More-Electric Aircraft Modeling in Simscape

Graham Dudgeon
Principal Product Manager – Electrical Technology

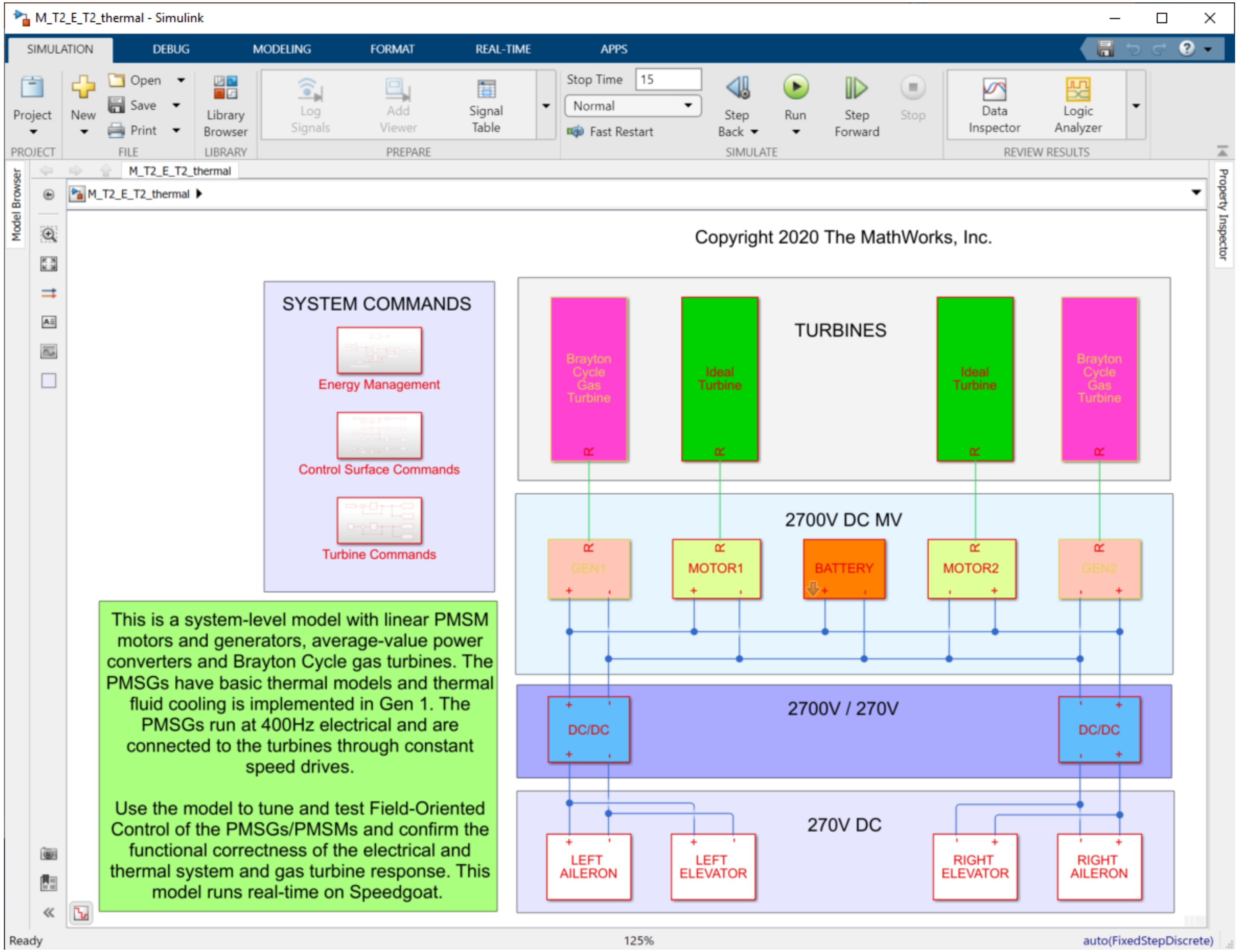
Overview

- Notional More-Electric Aircraft
- Local Solver
- Using Different Sample-Times for Different Networks
- Real-Time Simulation using SLRT and Speedgoat
- Creating an FMU Standalone Model

Notional More-Electric Aircraft

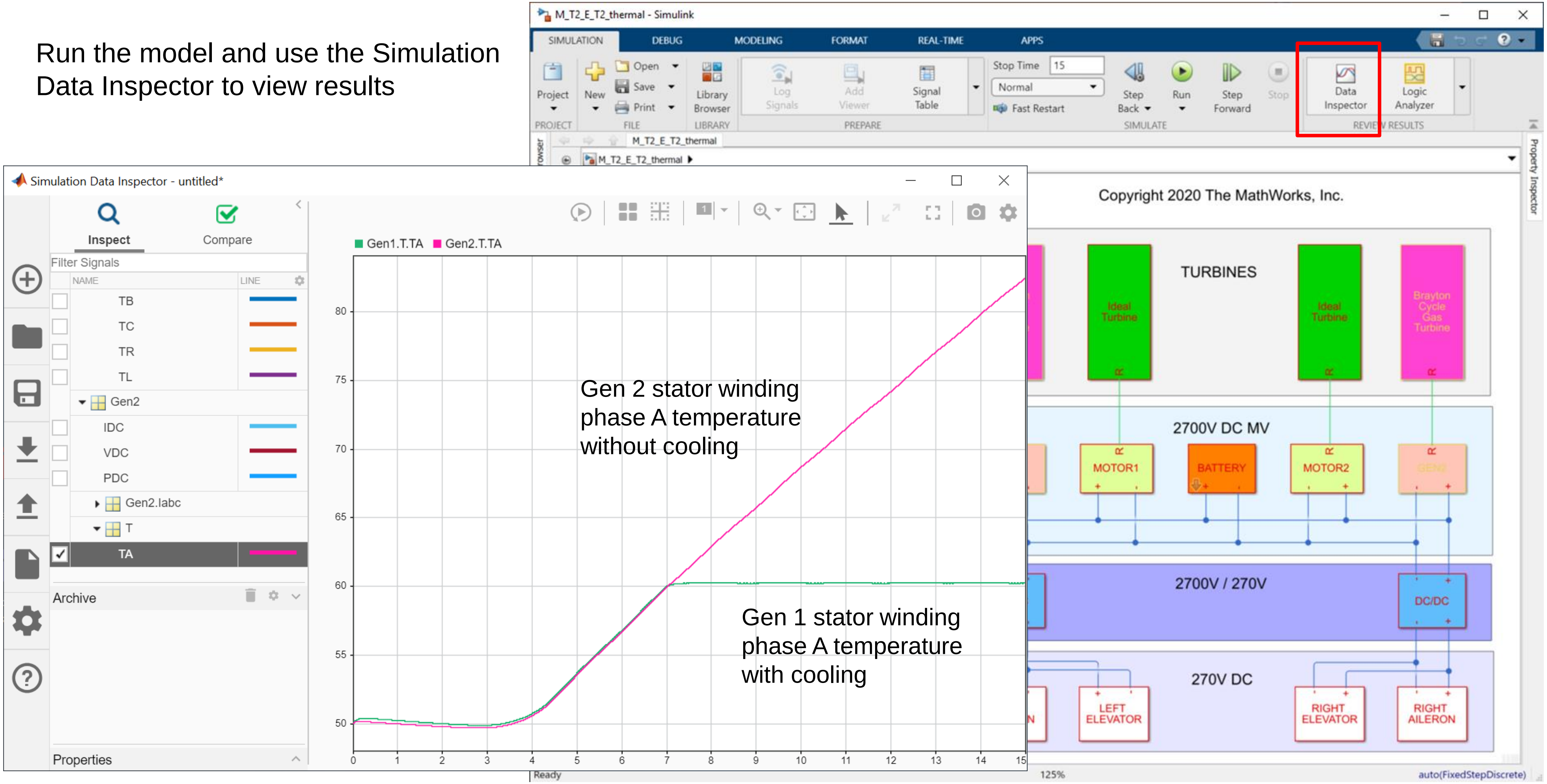
The model has the following components,

- Two Brayton Cycle Gas Turbines (gas, thermal and mechanical domains)
- Two PMSGs with FOC control (electrical and thermal domains)
- Two PMSMs with FOC control and ideal speed reference (electrical and mechanical domains)
- Gen 1 has a basic thermal model and thermal fluid cooling
- Gen 2 has a basic thermal model
- Two DC/DC converters
- Four PMSM actuators with average-value power converters and FOC control



