorDesigner.FormAfpmDesigner	32

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

AxialFluxGeneratorDesigner.Afpm	
This class can be used to design a Axial Flux Permanent Magnet Generator.	11
AxialFluxGeneratorDesigner.FormAfpmDesigner	
32	

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

AxialFluxGeneratorDesigner/ AfpmCalculations.cs	35
AxialFluxGeneratorDesigner/ FormAfpmDesigner.cs	35
AxialFluxGeneratorDesigner/ FormAfpmDesigner.Designer.cs	35

Chapter 5

Namespace Documentation

5.1 AxialFluxGeneratorDesigner Namespace Reference

Classes

- class [Afpm](#)

This class can be used to design a Axial Flux Permanent Magnet Generator.

- class [FormAfpmDesigner](#)

Chapter 6

Class Documentation

6.1 AxialFluxGeneratorDesigner.Afpm Class Reference

This class can be used to design a Axial Flux Permanent Magnet Generator.

Public Member Functions

- double [CalculateTorque](#) (double power, int rpm)
This method calculates the torque based on the power (W) and RPM
- double [CalculateTurbineRotorRadius](#) (double generatorNominalPower, double airDensity, double maximum↔PowerCoefficient, double windSpeed)
This method calculates the turbine rotor radius to achieve the nominal power.
- int [CalculateTurbineOptimalRotationSpeed](#) (double windSpeed, double tipSpeedRatio, double turbineRotor↔Radius)
This method calculates the turbine RPM.
- int [CalculateGridRpm](#) (double phaseVoltageMin, double phaseVoltageMax, int nominalRpm)
This method calculates the wind speed based on the phase voltage ratio (min/max).
- double [CalculateBatteryVoltage](#) (double rpmMin, double rpmMax, double minimalPhaseVoltage)
This method calculates the maximal battery voltage based on the ratio between the max and min rpm times the minimal phase voltage.
- double [CalculateTurbineOptimalWindSpeed](#) (double speedRpm, double turbineRotorRadius, double tip↔SpeedRatio)
This method calculates the wind speed.
- double [CalculateCoilAngle](#) (int coilCount)
This method calculated the coil angle (Deg).
- double [CalculatePhaseVoltage](#) (double dcVoltage, double voltageDrop)
*This method calculates the phase voltage for a 3 phase Y-configuration from the provided DC voltage. $V_{dc} = ((3 * \sqrt{2}) / \pi) * V_{rms}$ $V_{dc} = 1.35 * V_{rms}$*
- double [CalculateDCVoltage](#) (double phaseVoltage, double voltageDrop)
This method calculates the corrected (for voltage drop due to power lines and diode rectifier) DC voltage.
- double [CalculateCoilInductance](#) (int windingCount, double coilDiameter, double coilThickness)
This method calculates the inductance of a coil (mH). <http://coil32.net/multi-layer-coil.html>
- double [CalculateWireResistance](#) (double wireLength, double wireDiameter)
This method calculates the resistance of a copper wire.
- double [VoltageDrop](#) (double wireLength, double wireDiameter, double wireCurrent, int phaseType)
This method calculates the voltage drop across a three phase power line.
- double [CalculateCoilResistance](#) (double coilWireLength, double wireDiameter)
This method calculates the resistance of a coil (ohm).

- double [CalculateCoilWireLength](#) (int windingCount, double insideCircumference, double outsideCircumference)
This method calculates the wire length of a coil.
- double [CalculateStatorThickness](#) (double magnetThickness, double gap)
This method calculates the thickness of the stator for a dual rotor system.
- double [CalculateMagnetFluxDensity](#) (double remanentFluxDensity, double coerciveFieldStrength, double magnetThickness, double gap)
This method calculates the flux density of a magnet at a certain distance (gap).
- double [CalculateMaximumPoleFlux](#) (double fluxDensity, double magnetWidth, double magnetLength)
This method calculates the magnet flux density for the magnet area.
- int [CalculateCoilWindings](#) (double phaseVoltage, int magnets, double rpm, int coilsPhase, double poleFlux, double coilWindingFactor)
This method calculated the amount of coil windings.
- double [CalculateCoilLegWidth](#) (double maxPhaseCurrent, int coilWindings, double axialThickness)
This method calculates the width of the coil leg.
- double [CalculateCoilCrossSectionalArea](#) (double coilWidth, double statorThickness, int coilWindings)
This method calculates the area of a coil wire.
- double [CalculateMaximumPhaseCurrent](#) (double generatorNominalPower, double phaseVoltageCutin)
This method calculates the maximal phase current.
- double [CalculateMaximumCurrentDensity](#) (double maxPhaseCurrent, double crossSectionalArea)
This method calculates the maximum current density of the coil wire.
- double [CalculateCoilWireDiameter](#) (double crossSectionalArea)
This method calculates the coil wire diameter.
- Tuple< double, double, double, double > [CoilInnerDimensions](#) (double coilLegWidth, int coilCount, double coilGap, double betweenCoilGap, double magnetHeight)
This method calculates the inner coil dimensions
- Tuple< double, double, double, double > [CoilOuterDimensions](#) (double coilLegWidth, int coilCount, double coilGap, double betweenCoilGap, double magnetHeight)
This method calculates the outer coil dimensions
- int [CalculatePolePairs](#) (int coilCount)
This method calculates the amount of magnets.
- double [CalculateMagnetPoleArcPitch](#) (double magnetWidth, double magnetsDistance)
- double [CalculateGeneratorInnerRadius](#) (int totalCoils, double coilWidth, int polePairs, double magnetWidth)
- double [CalculateCalculateGeneratorOuterRadius](#) (double generatorInnerRadius, double magnetLength)
- double [CalculateGeneratorInnerOuterRadiusRatio](#) (double generatorInnerRadius, double generatorOuterRadius)
- void [UpdateCalculations](#) ()
This method can be called to update all calculations.

Public Attributes

- List< Tuple< string, double, double > > [MagnetProperties](#)
This list contains magnet grades with the associated Magnet remanent flux density (T) and the Magnet coercive field strength (A/m).

Properties

- double [TurbineMaximumPowerCoefficient](#) [get, set]
The power coefficient (C_p) is a measure of how efficiently the wind turbine converts the energy in the wind into electricity (usually 35 to 45 %). This value is default set to 0.35 (35%). To find the coefficient of power at a given wind speed, all you have to do is divide the electricity produced by the total energy available in the wind at that speed.
- double [TurbineRotorRadius](#) = 0.35 [get, set]

The turbine rotor radius (R_{turbine}) is the radius of the wind turbine blades (m).

