生成用于图像锐化的HDL代码

此示例显示如何使用Vision HDL Toolbox™实现基于FPGA的图像锐化模块。

Vision HDL Toolbox提供图像和视频处理算法，旨在用VHDL和Verilog（使用HDL Coder™）生成可读，可合成的代码。在FPGA上运行时生成的HDL代码（例如，Xilinx XC7Z045）可以以每秒60帧的速度处理1920x1080全分辨率图像。

此示例显示如何使用Vision HDL Toolbox生成锐化模糊图像的HDL代码。由于Vision HDL Toolbox算法可用作MATLAB®Systemobjects™和Simulink®块，因此可以从MATLAB或Simulink生成HDL代码。此示例显示了两个工作流程。

此示例使用：
图像处理工具箱
Simulink的
Vision HDL工具箱

在MATLAB中试一试

FPGA目标设计的工作流程是：

1.创建表示设计目标的行为模型;

2.使用适用于FPGA的算法，接口和数据类型复制设计，并支持HDL代码生成。

3.模拟两种设计并比较结果，以确认HDL优化设计符合目标。

4.从步骤2中创建的设计生成HDL代码。

对于MATLAB中的步骤2和3，您必须具有MATLAB，Vision HDL Toolbox和Fixed-Point Designer™。在Simulink中，您需要Simulink，Vision HDL Toolbox和定点设计器。在这两种情况下，您必须具有HDL编码器才能生成HDL代码。

行为模型

输入图像imgBlur显示在下图中的左侧。在右侧，使用图像处理工具箱™功能锐化图像`imfilter`。

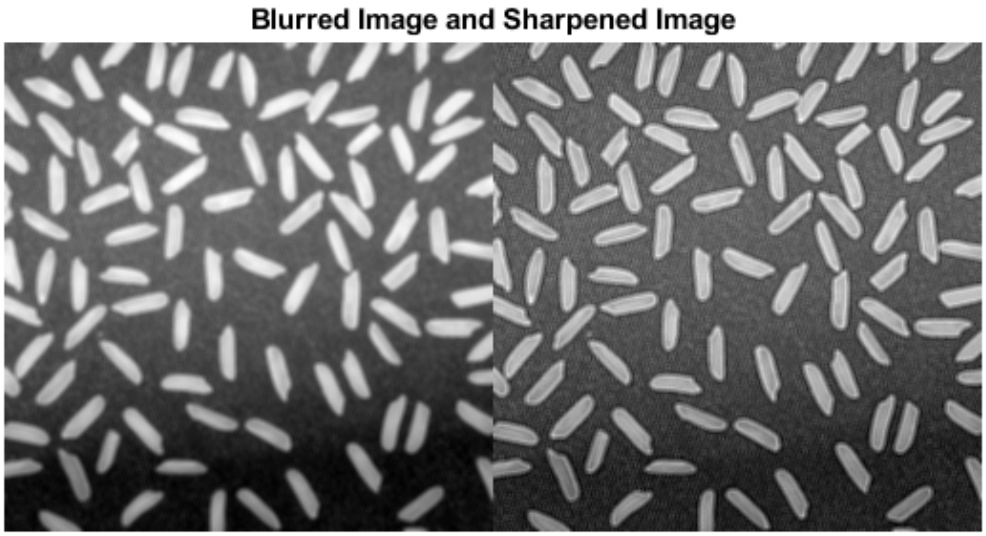
模拟时间打印为基准，以便将来进行比较。

```
imgBlur = imread ('riceblurred.png') ;
sharpCoeff = [0 0 0; 0 1 0; 0 0 0] -fspecial ('laplacian', 0.2) ;

f = @ () imfilter (imgBlur, sharpCoeff, 'symmetric') ;
fprintf ('经过时间是%.6f秒。\ n', timeit (f) ) ;

imgSharp = imfilter (imgBlur, sharpCoeff, 'symmetric') ;
数字
imshowpair (imgBlur, imgSharp, 'montage')
标题 ('模糊图像和锐化图像')
```