Notional More-Electric Aircraft

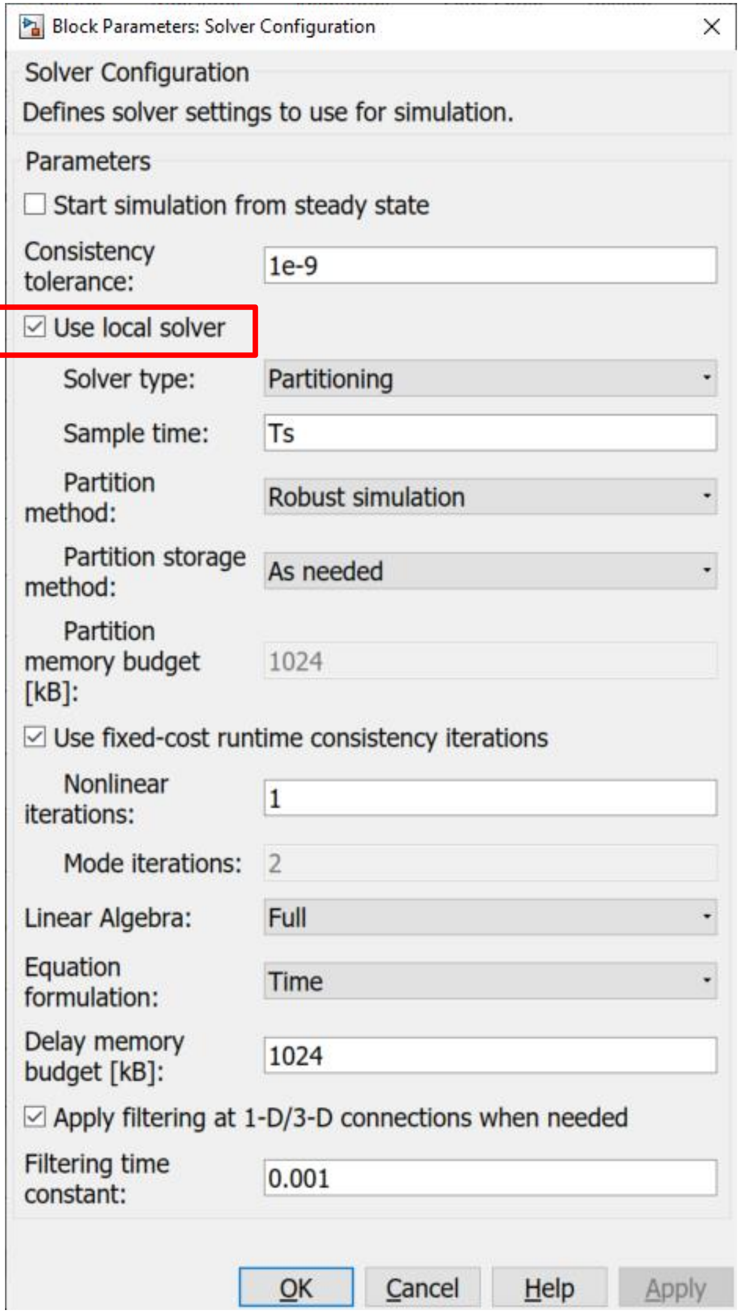
Run the model and use the Simulation Data Inspector to view results



Local Solver

- By selecting ‘local solver’ on the solver configuration block, the Simscape network will be simulated using a fixed-step solver, and the Simscape network is presented to Simulink as a discrete system.

- If ‘local solver’ is deselected, then the Simscape model is simulated using the main Simulink solver.

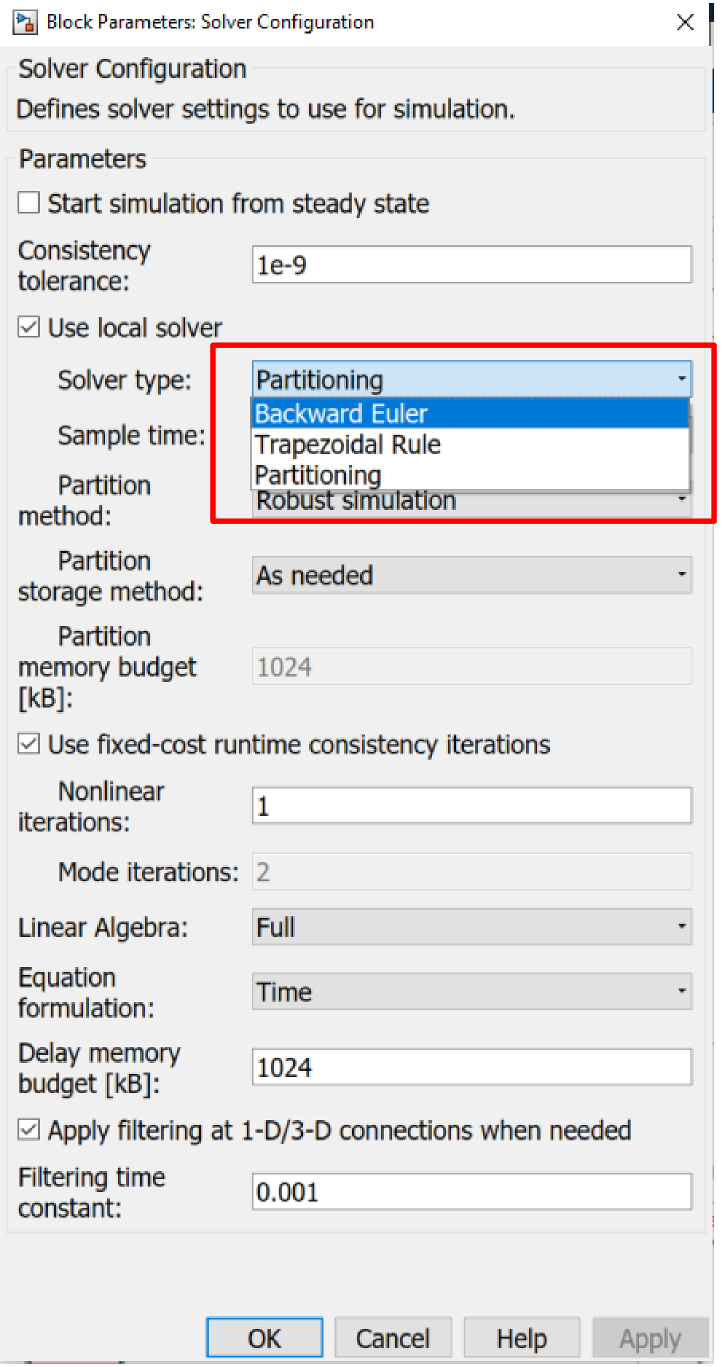


Local Solver

- There are three solver types.
  - Backward Euler
  - Trapezoidal Rule
  - Partitioning

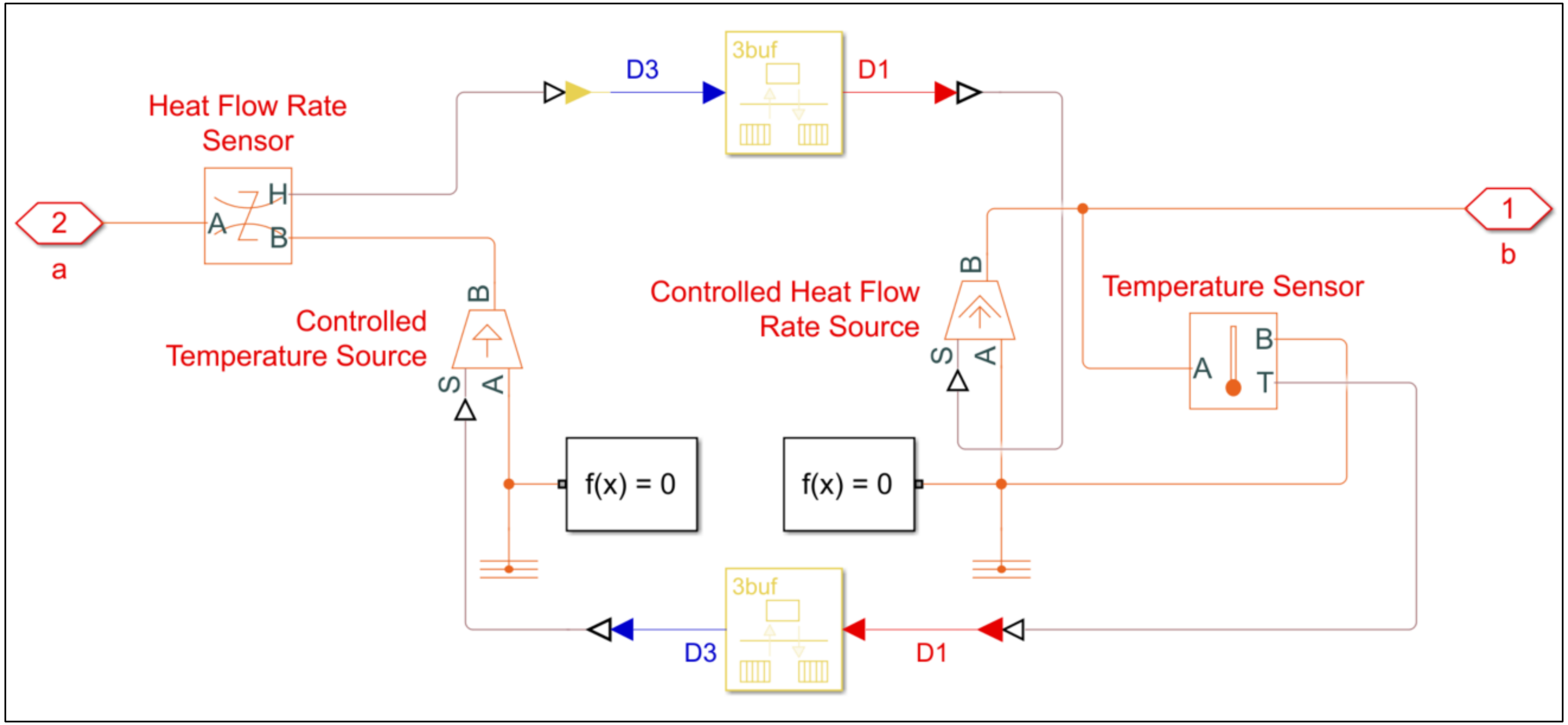
- Partitioning solver converts the entire system of equations for the Simscape network into several smaller sets of switched linear equations that are connected through nonlinear functions, and can lead to faster simulations for certain networks.

- NB: Not all networks can be simulated using Partitioning solver. e.g. highly non-linear systems and/or stiff systems.



