MAiTRIX

RNS TPU

Matrix Multiplier for AI and Science

Architectural Overview and Advantages

Introduction

Maitrix's RNS TPU provides a substantial increase in performance and power efficiency of hardware-based matrix multiplication versus available solutions. The RNS TPU is poised to solve today's power consumption and performance bottleneck issues of datacenter CPU's that perform AI processing. Furthermore, the RNS TPU is well positioned to perform other intense numerical computations such as weather prediction and web page rank more efficiently than conventional binary and floating point-based computers.

Maitrix's RNS TPU is the world's first modular arithmetic matrix processor; it represents a radical departure from classical binary arithmetic. In fact, the RNS TPU performs matrix arithmetic using a brand-new number system having unique and desirable properties for performing massive product summation, the main operation in AI. Arithmetic processing using this new number system is called *Modular Computation*.

A Brief History of Modular Computation

Modular computation has its roots in modular arithmetic and the residue number system (RNS). The earliest known publication of RNS appears to be M. Valach of Czechoslovakia in 1955. Early RNS work in the U.S. was published by H.L Garner in 1958 and Howard Aiken of Harvard University also in 1958.

The residue number system provides a unique method for performing addition, subtraction, and multiplication of integers without slowing due to carry from digit to digit. Because there is no carry in RNS, product accumulation in RNS can be performed in parallel, that is, each digit is operated on simultaneously. RNS is not a fixed-radix number system like binary and decimal and is more related to mixed-radix numbers, like the mixed radix numbers used to measure time and day. RNS is also based on modular arithmetic, where each digit resembles processing on a finite field. The main exception is that RNS is a true extendable number system that offers many unique advantages when it comes to massive product summation required by AI.

Today, the study of RNS is broad based and can be found in many academic journals. Unfortunately, the study of RNS is relegated to arcane topics, with virtually no practical application. As such, RNS based computing systems have not been deployed in commercial computing systems to date. The reason is there are many problems to be solved that are beyond the scope of this white paper. However, one important reason is the RNS system is known in academic circles as an 'integer only' number system, and thus it is not practical for general computation which must use fractional arithmetic. Maitrix has shown this notion to be false.

In 2012, after solving a large set of problems with RNS numbers, Digital System Research (DSR) introduced the first general purpose processor which performed general purpose computation using an advanced form of RNS. This was a first in history. DSR introduced the RNS fraction point as well as many other advancements which transforms RNS into a computer number system for general computation. In 2019, Maitrix took this work further, and invented the world's first error correction of arithmetic, and now, the world's first RNS systolic matrix multiplier that out-

THE FIRST MODULAR MATRIX PROCESSOR

A New Number System for Computation

performs classical binary systems in terms of power consumption, arithmetic precision, and speed. Computation using this new number system is called Modular Computation.

The Impact of Matrix Multiplication

Matrix multiplication is the principal operation in artificial intelligence (AI) and machine learning (ML) algorithms. Unfortunately, the number of operations required to perform a single matrix multiplication grows as a cube of the size of the matrix. For a matrix of size 256x256, more than 16 million multiplications and 16 million additions are required! In the context of AI, the number of fully connected hidden layers of modern AI algorithms has grown from only a few layers to millions, with each layer connecting many billions of parameters. Since each layer of processing represents a matrix multiply, the number of operations is staggering.