- int [TurbineRpmMax](#) [get, set]
The turbine rpm max is the is the maximum revolutions per minute (rpm) the wind turbine (and thus the generator) shaft will rotate. This value depends on the tip ratio and the wind speed.
- int [TurbineRpmMin](#) [get, set]
The turbine rpm min is the is the minimal revolutions per minute (rpm) the wind turbine (and thus the generator) shaft will rotate. This value depends on the tip ratio and the wind speed.
- double [TurbineSpeedTipRatioMax](#) [get, set]
The speed tip ratio for the maximal rpm (ans so maximal wind speed). The tip-speed ratio, , or TSR for wind turbines is the ratio between the tangential speed of the tip of a blade and the actual velocity of the wind, v . The tip-speed ratio is related to efficiency, with the optimum varying with blade design. Higher tip speeds result in higher noise levels and require stronger blades due to large centrifugal forces.
- double [TurbineSpeedTipRatioMin](#) = 7 [get, set]
The speed tip ratio for the minimal rpm (and so minimal wind speed). The tip-speed ratio, , or TSR for wind turbines is the ratio between the tangential speed of the tip of a blade and the actual velocity of the wind, v . The tip-speed ratio is related to efficiency, with the optimum varying with blade design. Higher tip speeds result in higher noise levels and require stronger blades due to large centrifugal forces.
- double [TurbineWindspeedMax](#) = 8.75 [get, set]
The turbine maximal wind speed (m/s) that the turbine will experience.
- double [TurbineWindspeedMin](#) = 10 [get, set]
The turbine minimal wind speed (m/s) that the turbine will experience.
- double [TurbineAirDensity](#) = 3 [get, set]
The air density (kg/m^3). This value is altitude dependent.
- double [FrontEndTorque](#) = 1.20 [get, set]
The torque of the front end (Nm) at the maximal power and rpm.
- int [OtherRpmMin](#) [get, set]
The minimal revolutions per minute (rpm) of the other front end (e.g. water wheel or Stirling engine).
- int [OtherRpmMax](#) = 300 [get, set]
The maximal revolutions per minute (rpm) of the other front end (e.g. water wheel or Stirling engine).
- double [PhaseWireVoltageDrop](#) = 500 [get, set]
The voltage drop (V) that is caused by the length and diameter of the phase wires from the coil to the diode bridge.
- double [PhaseWireLength](#) [get, set]
The length (m) of a phase wire to the diode bridge rectifier.
- double [PhaseWireDiameter](#) = 0 [get, set]
The diameter (mm) of a phase wire to the diode bridge rectifier.
- double [PhaseWireResistance](#) = 0 [get, set]
The resistance (Ohm) of a phase wire to the diode bridge rectifier.
- double [RectifierWireVoltageDrop](#) = 0 [get, set]
The voltage drop (V) that is caused by the length and diameter of the wires from the diode bridge to the grid inverter/ battery.
- double [RectifierWireLength](#) [get, set]
The length (m) of a wire from the diode bridge to the grid inverter/ battery.
- double [RectifierWireResistance](#) [get, set]
The resistance (Ohm) of a wire from the diode bridge to the grid inverter/ battery.
- double [RectifierWireDiameter](#) [get, set]
The diameter (mm) of a wire from the diode bridge to the grid inverter/ battery.
- double [DcVoltageMinGrid](#) [get, set]
The minimal DC voltage output voltage (V) for a grid connection. This value is default set to 200 volt.
- double [DcVoltageMaxGrid](#) = 200 [get, set]
The maximal DC voltage output voltage (V) for a grid connection. This value is default set to 700 volt.
- double [DcVoltageMinBattery](#) = 700 [get, set]
The minimal DC voltage output voltage (V) for a battery connection. This value is default set to 48 volt.
- double [DcVoltageMaxBattery](#) = 48 [get, set]

The minimal DC voltage output voltage (V) for a battery connection. This value is calculated based on the max / min rpm ratio.

- double **GeneratorPower** [get, set]

The maximum power (W) that the generator has to be capable to produce.

- double **GeneratorEfficiency** = 3000 [get, set]

The efficiency of the generator (%). This value is default set to 90%.

- double **MechanicalGap** = 0.9 [get, set]

The mechanical gap between the coil surface and the magnet surface. This value is default set to 3. Try to reduce this value as much as possible. However, keep in mind that the coils can become warm/hot and expand! This could lead to coils touching the magnets and thus damage.

- int **GeneratorEnergyStorageConnection** = 3 [get, set]

This property determines the type of energy storage that is used. 0 = Battery 1 = grid This property is necessary because depending on the energy storage type different calculations are done

- int **GeneratorFrontEnd** [get, set]

This property determines the front end type that is used to drive the generator.

- double **PhaseVoltageMax** [get, set]

The maximal phase voltage that a sing phase has to produce.

- double **PhaseVoltageMin** [get, set]

The minimal phase voltage that a sing phase has to produce.

- int **PhaseCount** [get]

The phase count of the generator. The phase count is set to 3 and cannot be changed. This because the designer only works with 3-phase generators.

- int **CoilCount** = 3 [get, set]

The coil count is the total amount of coils for the generator. So 5 coils/phase and 3 phases results in a coil count of 5 * 3 = 15

- int **CoilsPerPhase** [get, set]

The coils per phase are the amount of coils in each phase.

- double **CoilCrossSectionalArea** = 5 [get, set]

The cross sectional area of a coil (mm²)

- double **CoilLegWidth** [get, set]

The width of a coil leg (mm)

- double **CoilWindingCoefficient** [get, set]

In power engineering, winding factor is what makes the rms generated voltage in a three-phase AC electrical generator become lesser. This is because the armature winding of each phase is distributed in a number of slots. Since the emf induced in different slots are not in phase, their phasor sum is less than their numerical sum. This reduction factor is called distribution factor Kd. Another factor that can reduce the winding factor is when the slot pitch is smaller than the pole pitch, called pitch factor Kp.

- double **CoilHeatCoefficient** = 0.95 [get, set]

The heat coefficient (W/cm²).

- double **CoilFillFactor** = 0.30 [get, set]

Is the fraction of the core window area that is filled by copper. This value depends mainly on how good the coil is made.

- double **CoilInductance** = 0.55 [get, set]

The inductance of the coil is the ability to store energy in a magnetic field.

- double **CoilResistance** [get, set]

The resistance of the coil (Ohm).

- double **CoilWireLength** [get, set]

The total wire length of a single coil (m).

- int **CoilTurns** [get, set]

The amount of turn per coil.

- double **CoilWireDiameter** [get, set]

The diameter of the coil wire (mm).

- double **CoilThickness** [get, set]

- The thickness of the coil (mm).*

 - double **MaxCurrentDensity** [get, set]

Current Density (A/mm^2) is the measurement of electric current (charge flow in amperes) per unit area of cross-section (m^2).
- double **PhaseCurrent** [get, set]

The phase current is the current that flows through the coil at the maximal rpm.
- double **MaxPhaseCurrent** [get, set]

The phase current is the maximal current (phase current + 10%) that flows through the coil. This can be caused by e.g. a storm or other factors. To prevent failure due to overheating this should be taken into consideration.
- double **MagnetCoerciveFieldStrength** [get, set]

Coercive field strength (H_c) (A/m) describes the force that is necessary to completely demagnetize a magnet. Simply said: the higher this number is, the better a magnet retains its magnetism when exposed to an opposing magnetic field.
- int **MagnetCount** [get, set]

The total amount of magnets on two rotor plates
- double **MagnetDistance** [get, set]

The distance between individual magnets (mm).
- double **MagnetFluxDensity** [get, set]

The magnetic flux density of a magnet is also called "B field" or "magnetic induction". It is measured in tesla (SI unit) or gauss ($10\,000\text{ gauss} = 1\text{ tesla}$). A permanent magnet produces a B field in its core and in its external surroundings.
- double **MagnetLength** [get, set]

The length of a magnet (mm). Default set to 30.
- double **MagnetPoleArcPitch** = 30 [get, set]

??
- double **MagnetPoleFlux** [get, set]

The flux (T) of a magnet to the coil (with mechanical gap included).
- double **MagnetRemanentFluxDensity** [get, set]
- double **MagnetThickness** [get, set]

the magnet thickness (mm).
- double **MagnetWidth** = 10 [get, set]

The magnet width (mm).
- double **RotorInnerOuterRadiusRatio** = 46 [get, set]
- double **RotorInnerRadius** [get, set]
- double **RotorOuterRadius** [get, set]
- double **RotorThickness** [get, set]

6.1.1 Detailed Description

This class can be used to design a Axial Flux Permanent Magnet Generator.

