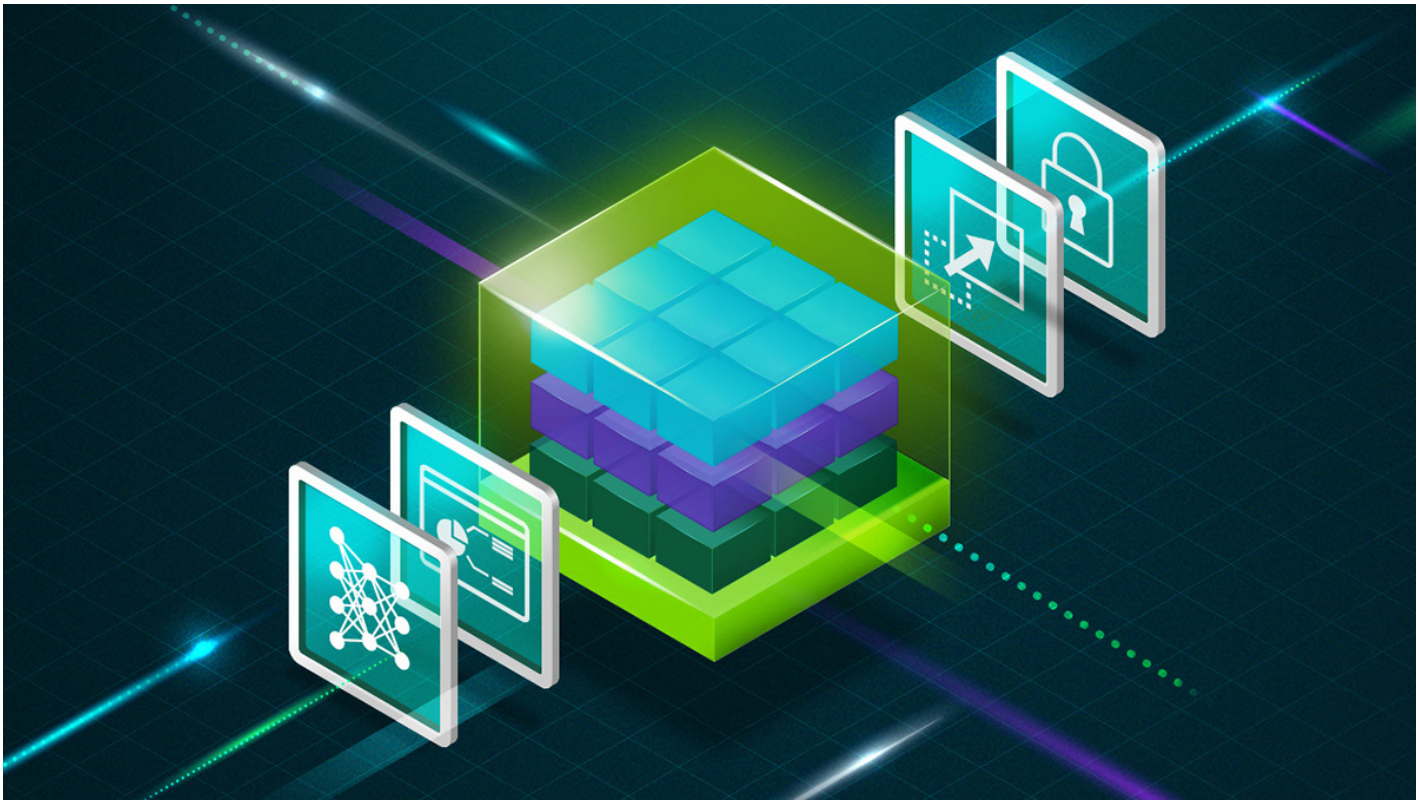
VIVIDSPARKS

GPU versus RacEr GP GPU FPGA for high productivity computing



Introduction

High-performance computing (HPC) is the use of parallel processing for the fast execution of advanced application programs. FPGAs have been shown to effectively accelerate certain types of computation useful for research and modeling [1], [2], [3]. However their utilization has been restricted by the development time and complexity for fine-grained implementations.

