EMTAT Physics Based Library Tutorials

Rev. 1 | Nathan Butler | Spring 2021

EMTAT Version 1.0.7

Table of Contents

[1.0 Introduction 1](#_Toc69650673)

[1.1 EMTAT Overview 1](#_Toc69650674)

[1.2 EMTAT Physics Based Modeling Concepts 1](#_Toc69650675)

[1.3 EMTAT Physics Based Examples File 1](#_Toc69650676)

[2.0 Physics Based Library Blocks 2](#_Toc69650677)

[3.0 Example 1: Multiple Motor Branches Model 4](#_Toc69650678)

[3.1 Overview 4](#_Toc69650679)

[3.2 Model 1: Resistor-Motor 4](#_Toc69650680)

[3.3 Model 2: Inverter-Motor 5](#_Toc69650681)

[3.4 Model 3: Inverter-Cable-Motor 6](#_Toc69650682)

[3.5 Model 4A: Boost Converter Example 7](#_Toc69650683)

[3.6 Model 4B: Boost Converter-Inverter-Motor Branch 8](#_Toc69650684)

[3.7 Model 5A: Motor with Control Knobs 8](#_Toc69650685)

[3.8 Model 5B: Multiple Motor Branches 9](#_Toc69650686)

[4.0 Example 2: 11](#_Toc69650687)

[Model 1A: Generator & Motor Comparison 11](#_Toc69650688)

[Model 1B: Rectifier & Inverter Comparison 11](#_Toc69650689)

[Model 2: Generator-Rectifier Interface 12](#_Toc69650690)

[Model 3: Generator-Rectifier-Boost Converter 12](#_Toc69650691)

[Model 4: Generator-Rectifier-Boost Converter-Inverter-Motor 13](#_Toc69650692)

[Model 5: Generator-Rectifier-Buck Converter-Inverter-Motor with Cables & Control Knobs 14](#_Toc69650693)

# 1.0 Introduction

## 1.1 EMTAT Overview

The Electrical Modeling and Thermal Analysis Toolbox (EMTAT) is a Simulink toolbox containing blocks that represent generic electrical components such as motors, batteries, resistors, and more. While designed primarily for use in Electric Aircraft Propulsion (EAP) systems, EMTAT blocks may be used in many other applications.

EMTAT is divided into two main libraries. The first being PowerFlow, which uses an analytical approach to modeling electrical systems. The second library uses Physics Based components that more closely represent hardware that can be selected by engineers. Because of this, these blocks tend to have more detailed settings and connections. For some EMTAT models, a numerical solver method must be used.

EMTAT can be downloaded at [GitHub - nasa/EMTAT](https://github.com/nasa/EMTAT) or going to https://github.com/nasa/EMTAT.

## 1.2 EMTAT Physics Based Modeling Concepts

This tutorial describes the blocks contained within the Physics Based EMTAT library and gives several examples showing how they can be used. There are some concepts used in modeling with these blocks that will be discussed here.

Many blocks in the Physics Based libraries have masks defined with a tool called iDesign, which allows the user to define desired properties of the component represented by that block. Clicking “Run iDesign” will update block parameters to meet the user-requested properties. These parameters will then be used in the simulation and can be referenced in the selection of hardware components.

EMTAT Physics Based blocks can be thought of as passing information both forward and backwards. Specifically, blocks in the same branch of a system will pass voltage “forward,” with voltage changes happening in each additional component. Current, then, can be thought of as getting passed “backwards.” This means that the current demanded by a component in a branch will be sent backwards as an input to the previous component in that branch. The input Idemand is received from the next block and the output Idemanded is sent backwards to the previous block. This layout will become clear after examples are viewed.

## 1.3 EMTAT Physics Based Examples File

This tutorial references examples contained in a Simulink file located at: [EMTAT/Examples/Physics Based/Physics Based Combined Examples at master · nbutler01/EMTAT (github.com)](https://github.com/nbutler01/EMTAT/tree/master/Examples/Physics%20Based/Physics%20Based%20Combined%20Examples). This file contains multiple subsystems with examples grouped inside of them. For example, the subsystem “Example 1: Multiple Motor Branch Model” has all models referenced in the Example 1 section of this guide. Double-clicking a desired subsystem will enter it and allow the user to view and run the examples.

Opening the file will bring up subsystems have been commented out to decrease runtime. Uncomment subsystems and double click on them to view and run the examples inside. Inputs have been colored orange on all examples. Other values have different coloring. For example, error outputs are colored yellow.

# 2.0 Physics Based Library Blocks

This section contains brief descriptions of the blocks contained in the Physics Based EMTAT library. Details regarding block inputs, outputs, and mask parameters can be found in the Appendix. Additionally, EMTAT comes with a block library found at: [EMTAT/Documentation at master · nbutler01/EMTAT (github.com)](https://github.com/nbutler01/EMTAT/tree/master/Documentation).

It is recommended that this library is referenced while using these examples.

**Current-Based Resistor**

This block represents a resistor with calculations based on current. Adjust resistance to achieve a desired voltage drop at a given current.

**Voltage-Based Resistor**

This block represents a resistor with calculations based on voltage drop. Adjust resistance to achieve a desired current flow at a given voltage drop.

**Cable**

The cable block represents an electrical transmission cable that sends power from one component to another with some losses in between. These losses are calculated using cable properties that can be specified by the user in the cable’s iDesign window.

**Motor**

The motor block represents a generic AC permanent magnet synchronous motor with electrical performance calculated in the dq0 space. Motor performance is dictated by the commanded torque and speed setpoints, as well s the line input voltage. All the performance calculations and state space conversions are contained inside the block, and the related inverter block provides realistic efficiency losses rather than calculate any specific waveforms.

**Generator**

The generator block represents a generic AC permanent magnet synchronous generator with electrical performance calculated in the dq0 space. Generator performance is dictated by the commanded current and speed setpoints. All performance calculations and state space conversions are contained inside the block, and the related rectifier block provides realistic efficiency losses rather than calculate any specific waveforms.

**Inverter**

This block represents a DC-to-AC inverter (for motor control) using 6 IGBTs (2 per phase) using PWM to create an output 3-phase signal. (The model is based off of papers from Intech and Infineon.) The primary purpose of this model is to calculate the power losses from a sample inverter, since the motor model performs all the voltage and phase calculations internally.

The iDesign relies on a set of Infineon IGBTs and parallel IGBT modules. It selects the smallest one usable for the inverter based on maximum current, which in turn is based on total inverter power and phase voltage. To use your own IGBT data sheets, you can use the instructions provided in the source from Infineon about calculating IGBT power losses. The block parameters are generally 2-element vectors defining lookup tables with regard to temperature. This allows approximation of the IGBT losses varying with temperature, based on the two temperatures for which information is given in Infineon's datasheets. Lookup tables with more than 2 temperature breakpoints may be usable in theory.

**Rectifier**

This block represents a DC-to-AC rectifier using 6 IGBTs (2 per phase) using PWM to take in a 3-phase signal. (The model is based off of papers from Intech and Infineon.) The primary purpose of this model is to calculate the power losses from a sample rectifier, since the generator model performs all the voltage and phase calculations internally.