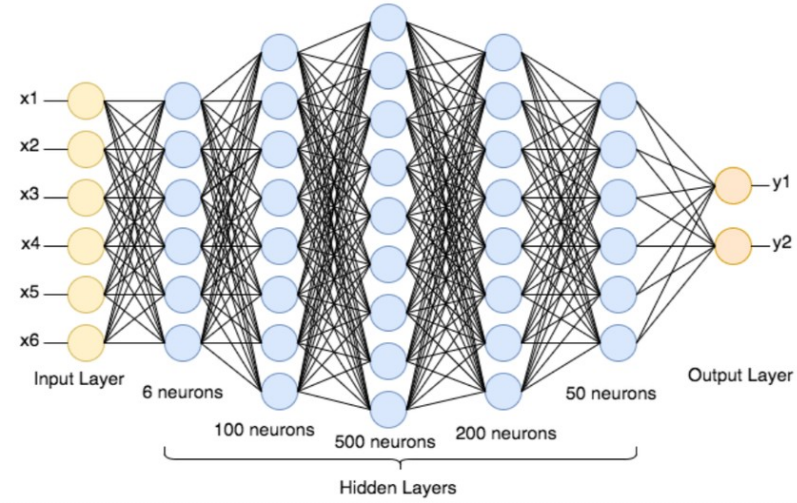
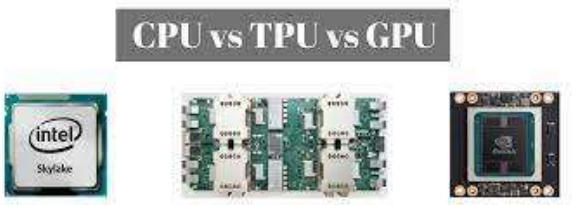


Figure 1 - Each line is a multiply and addition!

Conventional CPU's and software matrix libraries cannot perform at a level sufficient to make AI practical. This reality has led to an explosion of hardware accelerated solutions for training AI algorithms, and new number formats for increasing speed and reducing storage. Early Graphics Processing Units (GPU's) provided the sheer computational power to train AI algorithms but were not specifically designed for the task. However, Nvidia, a leading GPU processor company, has recently designed upgraded GPU processors tailored for AI training tasks that leverage existing GPU software infrastructure like CUDA. However, GPU's have not gained much use in the inference, or retrieval phase, of AI routines.



The most dedicated hardware solution for AI is known as a Tensor Processor Unit or TPU. The TPU is a hardware accelerator for matrix multiplication designed to meet the needs of AI computation. Many TPU designs target the retrieval phase of AI processing tasks, while other TPU's are more general purpose and serve to accelerate training phases as well. Most large technology companies now embrace embedding TPU technology into their product portfolio and into their silicon-based products. First in 2015, Google launched its TPU version 1 and has since launched version 2, 3, & 4. Nvidia, Intel and IBM have quickly followed with their TPU implementations embedded within their advanced server ready GPUs and CPUs.

The TPU targets matrix multiplication

Power Demands of AI Processing

Regardless of these hardware innovations, the amount of processing is so large for AI applications the power demands on data centers have become a major concern. In fact, a one study has estimated the power requirements for AI processing doubles every 3.4 months and was shown to increase more than 300,000 times in a span of six short years. As the use of AI continues to grow, the power demands of data centers will require more power than is currently generated world-wide in a very short time! Solutions to solve this power demand problem are desperately needed.

Power demands of AI data processing is not sustainable with existing technology

The RNS TPU will play an important role in decreasing power requirements for AI training algorithms. Stay tuned for future whitepapers which outline the significantly lower power demands for RNS based AI processors.

RNS TPU Architecture

Maitrix provides many configurations of the RNS TPU for various customer requirements; this includes TPU's for the training phases of AI. Unlike most solutions which use floating-point formats, the RNS TPU uses modular arithmetic and a brand-new form of non-binary computation to perform efficient matrix operations.