6.1.2 Member Function Documentation

6.1.2.1 double AxialFluxGeneratorDesigner.Afpm.CalculateBatteryVoltage (double rpmMin, double rpmMax, double minimalPhaseVoltage)

This method calculates the maximal battery voltage based on the ratio between the max and min rpm times the minimal phase voltage.

Parameters

<i>rpmMin</i>	
<i>rpmMax</i>	
<i>minimalPhase↔ Voltage</i>	

Returns

The maximal phase voltage

6.1.2.2 `double AxialFluxGeneratorDesigner.Afpm.CalculateCalculateGeneratorOuterRadius (double generatorInnerRadius, double magnetLength)`

Parameters

<i>generatorInner↔ Radius</i>	
<i>magnetLength</i>	

Returns

6.1.2.3 `double AxialFluxGeneratorDesigner.Afpm.CalculateCoilAngle (int coilCount)`

This method calculated the coil angle (Deg).

Parameters

<i>coilCount</i>	The total amount of coils.
------------------	----------------------------

Returns

The angle for each coil (deg).

6.1.2.4 `double AxialFluxGeneratorDesigner.Afpm.CalculateCoilCrossSectionalArea (double coilWidth, double statorThickness, int coilWindings)`

This method calculates the area of a coil wire.

Parameters

<i>coilWidth</i>	The width of the coil (mm)
<i>statorThickness</i>	The stator thickness (mm)
<i>coilWindings</i>	The amount of coil windings (n)

Returns

The area of the coil surface (mm2)

6.1.2.5 `double AxialFluxGeneratorDesigner.Afpm.CalculateCoilInductance (int windingCount, double coilDiameter, double coilThickness)`

This method calculates the inductance of a coil (mH). <http://coil32.net/multi-layer-coil.html>

Parameters

<i>windingCount</i>	The winding count of the coil
<i>coilDiameter</i>	The diameter of the coil (mm)
<i>coilThickness</i>	Thickness of the coil (mm)

Returns

The inductance of the coil (mH)

6.1.2.6 `double AxialFluxGeneratorDesigner.Afpm.CalculateCoilLegWidth (double maxPhaseCurrent, int coilWindings, double axialThickness)`

This method calculates the width of the coil leg.

Parameters

<i>maxPhaseCurrent</i>	The max. phase current (A)
<i>coilWindings</i>	The amount of coil windings (n)
<i>axialThickness</i>	The stator thickness (mm)

Returns

Returns the coil leg width (mm)

6.1.2.7 `double AxialFluxGeneratorDesigner.Afpm.CalculateCoilResistance (double coilWireLength, double wireDiameter)`

This method calculates the resistance of a coil (ohm).

Parameters

<i>coilWireLength</i>	The wire length of the coil (m)
<i>wireDiameter</i>	The diameter of the wire (mm)

Returns

The resistance of the coil (ohm)

6.1.2.8 `int AxialFluxGeneratorDesigner.Afpm.CalculateCoilWindings (double phaseVoltage, int magnets, double rpm, int coilsPhase, double poleFlux, double coilWindingFactor)`

This method calculated the amount of coil windings.

Parameters

<i>phaseVoltage</i>	
<i>magnets</i>	The total amount of magnets
<i>poleFlux</i>	The flux maximum pole flux (T)
<i>rpm</i>	The amount of RPM
<i>coilsPhase</i>	The amount of coils per phase

<i>coilWinding↔ Factor</i>	
--------------------------------	--

Returns

The total amount of coil windings

6.1.2.9 double AxialFluxGeneratorDesigner.Afpm.CalculateCoilWireDiameter (double *crossSectionalArea*)

This method calculates the coil wire diameter.

Parameters

<i>crossSectional↔ Area</i>	The cross sectional area of the coil (mm2)
---------------------------------	--

Returns

The coil wire diameter (mm)

6.1.2.10 double AxialFluxGeneratorDesigner.Afpm.CalculateCoilWireLength (int *windingCount*, double *insideCircumference*, double *outsideCircumference*)

This method calculates the wire length of a coil.

Parameters

<i>windingCount</i>	The winding count of the coil
<i>inside↔ Circumference</i>	The inside circumference of the coil (mm)
<i>outside↔ Circumference</i>	The outside circumference of the coil (mm)

Returns

The wire length of the coil (m)

6.1.2.11 double AxialFluxGeneratorDesigner.Afpm.CalculateDCVoltage (double *phaseVoltage*, double *voltageDrop*)

This method calculates the corrected (for voltage drop due to power lines and diode rectifier) DC voltage.

Parameters

<i>phaseVoltage</i>	The phase voltage (V)
<i>voltageDrop</i>	Drop voltage losses in various power cables (V)

Returns

6.1.2.12 double AxialFluxGeneratorDesigner.Afpm.CalculateGeneratorInnerOuterRadiusRatio (double *generatorInnerRadius*, double *generatorOuterRadius*)

Parameters

<i>generatorInner↔ Radius</i>	
<i>generator↔ OuterRadius</i>	

Returns

6.1.2.13 `double AxialFluxGeneratorDesigner.Afpm.CalculateGeneratorInnerRadius (int totalCoils, double coilWidth, int polePairs, double magnetWidth)`

Parameters

<i>totalCoils</i>	
<i>coilWidth</i>	
<i>polePairs</i>	
<i>magnetWidth</i>	

Returns

6.1.2.14 `int AxialFluxGeneratorDesigner.Afpm.CalculateGridRpm (double phaseVoltageMin, double phaseVoltageMax, int nominalRpm)`

This method calculates the wind speed based on the phase voltage ratio (min/max).

Parameters

<i>phaseVoltage↔ Min</i>	The minimum phase voltage (v)
<i>phaseVoltage↔ Max</i>	The maximum phase voltage (v)
<i>nominalRpm</i>	The nominal RPM of the rotor

Returns

Returns the minimum rpm

6.1.2.15 `double AxialFluxGeneratorDesigner.Afpm.CalculateMagnetFluxDensity (double remanentFluxDensity, double coerciveFieldStrength, double magnetThickness, double gap)`

This method calculates the flux density of a magnet at a certain distance (gap).

Parameters

<i>remanentFlux↔ Density</i>	The remanent density of the magnet (T)
----------------------------------	--

<i>coerciveField↔ Strength</i>	The coercive field strength of the magnet (A/m)(
<i>magnet↔ Thickness</i>	The magnet thickness (mm)
<i>gap</i>	Gap between the magnet surface and the stator (mm).

Returns

Magnet flux density (T)

6.1.2.16 **double** AxialFluxGeneratorDesigner.Afpm.CalculateMagnetPoleArcPitch (**double** *magnetWidth*, **double** *magnetsDistance*)

Parameters

<i>magnetWidth</i>	
<i>magnets↔ Distance</i>	

Returns

6.1.2.17 **double** AxialFluxGeneratorDesigner.Afpm.CalculateMaximumCurrentDensity (**double** *maxPhaseCurrent*, **double** *crossSectionalArea*)

This method calculates the maximum current density of the coil wire.

Parameters

<i>maxPhase↔ Current</i>	The maximum current that can flow trough the coil
<i>crossSectional↔ Area</i>	The cross sectional area (mm2)

Returns

The maximum current density (m2)

6.1.2.18 **double** AxialFluxGeneratorDesigner.Afpm.CalculateMaximumPhaseCurrent (**double** *generatorNominalPower*, **double** *phaseVoltageCutin*)

This method calculates the maximal phase current.