The iDesign relies on a set of Infineon IGBTs and parallel IGBT modules. It selects the smallest one usable for the rectifier based on maximum current, which in turn is based on total rectifier power and phase voltage. To use your own IGBT data sheets, you can use the instructions provided in the source from Infineon about calculating IGBT power losses. The block parameters are generally 2-element vectors defining lookup tables with regard to temperature. This allows approximation of the IGBT losses varying with temperature, based on the two temperatures for which information is given in Infineon's datasheets. Lookup tables with more than 2 temperature breakpoints may be usable in theory.

**Boost Converter**

This block represents a generic DC-DC boost converter controlled to produce a constant output voltage. The converter's efficiency is defined via a lookup table that is a function of input voltage and current. The input current is solved for, such that the power in equals the power out.

**Buck Converter**

This block represents a generic DC-DC buck converter controlled to produce a constant output voltage. The converter's efficiency is defined via a lookup table that is a function of input voltage and current. The input current is solved for, such that the power in equals the power out.

**Electrothermal Battery Model**

This block represents a physics-based battery model. The output voltage is a function of the current drawn from the battery ('Iout'), and the time integral of the current drawn from the battery, ('it', in Amp-hours). You need an integrator that takes in it\_dot, and you need to feed that integrated ‘it’ value into it\_base. The initial condition for it\_base should be zero, as this integrator should keep track of consumed charge, in amp-hours. If you put in a value other than 100% for SOC\_Init\_M, then you will see a difference between the it\_base (which always starts at zero), and ‘it’ as you get ‘it’ from the Outputs structure, which accounts for the difference in SOC\_Init\_M from 100%.

**Temperature Block**

This block is to be used in conjunction with other physics based, heat generating blocks. It takes an input of the heat generated by its associated functional block as well as the ambient temperature and calculates junction and surface temperature of the component.

**External Block: Steady State Newton-Raphson Solver with Jacobian Calculator**

This block is borrowed from the Toolbox for the Modeling and Analysis of Thermodynamic Systems. While not contained within the EMTAT Physics Based library, it is used in solving several examples.

The Steady State Newton Raphson Solver with Jacobian Calculator block supplies the inputs to iteratively drive the outputs of a system to zero. This is performed in three steps.

1) Jacobian calculator sends Perturbations to the plant then calculates the plant Jacobian based on the system response

2) The Newton Raphson will use the inverse of the Jacobian to iteratively step the system toward solution.

3) Once the solution has satisfied the conditions the Newton Raphson is shut off.

Simulation run time length should be selected as to give the Newton Raphson solver adequate time to develop a solution.

# 3.0 Example 1: Multiple Motor Branches Model

## 3.1 Overview

This series of examples shows how to build up a model that has multiple branches of motors. It demonstrates how to use the current-based resistor, motor, inverter, cable, and boost converter blocks. The examples largely build on each other, adding more blocks until reaching the final product in Model 5B.

## 3.2 Model 1: Resistor-Motor

This model includes a current-based resistor and a motor. The motor’s iDesign mask has been set to operate at a torque of 70 lbs-ft and a speed of 3000 rpm. These values are inputs to the motor block, which define the current demand output (Idemand).

The current demanded by the motor is then sent as an input to the resistor, where it is used to calculate voltage drop through the resistor. Finally, resistor voltage drop is subtracted from the source voltage and this value is input into the motor as V\_batt.

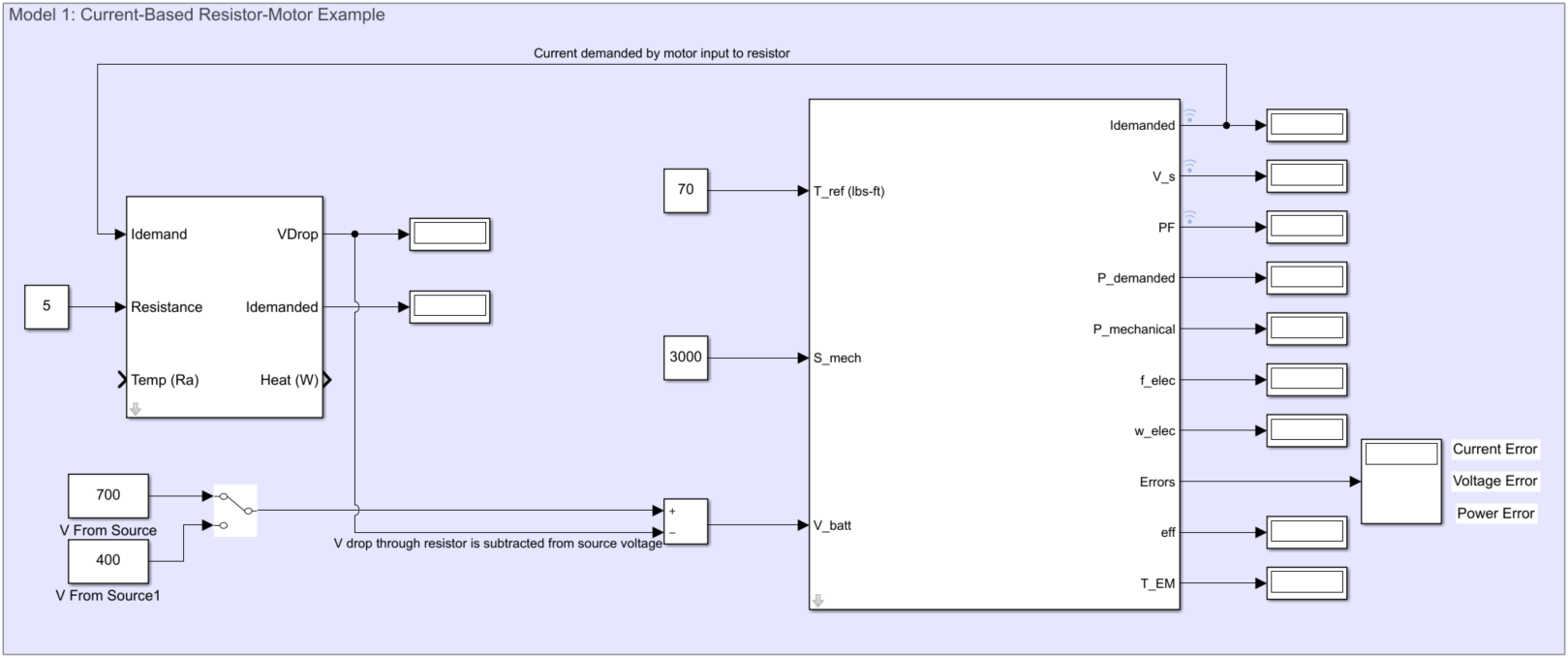


Figure 1: Current Based Resistor-Motor Example

Notes & Things to Try:

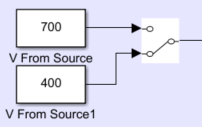
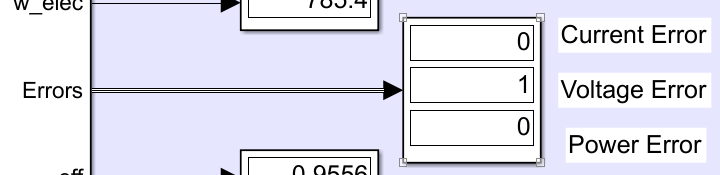
* Notice the switch on the V From Source input. When this switch is set to the lower voltage (400), an error appears in the Errors output of the motor block. The second error in the display represents a voltage error, meaning that the voltage supplied to V\_batt is less than the required stator voltage of the motor.