A new method of computation leads to a new hardware architecture

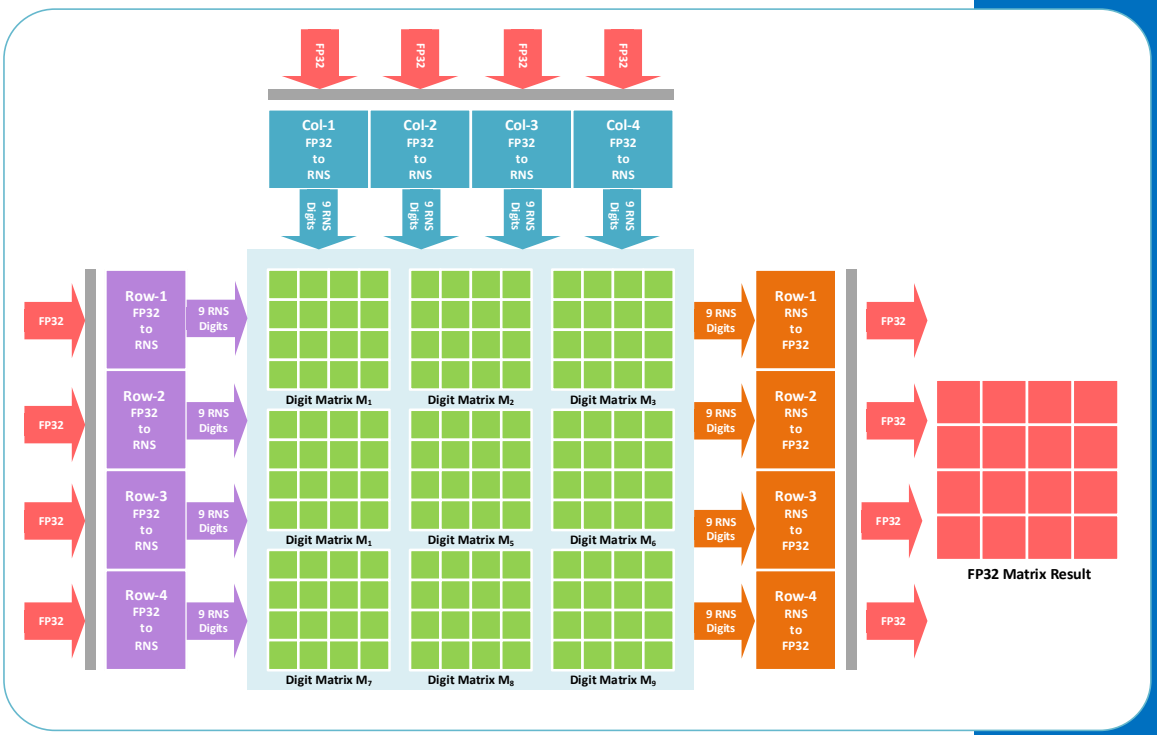


Figure 2 - RNS TPU with FP32 input and output

One such configuration is shown in Figure 2 for purpose of illustrating basic concepts of the RNS TPU. As shown in green, the RNS Matrix multiplier core is divided into nine separate matrix multiplier cores, each core performing a full matrix multiplication on a *single digit* of the RNS word. Such a configuration is not possible using binary arithmetic since there is no way to separate digits (bits) into their own matrix multiplier because of arithmetic carry. However, this is possible in the RNS TPU since the matrix computation is performed using a carry-free RNS arithmetic.

In Figure 3, the example RNS TPU machine word is shown re-arranged as a nine-digit wide value:

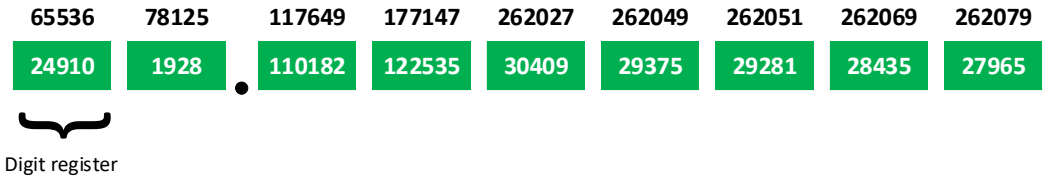


Figure 3 - RNS word using nine 18-bit wide digits

Figure 3 is provided to give an idea of an RNS number. The value encoded by the nine 18-bit wide digits of Figure 3 is ***12345678***. The modulus of each RNS digit is shown in black, and the value of each RNS digit is shown as white on green. Each green rectangle represents a "digit register" inside the TPU. To get the value of any RNS digit, simply take the number to be represented and perform a *mod* function using its modulus. For example, the value of the first digit is 12345678 % 65536 = 24910. The value of the first RNS digit when two numbers are added is A + B % M, where M is the modulus of the digit. No carry is generated to the next digit. The same rules apply to multiplication. The maximum integer value represented by the RNS word of Figure 3 is greater than 1.3×10^{47}. The word size width is double that needed to represent 32.32 values in the same way a 128-bit result is formed when two 64-bit binary numbers are multiplied.