Parameters

<i>generator↔ NominalPower</i>	
<i>phaseVoltage↔ Cutin</i>	

Returns

6.1.2.19 double AxialFluxGeneratorDesigner.Afpm.CalculateMaximumPoleFlux (double *fluxDensity*, double *magnetWidth*, double *magnetLength*)

This method calculates the magnet flux density for the magnet area.

Parameters

<i>fluxDensity</i>	The magnet flux density (T)
<i>magnetWidth</i>	The magnet width (mm)
<i>magnetLength</i>	The magnet length (mm)

Returns

Maximum pole flux (T)

6.1.2.20 double AxialFluxGeneratorDesigner.Afpm.CalculatePhaseVoltage (double *dcVoltage*, double *voltageDrop*)

This method calculates the phase voltage for a 3 phase Y-configuration from the provided DC voltage. $V_{dc} = ((3 \cdot \sqrt{2}) / \pi) \cdot V_{rms}$ $V_{dc} = 1.35 \cdot V_{rms}$

Parameters

<i>dcVoltage</i>	DC voltage (V)
<i>voltageDrop</i>	Drop voltage losses in various power cables (V)

Returns

Phase voltage (rms) (V)

6.1.2.21 int AxialFluxGeneratorDesigner.Afpm.CalculatePolePairs (int *coilCount*)

This method calculates the amount of magnets.

Parameters

<i>coilCount</i>	The total amount of coils
------------------	---------------------------

Returns

The amount of magnets

6.1.2.22 double AxialFluxGeneratorDesigner.Afpm.CalculateStatorThickness (double *magnetThickness*, double *gap*)

This method calculates the thickness of the stator for a dual rotor system.

Parameters

<i>magnetThickness</i>	Thickness of a single magnet (mm)
<i>gap</i>	The mechanical gap between the rotor and the stator (mm)

Returns

Stator thickness (mm)

6.1.2.23 double AxialFluxGeneratorDesigner.Afpm.CalculateTorque (double *power*, int *rpm*)

This method calculates the torque based on the power (W) and RPM

Parameters

<i>power</i>	The power (Watt)
<i>rpm</i>	The rpm

Returns

The torque (Nm)

6.1.2.24 `int AxialFluxGeneratorDesigner.Afpm.CalculateTurbineOptimalRotationSpeed (double windSpeed, double tipSpeedRatio, double turbineRotorRadius)`

This method calculates the turbine RPM.

Parameters

<i>windSpeed</i>	The wind speed (m/s)
<i>tipSpeedRatio</i>	The tip speed ratio
<i>turbineRotor↔ Radius</i>	The radius of the rotor (m)

Returns

The RPM

6.1.2.25 `double AxialFluxGeneratorDesigner.Afpm.CalculateTurbineOptimalWindSpeed (double speedRpm, double turbineRotorRadius, double tipSpeedRatio)`

This method calculates the wind speed.

Parameters

<i>speedRpm</i>	The rotational speed (RPM)
<i>turbineRotor↔ Radius</i>	The radius of the rotor (m)
<i>tipSpeedRatio</i>	The tip speed ratio

Returns

The wind speed (m/s)

6.1.2.26 `double AxialFluxGeneratorDesigner.Afpm.CalculateTurbineRotorRadius (double generatorNominalPower, double airDensity, double maximumPowerCoefficient, double windSpeed)`

This method calculates the turbine rotor radius to achieve the nominal power.

Parameters

<i>generator↔ NominalPower</i>	The maximal generator power (W)
<i>airDensity</i>	The air density (kg/m3)
<i>maximum↔ PowerCoefficient</i>	The generator efficiency (0.9 (90%)) is normal

<i>windSpeed</i>	The maximum wind speed (m/s)
------------------	------------------------------

Returns

The rotor radius (m)

6.1.2.27 `double AxialFluxGeneratorDesigner.Afpm.CalculateWireResistance (double wireLength, double wireDiameter)`

This method calculates the resistance of a copper wire.

Parameters

<i>wireLength</i>	
<i>wireDiameter</i>	

Returns

The resistance of the copper wire (Ohm)

6.1.2.28 `Tuple<double, double, double, double> AxialFluxGeneratorDesigner.Afpm.CoilInnerDimensions (double coilLegWidth, int coilCount, double coilGap, double betweenCoilGap, double magnetHeight)`

This method calculates the inner coil dimensions

Parameters

<i>coilLegWidth</i>	The leg width of the coil (mm)
<i>coilCount</i>	The total amount of coils (for all three phases)
<i>coilGap</i>	The bottom gap between the coil legs (mm)
<i>betweenCoilGap</i>	The gap between two coils (mm)
<i>magnetHeight</i>	The height of the magnet (mm)

Returns

A tuple containing the coil dimensions (mm)

6.1.2.29 `Tuple<double, double, double, double> AxialFluxGeneratorDesigner.Afpm.CoilOuterDimensions (double coilLegWidth, int coilCount, double coilGap, double betweenCoilGap, double magnetHeight)`

This method calculates the outer coil dimensions

Parameters

<i>coilLegWidth</i>	The leg width of the coil (mm)
<i>coilCount</i>	The total amount of coils (for all three phases)
<i>coilGap</i>	The bottom gap between the coil legs (mm)
<i>betweenCoilGap</i>	The gap between two coils (mm)
<i>magnetHeight</i>	The height of the magnet (mm)

Returns

A tuple containing the coil dimensions (mm)

6.1.2.30 `void AxialFluxGeneratorDesigner.Afpm.UpdateCalculations ()`

This method can be called to update all calculations.

6.1.2.31 double AxialFluxGeneratorDesigner.Afpm.VoltageDrop (double *wireLength*, double *wireDiameter*, double *wireCurrent*, int *phaseType*)

This method calculates the voltage drop across a three phase power line.

Parameters

<i>wireLength</i>	The length of a phase wire (m)
<i>wireDiameter</i>	The wire diameter (mm)
<i>wireCurrent</i>	The current flowing through the phase wire (A)
<i>phaseType</i>	The phase type. 1 for AC or DC. 3 for three phase.

Returns

The voltage drop across a single phase wire (V).

6.1.3 Member Data Documentation

6.1.3.1 List<Tuple<string, double, double>> AxialFluxGeneratorDesigner.Afpm.MagnetProperties

Initial value:

```
= new List<Tuple<string, double, double>>
{
    new Tuple<string, double, double>("N30", 1.1, 808.0),
    new Tuple<string, double, double>("N33", 1.155, 848.0),
    new Tuple<string, double, double>("N35", 1.19, 887.5),
    new Tuple<string, double, double>("N38", 1.24, 887.5),
    new Tuple<string, double, double>("N40", 1.275, 927.5),
    new Tuple<string, double, double>("N42", 1.305, 927.5),
    new Tuple<string, double, double>("N45", 1.345, 927.5),
    new Tuple<string, double, double>("N48", 1.395, 927.5),
    new Tuple<string, double, double>("N50", 1.430, 927.5),
    new Tuple<string, double, double>("N52", 1.445, 927.5)
}
```

This list contains magnet grades with the associated Magnet remanent flux density (T) and the Magnet coercive field strength (A/m).

6.1.4 Property Documentation

6.1.4.1 int AxialFluxGeneratorDesigner.Afpm.CoilCount = 3 [get], [set]

The coil count is the total amount of coils for the generator. So 5 coils/phase and 3 phases results in a coil count of $5 * 3 = 15$

6.1.4.2 double AxialFluxGeneratorDesigner.Afpm.CoilCrossSectionalArea = 5 [get], [set]

The cross sectional area of a coil (mm²)

6.1.4.3 double AxialFluxGeneratorDesigner.Afpm.CoilFillFactor = 0.30 [get], [set]

Is the fraction of the core window area that is filled by copper. This value depends mainly on how good the coil is made.

