In this assignment you will write an OpenSPL application that implements the convolution discussed in class (see page 2 of the CUDA part 5 lecture notes.)

\[(f \ast g)(x) = \int f(x - t)g(t)dt\]

Include an implementation of the same calculations in your CPU code in order to verify your results.

Pass the weights for \(g()\) by using three scalars.

When writing your kernel, pass a datatype for scalars, input and output streams into the kernel constructor from the manager so that the kernel can be reused to process a stream of either \texttt{int} or \texttt{float} by changing nothing more than the datatype passed in from the manager. Do this by adding a parameter to the kernel constructor like this:

```java
protected a5Kernel(KernelParameters parameters, DFEType type)
```

...and then construct it from your manager like this:

```java
Kernel kernel = new a5Kernel(manager.makeKernelParameters(s_kernelName), dfeFloat(8,24));
```

Use the following for your weights:

\[.25, .5, .25\]

Note that the first and last values in your output stream will be calculated using non-existent values. Your DFE will run. But it will generate garbage for those undefined values. We will disregard this problem... for this assignment.

If you are interested in fixing this problem now, read ahead in Chapter nine of the Max-Compiler tutorial.

1 Compiling and Testing Your Program

Build and test your program using the VM as discussed in class. Name your project as discussed in class: \texttt{z88888-a05}.

2 How to Hand in Your Program

Hand in your program using the SVN repo as discussed in class and in the appendix of the lecture notes.