System Overview

The system being modeled in this Simulink example is a variable-speed fan cooling a CPU which, depending on the model and the operating point, can generate variable amount of heat. Each component of the system is modeled using empirical data and technical specifications, and assumptions are stated in the following discussion regarding levels of modeling fidelity.

Compensation

A PID controller is used to enforce a temperature limit of 70.0°C on the processor case. Temperature limits are typically provided by the manufacturer and can be easily integrated into this model. In this example, the sensing location is on the case, so the limit used is for the case, however the sensor can be placed on the die and the limit accordingly set to the maximum junction temperature. To avoid overheating, the temperature limit set-point for the controller is specified as 60.0°C. The cooling fan used by this model is driven by a 12V electric motor. Thus, the controller output is saturated to a lower limit of 0V, indicating the motor is not running and there is no forced convective cooling, and an upper limit of 12V, which corresponds to the highest speed of the motor and therefore the highest convective heat transfer the system is capable of.

Actuation

The performance of the cooling fan used in this model is approximated using numbers in the typical range of consumer grade fans. The two values of interest for the fan are the RPM output from an input voltage and the corresponding volumetric flow rate at that RPM. For simplicity in this example, maximum values are provided for RPM and CFM and intermediate values are linearly interpolated depending on the voltage and RPM input, respectively. Fan performance plots can be obtained from manufacturers and these can be digitized into a data table using a plot digitization tool. [1] Using this plot digitization to lookup table technique, the model can better capture nonlinearities by adding more data points. In order to get mass flow rate, the International Standard Atmosphere model was used at sea level elevation to get an air density of 1.2250 kg/m^3. This data is available from the Aerospace Toolbox. [2] Finally, inertial effects need to be introduced to the fan model to account for the delay from voltage input to RPM output. To do this, a Simscape simulation was run by adapting the “Thermistor-Controlled Fan” example. [3] The DC motor with a rotational inertia of 1E-5 $kg \cdot m^2$ attached to it was run at 12V and the time to reach 63.2% of the steady state rotational velocity was computed and used as a first-order time constant.

Future simulation considerations can make use of noise data available for cooling fans. Additional requirements for noise levels can be added to test whether or not a specific fan model is capable of adequately cooling a CPU while staying sufficiently quiet for customer requirements.

Plant

The plant in this system is the thermal dynamics of the heat-generating CPU and the heat-dissipating heat sink. These dynamics are modeled by a finite element formulation of the heat equation on the 3D geometry. The geometry, meshing, and discretization in this example are done using MATLAB’s PDE Toolbox. The finite element matrices generated can then be solved using Simulink’s Descriptor State Space block. For more information on time-dependent finite-element formulations, please see the MathWorks documentation. [4]

The dominant mode of heat transfer in this scenario is due to convection and is governed by the equation:

$$\dot{q} = h(T_\infty - T)$$

The coefficient *h* is the convective heat transfer coefficient and can be found experimentally and varies as a function of mass flow rate. Again, for simplicity values are interpolated linearly between a free convection coefficient (no airflow) to a maximum forced convection coefficient. Like the fan performance plots, empirically derived values can be used to construct a lookup table for a mass flow dependent coefficient value. A convective boundary condition was constructed for the ducted airflow where forced convection coefficients would be applied on the internal fin surfaces, and free convection coefficients were applied on the external facing surfaces of the system.

The heat generation of the CPU can be controlled by the user in this formulation as an input Wattage. This demonstration provides examples of different loadings of the processor that lead to different heat generation, thus lending itself useful for design decisions in determining if a certain heat sink is suitable for a specific CPU.