In this work, we compare RacEr GP GPU FPGA based with NVIDIA GPU. The objective is to compare the productivity of each platform for a broad range of tasks. The benchmarks we chosen are:
- Batch generation of pseudo-random numbers.
- Dense square matrix multiplication.
- Sum of large vectors of random numbers.
- Second order N-body simulation.

The above benchmarks are coded in CUDA.

GPU Based HPC Architecture

The RTX 3090 is a member for the GeForce family of GPUs made by NVIDIA. It has 10496 core processors running at 1.7GHz and supports up to 24GB of external DDR6 RAM. Figure 1 shows its component arrangement and interfaces. RacEr GP GPU architecture is similar except RTX 3090 has 10496 CPUs in our case we have 512 CPUs running at 300MHz with DDR4 RAM of 32GB.

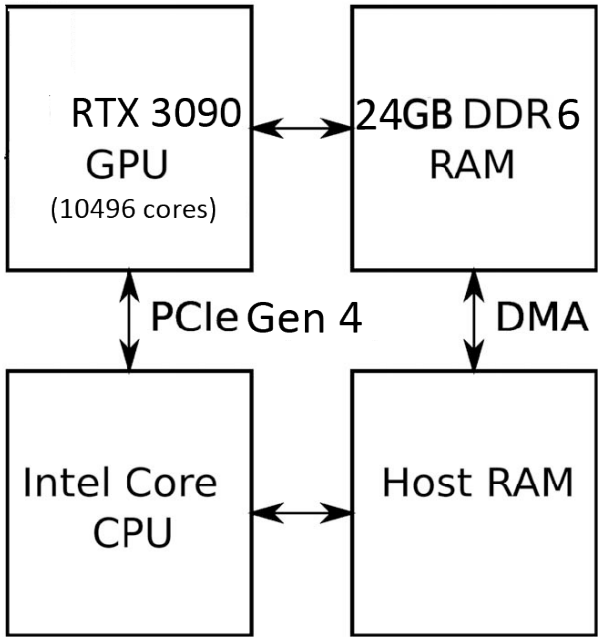


Figure 1: Structure of GTX285

Random Number Generation

We used a Mersenne twister [4] pseudo-random number generator (PRNG) to create batches of 32-bit random numbers. Both Nvidia and Convey provide a Mersennetwister as library functions. Whereas the NVIDIA PRNG is implemented as custom software on a fixed architecture, the Convey PRNG uses a pipeline shift-register architecture in custom firmware as part of their financial applications personality. Figure 2 shows the performance of the NVIDIA RTX 3090 and the RacEr GP GPU when generating increasingly large batches of random numbers. This performance is measured as an improvement over a single core CPU implementation of a Mersenne Twister.

The RacEr performs, on average, 88.9 times better than the CPU. The RTX 3090 performs 89.3 times better than the CPU. However because the GPU uses a batch processing architecture it is much more sensitive to the size of the batch than the RacEr pipeline architecture. Also the memory available to the FPGAs on the RacEr is 256 times greater than that available to the RTX 3090, so larger batches can be generated.