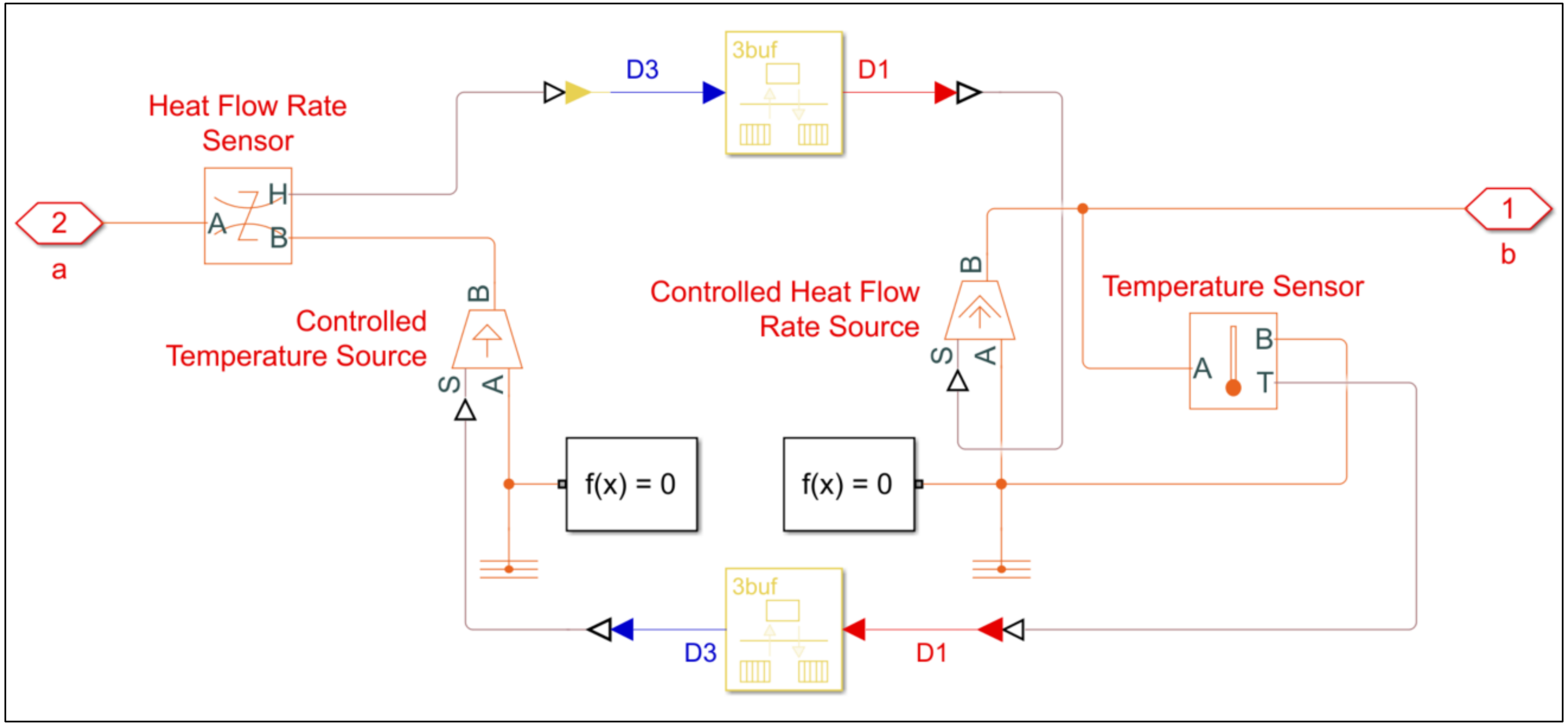
Using Different Sample-Times for Different Networks

- To run different sections of a physical system at different sample times, we first connect the networks via Simulink signals. As an example, connecting two systems via thermal ports is shown below.



Using Different Sample-Times for Different Networks

- Each solver configuration block is set to 'local solver' and appropriate sample times can be set. Rate Transition blocks then manage the transfer of data between the two networks.

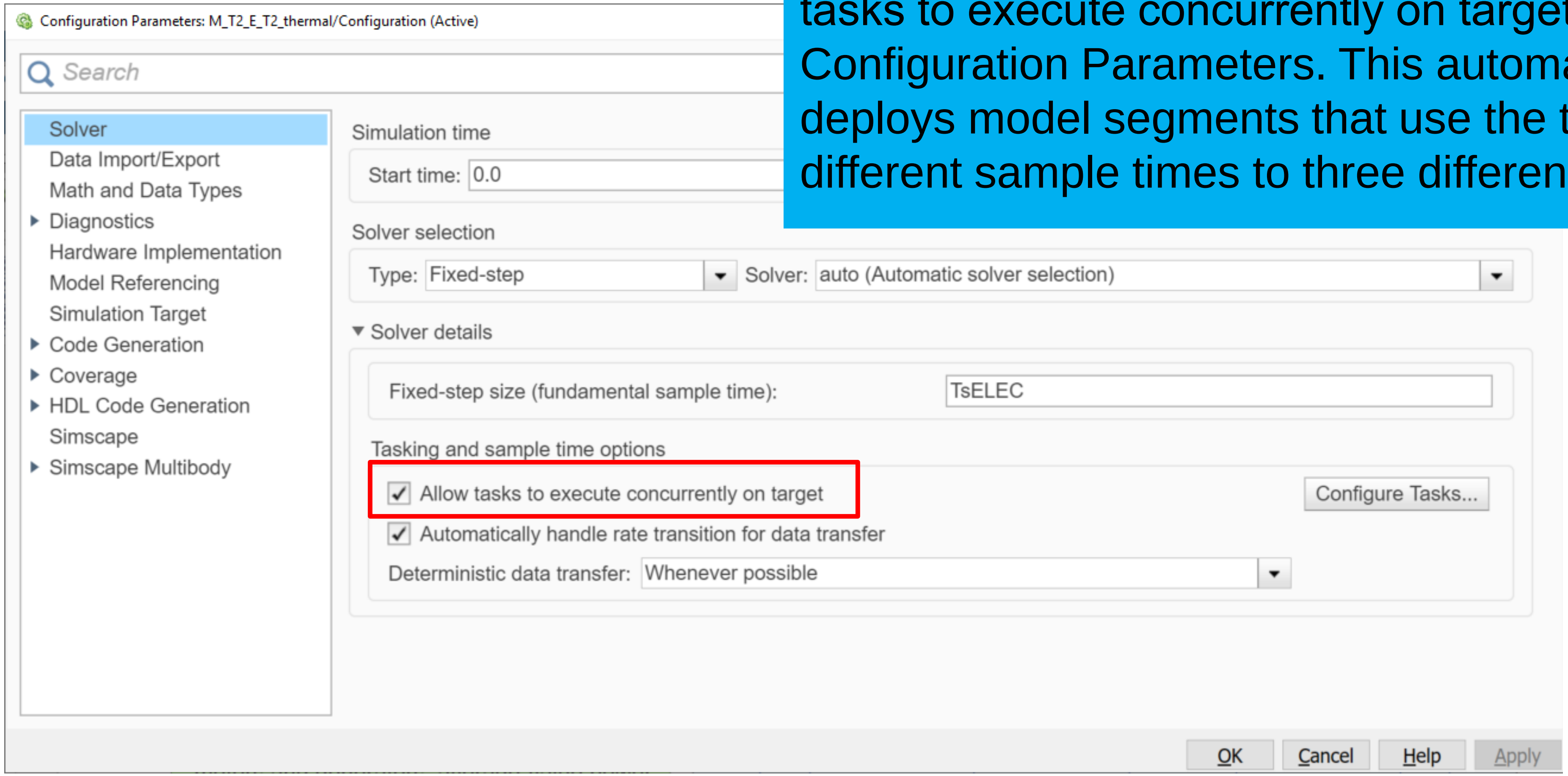


Using Different Sample-Times for Different Networks

- The more-electric aircraft model has three sample-times for the physical networks,
  - TsELEC for the electrical system
  - TsMECH for the Brayton-cycle gas turbines
  - TsTHERMAL for the thermal and thermal fluid systems

Real-Time Simulation using SLRT and Speedgoat

To take advantage of concurrent execution on multiple cores on Speedgoat, we select 'Allow tasks to execute concurrently on target' in Configuration Parameters. This automatically deploys model segments that use the three different sample times to three different cores.