经过的时间是`0.000399`秒。



HDL优化设计考虑因素

需要进行三项关键更改才能生成HDL代码。

- **使用HDL友好算法：** 图像处理工具箱中的功能不支持HDL代码生成。Vision HDL Toolbox提供专为高效HDL实现而设计的图像和视频处理算法。您可以使用MATLAB中的System对象和Simulink中的块来从这些算法生成HDL代码。这个例子中提供了两个工作流程。要设计基于FPGA的模块，请将图像处理工具箱中的功能替换为Vision HDL Toolbox中与HDL兼容的功能。此示例使用MATLAB中`imfilter`的`visionhdl.ImageFilterSystem`对象或`Image Filter`Simulink中的块替换行为模型。
- **使用流式像素接口：** 图像处理工具箱模型中的高级抽象功能。它们执行全帧处理，一次在一个图像帧上操作。然而，FPGA和ASIC实现执行像素流处理，一次在一个图像像素上操作。Vision HDL Toolbox块和System对象使用流式像素界面。`visionhdl.FrameToPixels`在MATLAB或中使用System对象`Frame To Pixels`在Simulink中阻止将全帧图像或视频转换为像素流。流式像素接口包括指示每个像素在帧中的位置的控制信号。在像素邻域上操作的算法使用内部存储器来存储最少数量的行。Vision HDL Toolbox提供流式像素接口和自动存储器实现，以解决针对FPGA和ASIC时的常见设计问题。有关Vision HDL Toolbox中System对象使用的流式像素协议的更多信息，请参阅流式像素界面。
- **使用定点数据表示：** Image Processing Toolbox中的函数在浮点或整数域中执行视频处理算法。Vision HDL Toolbox的系统对象和块需要定点数据来生成HDL代码，以针对FPGA和ASIC。将设计转换为定点会引入量化误差。因此，HDL友好模型可能会生成与行为模型略有不同的输出。对于大多数应用，容差范围内的小量化误差是可以接受的。您可以调整定点设置以满足您的要求。

在此示例中，我们使用静态图像作为源。该模型还能够处理连续视频输入。

从MATLAB生成HDL代码

要从MATLAB生成HDL，您的代码需要分为两个文件：测试平台和设计。设计文件用于在FPGA或ASIC中实现算法。测试平台文件将输入数据提供给设计文件并接收设计输出。

第1步：创建设计文件

函数ImageSharpeningHDLDesign.m接受像素流和由五个控制信号组成的控制结构，并返回修改的像素流和控制结构。

在此示例中，设计包含System对象`visionhdl.ImageFilter`。它是该`imfilter`功能的HDL友好型对应物。使用与之相同的系数和填充方法对其进行配置`imfilter`。

```
function [pixOut, ctrlOut] = ImageSharpeningHDLDesign (pixIn, ctrlIn)
 %ImageSharpeningHDLDesign使用
Vision HDL工具箱中的像素流%System对象实现算法

%Copyright 2015 The MathWorks, Inc。

% # codegen
persistent sharpeningFilter;
if isempty (sharpeningFilter)
    sharpCoeff = [0 0 0; 0 1 0; 0 0 0] -fspecial ('laplacian', 0.2) ;
    sharpeningFilter = visionhdl.ImageFilter (...
    'Coefficients', sharpCoeff, ...
    'PaddingMethod', 'Symmetric', ...
    'CoefficientsDataType', 'Custom', ...
    'CustomCoefficientsDataType', numerictype (1,16,  12) ) ;
结束

[pixOut, ctrlOut] = step (sharpeningFilter, pixIn, ctrlIn) ;
```