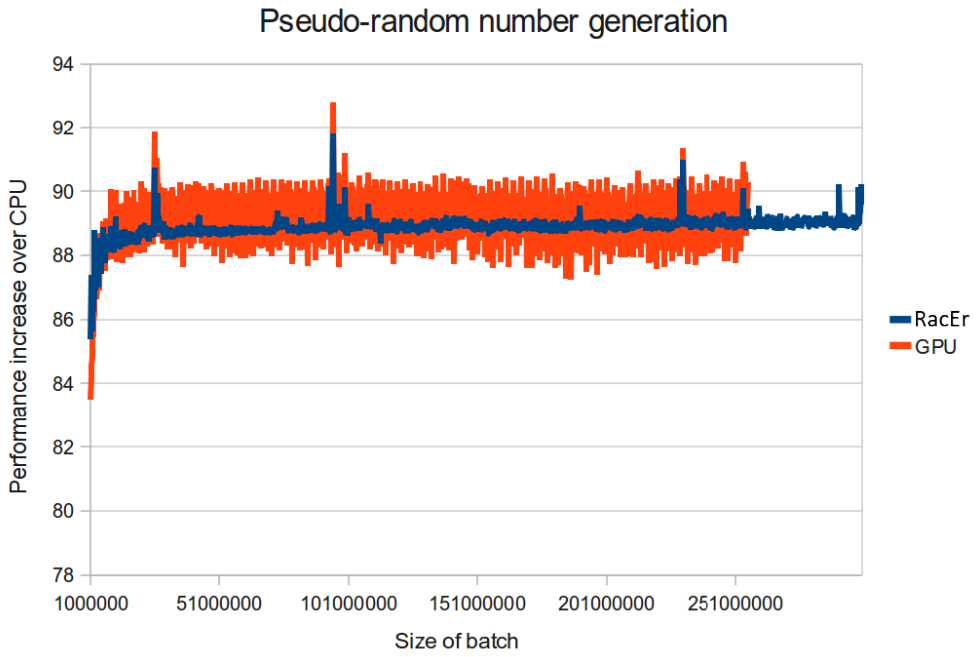


Figure 2: Performance of GPU and RacEr GP GPU FPGA architectures generating random numbers

2

Matrix Multiplication

Here we tested the performance of each architecture when multiplying two large square matrices. The Nvidia card performed the calculations on 240 single-precision hardware cores. The RacEr used 512 cores, however they were optimized for 32-bit calculations, depending on the experiment. Figure 3 shows the performance improvement over the reference GPU implementation of the same 32-bit matrices.

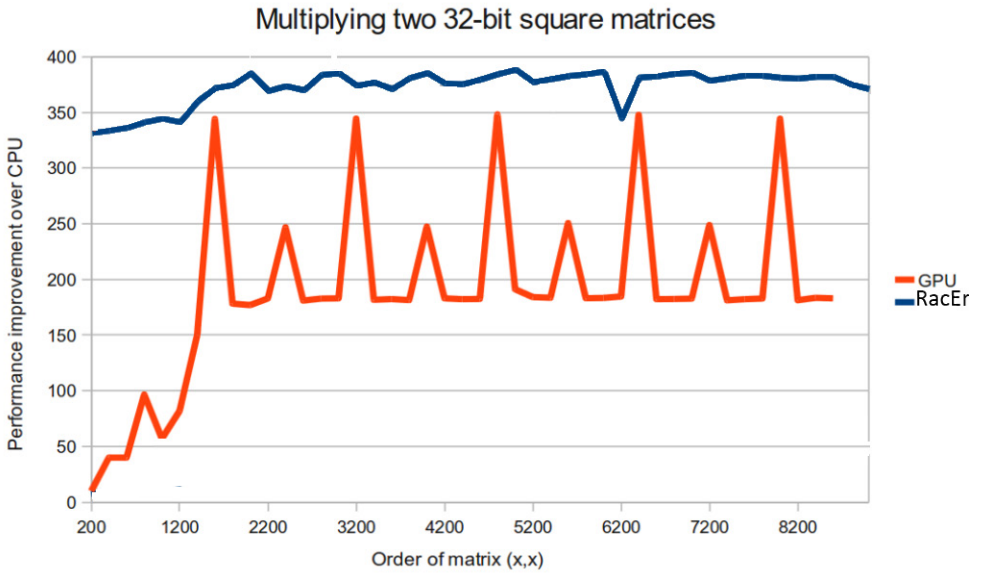


Figure 3: Performance of GPU and RacEr GP GPU FPGA architectures calculating product of two 32-bit matrices

The RacEr performs, on average, 226.8 times better than the CPU on 32-bit matrices. The GPU performs 190.4 times better on 32-bit matrices. The RacEr and GPU performance peaks occur when the width of the matrix is a multiple of the size of the available shared memory (16kb for every group of eight cores), allowing the use of a more efficient full tile matrix multiplication function.