Real-Time Simulation using SLRT and Speedgoat

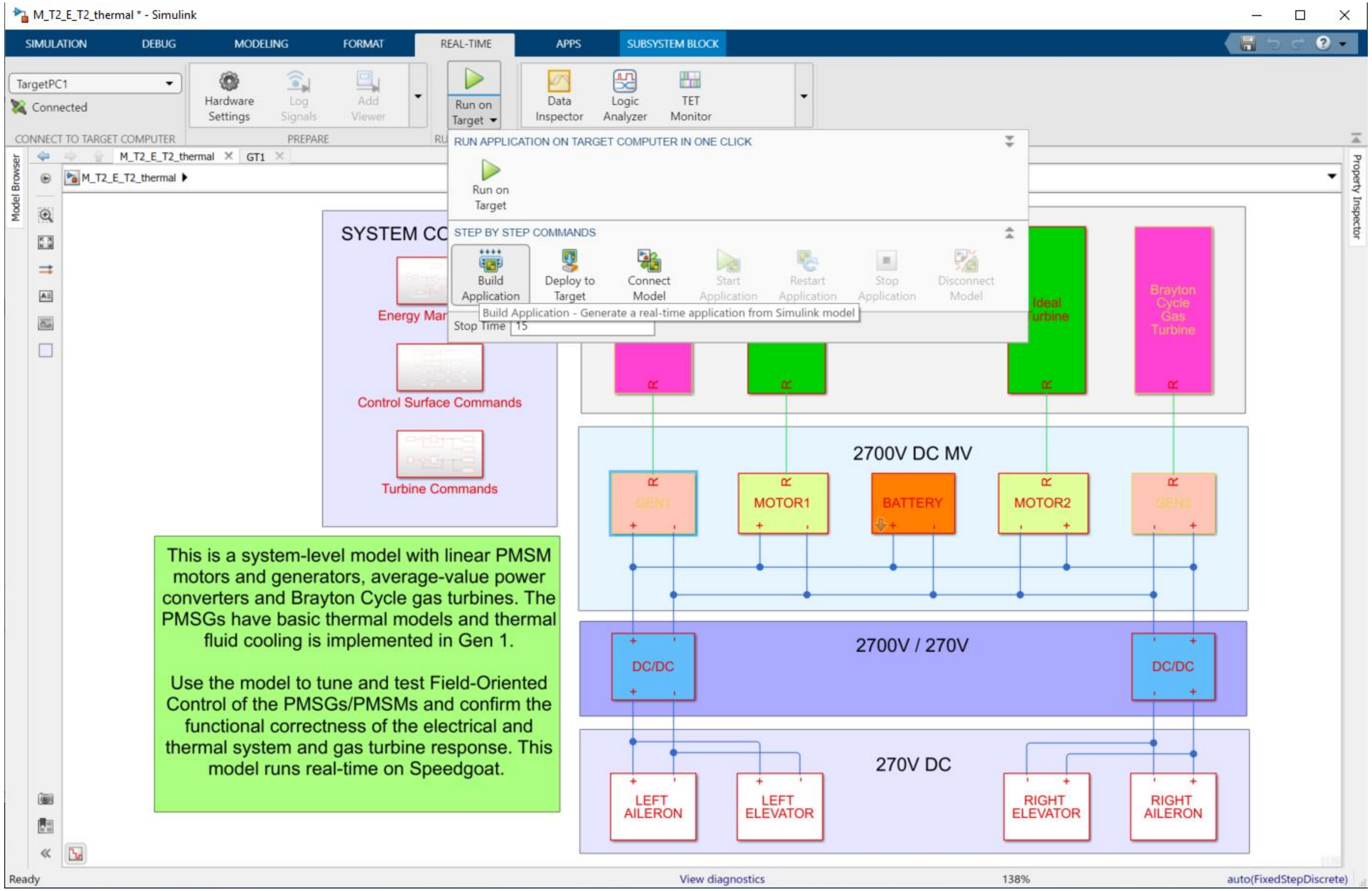
- To run the model on Speedgoat, we select the ‘Real-Time’ tab and build and deploy the model.

- Next, create an slrt object,

  >> tg = slrt;

- You can now run the simulation on Speedgoat

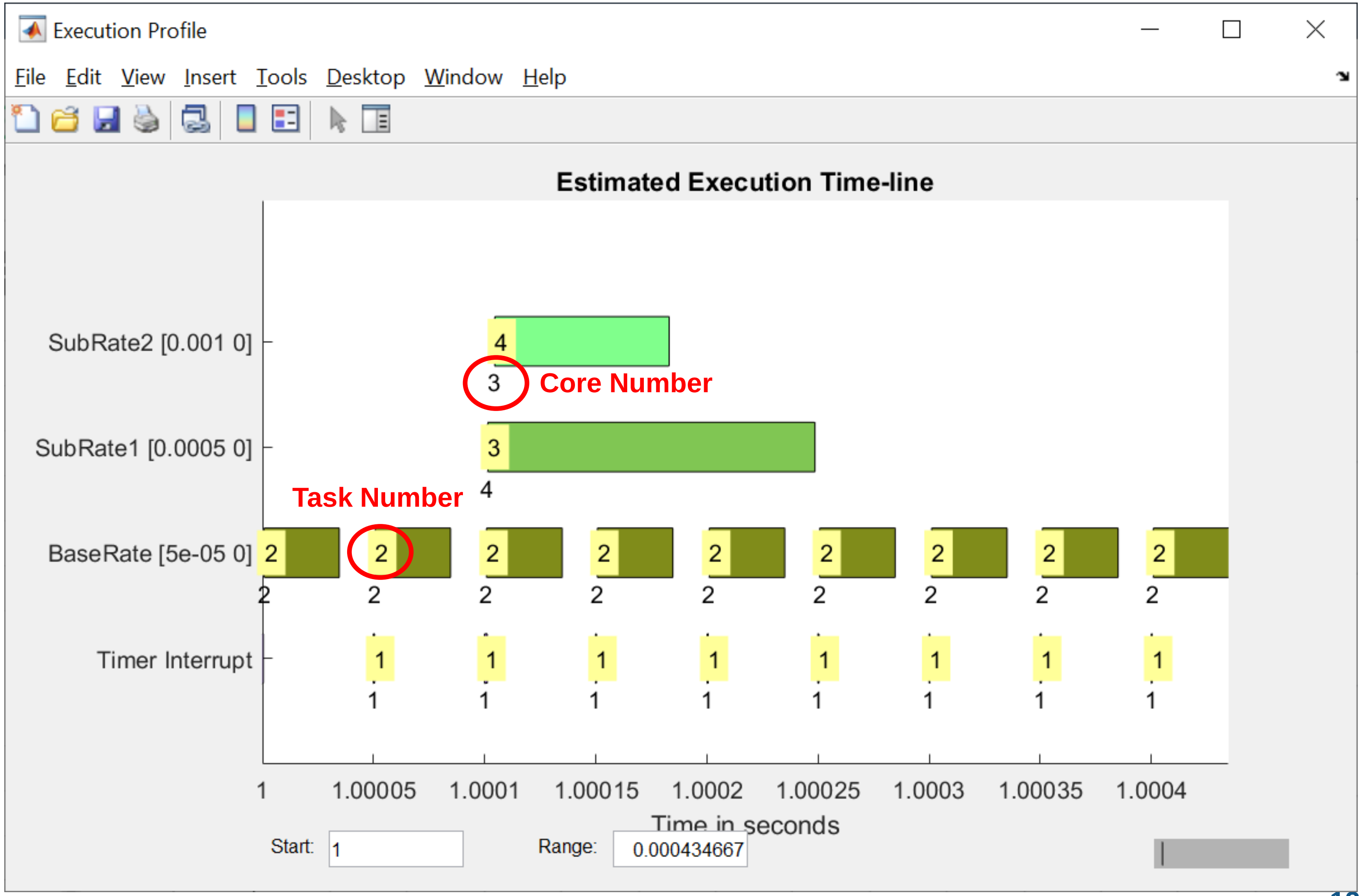
  >> tg.start



Real-Time Simulation using SLRT and Speedgoat

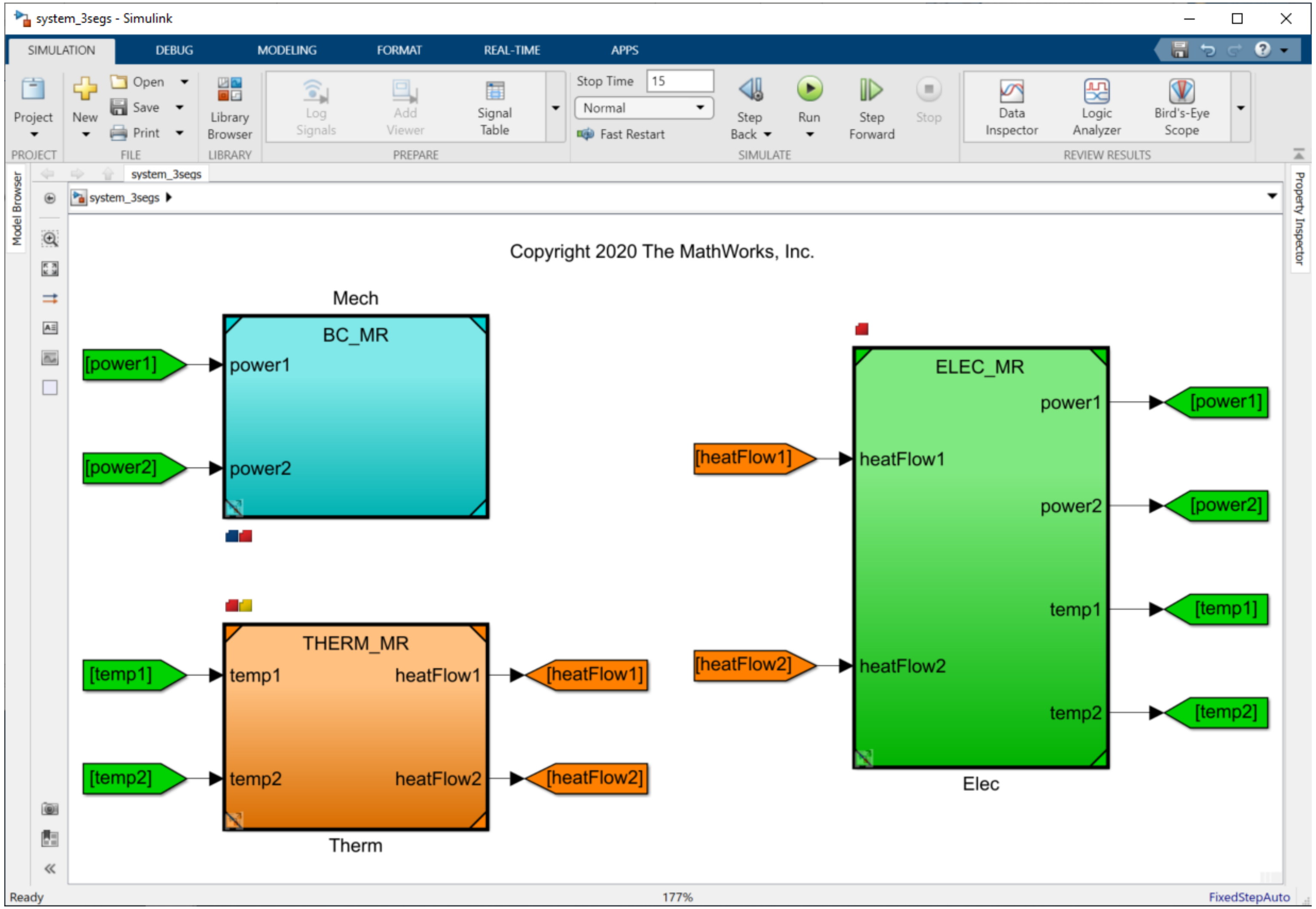
- Use the profiler to see the task-execution-time (TET) at different rates

```
>> tg.startProfiler;
>> tg.start;
>> profData = tg.getProfilerData;
>> profData.plot
```