Readers are encouraged to review RNS research to learn more about this unique number system. Note that Maitrix has advanced the basic RNS to a high degree, including the insertion of a fraction point in the RNS word of Figure 3. The takeaway is that adding and multiplying RNS numbers is performed digit by digit within the same modulus, without carry of information to any other digit. Therefore, all digits are operated in parallel without carry. The unique 'digit matrix' architecture of the RNS TPU as shown in Figure 2 is therefore possible.

Hybrid Number System TPU

A modern TPU may support single precision floating point (FP32), half precision floating point (FP16 or BFP16), and even 8-bit and 4-bit integers in different sections of the TPU. This is referred to as a hybrid or mixed-precision design. The goal of these designs is to support a word size that can accommodate an acceptable level of performance, while supporting the smallest word size to accelerate processing speed and reduce storage and power requirements.

The RNS TPU of Figure 2 supports single precision floating point as data inputs and outputs. It can also support any other floating-point format based on requirements. This allows the RNS TPU to be easily integrated into our customers products. The row and column converters, shown in purple and blue, perform what is called 'forward conversion'. The output converters shown in orange convert the RNS results back to floating point, a process known as reverse conversion. This matrix configuration performs the matric multiplication entirely in RNS format while maintaining compatibility with any binary CPU.

The RNS TPU supports carry-free arithmetic with high-speed and high accuracy summation. The RNS TPU provides conversion to and from binary formats like FP32 and FP16 for compatibility with existing

software. The advantages of a wide word, carry free computation is therefore supported with floating point formats used for efficient storage and complex calculations. The internal RNS TPU processing can be tuned for many applications and is both more efficient, faster, and more accurate than floating point processing in many cases.

Pure Systolic Matrix Multiplication

To perform high performance matrix multiplication, TPU designers typically arrange many multipliers into a square or rectangular matrix known as a *systolic array*. This provides for a high degree of parallelism, with performance surpassing most vector processing units included in high-end CPU's. Another advantage of the systolic arrangement is matrix inputs are passed from one multiplier to the next thereby reducing the number of memory data accesses. This is the secret of the modern TPU.

