# VPSDK : a portability library for extended arithmetic operations targetting a RISC-V Variable eXended Precision accelerator.

Jerome Fereyre[1], Alexandre Hoffmann[1]

[1]Univ. Grenoble Alpes, CEA, List
F-38000 Grenoble, France

**Abstract**

*We develop a RISC-V based accelerator called VXP (Variable eXtended Precision). It provides an efficient way to handle extended precision arithmetic operations. This helps to address convergence issues encountered using linear algebras solvers for scientific applications. In the current work we introduce a library called VPSDK. VPSDK is a general framework to develop application on this accelerated environment as well as in general purpose architectures.*

## Introduction

Linear algebra kernels, such as linear solvers or eigensolvers, are ubiquitous in both scientific and industrial applications. Theses kernels can be divided into two categories, direct solvers and iterative solvers. Direct solvers are usually faster but their memory requirement make them unusable for large problems. Iterative solvers, such as Krylov solvers, are thus the preferred choice for many industrial and scientific problems.

These methods are very sensitive to round-off errors, which leads to numerical instabilities, slower convergence, and, in some cases, divergence of the solver.

The convergence of iterative methods can be sped-up by using a *preconditioner*. However, simple preconditioners, such as Jacobi, often lead to limited speed-up. Sophisticated preconditioners, such as incomplete LU, lead to substential speedup, but are both computationally and memory intensive and require problem specific tuning.

Alternatively, using extended precision limits the impact of round-off errors and thus speed-up, and in some cases enable, the convergence of iterative methods. Figure 1 shows how numerical precision can speed-up the convergence of the BiConjugate Gradient (BiCG) method. This solution is easier and more scalable but the lack of hardware support introduces a very significant overhead. In our experiment a peak speed up of x800 may be provided by hardware supported scalar operations versus software variable precsion library such as the Multiple Precision Floating-Point Reliable (MPFR) library [1];

This motivated the development of the Variable eXtended Precision(VXP) (formerly called VRP) hardware accelerator. The VXP is based on the RISC-V Instruction Set Architecture (ISA), and provides hardware support for extended precision floating point

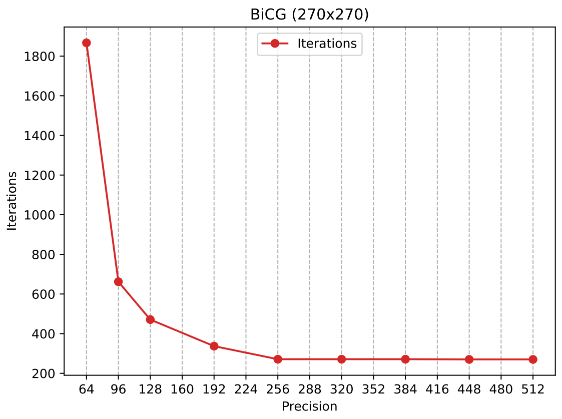

**Figure 1:** *BiConjugate gradient iteration count evolution when increasing computing precision*

numbers [2]. This support is provided through custom ISA extensions for loading, storing and manipulating extended precision numbers. This ISA implementation was developped by extended the CVA6 RISC-V core, adding a VPFPU which performs variable precision computation.

In order to use this extended ISA, an additional software layer is required. We provide a custom version of the standard BLAS libraries that supports variable precision hardware. We also provide a set of operators for manipulating variable precision scalars.

Three workflows are available. First, it is possible to compile code for RISC-V and to execute it on the VXP. Secondly, it is possible to compile code for RISC-V and to execute it on a standard Linux machine, on a modified RISC-V Spike model. Finally, it is possible to emulate variable precision using the GNU MPFR library. The three workflows are represented in Figure 2. Both the second and third workflows allow one to benefits variable precision even if the VXP hardware is not available.

## VPSDK overview and results

VPSDK is library composed by a set of C++ classes providing a common interface to support run in two environments. The first one uses MPFR as backend for variable precision support, and the second one will use the VXP accelerator.

This library defines a class called VPFloat that handles scalar variable precision numbers. This class provides operators for: simple arithmetic operations(+,-,*,/), variable assignments operators (=,+=,-=,*=,/=), and is used by additional functions power(`pow2`), square root (`sqrt`).