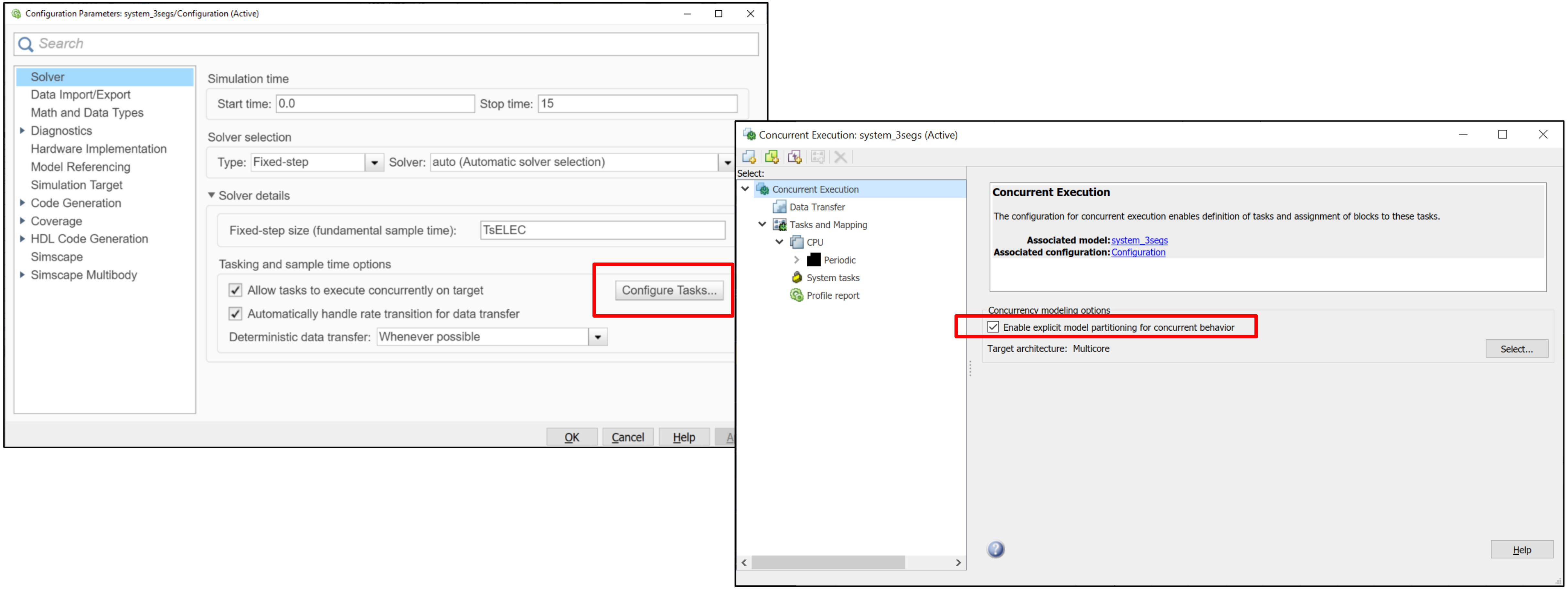
Real-Time Simulation using SLRT and Speedgoat

- For explicit (user-defined) task partitioning , we first architect the model using Model Reference.



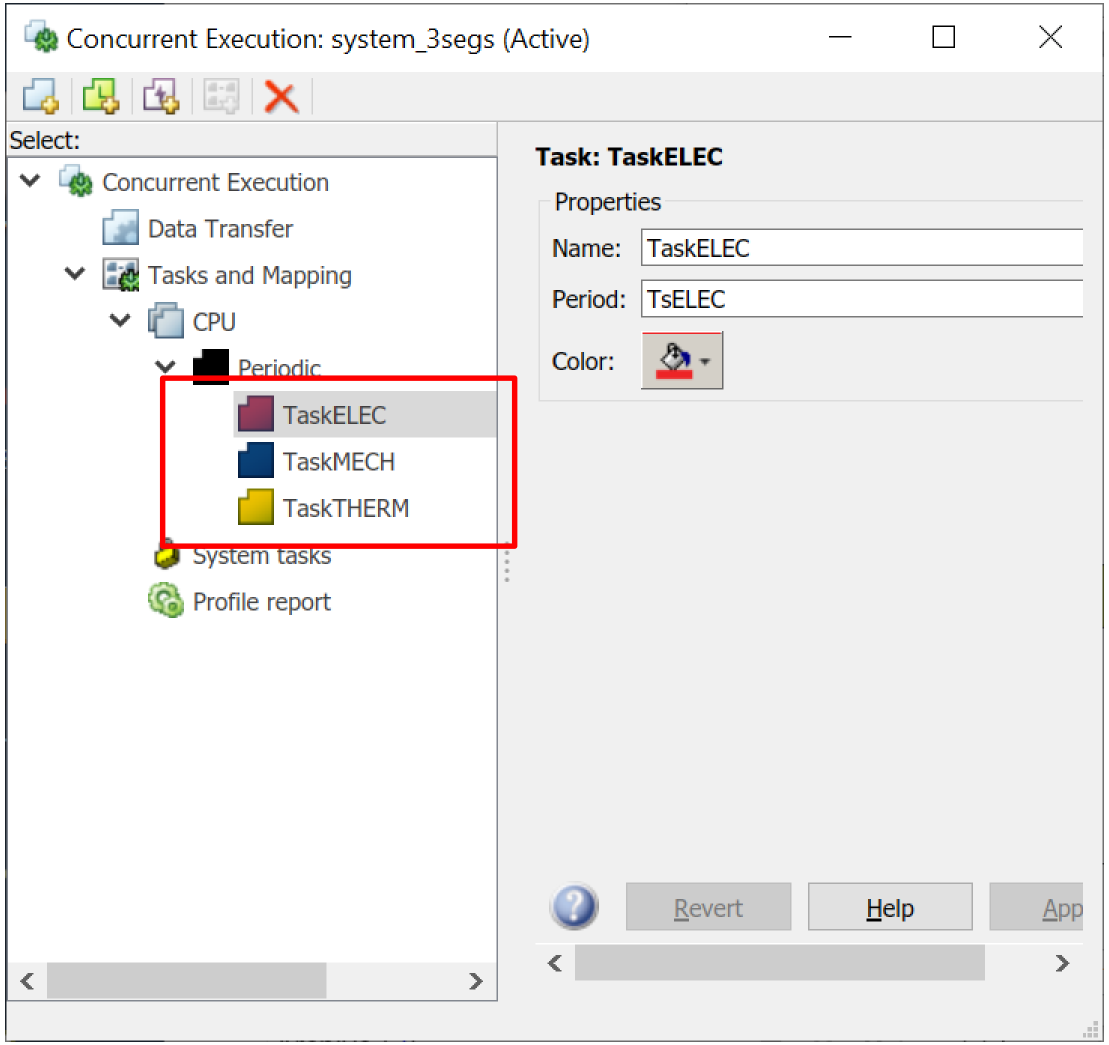
Real-Time Simulation using SLRT and Speedgoat