Figure 2: Example 1, Model 1 Input Voltage Switch

Figure 3: Example 1, Model Motor Errors

## 3.3 Model 2: Inverter-Motor

This example shows a more applicable setup by connecting a DC to AC inverter to the motor. In this case, the inverter’s inputs rely on V\_s, PF, and Idemand from the motor. Additionally, all motor inputs first pass through the inverter block before being sent on as motor inputs.

The inverter’s iDesign mask is updated to handle V\_batt and the expected power of the circuit. Notice the efficiencies of each block are near 1. This is expected if the inputs near the specifications entered in iDesign.

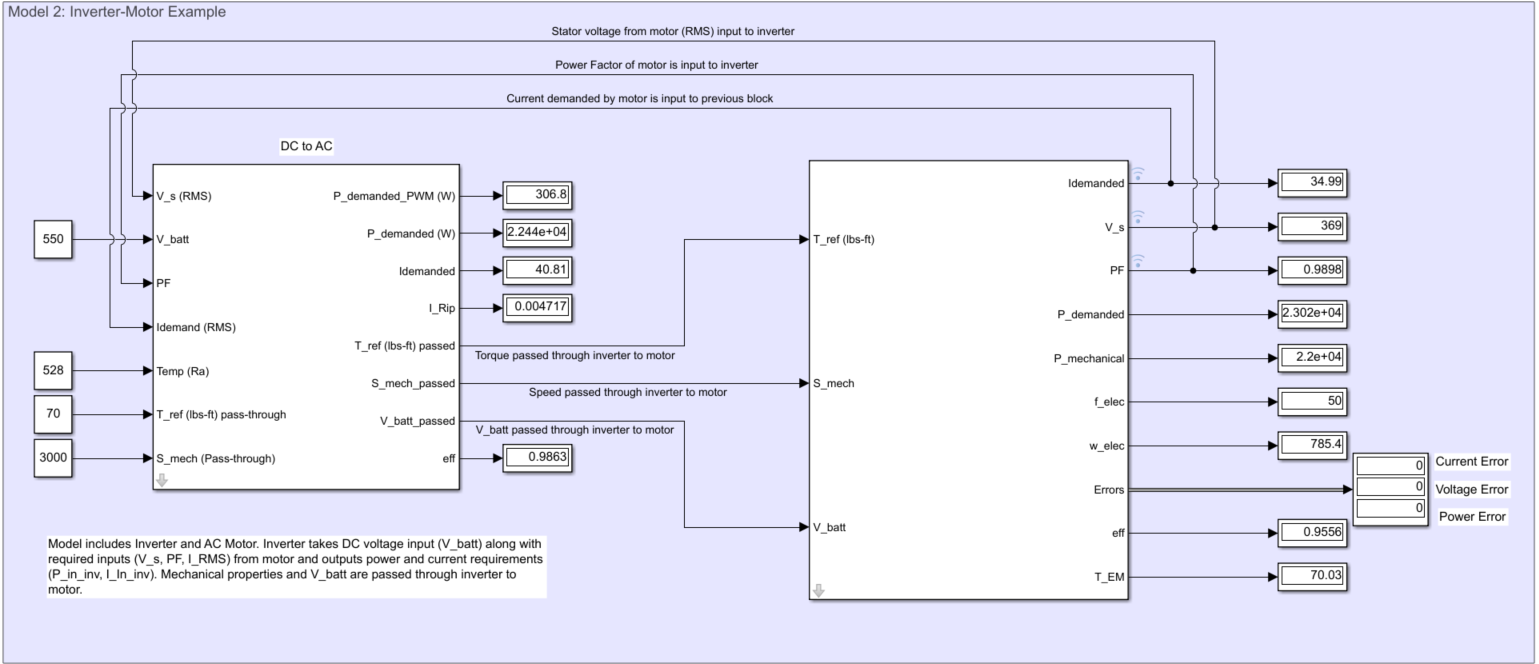


Figure 4: Inverter-Motor Example

Notes & Things to Try:

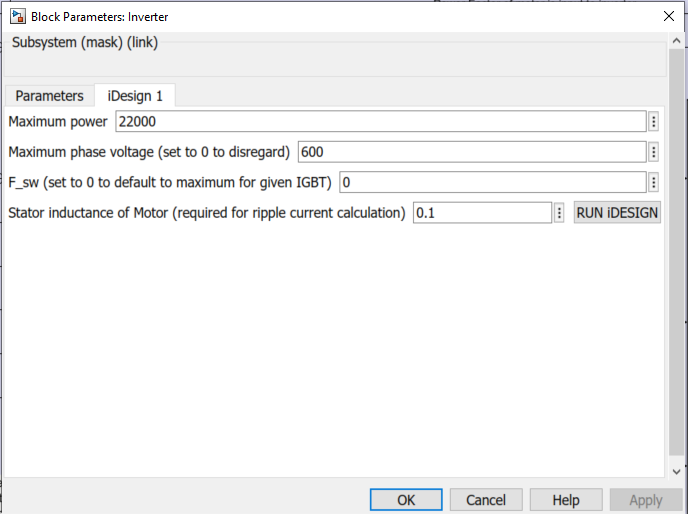
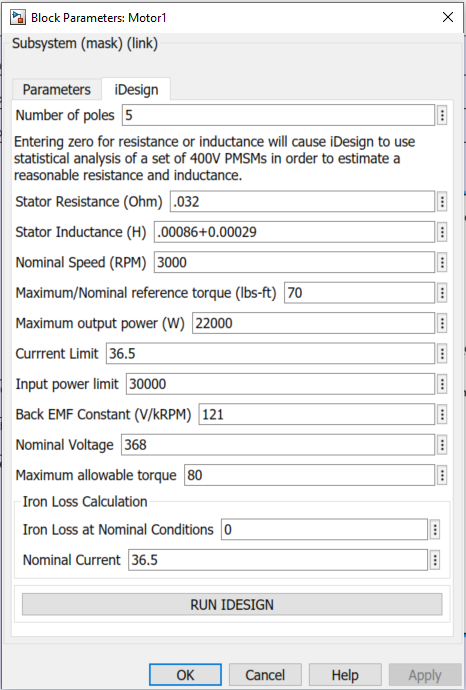
* The power and current demand outputs from the inverter would be sent backwards as inputs to additional components in its branch, if this were a more complex model.
* Notice that the iDesign parameter for power in both components are set to 22000W

Figure 5: Motor iDesign

Figure 6: Inverter iDesign



## 3.4 Model 3: Inverter-Cable-Motor

This example is similar to Model 2, the only difference is that a cable block has been inserted so some voltage will be lost between the inverter and motor. Idemanded from the motor is fed back into the previous component in the branch, in this case that is the cable. Idemanded by the cable then becomes the input to the inverter.