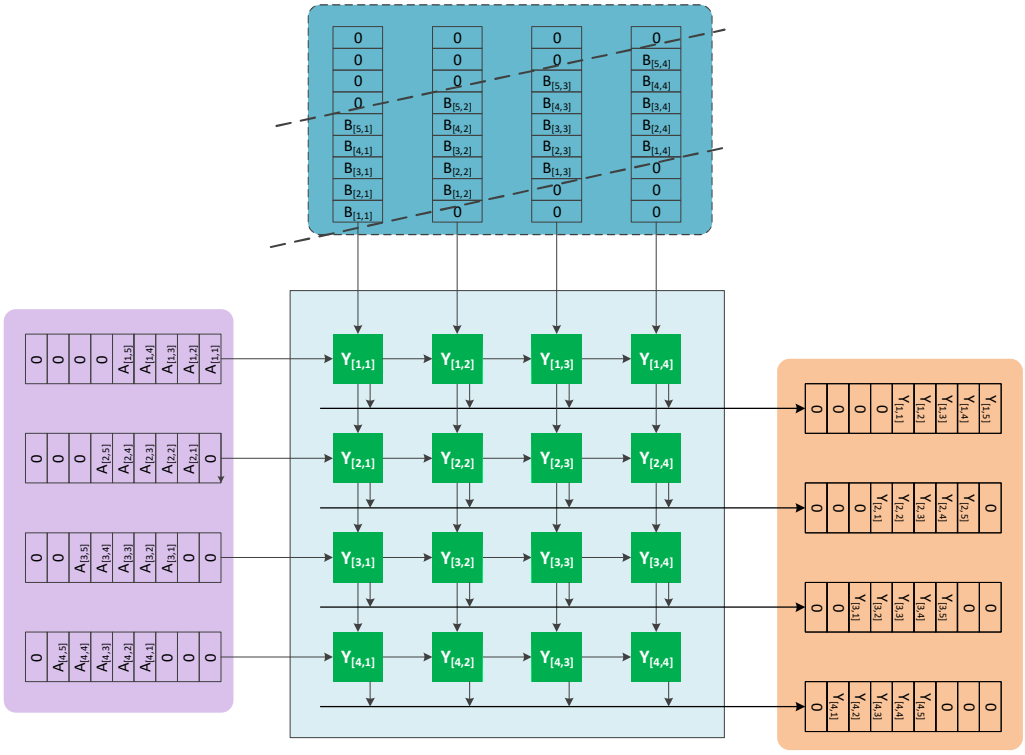


Figure 4 - Matrix Multiplication using Systolic Array

One such arrangement is a *full* systolic matrix multiplier as shown in Figure 4. This architecture provides a very efficient *single pass* matrix multiplication. Note that input matrix values are staggered as shown by the dotted line to allow for a continuous flow of matrices through the systolic array.

The data flow in the systolic structure of Figure 4 requires each processing element **Y** support both a multiplier *and an accumulator*. This is a problem for binary data types like FP32 and fixed-point. The reason is the accumulation, not the multiplication. Note that accumulation is not simply addition but is addition with a feedback loop. This feedback path is shown in Figure 5 with the adder shown in blue in Figure 5. Binary accumulation *feedback* is a bottleneck since fast floating-point addition (blue block of Figure 5) often requires multiple clocks to perform, and therefore each product to be accumulated must wait until the previous addition is complete.

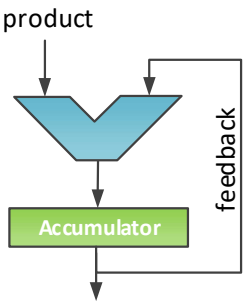


Figure 5

The bottleneck of Figure 5 is one reason TPU designers have abandoned the full systolic matrix multiplication of Figure 4. Instead, many TPU designers move the accumulation stage outside the systolic array. As a result, data buses connecting binary multipliers must transfer each product outside the systolic array, and product summation is performed using less adders than multipliers in the array. This decreases the efficiency and performance of such architectures by requiring more data buses and more data movement, since accumulation cannot be performed to compress (accumulate) the large number of multiplier products.

Fast Modular Accumulation

The RNS TPU does not suffer from the binary accumulator bottleneck. As previously mentioned, each digit of the RNS word is configured into its own systolic array like Figure 6. Within the array, both multiplication and accumulation are modular, not binary. For example, if the RNS digit width is chosen to be 8 bits, the output of a modular multiplier is only 8 bits, not 16-bits wide as in a conventional binary multiplier. In this example, the bit width of a modular accumulator is only 11-bits wide. The accumulation does not grow in bit width in an un-bounded fashion, even for thousands of products summed. It is only when all modular digits of the RNS word are taken together, the total accumulation word-size can be as wide as required.

