**Writing Exercise Notes**

Constant inputs are commonly used for flags that control how an algorithm executes and values that specify the sizes or types of data.

If you know that your primary inputs will not change at run time, overhead can be reduced in the generated code by specifying that the primary inputs are constant values.

Define constant inputs at the command line using the -args option of the codegen command:

-args {coder.Constant(constant-input)}

Here is an example.

Write a MATLAB function, myadd.m

function y = mcadd(u,v) %#codegen

y = u + v;

mcadd has two inputs, u and v. In this example, the user will specify that v is a constant with value 42.

Create a configuration object for MEX code generation.

cfg = coder.config('mex)

When generating code, the user specifies that the first input has the same type as the number 2. The user uses coder.Constant to specify that the second input will have constant value 42.

codegen -config cfg -args {2, coder.Constant(42)} mcadd

Call the MEX function giving u a value of 4. By default, you will have to provide a value for v when you call the MEX function and the value has to equal the constant value that you specified when you generated the MEX function

mcadd\_mex(2, 42)

ans =

46

You can control whether a generated MEX function signature will include constant inputs. If you will be using the same test file to run the original MATLAB® function and the MEX function, then the MEX function signature must contain the constant inputs. You can also control whether the run-time values of the constant inputs must match the compile-time values. Checking that the values match can slow down execution speed.

The configuration object property ConstantInputs controls whether you will have to provide a value for the input that is a constant when you call the MEX function. If ConstantInputs is 'CheckValues' (this is the default), the constant input must be provided when you call the MEX function and the value that you provide must be the same as the constant value.

mcadd\_mex(4, 42)

If ConstantInputs is 'IgnoreValues', the constant input must be provided when you call the MEX function but the value does not have to be the same as the constant value.

mcadd\_mex(4, 142)

If ConstantInputs is 'Remove', the constant value does not have to be provided when you call the function.

mcadd\_mex(4)

If you want to generate a MEX function so that you will not have to provide the constant value when you call the function, set ConstantInputs to 'Remove'

Create a code configuration object for MEX code generation.

cfg = coder.config('mex')

Configure ConstantInputs so that the MEX function does not include the constant input.

cfg.ConstantInputs = 'Remove';

Generate the MEX function for mcadd\_mex so that u has the type of the number 2 and v has constant value 42.

codegen -config cfg -args {2, coder.Constant(42)} mcadd

Call mcadd\_mex . Provide a value for u (make it 3). Don’t provide a value for v.

mcadd\_mex(3)

ans =

45