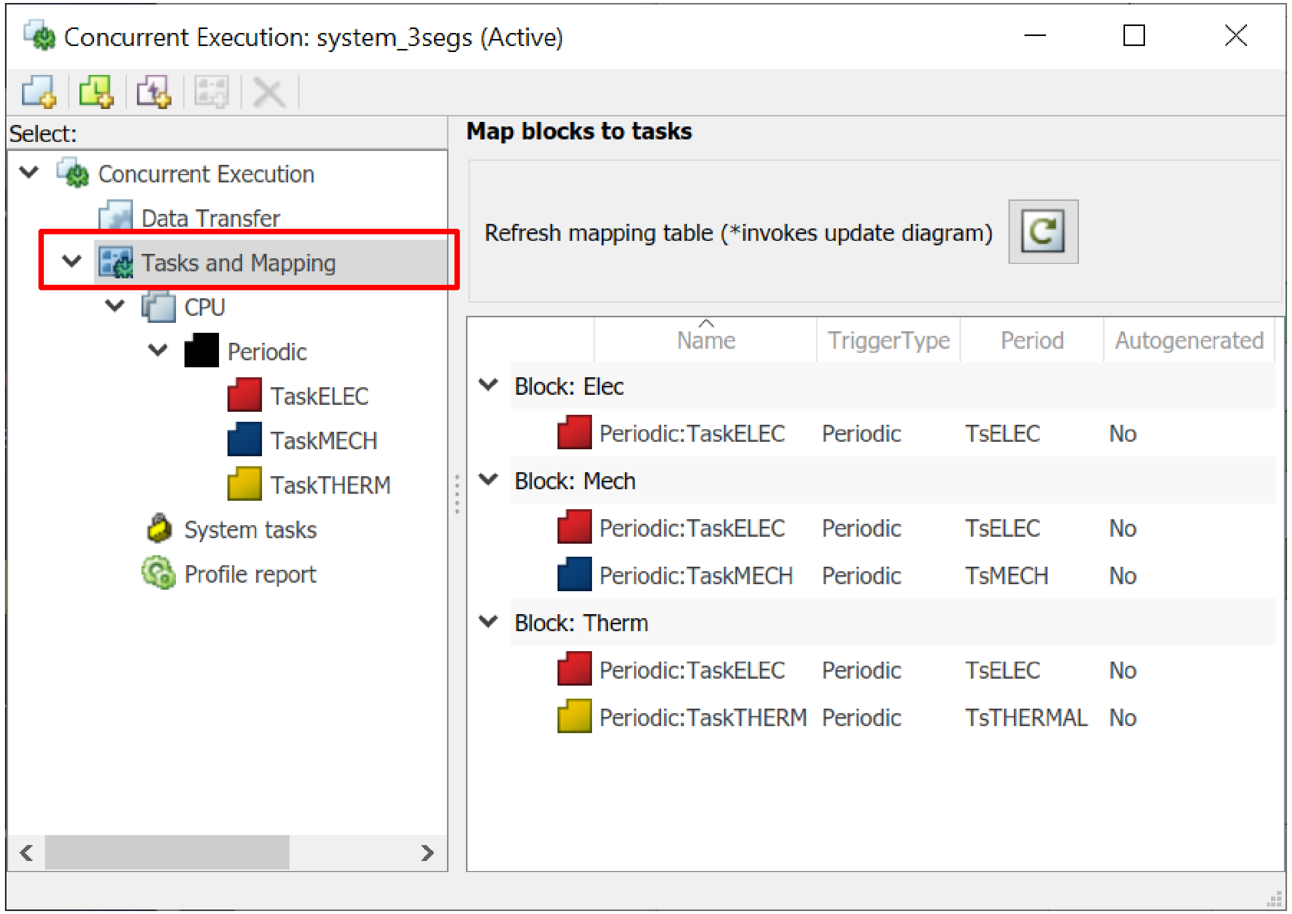
- Next, we select ‘configure tasks’ in ‘configuration parameters’, and select ‘enable explicit model partitioning for concurrent behavior’.



Real-Time Simulation using SLRT and Speedgoat

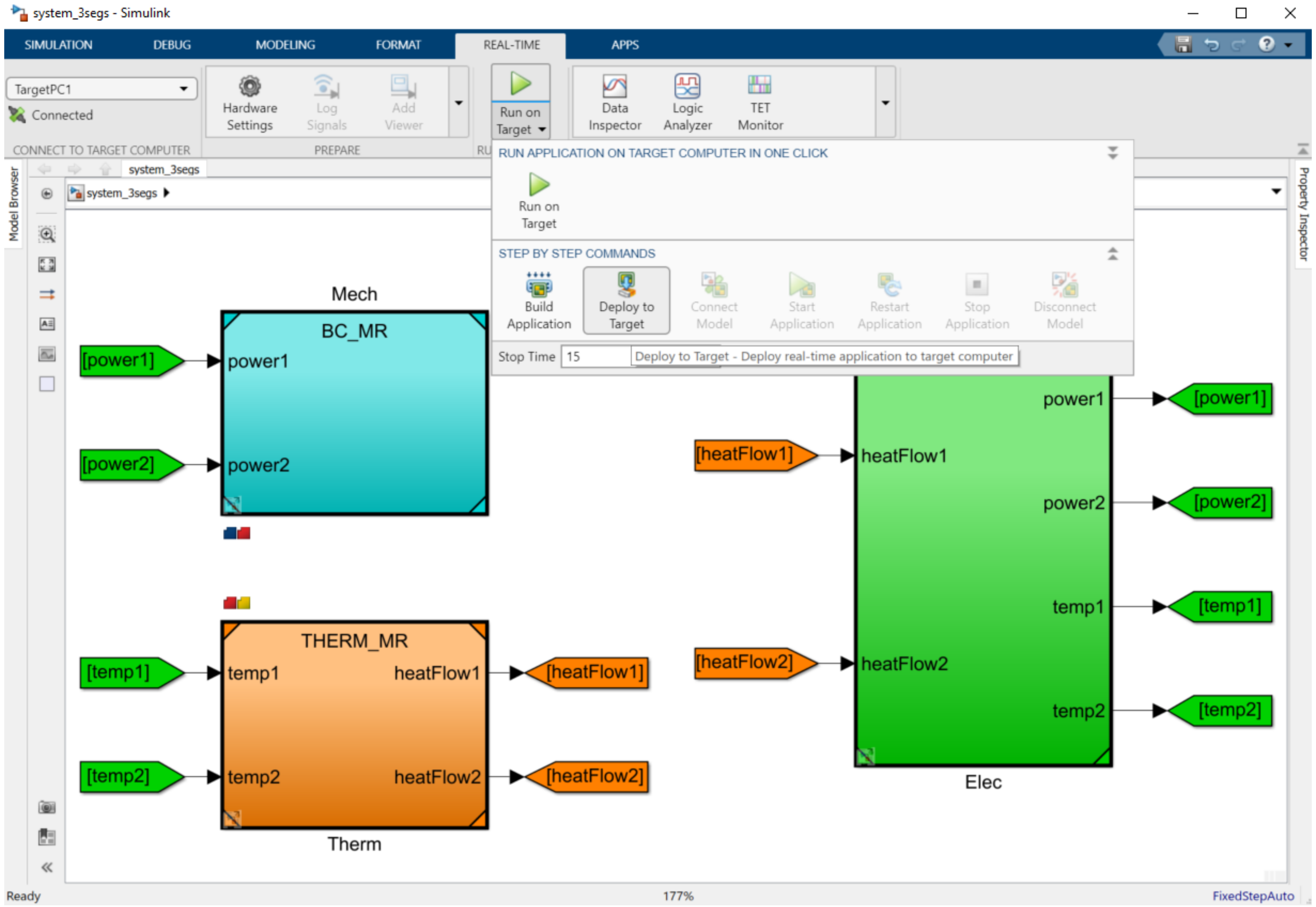
- Next, we define three periodic tasks, and map the tasks to the model reference blocks. Note that TaskELEC is the fastest rate, and needs to be set in each block.





Real-Time Simulation using SLRT and Speedgoat

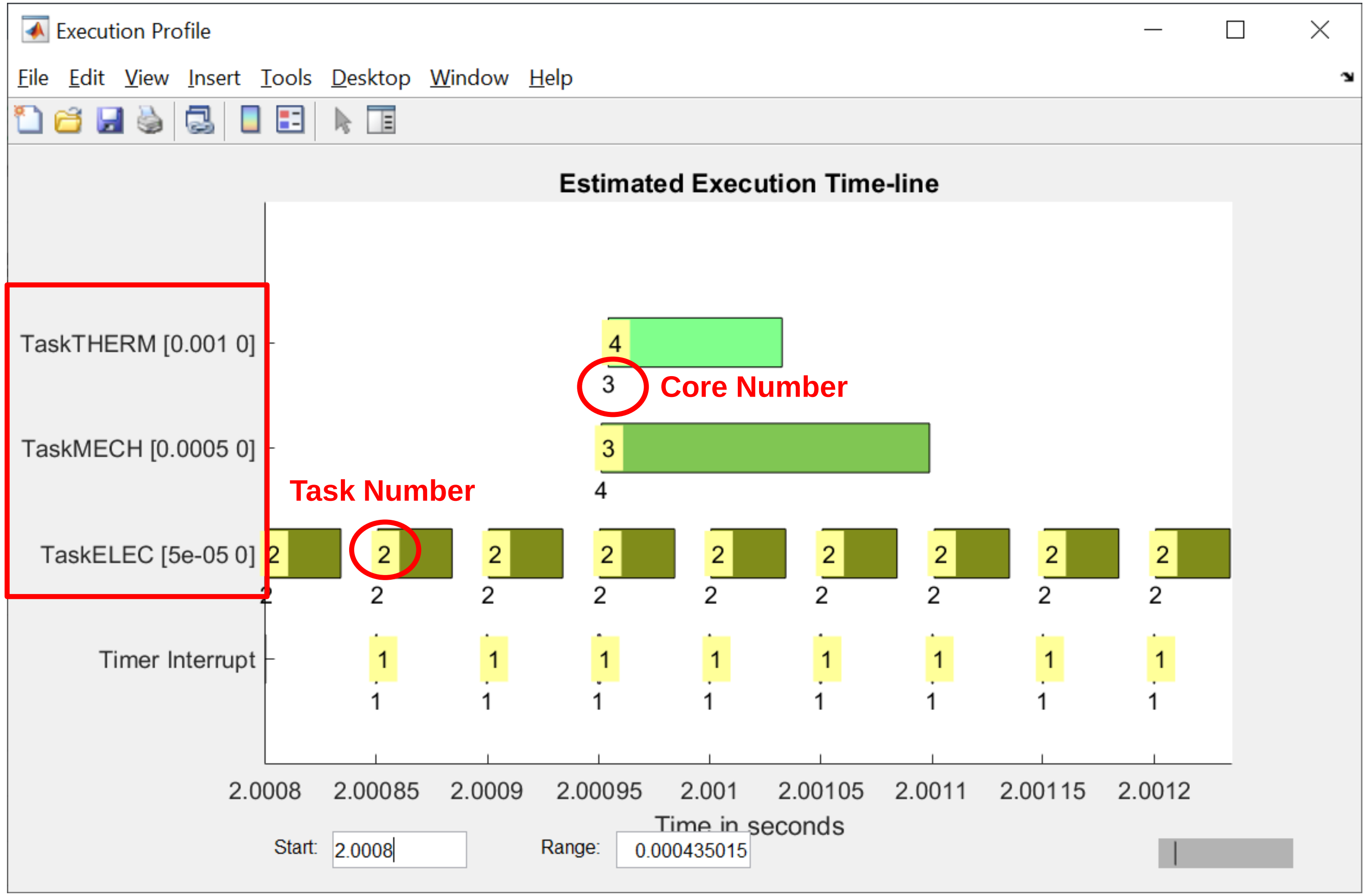
- To run the model on Speedgoat, we select the ‘Real-Time’ tab and build and deploy the model.
- Next, create an slrt object,
  >> tg = slrt;
- You can now run the simulation on Speedgoat
  >> tg.start