Additionally, voltage from the inverter must go through the cable before reaching the motor. Voltage will typically follow this trend of being passed forward through models. V\_s and PF from motor are still fed back into the inverter and T\_ref and S\_mech are still passed from inverter to motor.

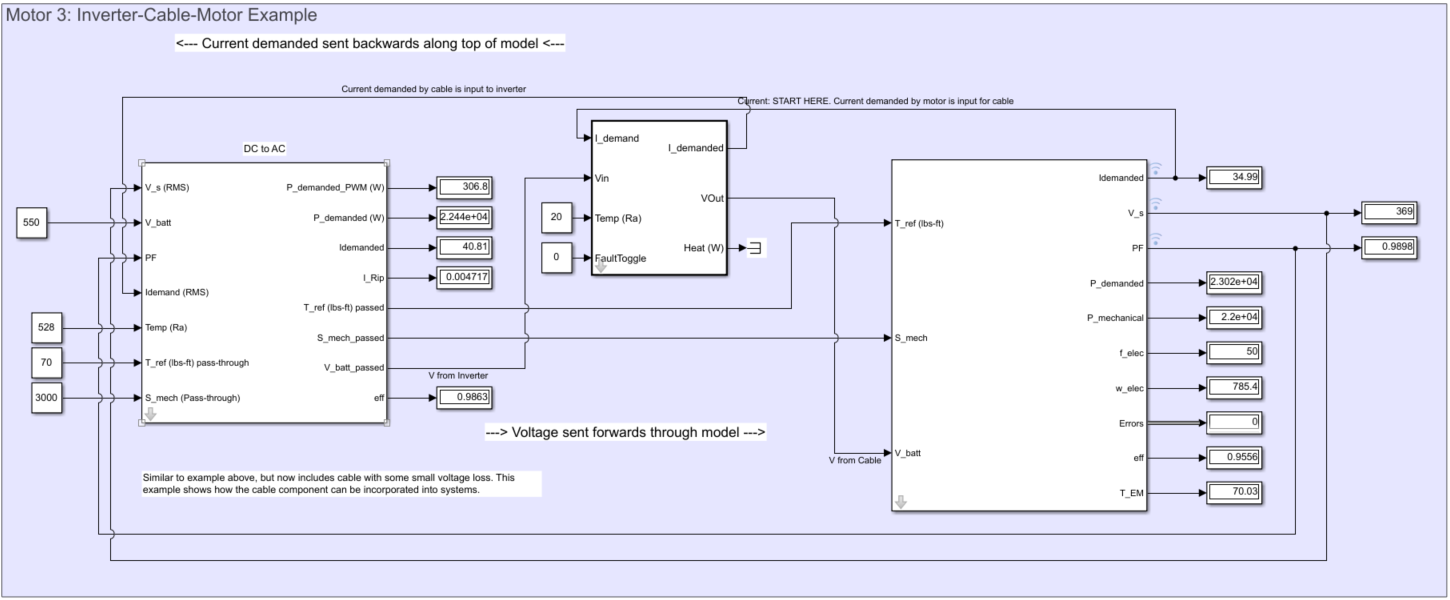


Figure 7: Inverter-Cable-Motor Example

Notes & Things to Try:

* Try adjusting the ambient temperature inputs to the cable and inverter and see how the system responds.
* The fault toggle input on the cable is currently 0. A value of 1 will set all cable outputs to 0, meaning the Idemand input to the inverter is 0 and V\_batt input to the motor is 0.

## 3.5 Model 4A: Boost Converter Example

This example demonstrates the use of the boost converter block. This includes the use of a Steady-State Newton Raphson Solver block that solves for input current and duty cycle by driving P\_Err and the difference between Vout and desired Vout to 0.

The iDesign mask on the boost converter sets the block to take an input of approximately 250V and output 540V at the current specified by the Idemand input (150A in this example). The solver then determines guesses for the required current demanded by the boost converter. In this case the final current demanded is approximately 350A.

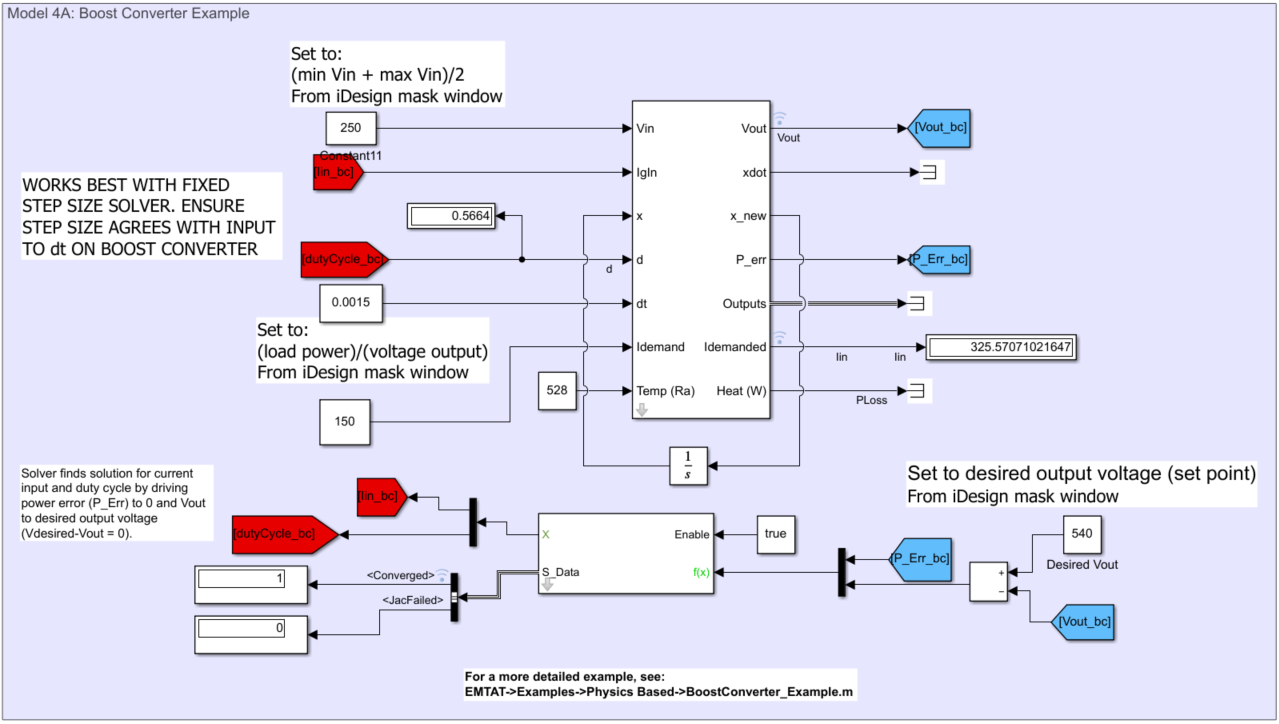


Figure 8: Boost Converter Example

Notes & Things to Try:

* A more detailed version of this example can be found at:

EMTAT->Examples->Physics Based->BoostConverter\_Example.m

* The ‘Converged’ output from the solver will indicate whether a solution has been found.
* Try changing the boost converter iDesign values to output a different voltage. Be sure to click RUN IDESIGN before closing the mask. Also, update the value of constant “Desired Vout” to match your new voltage. Run the model and see how current demanded changes.