第2步：创建测试台文件

测试台ImageSharpeningHDLTestBench.m读取模糊图像。该frm2pix对象将完整图像帧转换为像素流和控制结构。测试台调用设计函数一次ImageSharpeningHDLDesign处理一个像素。在处理整个像素流之后，pix2frm将输出像素流转换为全帧图像。测试台将输出图像与参考输出进行比较imgSharp。

```
...
[pixInVec, ctrlInVec] = step (frm2pix, imgBlur) ;
对于 p = 1: numPixPerFrm
    [pixOutVec (p) , ctrlOutVec (p) ] = ImageSharpeningHDLDesign (pixInVec (p) ,
结束
imgOut = step (pix2frm, pixOutVec, ctrlOutVec) ;

%比较结果
imgDiff = imabsdiff (imgSharp, imgOut) ;
fprintf ('对应像素之间的最大差异是%d。\ n', max (imgDiff (:)) ) ;
fprintf ('总共%d个像素不同。\ n', nnz (imgDiff) ) ;
...
```

第3步：模拟设计和验证结果

在生成HDL代码之前使用测试台模拟设计，以确保没有运行时错误。

```
ImageSharpeningHDLTestBench
```

```
对应像素之间的最大差异为1。
总共41248个像素是不同的。

模拟完成了368.091661秒。
```

测试台显示比较结果和模拟所花费的时间。由于量化误差和舍入误差，在总共256 * 256 = 65536像素中，imgOut的38554不同imgSharp。然而，强度的最大差异是1.在0到255的比例上，这种差异在视觉上是不明显的。

通过将MATLAB中的仿真时间与行为模型的仿真时间进行比较可以看出，像素流协议引入了显着的开销。您可以使用MATLAB Coder™加速MATLAB中的像素流模拟。请参阅使用MATLAB Coder加速像素流设计。

第4步：生成HDL代码

一旦您对FPGA目标模型的结果感到满意，您就可以使用HDL Coder从设计中生成HDL代码。您可以在HDL仿真器中运行生成的HDL代码，或将其加载到FPGA中并在物理系统中运行。

确保设计和测试平台文件位于相同的可写目录中。要生成HDL代码，请使用以下命令：

```
hdlcfg = coder.config ('hdl') ;
hdlcfg.TestBenchName = 'ImageSharpeningHDLTestBench' ;
hdlcfg.TargetLanguage = 'Verilog' ;
hdlcfg.GenerateHDLTestBench = false;
代码生成-config  hdlcfg  ImageSharpeningHDLDesign
```

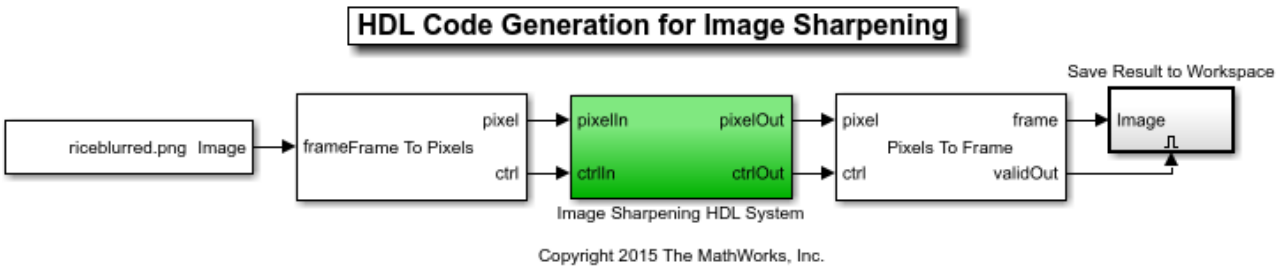
有关创建和配置MATLAB到HDL项目的教程，请参阅MATLAB入门到HDL工作流程。

从Simulink生成HDL代码

第1步：创建HDL优化模型

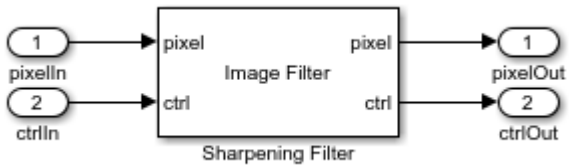
该ImageSharpeningHDLModel模型如下所示。

```
modelname = 'ImageSharpeningHDLModel' ;
open_system (MODELNAME) ;
set_param (modelname, 'Open', 'on') ;
```