Real-Time Simulation using SLRT and Speedgoat

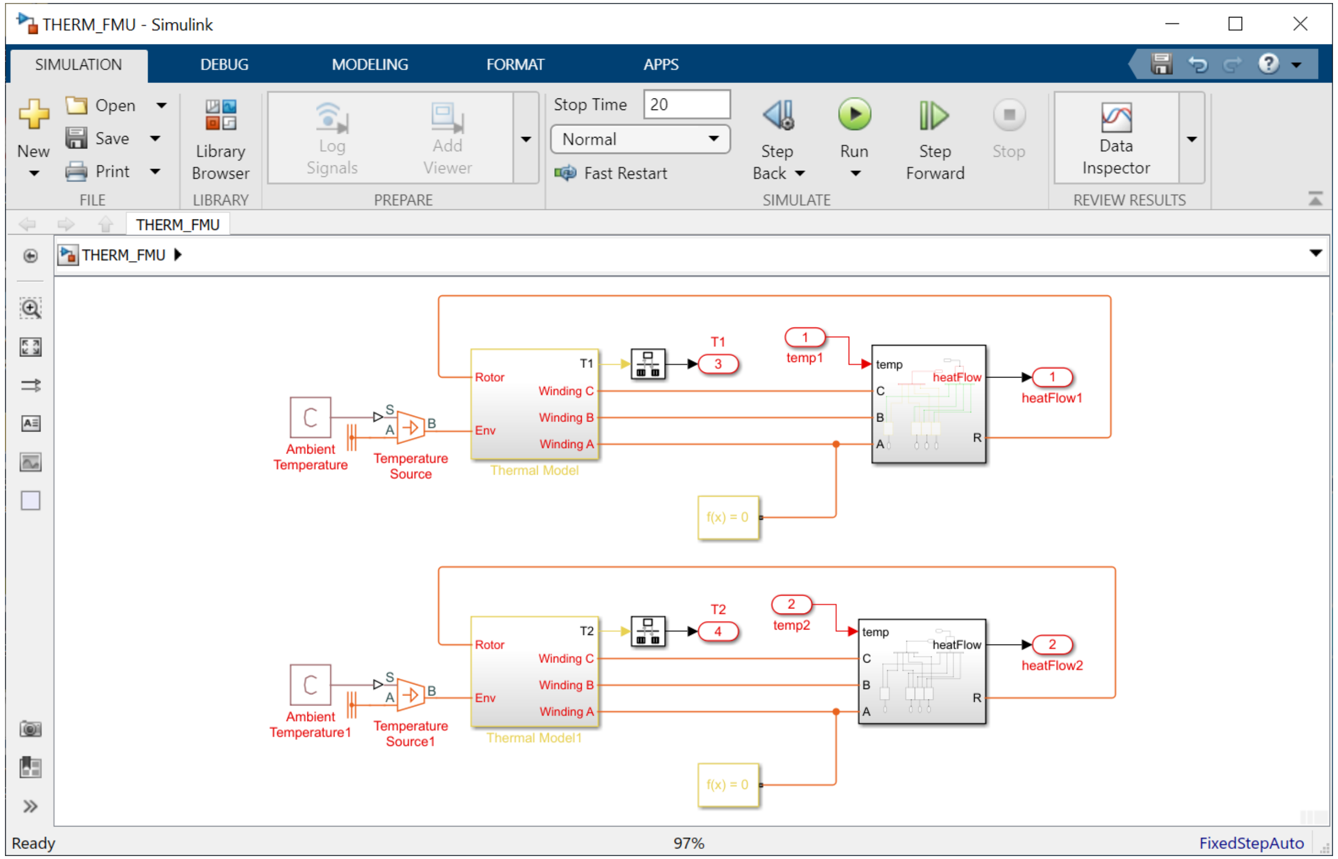
- Use the profiler to see the task-execution-time (TET) at different rates

```
>> tg.startProfiler;
>> tg.start;
>> profData = tg.getProfilerData;
>> profData.plot
```



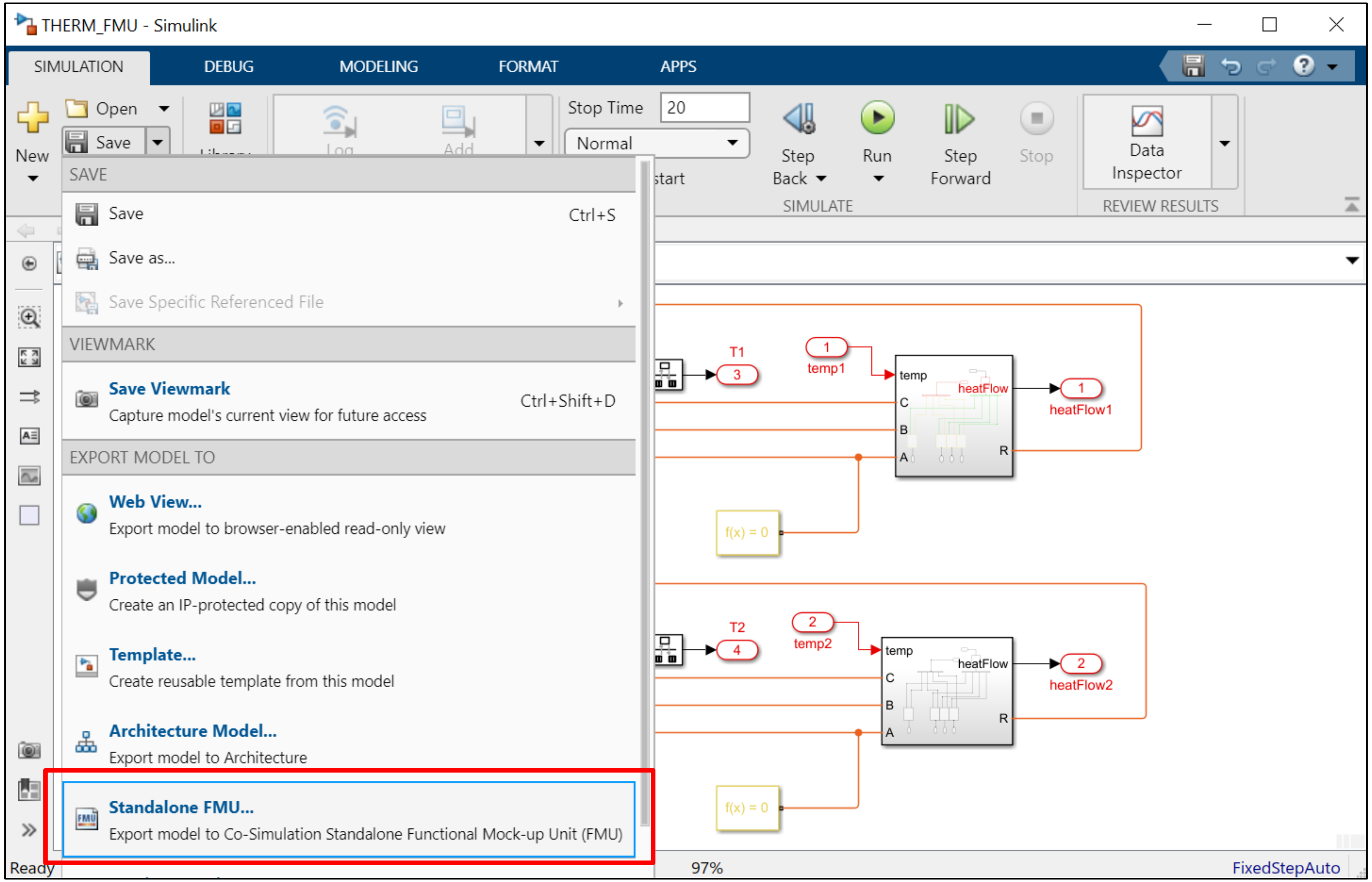
Creating an FMU Standalone model

- With Simulink Compiler, FMU standalone components can be generated
- First, create a model that contains the system you want to generate an FMU component for. In this case, we choose the thermal system.