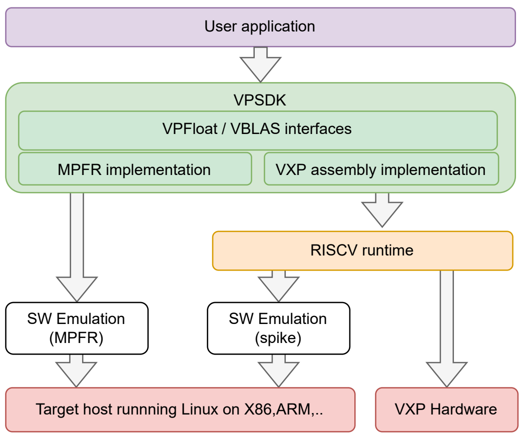

**Figure 2:** *VPSDK use*

A VPFloat object is declared with a triplet of integers: the size of the exponent, the total size of the variable precision number in memory and a stride used by hardware to deduce next VPFloat address in case of sequential declaration.

**Listing 1:** *VPFloat C code example*

```
# include <VPFloat.hpp>
using namespace VPFloatPackage;

void main() {
  // Matissa size is 128 bits
  int precision=128;
  // Exponent size is 7 bits
  int exponent_size=7;
  // Stride size is 1 vpfloat elements
  int stride_size = 1;
  int n = 10;

  int vpfloat_memory_size = precision
                          + exponent_size + stride;

  // Set the precision to use for further code
  VPFloatComputingEnvironment::set_precision(precision);

  // VPFloat array declaration
  VPFloatArray r_k( exponent_size,
                    vpfloat_memory_size,
                    stride_size,
                    n);

  // VPfloat scalar declaration
  VPFloat rs (exponent_size,
              vpfloat_memory_size, stride_size );

  rs = 10.0;

  for (int i = 1; i <= n; i++ ) {
      r_k[i-1] = rs / (double)i;
  }
}
```

Listing 1 presents an example of C/C++ code using the VPFloat++ interface

In addition to the scalar and array classes some BLAS primitives are available in an additional module called VBLAS. These functions that are available for the VXP, and they are also available for MPFR. Currently available functions are: `vgemvd, vscal, vcopy, vaxpy, vdot, vzero` and correspond to their BLAS counterparts.

Various linear algebra solvers such as Conjugate Gradient (CG), and BiConjugate Gradient (BiCG), were implemented using the VBLAS, making possible to run them on our VXP accelerated platform or on X86_64 platform with MPFR backend.

## Future

Future work involves the consolidation of our SDK as well as its extension to support the same features as other linear algebra packages.

This new version will allow us to address more general problems, like the Harmonic Wave Equation, on which we are currently working.

We also plan to add libquad support, in order to widen the the selection of variable precision software solutions and to compare them to the VXP solution.

This library will be made available as open source, as part of a TRISTAN European funded project.

## Acknowledgements

This work has been performed in the context of the EPI and TRISTAN projects. EPI has received funding from the European Union's Horizon research and innovation program under Grant Agreement EPI-SGA1: 826647. TRISTAN has received funding from the Key Digital Technologies Joint Undertaking (KDT JU) under Grant Agreement nr. 101095947.

## References

[1] Laurent Fousse et al. "MPFR: A Multiple-Precision Binary Floating-Point Library with Correct Rounding". In: *ACM Trans. Math. Softw.* 33.2 (June 2007), 13–es. ISSN: 0098-3500. DOI: `10.1145/1236463.1236468`. URL: `https://doi.org/10.1145/1236463.1236468`.

[2] Y. Durand et al. "Accelerating Variants of the Conjugate Gradient with the Variable Precision Processor". In: *2022 IEEE 29th Symposium on Computer Arithmetic (ARITH)*. Los Alamitos, CA, USA: IEEE Computer Society, Sept. 2022, pp. 51–57. DOI: `10.1109/ARITH54963.2022.00017`. URL: `https://doi.ieeecomputersociety.org/10.1109/ARITH54963.2022.00017`.