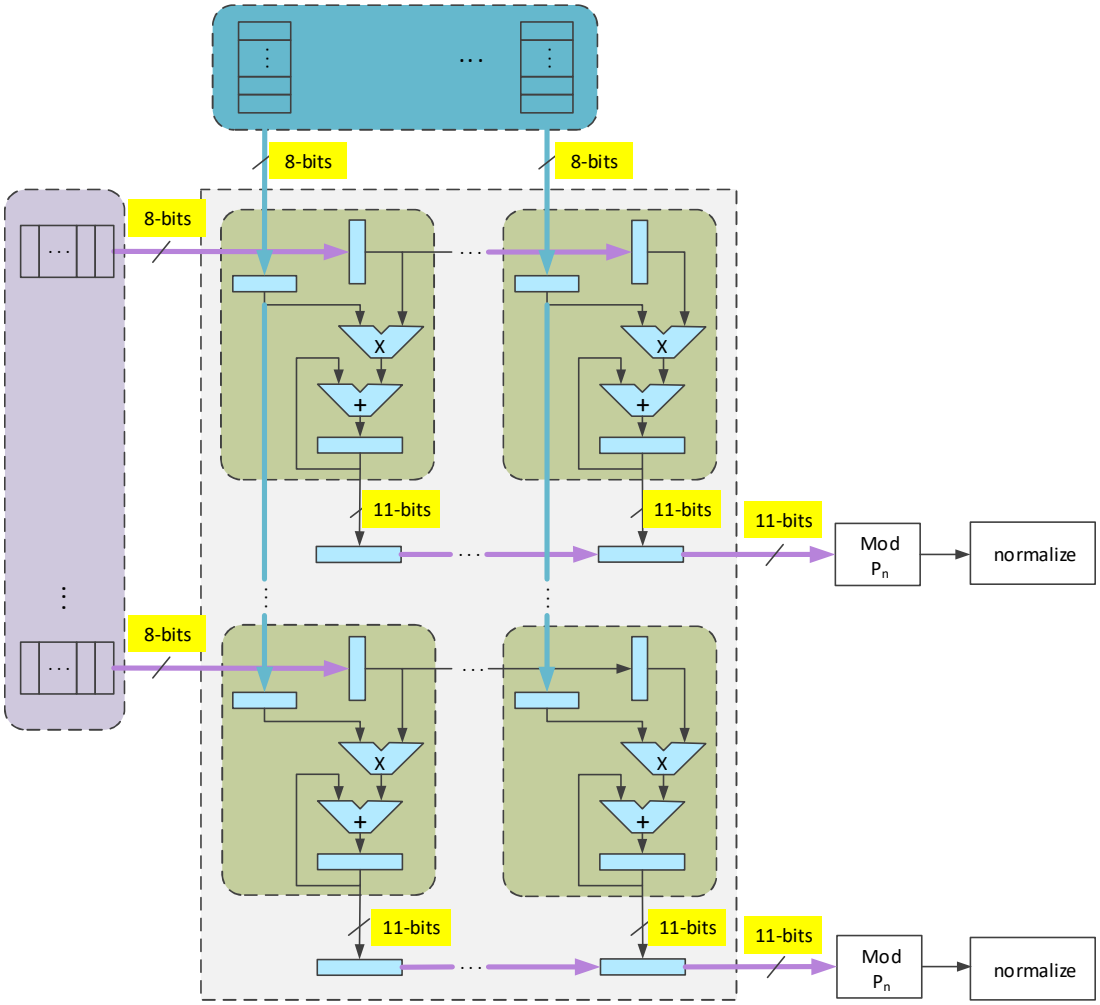


Figure 6 – RNS digit matrix data buses are narrow and efficient

High speed and narrow bus size is a characteristic of the RNS TPU. For example, both modular multiplication and modular accumulation process 8-bit numbers and produce a single 11-bit number as shown in Figure 6. The modular accumulation never exceeds eleven bits even though the modular accumulation of ten 8-bit digit matrix multipliers is more than 78-bits wide! The reason is the full accumulation of each final inner product lies in all ten digits taken together. This means the RNS TPU architecture supports extremely fast and precise operation and does so without suffering from routing delays or large accumulators that are needed to mitigate overflow when using binary arithmetic.

Increased Precision without Reduction of Speed

The enhanced precision of the RNS TPU does not come at a loss in speed because all RNS digit matrix multipliers are operated in parallel. If more precision is required, then the RNS word size is increased. This means more digit matrix multipliers are supported in the RNS TPU. No matter how many digits are required in your application, there is no transfer of data from any RNS digit to any other digit during matrix multiplication. Therefore, the speed of operation remains as fast as a single digit matrix!