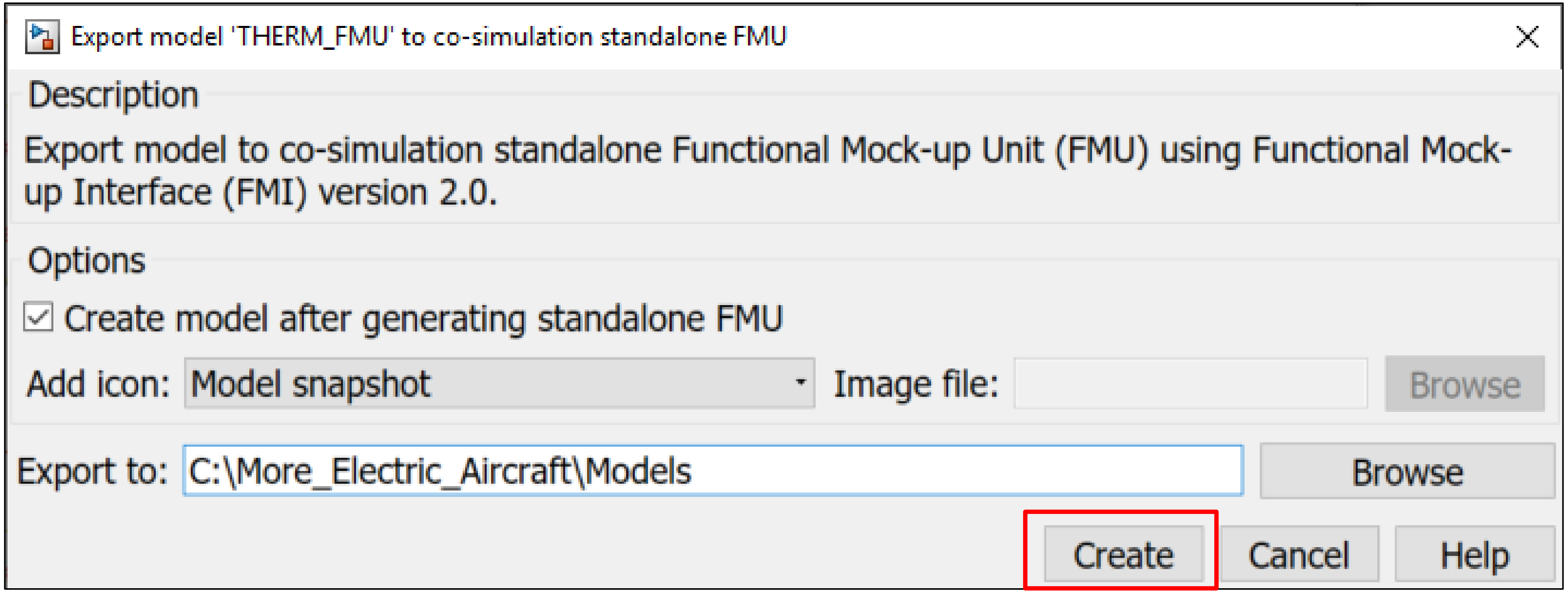
Creating an FMU Standalone model

- Next, select ‘Save’ and select ‘Standalone FMU’ on the dropdown.



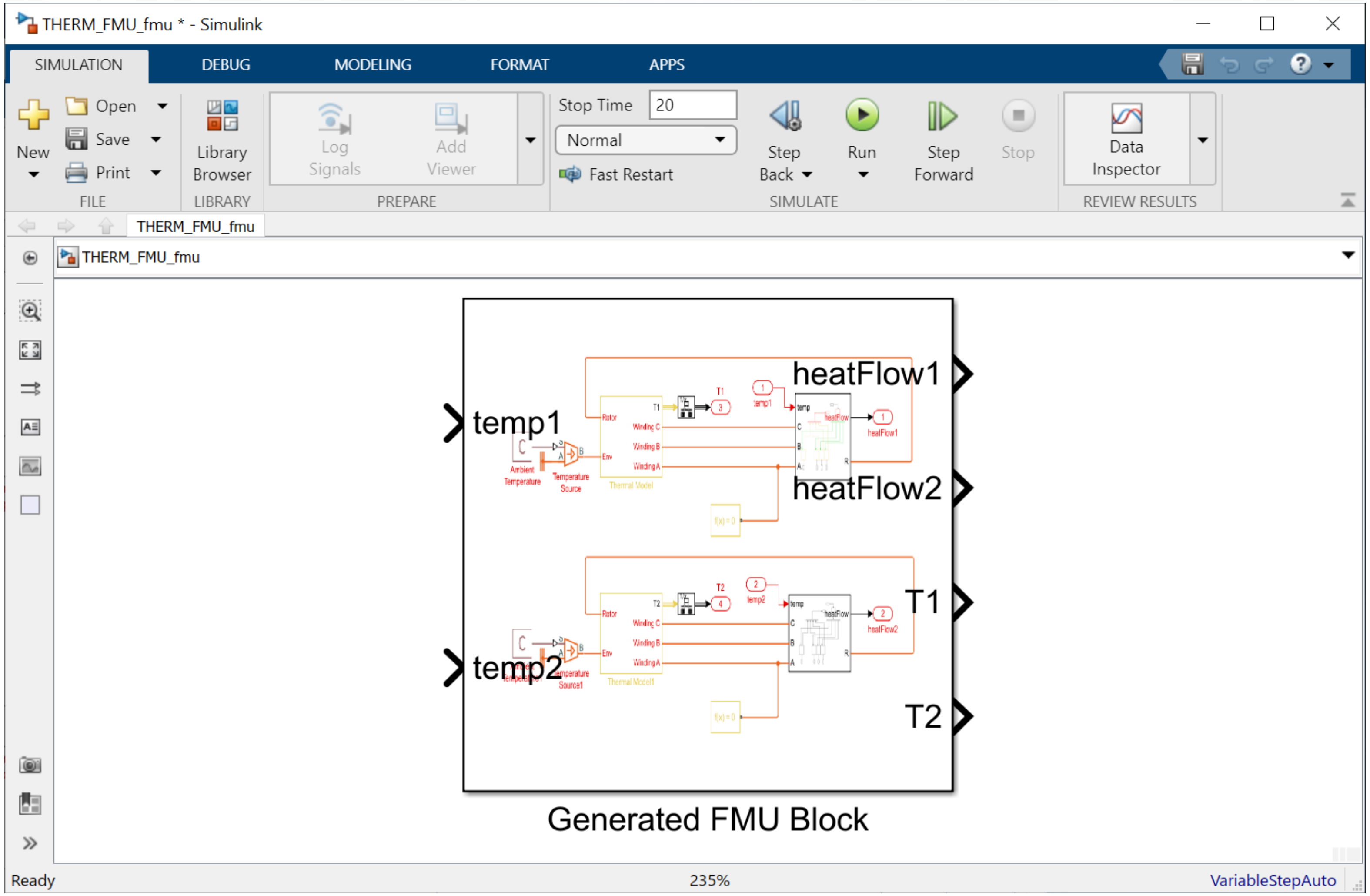
Creating an FMU Standalone model

- Next, select ‘Create’ on the FMU user interface.



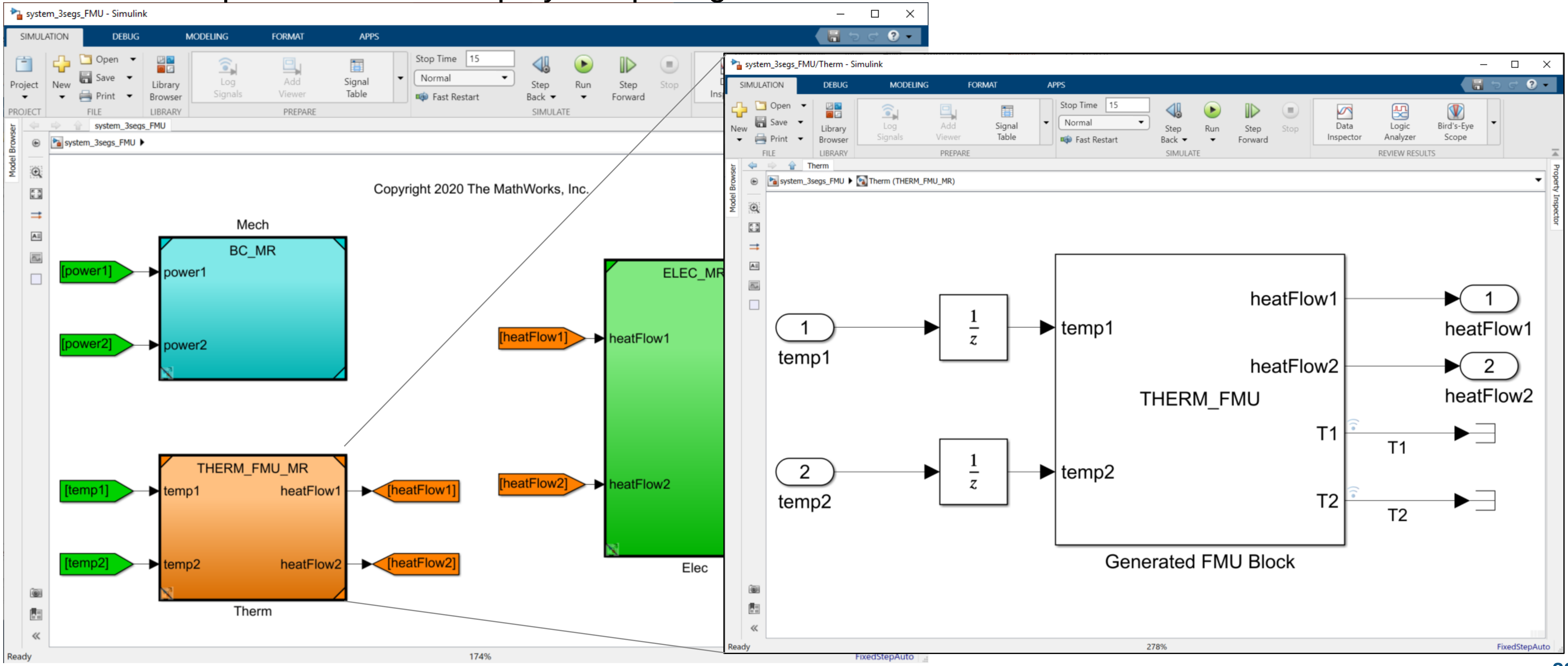
Creating an FMU Standalone model

- Once the FMU component is built, it is placed in a separate model.



Creating an FMU Standalone model

- Next, place the FMU component in the full system. Note this is a desktop-only simulation – FMU components do not deploy to Speedgoat.



Creating an FMU Standalone model

- Simulate and observe results