**Matlab Codegen Tutorial**

Juan Rojas

Nov 2012

**Building the MEX Function**

1. Specify size of input vector
   * load position.mat
   * z = position(1:2,1);
2. Generate the C code
   * codegen -report kalman02.m -args {z}
3. Run the Generated MEX function
   * Use a test file to load all that you need before calling the function
   * Use the “coder.runTest” function with this file
   * coder.runTest('test02,'kalman02')

**Building the C Code**

1. Place steps in bullet 1 for “Building the MEX Function” in a script “build02.m” such that:

% Load the position vector

load position.mat  
 % Get the first vector in the position matrix to use as an example input  
 z = position(1:2,1);

1. Generate C code only  
   codegen -c -d build02 -config coder.config('lib') -report kalman02.m -args {z}

* codegen opens kalman02 and translates the matlaba code into C source
* -c tells codegen to build bode only without compiling the code to an object file. This allows you to easily compare matlab and C++ code
* - config coder.config('lib') tells coegen to generate C code suitable for static libraries instead of generating the default MEX function.
* -d tells codegen to put the output in build02
* -report tells codegen to generate a report useful for debugging
* -args tells codegen to compile the file kalman01.m using the clss, size, and complexity of the sample input param z.

1. Run the build script bulid02

Key Points to Remember

### **Key Points to Remember**

* Back up your MATLAB code before you modify it.
* Decide on a naming convention for your files and save interim versions frequently. For example, this tutorial uses a two-digit suffix to differentiate the various versions of the filter algorithm.
* Use build scripts to build your files.
* Use test scripts to separate the pre- and post-processing from the core algorithm.
* Generate a MEX function before generating C code. Use this MEX function to simulate your algorithm in MATLAB to validate its operation and check for run-time errors.
* Use the -args option to specify input parameters at the command line.
* Use the -report option to create a code generation report.
* Use coder.typeof to specify variable-size inputs.
* Use the code generation configuration object (coder.config) to specify parameters for standalone C code generation.