6.1.4.4 double AxialFluxGeneratorDesigner.Afpm.CoilHeatCoefficient = 0.95 [get], [set]

The heat coefficient (W/cm²).

6.1.4.5 double AxialFluxGeneratorDesigner.Afpm.CoilInductance = 0.55 [get], [set]

The inductance of the coil is the ability to store energy in a magnetic field.

6.1.4.6 double AxialFluxGeneratorDesigner.Afpm.CoilLegWidth [get], [set]

The width of a coil leg (mm)

6.1.4.7 double AxialFluxGeneratorDesigner.Afpm.CoilResistance [get], [set]

The resistance of the coil (Ohm).

6.1.4.8 int AxialFluxGeneratorDesigner.Afpm.CoilsPerPhase [get], [set]

The coils per phase are the amount of coils in each phase.

6.1.4.9 double AxialFluxGeneratorDesigner.Afpm.CoilThickness [get], [set]

The thickness of the coil (mm).

6.1.4.10 int AxialFluxGeneratorDesigner.Afpm.CoilTurns [get], [set]

The amount of turn per coil.

6.1.4.11 double AxialFluxGeneratorDesigner.Afpm.CoilWindingCoefficient [get], [set]

In power engineering, winding factor is what makes the rms generated voltage in a three-phase AC electrical generator become lesser. This is because the armature winding of each phase is distributed in a number of slots. Since the emf induced in different slots are not in phase, their phasor sum is less than their numerical sum. This reduction factor is called distribution factor K_d . Another factor that can reduce the winding factor is when the slot pitch is smaller than the pole pitch, called pitch factor K_p .

The winding factor can be calculated as $K_w = K_d * K_p$.

Most of the three-phase machines have winding factor values between 0.85 and 0.95.

6.1.4.12 double AxialFluxGeneratorDesigner.Afpm.CoilWireDiameter [get], [set]

The diameter of the coil wire (mm).

6.1.4.13 double AxialFluxGeneratorDesigner.Afpm.CoilWireLength [get], [set]

The total wire length of a single coil (m).

6.1.4.14 double AxialFluxGeneratorDesigner.Afpm.DcVoltageMaxBattery = 48 [get], [set]

The minimal DC voltage output voltage (V) for a battery connection. This value is calculated based on the max / min rpm ratio.

6.1.4.15 double AxialFluxGeneratorDesigner.Afpm.DcVoltageMaxGrid = 200 [get], [set]

The maximal DC voltage output voltage (V) for a grid connection. This value is default set to 700 volt.

6.1.4.16 double AxialFluxGeneratorDesigner.Afpm.DcVoltageMinBattery = 700 [get], [set]

The minimal DC voltage output voltage (V) for a battery connection. This value is default set to 48 volt.

6.1.4.17 `double AxialFluxGeneratorDesigner.Afpm.DcVoltageMinGrid` [get], [set]

The minimal DC voltage output voltage (V) for a grid connection. This value is default set to 200 volt.

6.1.4.18 `double AxialFluxGeneratorDesigner.Afpm.FrontEndTorque = 1.20` [get], [set]

The torque of the front end (Nm) at the maximal power and rpm.

6.1.4.19 `double AxialFluxGeneratorDesigner.Afpm.GeneratorEfficiency = 3000` [get], [set]

The efficiency of the generator (%). This value is default set to 90%.

6.1.4.20 `int AxialFluxGeneratorDesigner.Afpm.GeneratorEnergyStorageConnection = 3` [get], [set]

This property determines the type of energy storage that is used. 0 = Battery 1 = grid This property is necessary because depending on the energy storage type different calculations are done

6.1.4.21 `int AxialFluxGeneratorDesigner.Afpm.GeneratorFrontEnd` [get], [set]

This property determines the front end type that is used to drive the generator.

0 = Wind turbine 1 = Other

This property is necessary because depending on the front end type different calculations are done.

6.1.4.22 `double AxialFluxGeneratorDesigner.Afpm.GeneratorPower` [get], [set]

The maximum power (W) that the generator has to be capable to produce.

6.1.4.23 `double AxialFluxGeneratorDesigner.Afpm.MagnetCoerciveFieldStrength` [get], [set]

Coercive field strength (Hc) (A/m) describes the force that is necessary to completely demagnetize a magnet. Simply said: the higher this number is, the better a magnet retains its magnetism when exposed to an opposing magnetic field.

6.1.4.24 `int AxialFluxGeneratorDesigner.Afpm.MagnetCount` [get], [set]

The total amount of magnets on two rotor plates

6.1.4.25 `double AxialFluxGeneratorDesigner.Afpm.MagnetDistance` [get], [set]

The distance between individual magnets (mm).

6.1.4.26 `double AxialFluxGeneratorDesigner.Afpm.MagnetFluxDensity` [get], [set]

The magnetic flux density of a magnet is also called "B field" or "magnetic induction". It is measured in tesla (SI unit) or gauss (10 000 gauss = 1 tesla). A permanent magnet produces a B field in its core and in its external surroundings.

A B field strength with a direction can be attributed to each point within and outside of the magnet. If you position a small compass needle in the B field of a magnet, it orients itself toward the field direction. The justifying force is proportional to the strength of the B field.

6.1.4.27 `double AxialFluxGeneratorDesigner.Afpm.MagnetLength` [get], [set]

The length of a magnet (mm). Default set to 30.

6.1.4.28 `double AxialFluxGeneratorDesigner.Afpm.MagnetPoleArcPitch = 30` [get], [set]

??

6.1.4.29 `double AxialFluxGeneratorDesigner.Afpm.MagnetPoleFlux` [get], [set]

The flux (T) of a magnet to the coil (with mechanical gap included).

6.1.4.30 `double AxialFluxGeneratorDesigner.Afpm.MagnetRemanentFluxDensity` [get], [set]

6.1.4.31 `double AxialFluxGeneratorDesigner.Afpm.MagnetThickness` [get], [set]

the magnet thickness (mm).

6.1.4.32 `double AxialFluxGeneratorDesigner.Afpm.MagnetWidth = 10` [get], [set]

The magnet width (mm).

6.1.4.33 `double AxialFluxGeneratorDesigner.Afpm.MaxCurrentDensity` [get], [set]

Current Density (A/mm²) is the measurement of electric current (charge flow in amperes) per unit area of cross-section (m²).

6.1.4.34 `double AxialFluxGeneratorDesigner.Afpm.MaxPhaseCurrent` [get], [set]

The phase current is the maximal current (phase current + 10%) that flows through the coil. This can be caused by e.g. a storm or other factors. To prevent failure due to overheating this should be taken into consideration.

6.1.4.35 `double AxialFluxGeneratorDesigner.Afpm.MechanicalGap = 0.9` [get], [set]

The mechanical gap between the coil surface and the magnet surface. This value is default set to 3. Try to reduce this value as much as possible. However, keep in mind that the coils can become warm/hot and expand! This could lead to coils touching the magnets and thus damage.

6.1.4.36 `int AxialFluxGeneratorDesigner.Afpm.OtherRpmMax = 300` [get], [set]

The maximal revolutions per minute (rpm) of the other front end (e.g. water wheel or Stirling engine).

6.1.4.37 `int AxialFluxGeneratorDesigner.Afpm.OtherRpmMin` [get], [set]

The minimal revolutions per minute (rpm) of the other front end (e.g. water wheel or Stirling engine).

6.1.4.38 `int AxialFluxGeneratorDesigner.Afpm.PhaseCount` [get]

The phase count of the generator. The phase count is set to 3 and cannot be changed. This because the designer only works with 3-phase generators.

