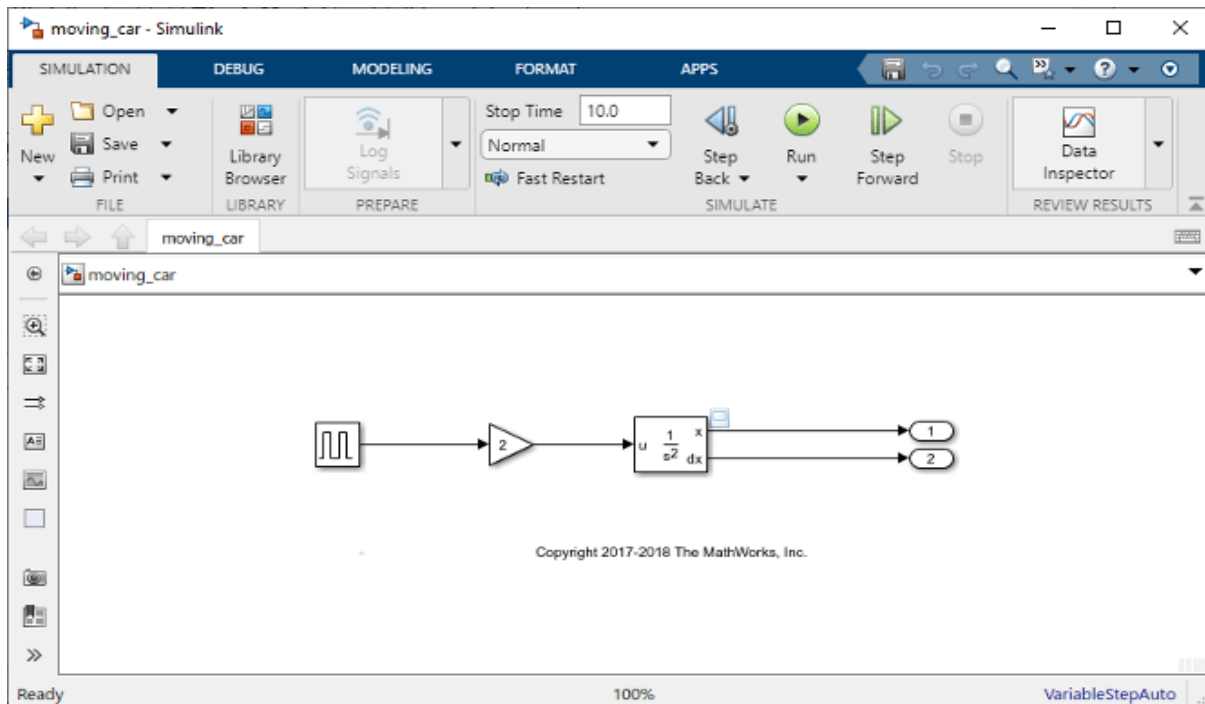



MathWorks C++ Design Question

Simulink® is a graphical modeling and simulation environment for dynamic systems. You can create block diagrams, where blocks represent parts of a system. A block can represent a physical component, a small system, or a function. A Simulink block defines a mathematical relationship between its input and output.



The figure above shows a simple Simulink model for a moving car. More details can be found [here](#).

In the figure, the element  is a gain block, which multiplies its input by 2 (hence '2' on the block). Simulink provides a library of blocks, each block with different functional behavior. Simulink users can create their own blocks with user specified functionality.

Blocks in the above model are connected through port connections, denoted by solid arrows in the figure. A source block feeds data to its destination block(s) using these port connections. More details about Simulink block diagrams can be found [here](#). Simulink users can also control functional behavior of blocks using block parameters.

As a model increases in size and complexity, you can simplify it by grouping blocks into subsystems. A subsystem is a set of blocks that you group into a single Subsystem block. More details about subsystems can be found [here](#).

A typical Simulink user workflow consists of

1. creating a new model in the editor,
2. adding blocks to the model,
3. connecting the blocks using port connections,
4. (optional) grouping a set of blocks inside a subsystem,
5. (optional) setting up block parameters,
6. compile the model, and
7. either simulate the model or generate code from it.

Users can add arbitrary number of blocks to the model. There is a fixed number of ports for a given type of block and the user cannot add new ports. Similarly, block parameters on a given type of block cannot be added or deleted. Only their values can be changed before compilation.

As part of the Simulink core architecture team, we are responsible for providing a set of C++ classes and associated API (Application Programming Interface) functions that Simulink developers can use in Simulink compiler development. Key requirements from Simulink compile developers include:

1. Query the model state during compilation. A few examples queries:
 - a. Find all the root blocks in the model. A root block is not enclosed in any subsystem block.
 - b. Find parent subsystem for a given block. A parent subsystem is the immediate subsystem block that encloses the given block.
 - c. Find all children blocks of a given subsystem block. A child block is any block immediately enclosed by the subsystem.
 - d. Find all *descendant* blocks of a given subsystem block. Let us call the input subsystem block *ssblk*. A descendant block can be a child of any descendant subsystem block *ssblk*.
 - e. Given a model (or a subsystem), find all the gain blocks inside that model.
 - f. Given a source block, transitively find all the destination blocks it reaches through port connections.
2. Transform the model. A transformation alters the model state. Transformations can
 - a. add a block between two ports,
 - b. delete a block and join its input and output ports,
 - c. Replace a block with another block.

These transformations should also work for subsystem blocks.

Model state queries are much more frequent than transformations during model compilation.

Please design a set of classes and associated API that Simulink feature developers can use to **query** the model state during compilation. Write C++ code and any test cases that implement your design. You could, for example, implement one or more classes that represent the model in the above figure and then in the test case, write a for loop that iterates over all the gain blocks in the model. Please explain your choices such as the classes, data-structures, C++ features you use in your design.