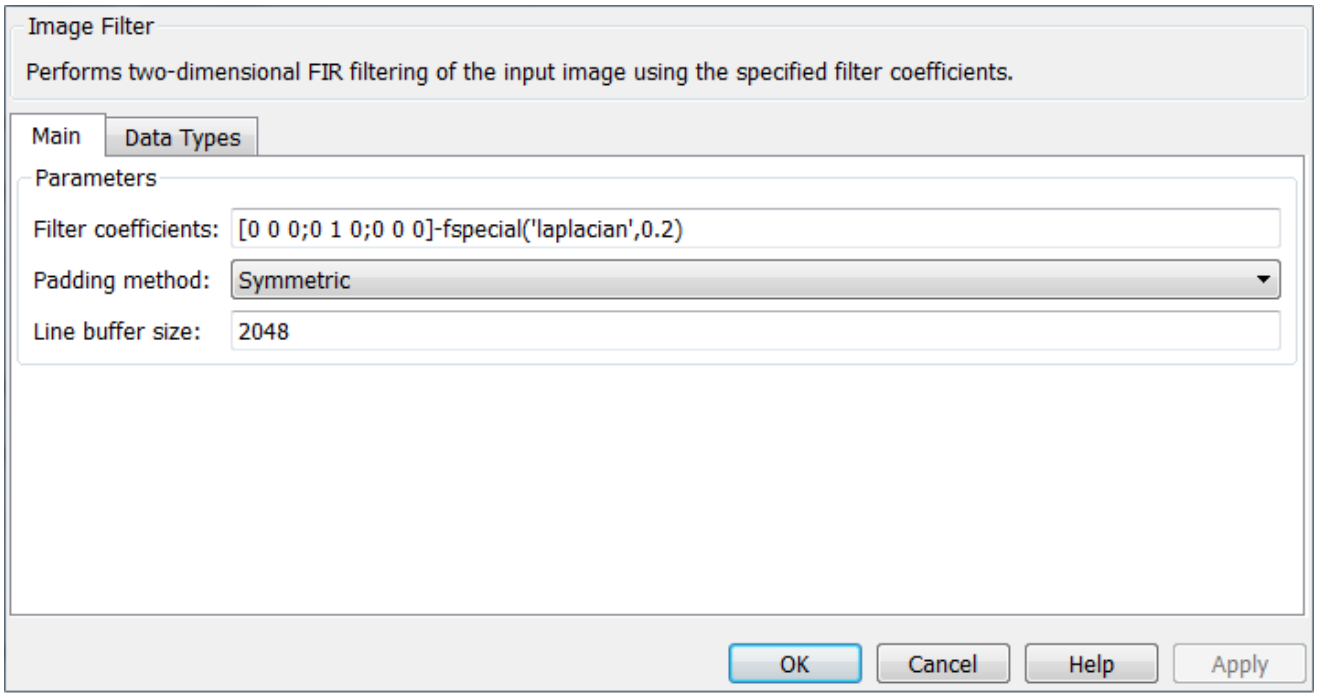


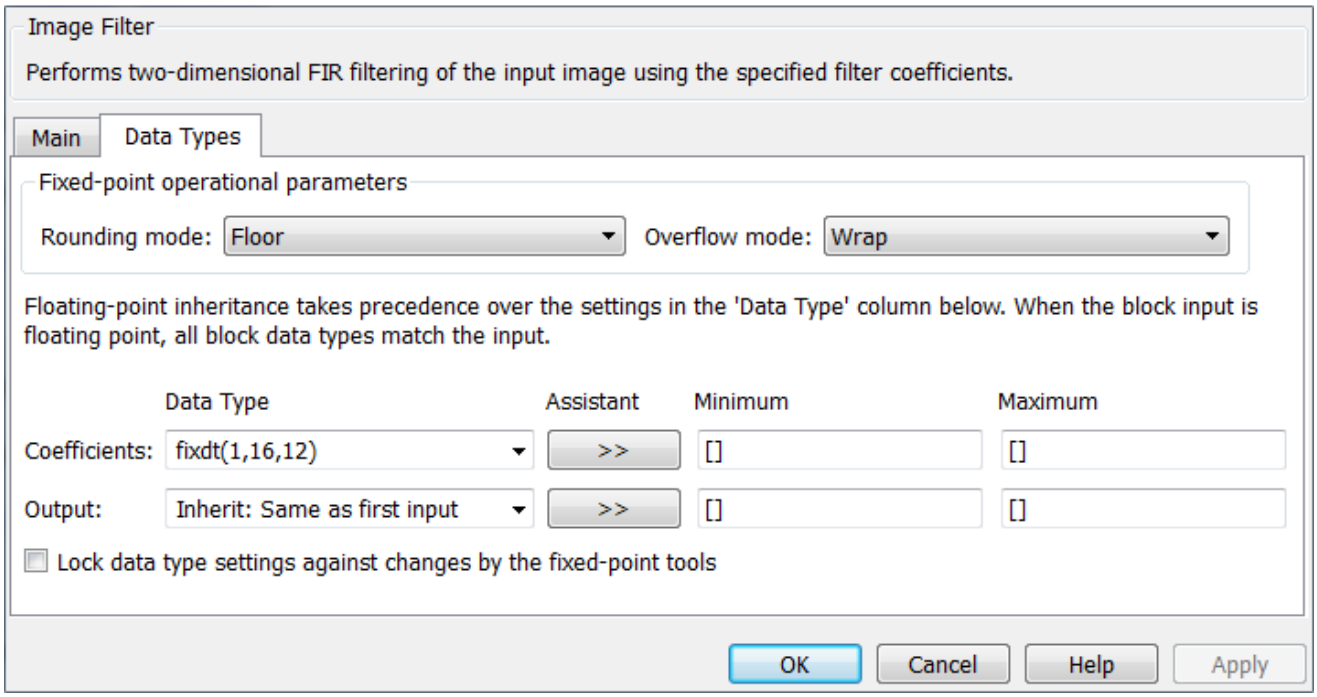
该模型读入模糊图像。帧到像素块将全帧图像转换为像素流，像素到帧块将像素流转换回全帧图像。图像锐化HDL系统包含一个图像过滤器块，它是imfilter行为模型中呈现的功能的Vision HDL Toolbox中与HDL友好的对应物。

```
set_param (modelname, 'Open', 'off') ;
set_param ([modelname '/ Image Sharpening HDL System' ], 'Open', 'on') ;
```



使用与行为模型中相同的锐化系数和填充方法配置图像滤镜块，如下面的蒙版所示。





第2步：模拟设计和验证结果

```
抽搐
SIM (MODELNAME) ;
TOC
```

经过的时间是20.761860秒。

Simulink利用C代码生成来加速仿真。因此，它比MATLAB仿真快得多，尽管仍比行为模型慢。

模拟在工作空间中创建一个名为imgOut的新变量。使用以下命令imgOut与imSharp行为模型生成的命令进行比较。

```
imgDiff = imabsdiff (imgSharp, imgOut) ;
fprintf ('对应像素之间的最大差异是%d。\ n', max (imgDiff (:)) ) ;
fprintf ('总共%d个像素不同。\ n', nnz (imgDiff) ) ;
```

对应像素之间的最大差异为1。

总共41248个像素是不同的。

由于量化误差和舍入误差，总共256 * 256 = 65536像素，38554 `imgOut`不同`imgSharp`。然而，强度的最大差异是1.在0到255的比例上，这种差异在视觉上是不明显的。（这与“从MATLAB生成HDL代码”部分的步骤3中提供的解释相同。）

第3步：生成HDL代码

一旦您对FPGA目标模型的结果感到满意，您就可以使用HDL Coder从设计中生成HDL代码。您可以在HDL仿真器中运行生成的HDL代码，或将其加载到FPGA中并在物理系统中运行。

使用以下命令从Image Sharpening HDL System生成HDL代码：

```
makehdl('ImageSharpeningHDLModel / Image Sharpening HDL System')
```

```
set_param([modelname '/ Image Sharpening HDL System' ], 'Open', 'off') ;
close_system(MODELNAME, 0) ;全部
关闭;
```