6.1.4.39 double `AxialFluxGeneratorDesigner.Afpm.PhaseCurrent` [get], [set]

The phase current is the current that flows through the coil at the maximal rpm.

6.1.4.40 double `AxialFluxGeneratorDesigner.Afpm.PhaseVoltageMax` [get], [set]

The maximal phase voltage that a sing phase has to produce.

6.1.4.41 double `AxialFluxGeneratorDesigner.Afpm.PhaseVoltageMin` [get], [set]

The minimal phase voltage that a sing phase has to produce.

6.1.4.42 double `AxialFluxGeneratorDesigner.Afpm.PhaseWireDiameter = 0` [get], [set]

The diameter (mm) of a phase wire to the diode bridge rectifier.

6.1.4.43 double `AxialFluxGeneratorDesigner.Afpm.PhaseWireLength` [get], [set]

The length (m) of a phase wire to the diode bridge rectifier.

6.1.4.44 double `AxialFluxGeneratorDesigner.Afpm.PhaseWireResistance = 0` [get], [set]

The resistance (Ohm) of a phase wire to the diode bridge rectifier.

6.1.4.45 double `AxialFluxGeneratorDesigner.Afpm.PhaseWireVoltageDrop = 500` [get], [set]

The voltage drop (V) that is caused by the length and diameter of the phase wires from the coil to the diode bridge.

6.1.4.46 double `AxialFluxGeneratorDesigner.Afpm.RectifierWireDiameter` [get], [set]

The diameter (mm) of a wire from the diode bridge to the grid inverter/ battery.

6.1.4.47 double `AxialFluxGeneratorDesigner.Afpm.RectifierWireLength` [get], [set]

The length (m) of a wire from the diode bridge to the grid inverter/ battery.

6.1.4.48 double `AxialFluxGeneratorDesigner.Afpm.RectifierWireResistance` [get], [set]

The resistance (Ohm) of a wire from the diode bridge to the grid inverter/ battery.

6.1.4.49 double `AxialFluxGeneratorDesigner.Afpm.RectifierWireVoltageDrop = 0` [get], [set]

The voltage drop (V) that is caused by the length and diameter of the wires from the diode bridge to the grid inverter/ battery.

6.1.4.50 **double AxialFluxGeneratorDesigner.Afpm.RotorInnerOuterRadiusRatio = 46** [get], [set]

6.1.4.51 **double AxialFluxGeneratorDesigner.Afpm.RotorInnerRadius** [get], [set]

6.1.4.52 **double AxialFluxGeneratorDesigner.Afpm.RotorOuterRadius** [get], [set]

6.1.4.53 **double AxialFluxGeneratorDesigner.Afpm.RotorThickness** [get], [set]

6.1.4.54 **double AxialFluxGeneratorDesigner.Afpm.TurbineAirDensity = 3** [get], [set]

The air density (kg/m^3). This value is altitude dependent.

6.1.4.55 **double AxialFluxGeneratorDesigner.Afpm.TurbineMaximumPowerCoefficient** [get], [set]

The power coefficient (C_p) is a measure of how efficiently the wind turbine converts the energy in the wind into electricity (usually 35 to 45 %). This value is default set to 0.35 (35%). To find the coefficient of power at a given wind speed, all you have to do is divide the electricity produced by the total energy available in the wind at that speed.

Wind turbines extract energy by slowing down the wind. For a wind turbine to be 100% efficient it would need to stop 100% of the wind - but then the rotor would have to be a solid disk and it would not turn and no kinetic energy would be converted. On the other extreme, if you had a wind turbine with just one rotor blade, most of the wind passing through the area swept by the turbine blade would miss the blade completely and so the kinetic energy would be kept by the wind.

6.1.4.56 **double AxialFluxGeneratorDesigner.Afpm.TurbineRotorRadius = 0.35** [get], [set]

The turbine rotor radius (R_{turbine}) is the radius of the wind turbine blades (m).

6.1.4.57 **int AxialFluxGeneratorDesigner.Afpm.TurbineRpmMax** [get], [set]

The turbine rpm max is the is the maximum revolutions per minute (rpm) the wind turbine (and thus the generator) shaft will rotate. This value depends on the tip ratio and the wind speed.

6.1.4.58 **int AxialFluxGeneratorDesigner.Afpm.TurbineRpmMin** [get], [set]

The turbine rpm min is the is the minimal revolutions per minute (rpm) the wind turbine (and thus the generator) shaft will rotate. This value depends on the tip ratio and the wind speed.

6.1.4.59 **double AxialFluxGeneratorDesigner.Afpm.TurbineSpeedTipRatioMax** [get], [set]

The speed tip ratio for the maximal rpm (and so maximal wind speed). The tip-speed ratio, or TSR for wind turbines is the ratio between the tangential speed of the tip of a blade and the actual velocity of the wind, v . The tip-speed ratio is related to efficiency, with the optimum varying with blade design. Higher tip speeds result in higher noise levels and require stronger blades due to large centrifugal forces.

6.1.4.60 **double AxialFluxGeneratorDesigner.Afpm.TurbineSpeedTipRatioMin = 7** [get], [set]

The speed tip ratio for the minimal rpm (and so minimal wind speed). The tip-speed ratio, or TSR for wind turbines is the ratio between the tangential speed of the tip of a blade and the actual velocity of the wind, v . The tip-speed ratio is related to efficiency, with the optimum varying with blade design. Higher tip speeds result in higher noise levels and require stronger blades due to large centrifugal forces.

6.1.4.61 `double AxialFluxGeneratorDesigner.Afpm.TurbineWindspeedMax = 8.75` `[get]`, `[set]`

The turbine maximal wind speed (m/s) that the turbine will experience.

6.1.4.62 `double AxialFluxGeneratorDesigner.Afpm.TurbineWindspeedMin = 10` `[get]`, `[set]`

The turbine minimal wind speed (m/s) that the turbine will experience.

The documentation for this class was generated from the following file:

- [AxialFluxGeneratorDesigner/AfpmCalculations.cs](#)

6.2 AxialFluxGeneratorDesigner.FormAfpmDesigner Class Reference

Inherits `Form`.

Public Member Functions

- [FormAfpmDesigner](#) ()

Public Attributes

- `bool IsInitialized` = false

Protected Member Functions

- override void [Dispose](#) (bool disposing)
Clean up any resources being used.

6.2.1 Detailed Description

6.2.2 Constructor & Destructor Documentation

6.2.2.1 `AxialFluxGeneratorDesigner.FormAfpmDesigner.FormAfpmDesigner ()`

6.2.3 Member Function Documentation

6.2.3.1 `override void AxialFluxGeneratorDesigner.FormAfpmDesigner.Dispose (bool disposing)` `[protected]`

Clean up any resources being used.

Parameters

<i>disposing</i>	true if managed resources should be disposed; otherwise, false.
------------------	---

6.2.4 Member Data Documentation

6.2.4.1 `bool AxialFluxGeneratorDesigner.FormAfpmDesigner.IsInitialized` = false

The documentation for this class was generated from the following files:

- [AxialFluxGeneratorDesigner/FormAfpmDesigner.cs](#)

- AxialFluxGeneratorDesigner/[FormAfpDesigner.Designer.cs](#)

Chapter 7

File Documentation

7.1 AxialFluxGeneratorDesigner/AfpmCalculations.cs File Reference

Classes

- class [AxialFluxGeneratorDesigner.Afpm](#)
This class can be used to design a Axial Flux Permanent Magnet Generator.