## 3.6 Model 4B: Boost Converter-Inverter-Motor Branch

This model adds a boost converter to the previously defined Inverter-Cable-Motor model. The setup for the boost converter similar to Model 4A, the main differences being that Vout from the boost converter is sent to the next cable block, and then through the rest of the model. Additionally, Idemand into the boost converter is received from the cable block.

It can be observed in this larger model that current demand is sent backwards through the circuit, starting from the motor block and ending with the Idemanded output from the boost converter. The opposite is true for voltage, which starts at the Vin input to the boost converter and ends with Vbatt to the motor.

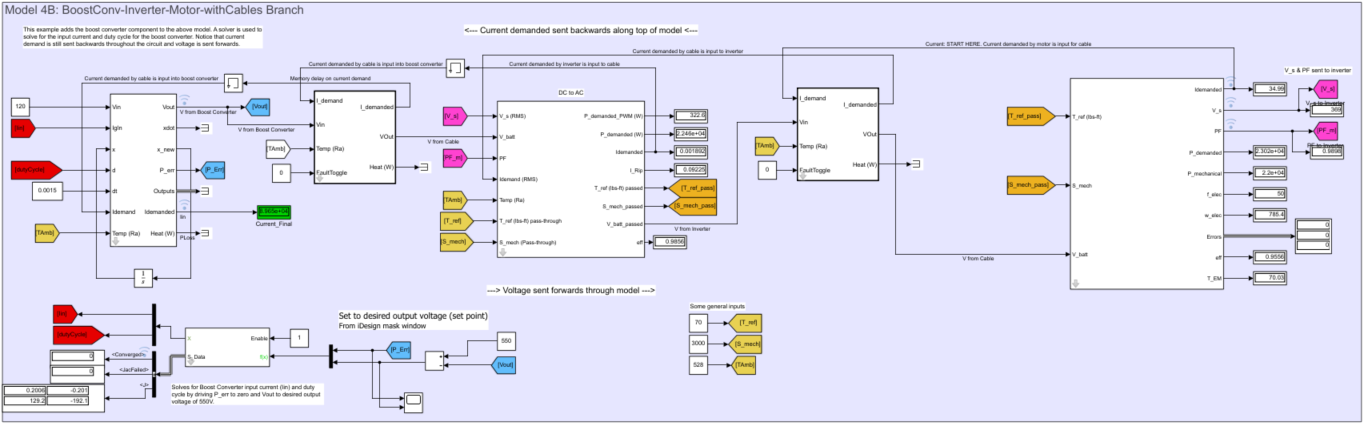


Figure 9: Boost Converter-Inverter-Motor Example

Notes & Things to Try:

* Memory blocks are necessary to avoid algebraic loops and solver errors for the boost converter calculations.
* Goto/From blocks are used to simplify the look of the model by reducing cluttered lines.
* The scope block on the solver inputs shows P\_Err and (Desired Voltage minus Vout ) being driven to zero by the solver.

## 3.7 Model 5A: Motor with Control Knobs

This example simply provides control knobs for the speed and torque inputs to the motor block. Adjust the control knobs and notice the change in Idemanded and V\_s outputs.

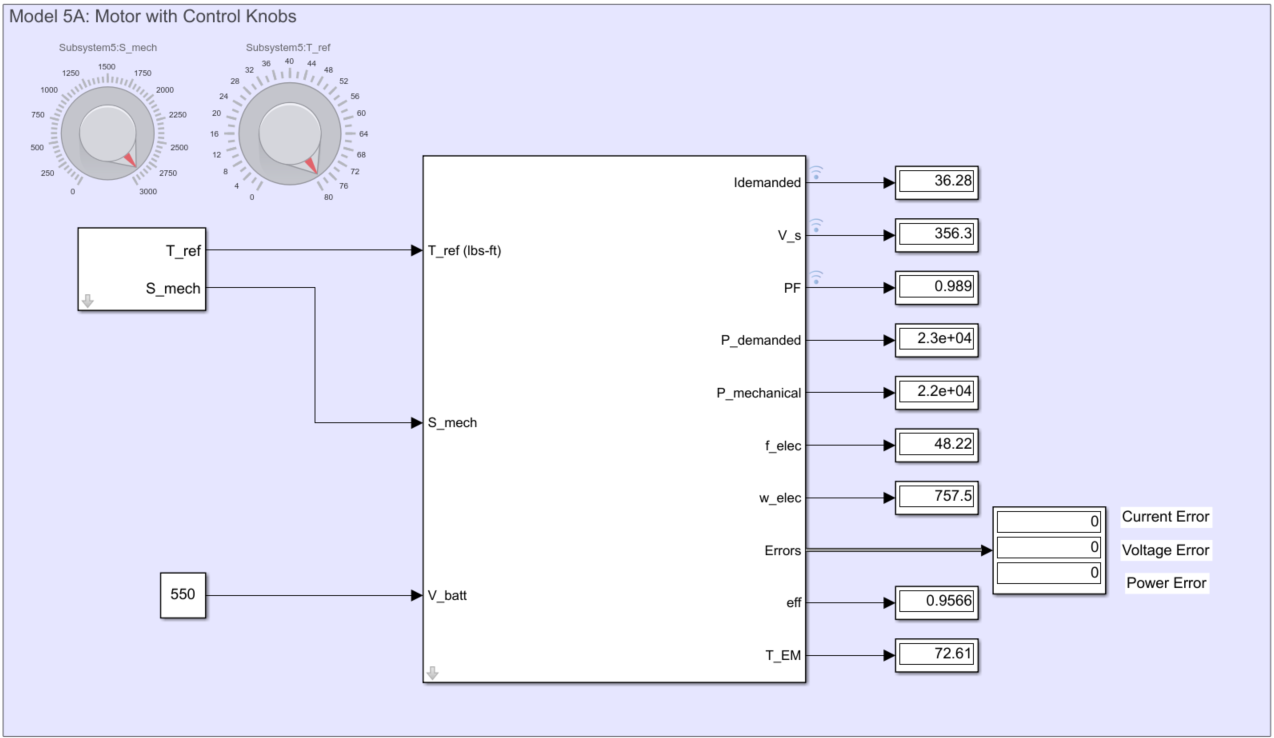


Figure 10: Motor with Control Knobs Example

## 3.8 Model 5B: Multiple Motor Branches

The final stage of these example models takes the Boost Converter – Inverter – Motor branch and adds an additional Inverter – Motor branch. This shows how multiple motors can be controlled using the EMTAT Physics Based library blocks. The current demanded by each branch is summed and then input to the boost converter, while voltage supplied by the boost converter is sent forward to each branch. Voltage losses can be seen resulting from each cable block.