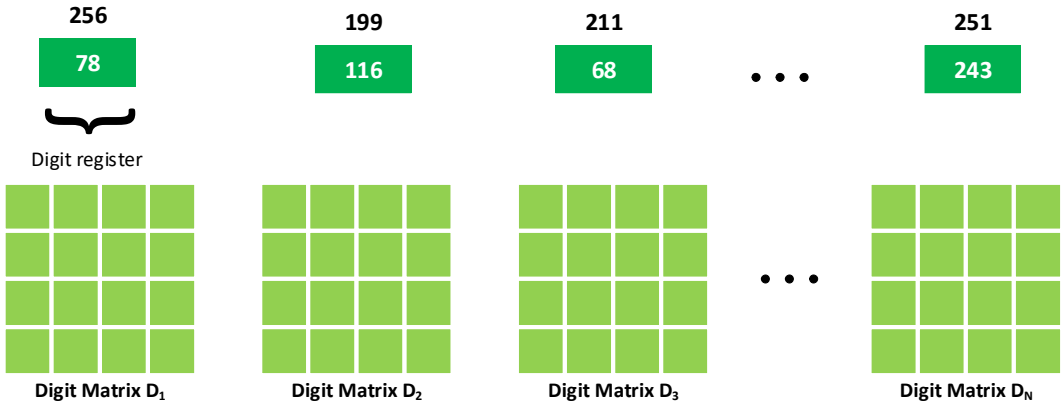


Figure 7 - Small digit width provides speed, increasing digits provides range

In practice, the width of each RNS digit is tailored for a given application. For AI training, a smaller digit bit-width is chosen for increased speed. As shown in Figure 7, an 8-bit digit width can be designed into the RNS TPU. Using ten RNS digits each 8-bits wide, 39 bits of resolution can be supported, which is approximately a 20.20-bit fractional number which exceeds FP16! If more range is required, the system can support more 8-bit digits, or alternatively the design can adopt 9-bit RNS digits. The number of RNS digits operated in parallel can be tailored to meet the range and precision of the AI application. The inner-product result is converted to FP16 or other formats for storage or other processing tasks.

Zero Error Matrix Multiplication

Arithmetic using floating point numbers is laden with errors. The reason is floating point arithmetic is forced to perform rounding on nearly every multiplication and addition. For large matrices, where the number of products accumulated are greater than 100 or even 1000, the effect of rounding is significant. When this rounding error is propagated from layer to layer in AI training, it can lead to poor learning.

The RNS TPU performs with exact precision and zero error. During matrix multiply, there is no loss of precision due to rounding because all accumulation and multiplication is performed in an extended word size. Only a single rounding operation is performed on each inner product of the matrix multiply result. While it is possible to operate using extended precision in binary, it is seldom used because of the long carry delays of each calculation. Because RNS is carry free, the RNS TPU does not suffer from the problem with carry.

For a small 8x8 matrix with a random data spread of +-1.000, the RNS TPU supporting an internal 32.32 fixed-point internal format can calculate matrix inner products more accurately than FP32 over 99.94% of the time! This even considers the input matrix is delivered as FP32 random numbers to start with. While there is generally a small conversion error which occurs when converting FP32 to RNS, the RNS calculation is error free, so the final answer is more accurate than FP32 alone. When measuring the accumulated error versus an FP64 "gold standard", the RNS TPU is more accurate than a straight FP32 calculation by 2.6 times.

Higher precision arithmetic is a requirement for certain portions of AI training algorithms, especially with accumulation of inner products. For example, many TPUs use BFP16 format to speed portions of the matrix multiply, then sum the results of the (partial) matrix multiplication using FP32. Both tasks are obsolete with the RNS TPU. Matrix multiplication is fast because of carry-free arithmetic, and the accumulation of partial matrix results is not required since it is performed in a modular, carry-free fashion, and within a true systolic matrix. Rapid advancement of AI algorithms is underway in many commercial labs and academic institutions throughout the world. Self-training AI models are finding applications into the workplace and industry. These new techniques will require the processing speed, efficiency and accuracy delivered by the RNS TPU.

Linear Multiplier Allocation

Another advantage of the RNS TPU architecture is related to efficient multiplier resource utilization. To increase bit width precision using the RNS TPU, a linear growth of multiplier resources versus word size in bits is required. Alternatively, a squared increase in multiplier resources is required by binary arithmetic as precision increases. Below is a table showing the maximum number of multiplier resources allocated using a Cyclone-IV FPGA for a specific systolic matrix multiplier design. The binary circuit failed to utilize all FPGA multipliers, and the number of DSP slices (unit multipliers) for each binary multiplier is greater than required for the RNS TPU.