Scalar Sum of Products

Determining the scalar sum of a vector is a "reduce" operation requiring a partially synchronous tree process architecture (see Figure 4). Figure 5 shows the performance improvement over the reference GPU implementation of the 32-bit vectors.

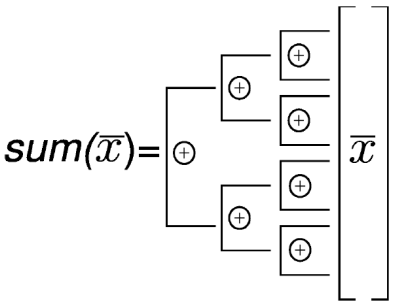


Figure 4: Process architecture for a "Reduce" operation

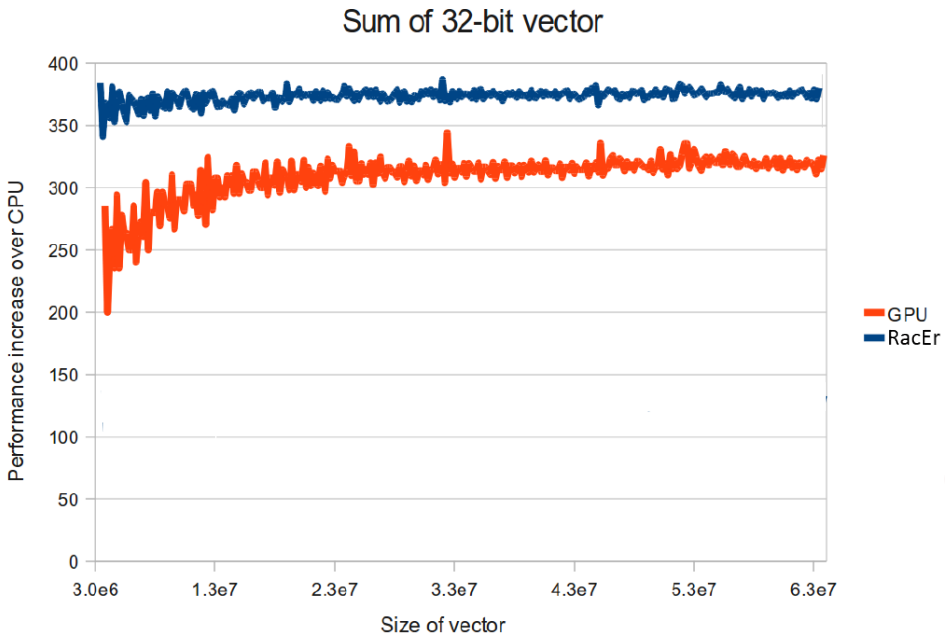


Figure 5: Performance of GPU and RacEr GP GPU FPGA architectures calculating sum of 32-bit Vector

The RacEr performs, on average, 515.87 times faster than the GPU for 32-bit vectors.

N-Body Simulation

As a final benchmark a two-dimensional, second-order simulation of the gravitational forces between a number of bodies of different mass was employed. The execution cycle is:

1) For each body, *i*, calculate the vector gravitational force, $F_{i,j}$, that is acting on it from every other body, *j*,$j \neq i$, using the following equation:

$$F_{i,j} = \frac{m_j(\overline{x_i} - \overline{x_j})}{|\overline{x_i} - \overline{x_j}|^2} \qquad (1)$$

2) For each body, *i*, sum the vector gravitational forces, $F_{i,j}$

$$F_i = m_i G \sum_{j \neq i} F_{i,j} \qquad (2)$$

Where *m* is the mass of each body, *x* is the position of each body and *G* is a scaling factor (we use *G* =0.001).