Control knobs have been added that control speed and torque settings for each motor. These controls are located in the bottom left corner of the model. By tweaking these, changes in voltage, current, and efficiencies can be viewed in the different components.

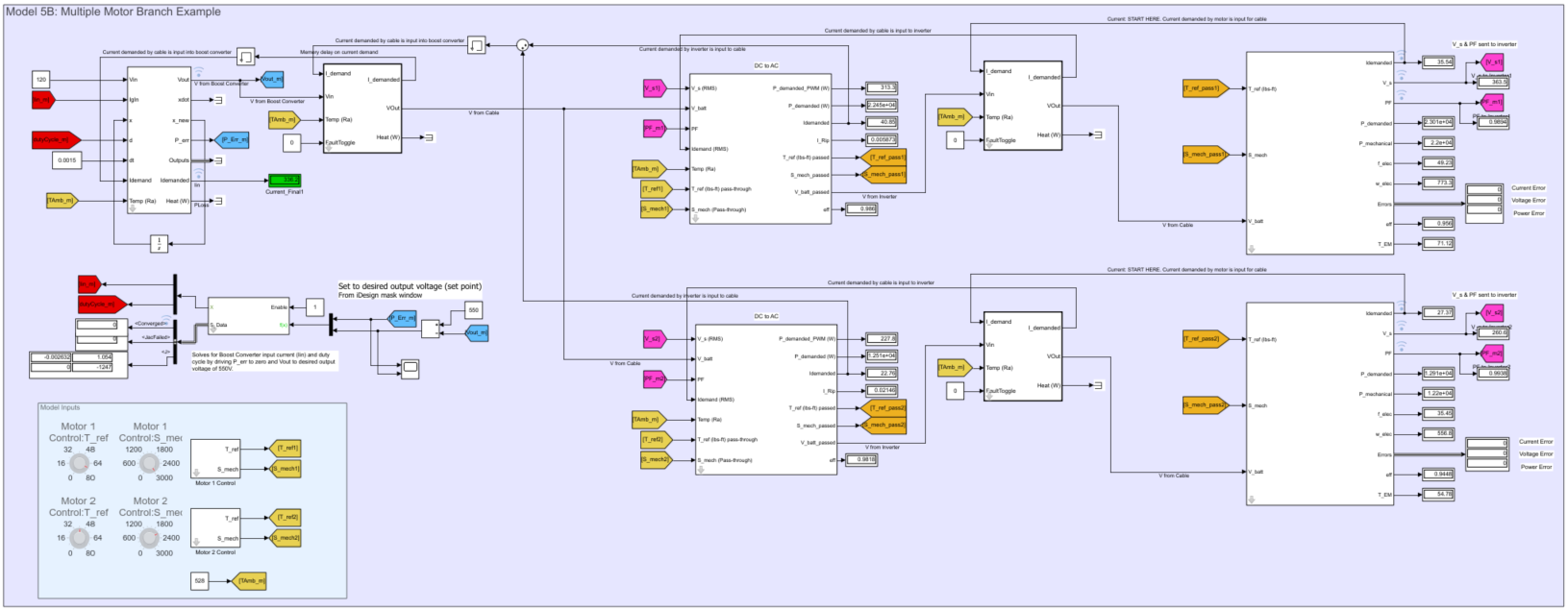


Figure 12: Multiple Motor Branches Example

Notes & Things to Try:

* Once again, Goto/From blocks are used to reduce clutter.
* Adjust the T\_Amb input in the control box in the bottom left corner to see how different components react.
* This example should demonstrate how additional motor branches can be added without major changes to the rest of the model.
* Try adjusting speed and torque settings for the different branches and see how overall power demand changes.
* Feel free to make some adjustments to the larger model that you made in previous examples and observe how they affect the larger model.

# 4.0 Example 2:

## Model 1A: Generator & Motor Comparison

This example compares the generator block with the motor block. Current, speed, and torque are highlighted to show the relationship between these values. The generator takes in speed and current demand and outputs a torque demand, which can be sent to a shaft or a similar component. Alternatively, the motor will take torque and speed inputs and output a current demand.

Notice that the torque output value from the generator is entered as an input to the motor. With both blocks running at the same speed, it can be observed that the motor’s current demand equals the generator current input.

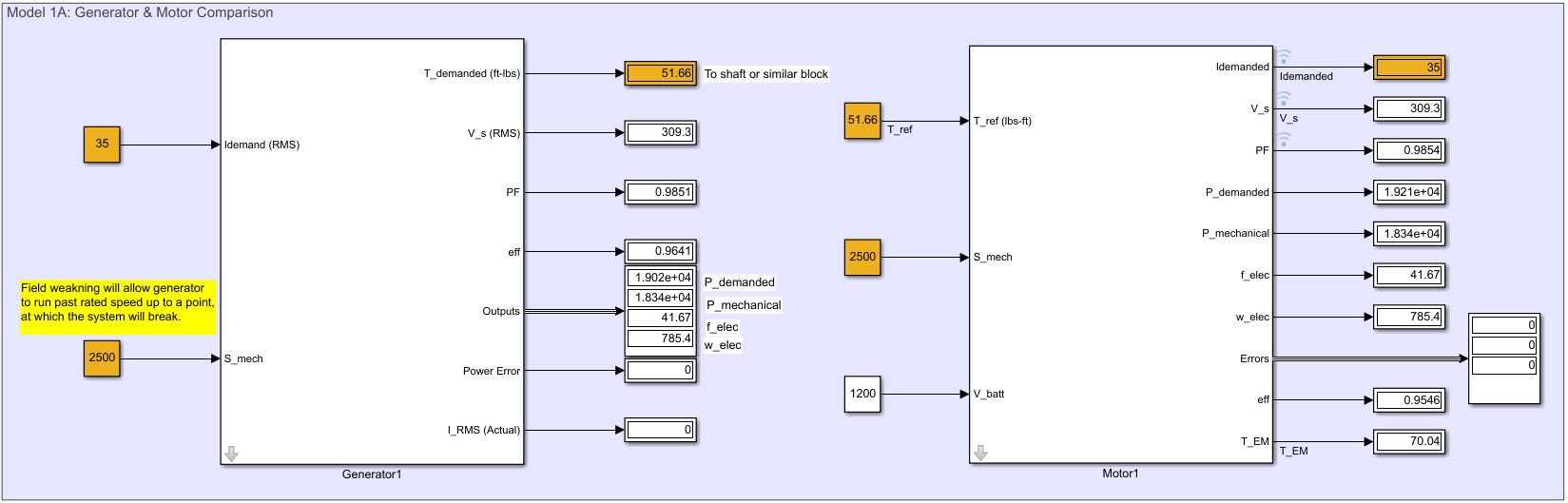


Figure 13: Generator vs Motor Example

Notes & Things to Try:

* Try adjusting some input values to see how the outputs behave. Note the power error on the generator, which is set to 1 when P\_demanded > P\_mechanical.
* Field weakening allows the generator to run past its rated speed up to a point, at which the system will break.