Table 1

Arithmetic	Matrix Size	# of mults	Speed (MHz)	MACs ($x10^6$/s)
Binary	6x6	576 (84%)	76	2,736
RNS	9x9	648 (95%)	287	23,247

Because of lack of carry in the RNS TPU, the overall speed of the FPGA based matrix multiplier is significantly faster with the RNS TPU than with traditional binary arithmetic, such as floating point. The resulting performance increase of the RNS TPU over an equivalent binary circuit is 9 times as shown in Table 1 for this specific FPGA test!

Lower Power Dissipation without Sacrifice

Many hardware designers claim their hardware AI accelerators will reduce power yet provide high performance. All our competitors' solutions are based on binary arithmetic; to reduce power, they depend on reducing the word size of AI processing to a bare minimum.

While power reduction can be gained through reduction of word-size, such reduction contributes to loss of precision, which cannot be guaranteed to work for some AI training algorithms now and in the future. High numeric precision is generally not needed for inference phases of AI algorithms, however, FP32 is required for certain portions of the training phases in AI algorithms.

RNS processing allows a more efficient approach to scaling precision by simply increasing or decreasing the number of RNS digits. This does not require partitioning of the internal architecture to support significant differences in numeric format. This leads to a more efficient power profile when optimizing AI hardware for a specific application. However, modern binary formats are still used once processing

is complete. To save power, the RNS TPU converts highly accurate results to the best binary format for storage and other operations.

Many more advantages of the RNS TPU

The RNS TPU provides many additional advantages that would make this whitepaper too long to explain in detail. The following advantages are therefore summarized:

Forward Error Correction of Arithmetic: Maitrix has pioneered the world's first error correction of arithmetic. Maitrix can include its advanced forward error correction as a module to the output to any of our TPU matrix multipliers. Using this configuration, it is possible to operate the RNS TPU in deep space applications. Moreover, by using error correcting arithmetic, it is possible to significantly over-clock RNS TPU circuits to provide the fastest speed in silicon IC arithmetic in the world.

Matrix times a vector configuration: Maitrix provides other matrix hardware configurations, including matrix times vector. Maitrix also provides hardware capable of providing more complex matrix equation solutions and is currently developing full matrix solutions for AI and scientific applications.

For advanced AI applications, the entire AI matrix training iteration can be performed in the modular number system without conversion to binary. To exploit these opportunities, Maitrix is developing both pooling and non-linear activation functions entirely in RNS.

Matrix processor solutions for scientific applications: Many existing and emerging scientific applications require more precision than standard double precision floating-point numbers. In these cases, arbitrary precision libraries are commonly employed to process very wide word widths using software. Some of these applications involve matrix computation of 5000x5000 matrices, requiring many trillions of calculations. The calculations can take days or even longer on the most advanced super-computers available. Because binary arithmetic requires carry and processes a single n-bit multiplication as $O(n^2)$, the cost for matrix multiplication can explode to over $O(n^4)$. Clearly, this is not desirable.

Modular Computation will help alleviate this huge computational bottleneck by configuring matrix multipliers to support very-wide RNS word width. Because RNS is carry-free, very fast operation can be attained on ultra-wide word values. Maitrix is busy evaluating and prototyping ultra-wide word processors with full TPU support.

Matrix processor solutions for other numerical fields: Maitrix excels at high-precision, carry-free fixed-point arithmetic. Maitrix also supports very wide word integer calculations for applications such as factorization and cryptography.

Modular RNS fractions exhibits properties that binary fractions do not. For example, RNS fractions can support exact representations for 1/3, 1/5, 1/7, etc., and combinations of these ratios. This means the RNS fraction is far richer in terms of fractional denominators than binary fractions. For example, binary only supports N number of binary fractional denominators for an N-bit fractional representation. RNS fractional ratios are combinatorial versus the number of fractional moduli. