
ホーム > Workshop on Large-scale Parallel Numerical Computing Technology (LSPANC 2020 January)

# Workshop on Large-scale Parallel Numerical Computing Technology (LSPANC 2020 January)

## — HPC and Computer Arithmetics toward Minimal-Precision Computing —

**January 29 – 30, 2020**

**RIKEN Center for Computational Science (R-CCS), Kobe, Japan**

## Overview

In numerical computations, the precision of floating-point computations is a key factor to determine the performance (speed and energy-efficiency) as well as the reliability (accuracy and reproducibility). However, the precision generally plays a contrary role for both. Therefore, the ultimate concept for maximizing both at the same time is the minimal-precision computation through precision-tuning, which adjusts the optimal precision for each operation and data. Several studies have been already conducted for it so far, but the scope of those studies is limited to the precision-tuning alone.

In 2019, we have just started the Minimal-Precision Computing project [1] to propose a more broad concept of the minima-precision computing system with precision-tuning, involving both hardware and software stack. Specifically, our system combines (1) a precision-tuning

https://www.r-ccs.riken.jp/labs/lpnctrt/lspanc2020jan/

# Day 1

**Session 1: Plenary talks**

**9:30-9:40** Opening

**9:40-10:10** "Overview of minimal-precision computing and (weak)-numerical reproducibility" Toshiyuki Imamura (RIKEN Center for Computational Science)

**10:10-10:50** "Precision Auto-Tuning and Control of Accuracy in Numerical Applications" Fabienne Jézéquel (Sorbonne University)

10:50-11:00 Coffee Break

**11:00-11:40** "Cygnus: GPU meets FPGA for HPC" Taisuke Boku (University of Tsukuba)

**11:40-12:10** "Data-flow Compiler for Stream Computing Hardware on FPGA" Kentaro Sano (RIKEN Center for Computational Science

12:10-13:20 Lunch Break

**Session 2: FPGA technologies**

**13:20-13:50** "Exploring HLS with arbitrary precision with the Nymble compiler" Jens Huthmann (RIKEN Center for Computational Science)

**13:50-14:20** ""CIRCUS": Pipelined Inter-FPGA Communication with Computation in OpenCL on Cygnus Supercomputer" Norihisa Fujita (University of Tsukuba)

14:20-14:40 Coffee Break

**14:40-15:10** "Precision Tuning of the Arithmetic Units in Matrix Multiplication on FPGA" Yiyu Tan (RIKEN Center for Computational Science)

**Session 3: Mixed-precision and applications (1)**

**15:10-15:40** "Investigation into the convergence behavior of the mixed-precision GMRES(m) method using FP64 and FP32" Takeshi Fukaya (Hokkaido University)

15:40-15:50 Coffee Break

**15:50-16:20** "Implementation binary128 version of semidefinite programming solver" Maho Nakata (RIKEN ACCC) and Naohito Nakasato (University of Aizu)

**16:20-17:00** "An a posteriori verification method for generalized real-symmetric eigenvalue problems in large-scale electronic state calculations" Takeo Hoshi (Tottori University)

**17:00-17:45 Open Discussion**

# Day2

**Session 4: Numerical verification (1)**

**10:40-11:10** "More system independent usage of numerical verification algorithms written in high-level programming languages" Kai Torben Ohlhus (Tokyo Woman's Christian University)

11:10-11:20 Coffee Break

**Session 5: Accurate numerical libraries**

**11:20-12:00** "Hierarchical and modular approach for reproducible and accurate linear algebra algorithms" Roman Iakymchuk (Sorbonne University and Fraunhofer ITWM)

**12:00-12:30** "Accurate BLAS implementations: OzBLAS and BLAS-DOT2" Daichi Mukunoki (RIKEN Center for Computational Science)

12:30-13:30 Lunch Break

**Session 6: Mixed-precision and applications (2)**

**13:30-14:10** "Performance Evaluation of Scientific Applications with Posit by using OpenCL" Yuki Murakami (University of Aizu)

**14:10-14:40** "Using Field-Programmable Gate Arrays to Explore Different Numerical Representation: A Use-Case on POSITs" Artur Podobas (RIKEN Center for Computational Science)

14:40-15:00 Coffee Break

**15:00-15:30** "How (not) to cheat in HPL-AI" Shuhei Kudo (RIKEN Center for Computational Science)

**15:30-16:00** "Double-precision FPUs in High-Performance Computing: an Embarrassment of Riches?" Jens Domke (RIKEN Center for Computational Science)

16:00-16:10 Coffee Break

**Session 7: Numerical verification (2)**

**16:10-16:50** "Verified Numerical Computations on Supercomputers" Takeshi Ogita (Tokyo Woman's Christian University)

**16:50-17:20 Discussion**

17:20-17:30 Closing

# Lunch

Cafeteria
“FOCUS” @1F

Daily set menus,  soba/udon noodles and a curry rice.
JPY 500～600

# *Overview of minimal-precision computing and (weak)-numerical reproducibility”*

*Workshop on Large-scale Parallel Numerical Computing Technology (LSPANC 2020 January) — HPC and Computer Arithmetics toward Minimal-Precision Computing —,*
R-CCS, Kobe, Japan, 29 January 2020



**RIKEN Center for Computational Science (R-CCS) (Japan)**
Toshiyuki Imamura, Daichi Mukunoki, Yiyu Tan, Atsushi Koshiba, Jens Huthmann, Kentaro Sano

**University of Tsukuba, CCS(Japan)**
Norihisa Fujita, Taisuke Boku

**Sorbonne University, CNRS, LIP6 (France)**
Fabienne Jézéquel, Stef Graillat, Roman Iakymchuk

# Minimal Precision Computing

- In numerical computations,
  - Precision of floating-point computations is a key factor to determine the performance (speed, energy-efficiency, reliability, accuracy, and reproducibility)
- However,
  - Precision generally plays a contrary role to the computing throughput and speed.
  - The ultimate concept for maximizing both at the same time is the minimal-precision computing through precision-tuning, which adjusts the optimal precision for each operation and data.

***FP32 or FP64 are believed indispensable for the science simulation, but sometimes 24bits or 40bits look enough…***

***Anticipate to maximize utilization of FP16 or BF16, but enough or insufficient?***

***Recent problems in a silicon renography, no more transistors can be installed on a die. Reduced-precision is necessary. How many bits are significant ?***

# Minimal Precision Computing

- **Since 2019,**

  RIKEN, Sorbonne University, and Univ. Tsukuba have started a new collaborative work; the Minimal-Precision Computing project [1] to **propose a broader concept of the minimal-precision computing system with precision-tuning, involving both hardware and software stack.**

- **Combining HW/SW stacks**

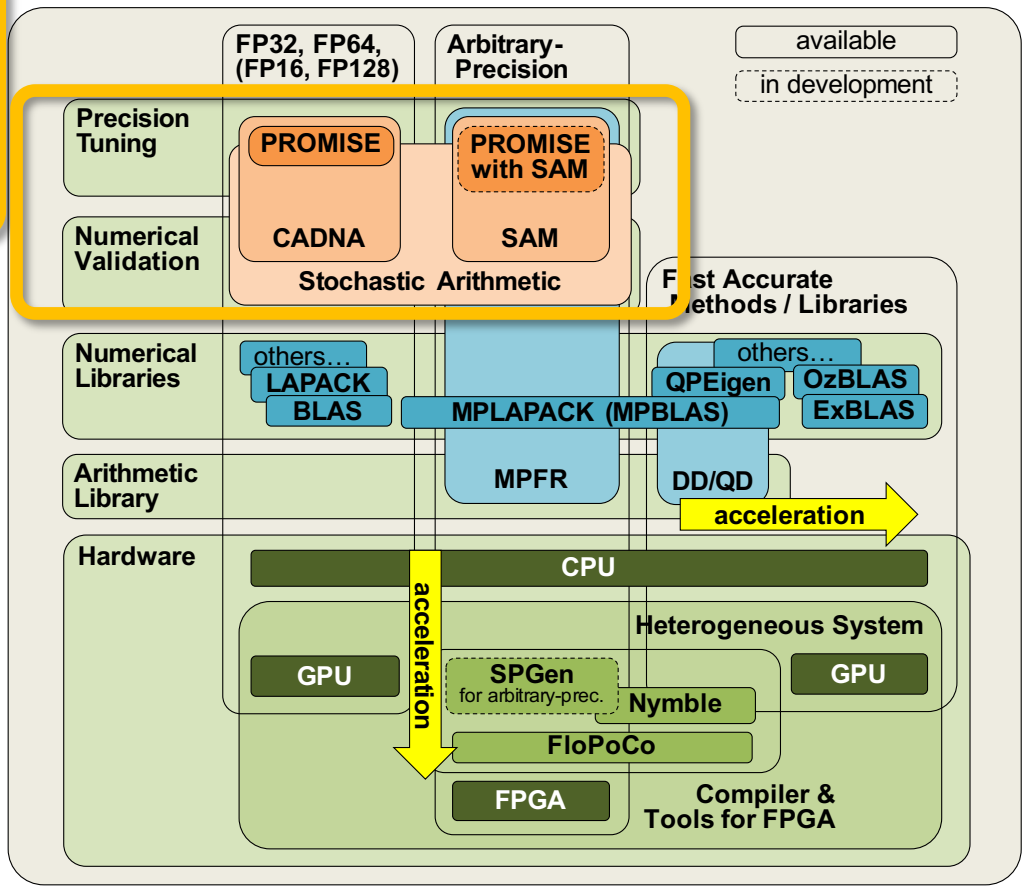
  (1) a precision-tuning method based on Discrete Stochastic Arithmetic (DSA), (2) arbitrary-precision arithmetic libraries, (3) fast and accurate numerical libraries, and (4) Field-Programmable Gate Array (FPGA) with High-Level Synthesis (HLS).

*[1] Daichi Mukunoki, Toshiyuki Imamura, Yiyu Tan, Atsushi Koshiba, Jens Huthmann, Kentaro Sano, Fabienne Jézéquel, Stef Graillat, Roman Iakymchuk, Norihisa Fujita, Taisuke Boku: Minimal-Precision Computing for High-Performance, Energy-Efficient, and Reliable Computations, SC19 research poster session, 2019.*

# System Overview

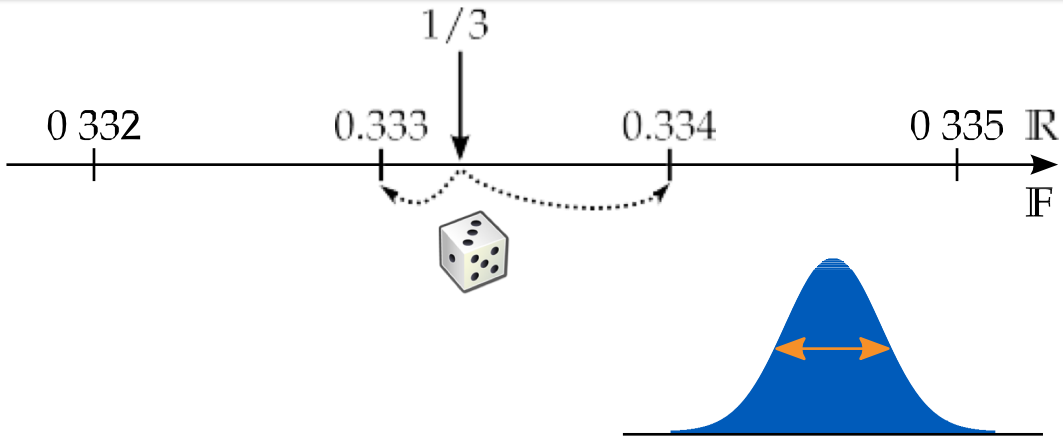
Main software/hardware components for minimal-precision computing system:

1. **Arbitrary-precision arithmetic library**
   - MPFR (GNU)
2. **Precision-tuning method based on stochastic arithmetic**
   - Stochastic libraries: CADNA & SAM (Sorbonne U.)
   - Precision-tuner: PROMISE (Sorbonne U.)
3. **Fast & accurate numerical libraries**
   - Accurate BLAS: ExBLAS (Sorbonne U.), OzBLAS (TWCU/RIKEN)
   - Quadruple-precision BLAS and Eigen solver: QPBLAS/QPEigen (JAEA/RIKEN)
   - Other open source (QD, MPLAPACK, etc.)
4. **Heterogeneous system with FPGA**
   - FPGA-GPU-CPU system: “Cygnus” (U. Tsukuba)
   - Compilers: SPGen (RIKEN), Nymble (TU Darmstadt/RIKEN)



# Discrete Stochastic Arithmetic (DSA)

[Vignes, 2004]




- each operation executed **3 times** with a random rounding mode
- **number of correct digits in the results estimated using Student's test with the probability 95%**
- estimation may be invalid if both operands in a multiplication or a divisor are not significant.
  ⇒ control of multiplications and divisions: *self-validation* of DSA.
- in DSA rounding errors are assumed centered.
  even if they are not rigorously centered, the accuracy estimation can be considered correct up to 1 digit.

# Implementation of DSA

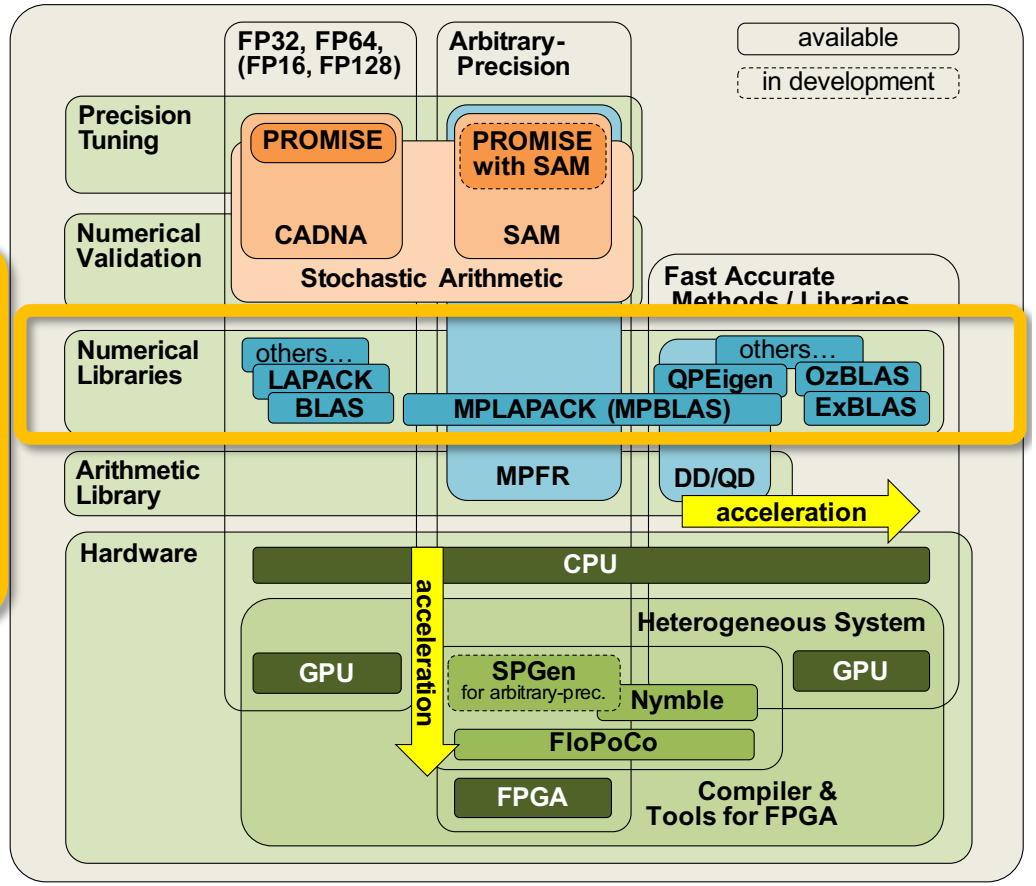
- CADNA: for programs in single and/or double precision
  http://cadna.lip6.fr
- SAM: for arbitrary precision programs (based on MPFR)
  http://www-pequan.lip6.fr/~jezequel/SAM

- estimate accuracy and detect numerical instabilities
- provide stochastic types (3 classic type variables and 1 integer):
  - `float_st` in single precision
  - `double_st` in double
  - precision `mp_st` in arbitrary precision
- all operators and mathematical functions overloaded
  ⇒ few modifications in user programs

# System Overview

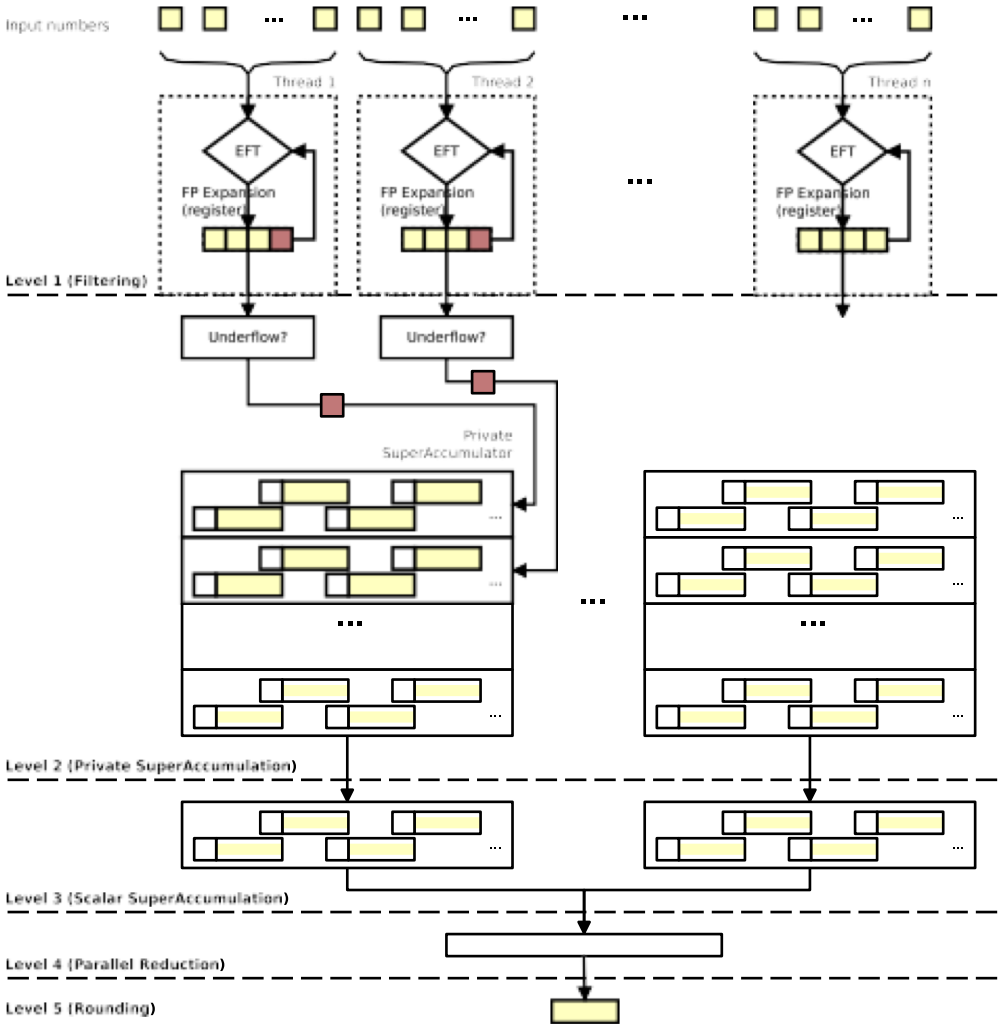
Main software/hardware components for minimal-precision computing system:

1. **Arbitrary-precision arithmetic library**
   - MPFR (GNU)
2. **Precision-tuning method based on stochastic arithmetic**
   - Stochastic libraries: CADNA & SAM (Sorbonne U.)
   - Precision-tuner: PROMISE (Sorbonne U.)
3. **Fast & accurate numerical libraries**
   - Accurate BLAS: ExBLAS (Sorbonne U.), OzBLAS (TWCU/RIKEN)
   - Quadruple-precision BLAS and Eigen solver: QPBLAS/QPEigen (JAEA/RIKEN)
   - Other open source (QD, MPLAPACK, etc.)
4. **Heterogeneous system with FPGA**
   - FPGA-GPU-CPU system: “Cygnus” (U. Tsukuba)
   - Compilers: SPGen (RIKEN), Nymble (TU Darmstadt/RIKEN)



# Accurate/ Reproducible BLAS(ExBLAS)

## Highlights of the Algorithm




- Parallel algorithm with 5-levels
- Suitable for today's parallel architectures
- **Based on FPE with EFT and Kulisch accumulator**
- Guarantees "inf" precision

→ **bit-wise reproducibility**

# Accurate/ Reproducible BLAS(OzBLAS)

## Accurate & reproducible dot-product ( $x^{\mathrm{T}}y$ )

The vectors can be split recursively until $\underline{x}^{(p)}$ and $\underline{y}^{(q)}$ become zero

$$x = x^{(1)} + x^{(2)} + x^{(3)} + \cdots + x^{(p\text{-}1)} + \underline{x}^{(p)}$$

$$y = y^{(1)} + y^{(2)} + y^{(3)} + \cdots + y^{(q\text{-}1)} + \underline{y}^{(q)}$$

$x^{\mathrm{T}}y$ is transformed to the sum of multiple dot-products

$$\begin{aligned} x^{\mathrm{T}}y = {} & (x^{(1)})^{\mathrm{T}}y^{(1)} + (x^{(1)})^{\mathrm{T}}y^{(2)} + (x^{(1)})^{\mathrm{T}}y^{(3)} + \cdots + (x^{(1)})^{\mathrm{T}}y^{(q\text{-}1)} \\ & + (x^{(2)})^{\mathrm{T}}y^{(1)} + (x^{(2)})^{\mathrm{T}}y^{(2)} + (x^{(2)})^{\mathrm{T}}y^{(3)} + \cdots + (x^{(2)})^{\mathrm{T}}y^{(q\text{-}1)} \\ & + (x^{(3)})^{\mathrm{T}}y^{(1)} + (x^{(3)})^{\mathrm{T}}y^{(2)} + (x^{(3)})^{\mathrm{T}}y^{(3)} + \cdots + (x^{(3)})^{\mathrm{T}}y^{(q\text{-}1)} \\ & + \cdots \\ & + (x^{(p\text{-}1)})^{\mathrm{T}}y^{(1)} + (x^{(p\text{-}1)})^{\mathrm{T}}y^{(2)} + (x^{(p\text{-}1)})^{\mathrm{T}}y^{(3)} + \cdots + (x^{(p\text{-}1)})^{\mathrm{T}}y^{(q\text{-}1)} \end{aligned}$$

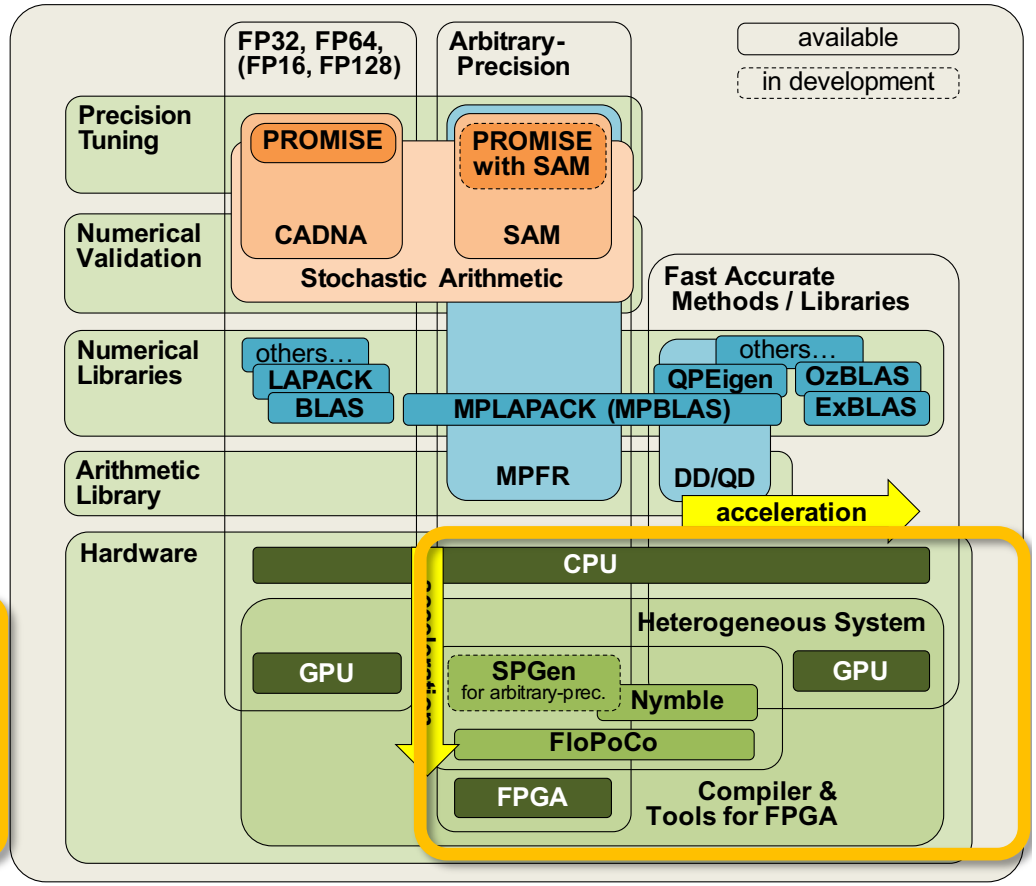
Those computations can be performed using standard BLAS (e.g., MKL, OpenBLAS, cuBLAS)

**Productive & High-performance**

# System Overview

Main software/hardware components for minimal-precision computing system:

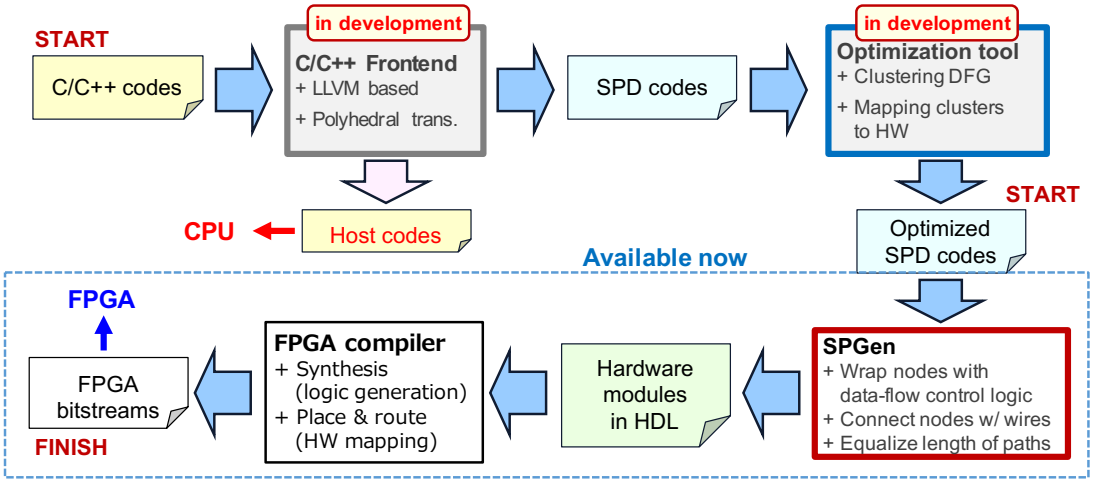
1. **Arbitrary-precision arithmetic library**
   - MPFR (GNU)
2. **Precision-tuning method based on stochastic arithmetic**
   - Stochastic libraries: CADNA & SAM (Sorbonne U.)
   - Precision-tuner: PROMISE (Sorbonne U.)
3. **Fast & accurate numerical libraries**
   - Accurate BLAS: ExBLAS (Sorbonne U.), OzBLAS (TWCU/RIKEN)
   - Quadruple-precision BLAS and Eigen solver: QPBLAS/QPEigen (JAEA/RIKEN)
   - Other open source (QD, MPLAPACK, etc.)
4. **Heterogeneous system with FPGA**
   - FPGA-GPU-CPU system: “Cygnus” (U. Tsukuba)
   - Compilers: SPGen (RIKEN), Nymble (TU Darmstadt/RIKEN)



# FPGA Performance enhancement

- SPGen (RIKEN)
  - **a compiler to generate HW module codes in Verilog-HDL for FPGA** from input codes in Stream Processing Description (SPD) Format.
  - a data-flow graph representation, which is suitable for FPGA.
  - it supports FP32 only, but we are going to extend SPGen to support **arbitrary-precision floating-point.**

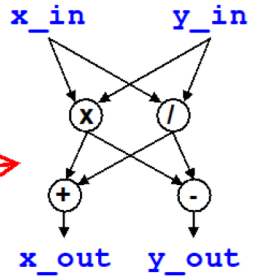
- Nymble (TU Darmstadt, RIKEN)
  - another compiler project for FPGA. It **directly accepts C codes** and has already started to support **arbitrary-precision.**




**Module definition with data-flow graph**
by describing formulae of computation

```
Name      PE;      ### Define pipeline "PE"
Main_In   {in:: x_in,  y_in};
Main_Out  {out::x_out, y_out};

EQU  eq1,  t1    = x_in * y_in;
EQU  eq2,  t2    = x_in / y_in;
EQU  eq3,  x_out = t1 + t2;
EQU  eq3,  y_out = t1 - t2;
```

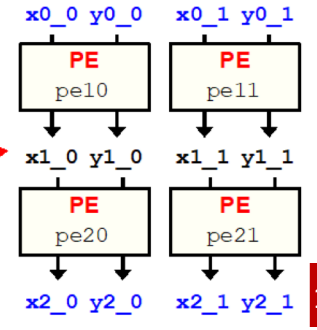



**Module definition with hardware structure**
by describing connections of modules

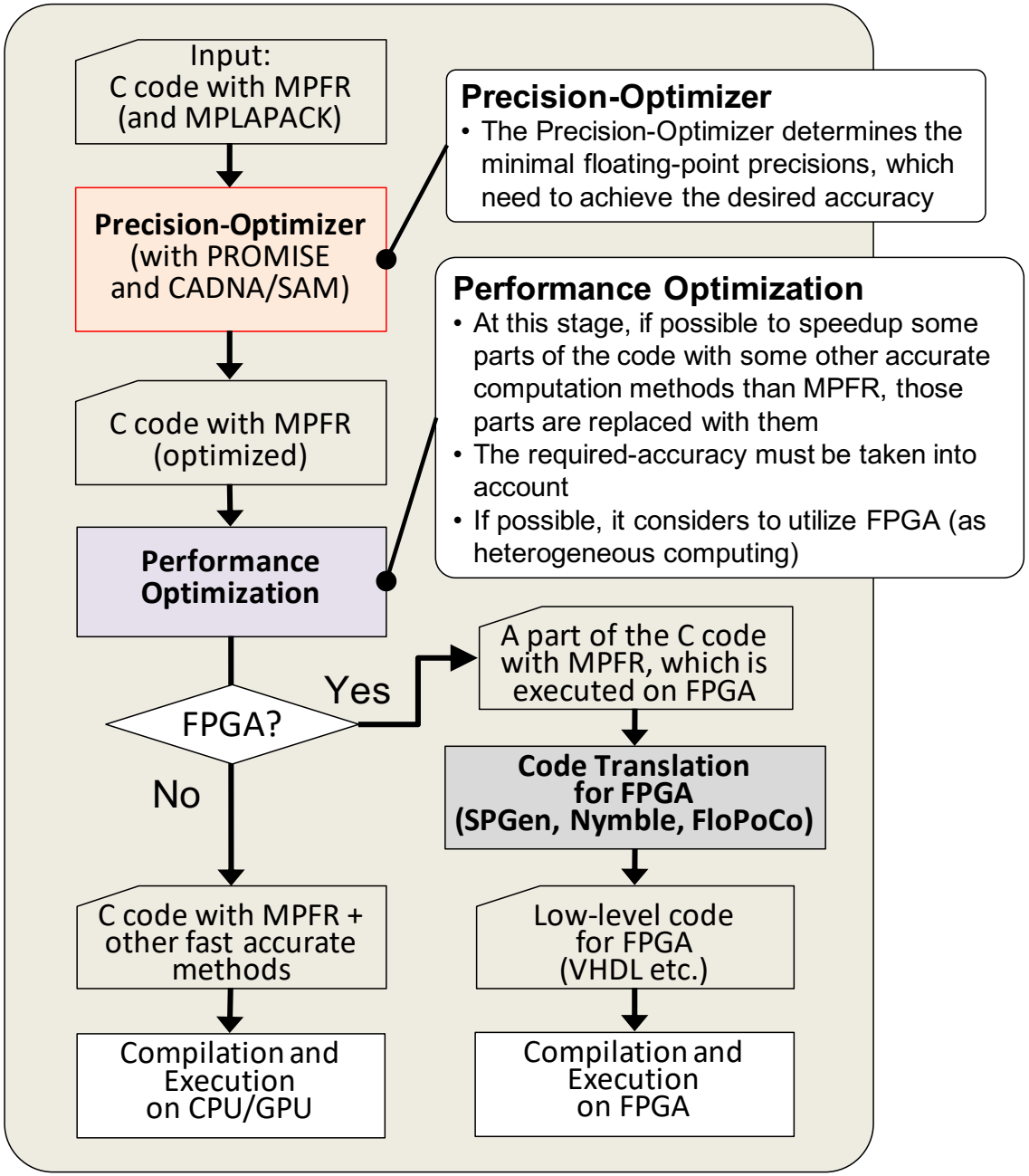
```
Name      Core;     ### Define IP core "Core"
Main_In   {in:: x0_0, x0_1, y0_0, y0_1};
Main_Out  {out::x2_0, x2_1, y2_0, y2_1};

### Description of parallel pipelines for t=0
HDL pe10, 123, (x1_0, y1_0) = PE(x0_0, y0_0);
HDL pe11, 123, (x1_1, y1_1) = PE(x0_1, y0_1);

### Description of parallel pipelines for t=1
HDL pe20, 123, (x2_0, y2_0) = PE(x1_0, y1_0);
HDL pe21, 123, (x2_1, y2_1) = PE(x1_1, y1_1);
```