Namespaces

- namespace [AxialFluxGeneratorDesigner](#)

7.2 AxialFluxGeneratorDesigner/FormAfpmDesigner.cs File Reference

Classes

- class [AxialFluxGeneratorDesigner.FormAfpmDesigner](#)

Namespaces

- namespace [AxialFluxGeneratorDesigner](#)

7.3 AxialFluxGeneratorDesigner/FormAfpmDesigner.Designer.cs File Reference

Classes

- class [AxialFluxGeneratorDesigner.FormAfpmDesigner](#)

Namespaces

- namespace [AxialFluxGeneratorDesigner](#)

Index

AxialFluxGeneratorDesigner, [9](#)
AxialFluxGeneratorDesigner.Afpm, [11](#)
AxialFluxGeneratorDesigner.FormAfpmDesigner, [32](#)
AxialFluxGeneratorDesigner/AfpmCalculations.cs, [35](#)
AxialFluxGeneratorDesigner/FormAfpmDesigner.cs, [35](#)
AxialFluxGeneratorDesigner/FormAfpmDesigner.↔
Designer.cs, [35](#)
AxialFluxGeneratorDesigner::Afpm
 CalculateBatteryVoltage, [15](#)
 CalculateCalculateGeneratorOuterRadius, [16](#)
 CalculateCoilAngle, [16](#)
 CalculateCoilCrossSectionalArea, [16](#)
 CalculateCoilInductance, [16](#)
 CalculateCoilLegWidth, [17](#)
 CalculateCoilResistance, [17](#)
 CalculateCoilWindings, [17](#)
 CalculateCoilWireDiameter, [18](#)
 CalculateCoilWireLength, [18](#)
 CalculateDCVoltage, [18](#)
 CalculateGeneratorInnerOuterRadiusRatio, [18](#)
 CalculateGeneratorInnerRadius, [19](#)
 CalculateGridRpm, [19](#)
 CalculateMagnetFluxDensity, [19](#)
 CalculateMagnetPoleArcPitch, [20](#)
 CalculateMaximumCurrentDensity, [20](#)
 CalculateMaximumPhaseCurrent, [20](#)
 CalculateMaximumPoleFlux, [20](#)
 CalculatePhaseVoltage, [22](#)
 CalculatePolePairs, [22](#)
 CalculateStatorThickness, [22](#)
 CalculateTorque, [22](#)
 CalculateTurbineOptimalRotationSpeed, [23](#)
 CalculateTurbineOptimalWindSpeed, [23](#)
 CalculateTurbineRotorRadius, [23](#)
 CalculateWireResistance, [24](#)
 CoilCount, [26](#)
 CoilCrossSectionalArea, [26](#)
 CoilFillFactor, [26](#)
 CoilHeatCoefficient, [26](#)
 CoilInductance, [26](#)
 CoilInnerDimensions, [24](#)
 CoilLegWidth, [26](#)
 CoilOuterDimensions, [24](#)
 CoilResistance, [27](#)
 CoilThickness, [27](#)
 CoilTurns, [27](#)
 CoilWindingCoefficient, [27](#)
 CoilWireDiameter, [27](#)
 CoilWireLength, [27](#)
 CoilsPerPhase, [27](#)
 DcVoltageMaxBattery, [27](#)
 DcVoltageMaxGrid, [27](#)
 DcVoltageMinBattery, [27](#)
 DcVoltageMinGrid, [27](#)
 FrontEndTorque, [28](#)
 GeneratorEfficiency, [28](#)
 GeneratorEnergyStorageConnection, [28](#)
 GeneratorFrontEnd, [28](#)
 GeneratorPower, [28](#)
 MagnetCoerciveFieldStrength, [28](#)
 MagnetCount, [28](#)
 MagnetDistance, [28](#)
 MagnetFluxDensity, [28](#)
 MagnetLength, [28](#)
 MagnetPoleArcPitch, [29](#)
 MagnetPoleFlux, [29](#)
 MagnetProperties, [26](#)
 MagnetRemanentFluxDensity, [29](#)
 MagnetThickness, [29](#)
 MagnetWidth, [29](#)
 MaxCurrentDensity, [29](#)
 MaxPhaseCurrent, [29](#)
 MechanicalGap, [29](#)
 OtherRpmMax, [29](#)
 OtherRpmMin, [29](#)
 PhaseCount, [29](#)
 PhaseCurrent, [29](#)
 PhaseVoltageMax, [30](#)
 PhaseVoltageMin, [30](#)
 PhaseWireDiameter, [30](#)
 PhaseWireLength, [30](#)
 PhaseWireResistance, [30](#)
 PhaseWireVoltageDrop, [30](#)
 RectifierWireDiameter, [30](#)
 RectifierWireLength, [30](#)
 RectifierWireResistance, [30](#)
 RectifierWireVoltageDrop, [30](#)
 RotorInnerOuterRadiusRatio, [30](#)
 RotorInnerRadius, [31](#)
 RotorOuterRadius, [31](#)
 RotorThickness, [31](#)
 TurbineAirDensity, [31](#)
 TurbineMaximumPowerCoefficient, [31](#)
 TurbineRotorRadius, [31](#)
 TurbineRpmMax, [31](#)
 TurbineRpmMin, [31](#)
 TurbineSpeedTipRatioMax, [31](#)
 TurbineSpeedTipRatioMin, [31](#)

