An FPGA API for VSIPL ++

Ben Cordes, Miriam Leeser, Joe Tarkoff
{bcordes, mleeser, jtarkoff}@ece.neu.edu
Dept. Electrical and Computer Engineering
Northeastern University, Boston, MA

VSIPL++ is a language designed for developers of high performance embedded computing applications. Many such applications benefit from a mix of hardware and software resources being used to solve these problems. FPGAs are a popular hardware platform due to their flexibility and high performance on kernels that exhibit a great deal of parallelism.

To date, there is no support for developing implementations that exploit software and FPGA hardware together. To fill this gap, we are developing an API to allow FPGA hardware to communicate with VSIPL++ software running on a host processor. We plan to support COTS FPGA hardware. This includes FPGA boards available from Annapolis Microsystems, Cray, Mercury Computers and SGI. All of these boards require similar functional calls such as initialization, board open, programming the FPGA, and board close. This board overhead will be wrapped into the VSIPL++ API, and will use common software calls independent of the particular board being used. The API will also support transmission and reception of data between the FPGA board and the VSIPL++ application.

Concurrency

VSIPL++ must be aware that a function is being run in hardware rather than in software. To achieve maximum acceleration, hardware and software functions should run concurrently. Support needs to be provided for making the software aware that data generated from the hardware implementation are available and for streaming data to the hardware implementation.

Fixed and Floating Point Data Types

FPGA implementations achieve speedup through parallelism. Such parallelism is enhanced by using operands with a small bit width, and by using fixed point operations. VSIPL++ only supports floating point data types, and has no notion of variable bit widths. To support a mix of hardware and software, both fixed and floating point data must be supported. We intend to address this by doing floating to fixed point conversion (and vice versa), in hardware. `Float2fix` and `fix2float` modules are available as part of the Variable Precision Floating Point Library developed at Northeastern University [1]. These will be integrated with the hardware components on the FPGA so that the interface to VSIPL++ is all floating point.

Example: Pulse Compression

To illustrate these concepts and our API, we are implementing pulse compression in VSIPL++ and in a mix of VSIPL++ and FGPA hardware. The basic building blocks of pulse compression are shown in Figure 1. Sample inputs and outputs are shown in Figure 2. Note that one of the input signals is the delayed version (echo) of the other.

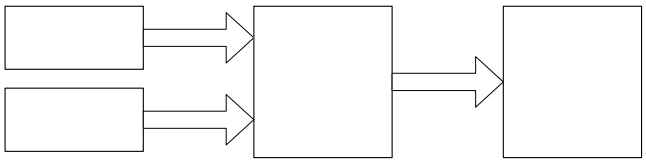

Figure 1: Pulse Compression Block Diagram

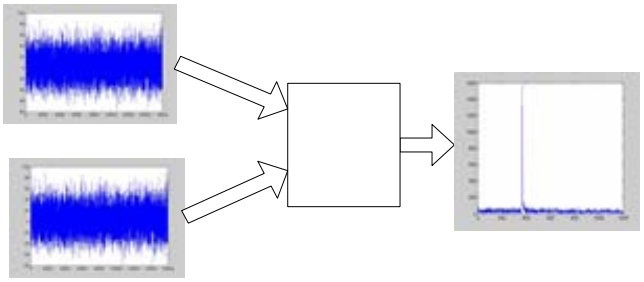

Figure 2: Sample Inputs and Outputs for Pulse Compression

Our first implementation will be completely in VSIPL++. Next we will move the FFT functionality to the FPGA and do the remainder of the computation in VSIPL++. We will illustrate the API, concurrent execution, and our handling of fixed point and floating point data types. We also plan to demonstrate the use of our API and hardware and software solutions on several different COTS platforms.

References

[1] Variable Precision Floating Point Library. Northeastern University. Available from: http://www.ece.neu.edu/groups/rpl/projects/floatingpoint/