# Minimal-Precision Computing - System Workflow



# What does Reproducibility refer to ?

In computational science, reproducibility is considered from several view-points depending on the context and demand.

- ***Bit-level reproducibility***
  is **the capability to reproduce the bit-wise identical result** with the same input on any HW/SW configuration. **No general approach** for any floating-point computation has been proposed yet. It is **non-realistic** to support bit-level reproducibility **on all floating-point computations** through the existing approaches.
- ***Weak numerical reproducibility[2]***
  the reproducibility, (up to a high probability) of **the computation result with a certain accuracy demanded by the user**. The underlying numerical validation is performed using **a statistical approach** that estimates with a high probability **the number of correct digits** in the computation result.

  → ***The extension of our minimal-precision computing scheme, which validates the accuracy (demanded by the user) of the result through the minimal-precision use.***

*[2] T. Imamura, D. Mukunoki, R. Iakymchuk, F. Jézéquel, S. Graillat, "Numerical reproducibility based on minimal-precision validation", Computational Reproducibility at Exascale Workshop (CRE2019) , conjunction with SC19, Denver, CO, USA, Nov. 2019*

# Weak Numerical Reproducibility on Minimal-Precision Computing

1. **The minimal-precision computing system => a black box**
   - Though ***different paths for execution may be used*** either to speed up computations and/or ensure energy-efficiency, required precision is guaranteed.
2. **Validation of the requested accuracy of the computation demanded by the user**
   - If the computation method can achieve the required result, any methods, any computation environments, and any computation conditions can be accepted.
   - ***No longer need to develop some reproducible variant(s)*** for each computation method or mathematical problem.
3. **Comparing with re-playable and re-traceable methods**
   - easier to adapt to different (parallel) architectures.
   - Existing methods and software ***for ensuring bit-level reproducibility are still able to contribute to ensure the demanded accuracy***, if such method relies on some accurate method.

# Minimal Precision Computing

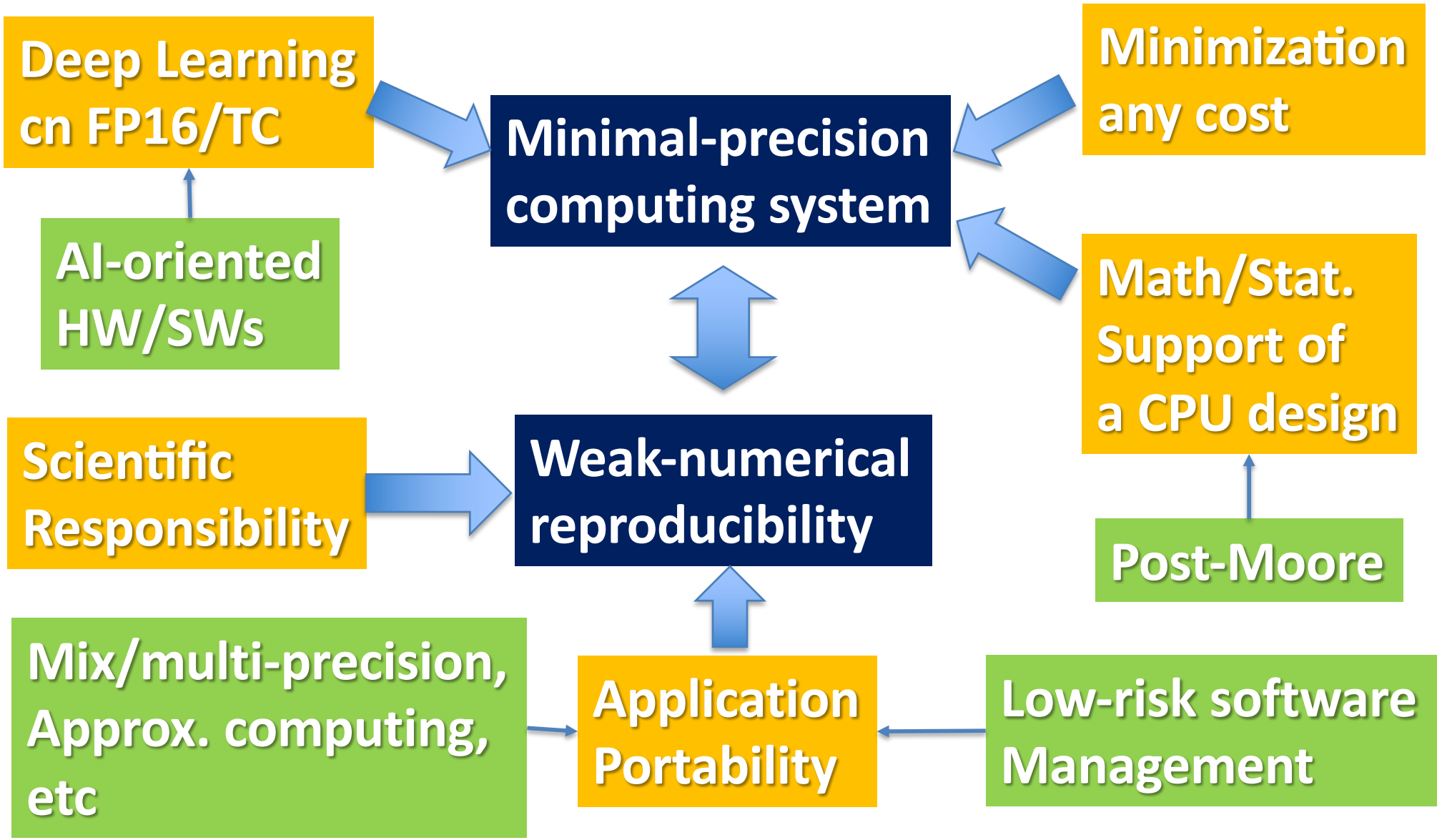
- In numerical computations,
  - Precision of floating-point computations is a key factor to determine the performance (speed, energy-efficiency, reliability, accuracy, and reproducibility)
- However,
  - Precision generally plays a contrary role to the computing throughput and speed.
  - The ultimate concept for maximizing both at the same time is the minimal-precision computing through precision-tuning, which adjusts the optimal precision for each operation and data.