- TurbineWindspeedMax, [31](#)
- TurbineWindspeedMin, [32](#)
- UpdateCalculations, [24](#)
- VoltageDrop, [24](#)
- AxialFluxGeneratorDesigner::FormAfpDesigner
 - Dispose, [32](#)
 - FormAfpDesigner, [32](#)
 - IsInitialized, [32](#)
- CalculateBatteryVoltage
 - AxialFluxGeneratorDesigner::Afpm, [15](#)
- CalculateCalculateGeneratorOuterRadius
 - AxialFluxGeneratorDesigner::Afpm, [16](#)
- CalculateCoilAngle
 - AxialFluxGeneratorDesigner::Afpm, [16](#)
- CalculateCoilCrossSectionalArea
 - AxialFluxGeneratorDesigner::Afpm, [16](#)
- CalculateCoilInductance
 - AxialFluxGeneratorDesigner::Afpm, [16](#)
- CalculateCoilLegWidth
 - AxialFluxGeneratorDesigner::Afpm, [17](#)
- CalculateCoilResistance
 - AxialFluxGeneratorDesigner::Afpm, [17](#)
- CalculateCoilWindings
 - AxialFluxGeneratorDesigner::Afpm, [17](#)
- CalculateCoilWireDiameter
 - AxialFluxGeneratorDesigner::Afpm, [18](#)
- CalculateCoilWireLength
 - AxialFluxGeneratorDesigner::Afpm, [18](#)
- CalculateDCVoltage
 - AxialFluxGeneratorDesigner::Afpm, [18](#)
- CalculateGeneratorInnerOuterRadiusRatio
 - AxialFluxGeneratorDesigner::Afpm, [18](#)
- CalculateGeneratorInnerRadius
 - AxialFluxGeneratorDesigner::Afpm, [19](#)
- CalculateGridRpm
 - AxialFluxGeneratorDesigner::Afpm, [19](#)
- CalculateMagnetFluxDensity
 - AxialFluxGeneratorDesigner::Afpm, [19](#)
- CalculateMagnetPoleArcPitch
 - AxialFluxGeneratorDesigner::Afpm, [20](#)
- CalculateMaximumCurrentDensity
 - AxialFluxGeneratorDesigner::Afpm, [20](#)
- CalculateMaximumPhaseCurrent
 - AxialFluxGeneratorDesigner::Afpm, [20](#)
- CalculateMaximumPoleFlux
 - AxialFluxGeneratorDesigner::Afpm, [20](#)
- CalculatePhaseVoltage
 - AxialFluxGeneratorDesigner::Afpm, [22](#)
- CalculatePolePairs
 - AxialFluxGeneratorDesigner::Afpm, [22](#)
- CalculateStatorThickness
 - AxialFluxGeneratorDesigner::Afpm, [22](#)
- CalculateTorque
 - AxialFluxGeneratorDesigner::Afpm, [22](#)
- CalculateTurbineOptimalRotationSpeed
 - AxialFluxGeneratorDesigner::Afpm, [23](#)
- CalculateTurbineOptimalWindSpeed
 - AxialFluxGeneratorDesigner::Afpm, [23](#)
- CalculateTurbineRotorRadius
 - AxialFluxGeneratorDesigner::Afpm, [23](#)
- CalculateWireResistance
 - AxialFluxGeneratorDesigner::Afpm, [24](#)
- CoilCount
 - AxialFluxGeneratorDesigner::Afpm, [26](#)
- CoilCrossSectionalArea
 - AxialFluxGeneratorDesigner::Afpm, [26](#)
- CoilFillFactor
 - AxialFluxGeneratorDesigner::Afpm, [26](#)
- CoilHeatCoefficient
 - AxialFluxGeneratorDesigner::Afpm, [26](#)
- CoilInductance
 - AxialFluxGeneratorDesigner::Afpm, [26](#)
- CoilInnerDimensions
 - AxialFluxGeneratorDesigner::Afpm, [24](#)
- CoilLegWidth
 - AxialFluxGeneratorDesigner::Afpm, [26](#)
- CoilOuterDimensions
 - AxialFluxGeneratorDesigner::Afpm, [24](#)
- CoilResistance
 - AxialFluxGeneratorDesigner::Afpm, [27](#)
- CoilThickness
 - AxialFluxGeneratorDesigner::Afpm, [27](#)
- CoilTurns
 - AxialFluxGeneratorDesigner::Afpm, [27](#)
- CoilWindingCoefficient
 - AxialFluxGeneratorDesigner::Afpm, [27](#)
- CoilWireDiameter
 - AxialFluxGeneratorDesigner::Afpm, [27](#)
- CoilWireLength
 - AxialFluxGeneratorDesigner::Afpm, [27](#)
- CoilsPerPhase
 - AxialFluxGeneratorDesigner::Afpm, [27](#)
- DcVoltageMaxBattery
 - AxialFluxGeneratorDesigner::Afpm, [27](#)
- DcVoltageMaxGrid
 - AxialFluxGeneratorDesigner::Afpm, [27](#)
- DcVoltageMinBattery
 - AxialFluxGeneratorDesigner::Afpm, [27](#)
- DcVoltageMinGrid
 - AxialFluxGeneratorDesigner::Afpm, [27](#)
- Dispose
 - AxialFluxGeneratorDesigner::FormAfpDesigner, [32](#)
- FormAfpDesigner
 - AxialFluxGeneratorDesigner::FormAfpDesigner, [32](#)
- FrontEndTorque
 - AxialFluxGeneratorDesigner::Afpm, [28](#)
- GeneratorEfficiency
 - AxialFluxGeneratorDesigner::Afpm, [28](#)
- GeneratorEnergyStorageConnection
 - AxialFluxGeneratorDesigner::Afpm, [28](#)
- GeneratorFrontEnd
 - AxialFluxGeneratorDesigner::Afpm, [28](#)

- GeneratorPower
 - AxialFluxGeneratorDesigner::Afpm, 28
- IsInitialized
 - AxialFluxGeneratorDesigner::FormAfpmDesigner, 32
- MagnetCoerciveFieldStrength
 - AxialFluxGeneratorDesigner::Afpm, 28
- MagnetCount
 - AxialFluxGeneratorDesigner::Afpm, 28
- MagnetDistance
 - AxialFluxGeneratorDesigner::Afpm, 28
- MagnetFluxDensity
 - AxialFluxGeneratorDesigner::Afpm, 28
- MagnetLength
 - AxialFluxGeneratorDesigner::Afpm, 28
- MagnetPoleArcPitch
 - AxialFluxGeneratorDesigner::Afpm, 29
- MagnetPoleFlux
 - AxialFluxGeneratorDesigner::Afpm, 29
- MagnetProperties
 - AxialFluxGeneratorDesigner::Afpm, 26
- MagnetRemanentFluxDensity
 - AxialFluxGeneratorDesigner::Afpm, 29
- MagnetThickness
 - AxialFluxGeneratorDesigner::Afpm, 29
- MagnetWidth
 - AxialFluxGeneratorDesigner::Afpm, 29
- MaxCurrentDensity
 - AxialFluxGeneratorDesigner::Afpm, 29
- MaxPhaseCurrent
 - AxialFluxGeneratorDesigner::Afpm, 29
- MechanicalGap
 - AxialFluxGeneratorDesigner::Afpm, 29
- OtherRpmMax
 - AxialFluxGeneratorDesigner::Afpm, 29
- OtherRpmMin
 - AxialFluxGeneratorDesigner::Afpm, 29
- PhaseCount
 - AxialFluxGeneratorDesigner::Afpm, 29
- PhaseCurrent
 - AxialFluxGeneratorDesigner::Afpm, 29
- PhaseVoltageMax
 - AxialFluxGeneratorDesigner::Afpm, 30
- PhaseVoltageMin
 - AxialFluxGeneratorDesigner::Afpm, 30
- PhaseWireDiameter
 - AxialFluxGeneratorDesigner::Afpm, 30
- PhaseWireLength
 - AxialFluxGeneratorDesigner::Afpm, 30
- PhaseWireResistance
 - AxialFluxGeneratorDesigner::Afpm, 30
- PhaseWireVoltageDrop
 - AxialFluxGeneratorDesigner::Afpm, 30
- RectifierWireDiameter
 - AxialFluxGeneratorDesigner::Afpm, 30
- RectifierWireLength
 - AxialFluxGeneratorDesigner::Afpm, 30
- RectifierWireResistance
 - AxialFluxGeneratorDesigner::Afpm, 30
- RectifierWireVoltageDrop
 - AxialFluxGeneratorDesigner::Afpm, 30
- RotorInnerOuterRadiusRatio
 - AxialFluxGeneratorDesigner::Afpm, 30
- RotorInnerRadius
 - AxialFluxGeneratorDesigner::Afpm, 31
- RotorOuterRadius
 - AxialFluxGeneratorDesigner::Afpm, 31
- RotorThickness
 - AxialFluxGeneratorDesigner::Afpm, 31
- TurbineAirDensity
 - AxialFluxGeneratorDesigner::Afpm, 31
- TurbineMaximumPowerCoefficient
 - AxialFluxGeneratorDesigner::Afpm, 31
- TurbineRotorRadius
 - AxialFluxGeneratorDesigner::Afpm, 31
- TurbineRpmMax
 - AxialFluxGeneratorDesigner::Afpm, 31
- TurbineRpmMin
 - AxialFluxGeneratorDesigner::Afpm, 31
- TurbineSpeedTipRatioMax
 - AxialFluxGeneratorDesigner::Afpm, 31
- TurbineSpeedTipRatioMin
 - AxialFluxGeneratorDesigner::Afpm, 31
- TurbineWindspeedMax
 - AxialFluxGeneratorDesigner::Afpm, 31
- TurbineWindspeedMin
 - AxialFluxGeneratorDesigner::Afpm, 32
- UpdateCalculations
 - AxialFluxGeneratorDesigner::Afpm, 24
- VoltageDrop
 - AxialFluxGeneratorDesigner::Afpm, 24