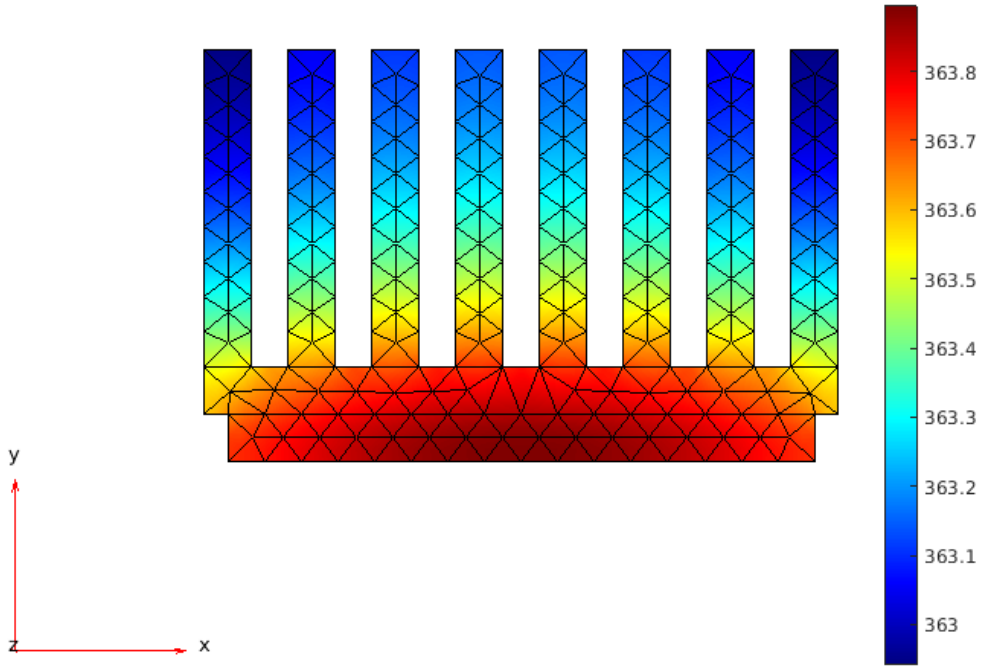

Figure 1: Cross section of heat sink and chip casing, showing temperature gradient colder near fin tips

Sensor

The sensor temperature dynamics, like the fan inertial effects, are model by a first order time constant. For this example, a time constant of 15 seconds is used, and this value can be taken directly from manufacturer specifications.

Model Order Reduction

This demonstration model represents just one way of modeling a physical system. Different levels of modeling fidelity need to considered and situational requirements of accuracy and computational cost

should be weighed against each other. Each engineering design process will have accuracy requirements for which a specific resolution level will correspond; any higher accuracy computations would be a waste of resources. Finding this level of accuracy and reducing unnecessary computations can be accomplished through model order reduction techniques, which will be explored in future contributions to this repository.

Running the Model

This example is organized inside a Simulink Project. Once the project is opened, running “top_script” will load all the variables needed for the simulation and then execute a parameter sweep. Six scenarios are included in this example. The six simulations run varied levels of heat generation from the CPU as input to the plant subsystem.

Upon completion of the parameter sweep, the Simulation Manager will launch. The socket temperature signal is logged using Simulink Data Inspector and can be visualized by selecting the simulations of interest and clicking the “Show Results” option. The Signal Editor block, data logging, and adding additional systems to the model can all provide opportunities to expand on physical system modeling and the engineering design decision process that goes along with it.

Note, there is an included “debug” plant model that uses the same parameters, but a simpler geometry with fewer mesh nodes to reduce the number of degrees of freedom. This is useful for debugging the model and iterating on other parameters in the system outside of the plant. This feature can be utilized by commenting out “heat_sink_defs” in the “top_script” and replacing it with “heat_sink_defs_DEBUG”.

References

[1] A. Rohatgi, "WebPlotDigitizer 4.4," [Online]. Available: https://apps.automeris.io/wpd/.

[2] Tha MathWorks, Inc., "Use International Standard Atmosphere model," [Online]. Available: https://www.mathworks.com/help/aerotbx/ug/atmosisa.html.

[3] The MathWorks, Inc., "Thermistor-Controlled Fan," [Online]. Available: https://www.mathworks.com/help/physmod/sps/ug/thermistor-controlled-fan.html.

[4] The MathWorks, Inc, "Finite Element Method Basics," [Online]. Available: https://www.mathworks.com/help/pde/ug/basics-of-the-finite-element-method.html.