***FP32 or FP64 are believed indispensable for the science simulation, but sometimes 24bits or 40bits look enough…***

***Anticipate to maximize utilization of FP16 or BF16, but enough or insufficient?***

***Recent problems in a silicon renography, no more transistors can be installed on a die. Reduced-precision is necessary. How many bits are significant ?***

# Minimal-precision Computing
# Weak-numerical Reproducibility



# Conclusion

- A new minimal-precision computing
- A new concept of weak numerical reproducibility

  ***the reproducibility, (up to a high probability) of the computation result with a certain accuracy demanded by the user.***

  A systematic approach for the minimal-precision system.

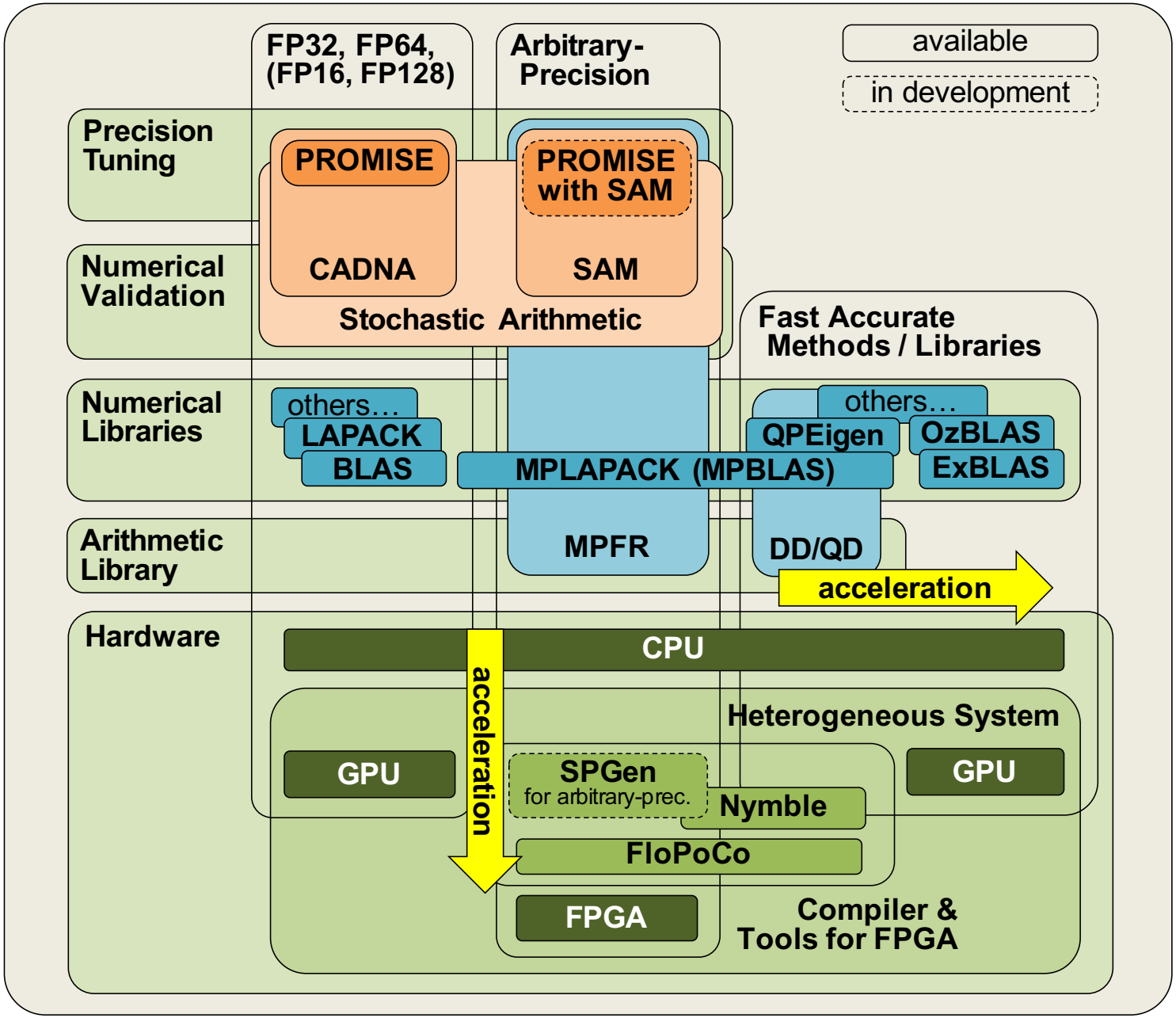
- The minimal-precision computing system is going to be built up, hopefully, soon…
- The concept of weak numerical reproducibility covers sort of the demands for responsibility in computational sciences.
- Besides, if it has been realized with new hardware like FPGAs, the minimal-precision computing system can address the demands for accuracy, high-performance, and energy efficient computation as well.
- Future work is ***Demonstration of weak numerical reproducibility.***

# ACKNOWLEDGMENT


- ***The authors would like to thank***

- ***Partially supported by***
  - the European Unions Horizon 2020 research, innovation programme under the Marie Skodowska-Curie grant agreement via the Robust project No. 842528,
  - the Japan Society for the Promotion of Science (JSPS) KAKENHI Grant No. 19K20286,
  - Multidisciplinary Cooperative Research Program in CCS, University of Tsukuba.