## Model 1B: Rectifier & Inverter Comparison

This example compares the Rectifier block with the inverter block. Like the inverter, the rectifier receives V\_s and PF inputs from the generator. The rectifier’s Idemand input comes from a component down the branch, rather than from an attached generator.

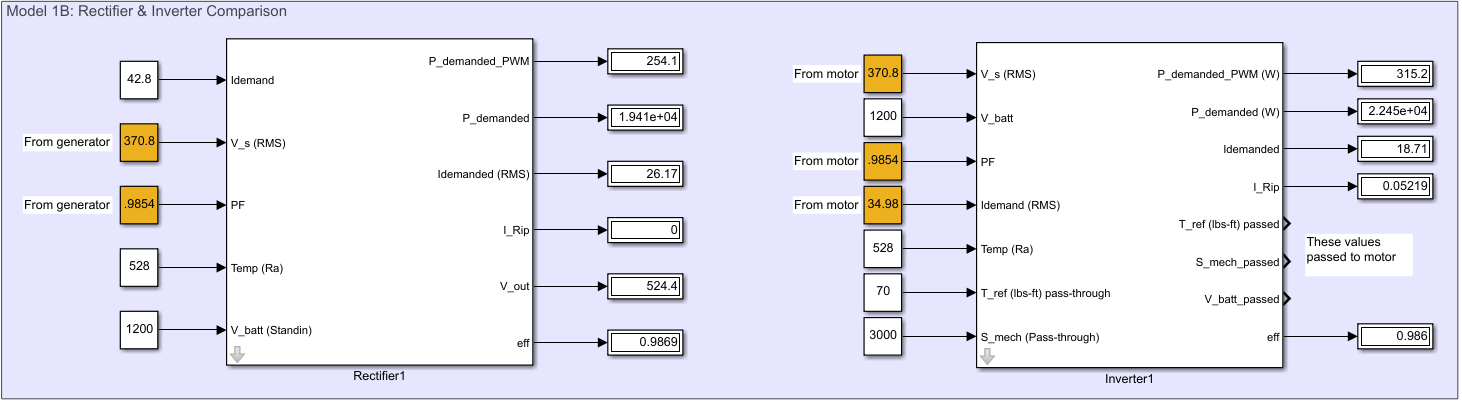


Figure 14: Rectifier vs Inverter Example

Notes & Things to Try:

* Both components receive V\_s and PF inputs from the generator/motor. The rectifier is needed to convert the AC generator outputs to DC values, whereas the inverter does the opposite
* The rectifier outputs current in RMS to match the generator input

## Model 2: Generator-Rectifier Interface

This model demonstrates the interface between the generator and the rectifier. The generator acts as a voltage source, sending V\_s to the rectifier. Current demanded by the rectifier is sent to the generator as an input, along with speed. Power factor is also sent from the generator forward to the rectifier.

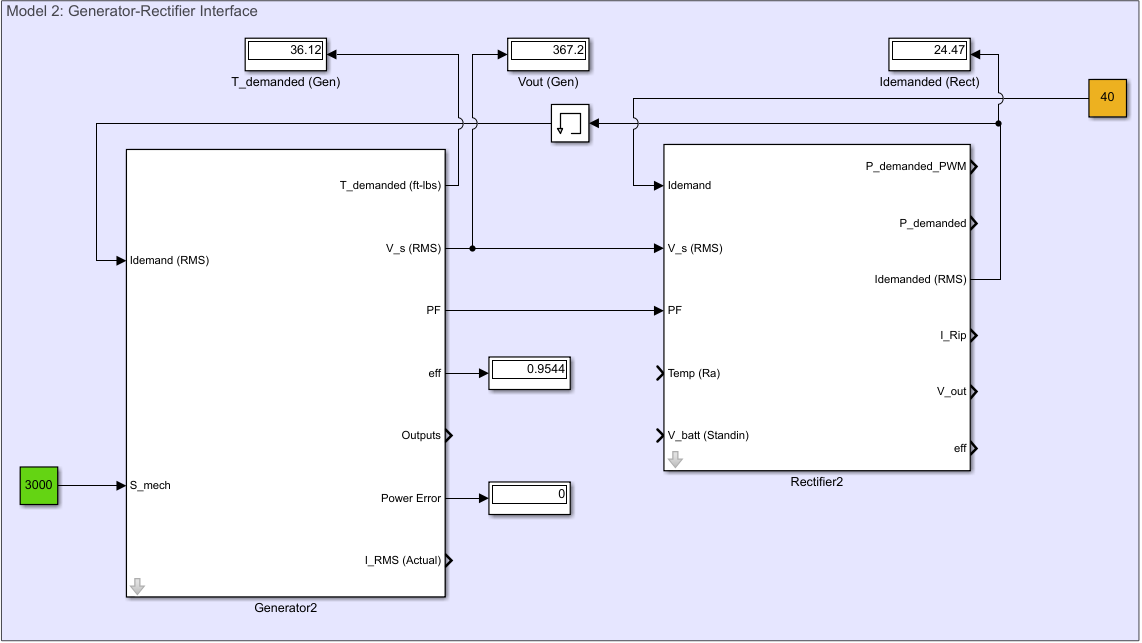


Figure 15: Generator-Rectifier Interface Example

Notes & Things to Try:

* A memory block is necessary between the components to prevent an algebraic loop.

## Model 3: Generator-Rectifier-Boost Converter

This model adds a boost converter to the previously defined Generator-Rectifier model. The setup for the boost converter similar to Model 4A in Example 1, the main difference is Vin to the boost converter is received from the Rectifier block. Additionally, Idemand into the boost converter is received from the added cable block.

It can be observed in this larger model that current demand is sent backwards through the model, starting from the cable block and ending with the Idemand input to the generator. The opposite is true for voltage, which is output by the generator and then passed forward along the branch.