3) Synchronise each thread

4) Calculate the new velocities of each body using the following equation:

$$\overline{x}_{t+1,i} = \overline{x}_{t,i} + \overline{v}_{t+1,i}.t \quad (3)$$

5) Calculate the new positions of each body using the following equation:

$$\overline{v}_{t+1,i} = \overline{v}_{t,i} + \frac{\overline{F}_i}{m_i}.t \quad (4)$$

Where t is the unit time, (we use t = 1).

6) Synchronise each thread

7) Repeat to (1) 100 times.

This task has been chosen in part because it is a common scientific model and in part because it requires a fully synchronised mesh process architecture. All the calculations were performed to 32-bit precision. Figure 6 shows the performance improvement over a reference GPU implementation.

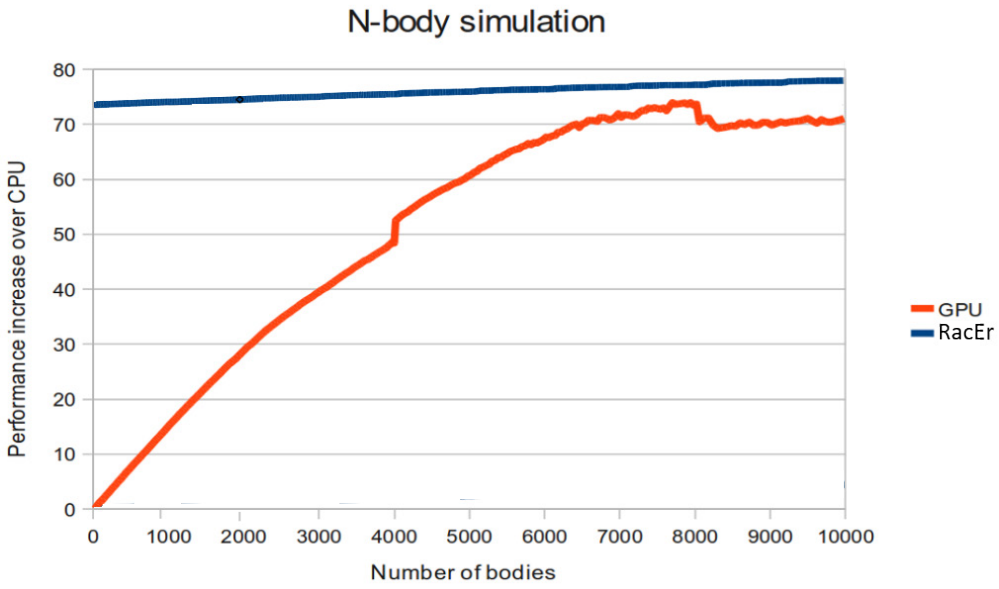


Figure 6: Performance of GPU and RacEr GP GPU FPGA architectures doing N-body simulations

For this benchmark, the performance of the RacEr 67.78 is on average greater than the GPU.

Conclusion

We have evaluated the performance of the RacEr GP GPU and the NVIDIA RTX 3090 against a range of tasks common to scientific computing. In all cases, RacEr GPU outperformed an equivalent GPU implementation.

References

[1] S. Craven and P. Athanas, "Examining the viability of FPGA supercomputing," EURASIP Journal on Embedded systems, vol. 2007, no. 1, pp. 13, 2007.

[2] T. Takagi and T. Maruyama, "Accelerating HMMER search using FPGA," Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on, pp. 332–337, 2009.

[3] M. Chiu and M. C. Herbordt, "Efficient particle-pair filtering for acceleration of molecular dynamics simulation," Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on, 2009.

[4] M. Matsumoto and T. Nishimura, "Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator," ACM Transactions on Modeling and Computer Simulation (TOMACS), vol. 8, no. 1, pp. 3–30, 1998.