# Minimal-Precision Computing - System Stack



# Minimal-precision computing 1/2

- Precision is a key factor to determine the performance (speed & energy) as well as the reliability (accuracy & reproducibility) but precision plays a contrary role for both
- The ultimate concept for both is the ***minimal-precision computing through precision-tuning***, which adjusts the optimal precision for each operation and data: it is **reliable** (robust) and **sustainable** as it ensures the requested accuracy of the result as well as is **high-performance** and **energy-efficient**
- While the scope of existing studies is limited to the precision-tuning alone, **our project** aims to **propose a broader concept of the computing system with precision-tuning**, involving both hardware & software stack

# Reproducible Conjugate Gradient

## Sources

- Identify sources of non-reproducibility: dot (parallel reduction), axpy, and spmv

## Solutions

- Combine sequential executions, reorganization of operations, and arithmetic solutions

→ aiming for lighter or lightweight approaches

- axpy is made reproducible thanks to fma
- spmv computes blocks of rows in parallel, but with $a * b + / - c * d$

→ ensure deterministic execution with explicit fmas

- dot –> apply ExBLAS and FPE-based approaches

# Accurate/ Reproducible Parallel Arithmetics

***Fix the Order of Computations***

Sequential mode: intolerably costly at large-scale systems

Fixed reduction trees: substantial communication overhead

→ Example: Intel **C**onditional **N**umerical **R**eproducibility in MKL (~ 2*x* for datum, no accuracy guarantees)

***Eliminate/Reduce the Rounding Errors***

Fixed-point arithmetic: limited range of values

Fixed FP expansions with Error-Free Transformations (EFT)

→ Example: double-double or quad-double (Briggs, Bailey, Hida, Li) (work well on a set of relatively close numbers)

"Infinite" precision: reproducible independently from the inputs

→ Example: Kulisch accumulator (considered inefficient)

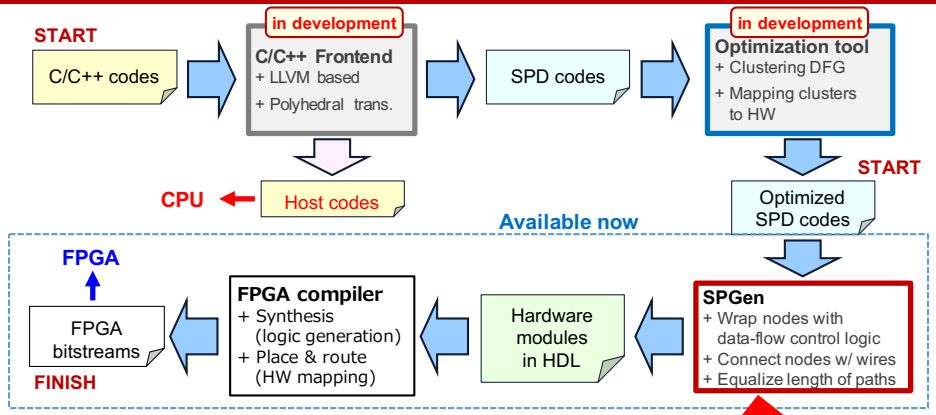
***Libraries***

ExBLAS: Exact BLAS (Iakymchuk et al.) ReproBLAS: Reproducible BLAS (Demmel et al.)
RARE-BLAS: Reproducible Accurately Rounded and Efficient BLAS (Chohra et al.)

# Other HW/SW Components

## SPGen (RIKEN)

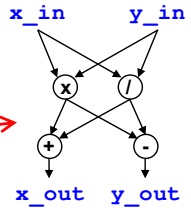
- SPGen (Stream Processor Generator) [14] is a compiler to generate HW module codes in Verilog-HDL for FPGA from input codes in Stream Processing Description (SPD) Format. The SPD uses a data-flow graph representation, which is suitable for FPGA.
- It supports FP32 only, but we are going to extend SPGen to support arbitrary-precision floating-point. Currently, there is no FPGA compiler supporting arbitrary-precision.




**Module definition with data-flow graph**
by describing formulae of computation

```
Name      PE;      ### Define pipeline "PE"
Main_In   {in:: x_in,  y_in};
Main_Out  {out::x_out, y_out};

EQU  eq1, t1    = x_in * y_in;
EQU  eq2, t2    = x_in / y_in;
EQU  eq3, x_out = t1 + t2;
EQU  eq3, y_out = t1 - t2;
```

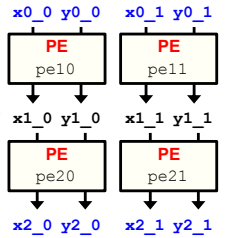



**Module definition with hardware structure**
by describing connections of modules

```
Name      Core;     ### Define IP core "Core"
Main_In   {in:: x0_0, x0_1, y0_0, y0_1};
Main_Out  {out::x2_0, x2_1, y2_0, y2_1};

### Description of parallel pipelines for t=0
HDL pe10, 123, (x1_0, y1_0) = PE(x0_0, y0_0);
HDL pe11, 123, (x1_1, y1_1) = PE(x0_1, y0_1);

### Description of parallel pipelines for t=1
HDL pe20, 123, (x2_0, y2_0) = PE(x1_0, y1_0);
HDL pe21, 123, (x2_1, y2_1) = PE(x1_1, y1_1);
```

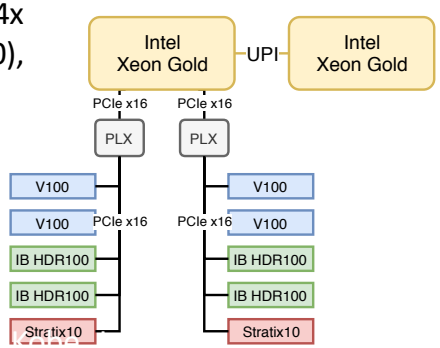



## Nymble (TU Darmstadt, RIKEN)

- Nymble [15] is another compiler project for FPGA. It directly accepts C codes and has already started to support arbitrary-precision.
- It is more suited for non-linear memory access pattern, like with graph based data structures.

## Cygnus (University of Tsukuba)

- Cygnus is the world first supercomputer system equipped with both GPU (4x Tesla V100) and FPGA (2x Stratix 10), installed in CCS, University of Tsukuba
- Each Stratix 10 FPGA has four external links at 100Gbps. 64 FPGAs make 8x8 2D-Torus network for communication
- This project targets such a heterogeneous system with FPGA.




# Minimal Precision Computing

- ***The minimal-precision computing***

  high-performance and energy-efficient as well as reliable (accurate, reproducible, and validated) computations

- ***systematic approach combining internally***

  1. a precision-tuning method based on Discrete Stochastic Arithmetic (DSA),
  2. arbitrary-precision arithmetic libraries,
  3. fast and accurate numerical libraries, and
  4. Field-Programmable Gate Array (FPGA) with High-Level Synthesis (HLS)

- Reliable, General, Comprehensive, High-performance, Energy-efficient, Realistic

# Outline

- What does Reproducibility refer to ?
  - Weak-numerical reproducibility
- Minimal-precision computing
- System Overview of Minimal-precision computing
  - Discrete Stochastic Arithmetic (DSA)
  - ExBLAS: Accurate/ Reproducible Parallel Arithmetics
  - Other components
- Discussions
  - Weak Numerical Reproducibility on Minimal-Precision Computing
- Conclusion