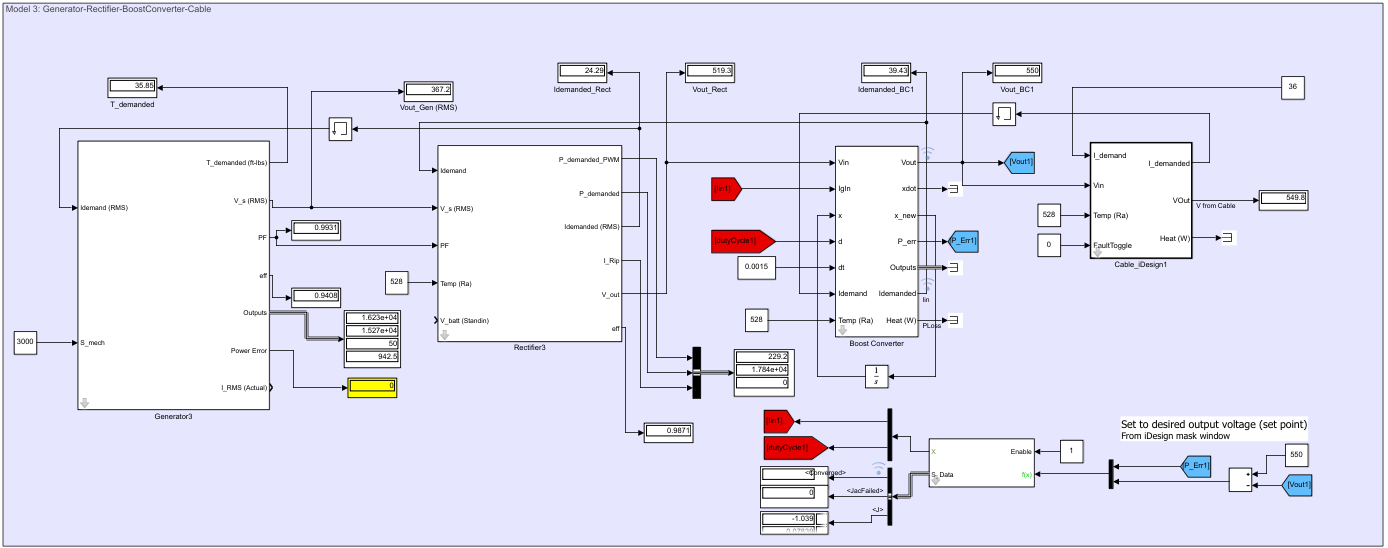


Figure 16: Generator-Rectifier-Buck Converter Example

Notes & Things to Try:

* The insertion of the boost converter is a step towards the coming models as it is often necessary to step up voltage out of the generator
* The boost converter will solve for the input current necessary to supply the output voltage defined in its iDesign mask

## Model 4: Generator-Rectifier-Boost Converter-Inverter-Motor

This model strings together a branch that begins with a generator source block and ends with a motor running at a defined speed and torque. The motor block demands a current to run at this state, which is then passed backwards through each block before being input into the generator. The generator block outputs a voltage that passes through the rectifier and to a boost converter, where it is stepped up before eventually reaching the motor block.

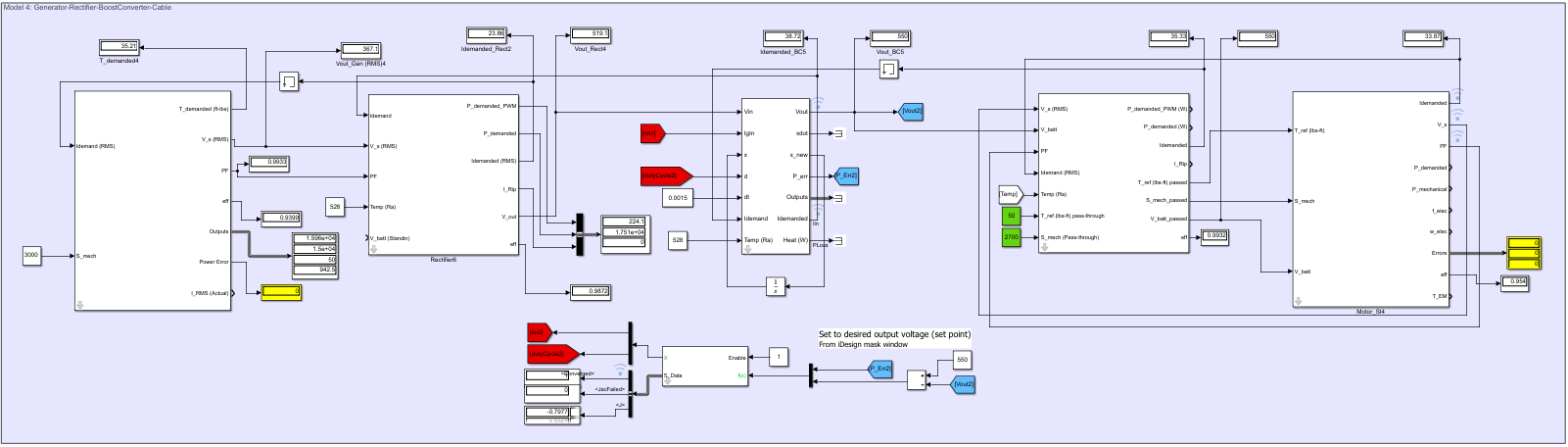


Figure 17: Generator-Rectifier-Boost Converter-Inverter-Motor Example

Notes & Things to Try:

* In these larger models the flow of current and voltage is more apparent. This example is set up such that current is passed along the top of the blocks, starting with the demand from the motor and ending with the input to the generator.

## Model 5: Generator-Rectifier-Buck Converter-Inverter-Motor with Cables & Control Knobs

This model builds on Model 4, with added cable blocks between each component. These cables see minor voltage drop, which can be altered by changing the temperature inputs. Additionally, controls are included that can be used to adjust the speed and torque inputs to the motor and the speed input to the generator.

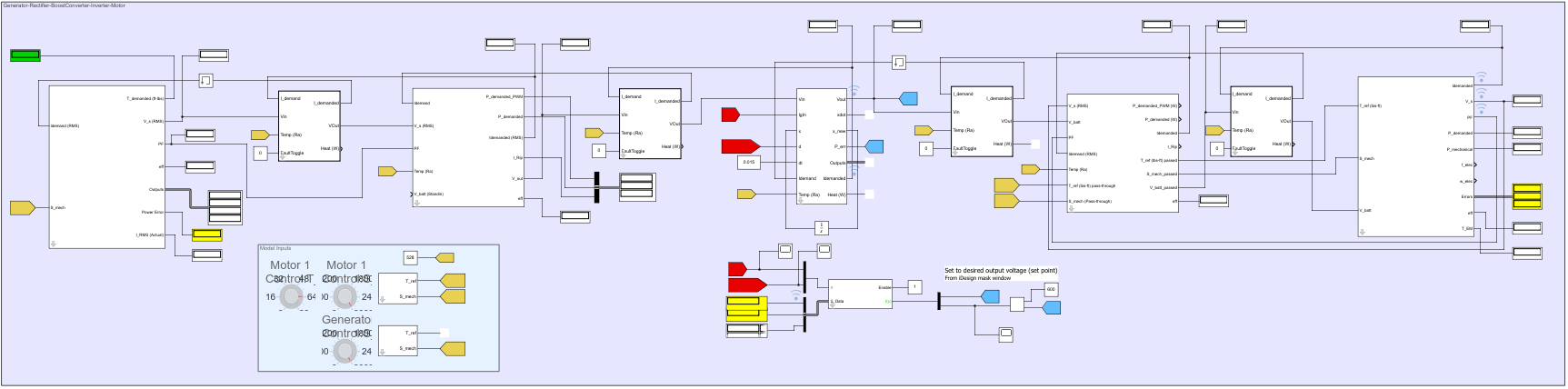


Figure 18: Model 4 with Added Cables and Control Knobs

Notes & Things to Try:

* Try adjusting speed and torque settings for the motor and generator. Take note of the displays highlighted in yellow, as they will indicate whether errors have occurred. The displays on the NR solver should have a 1 for “converged” if the model successfully solves.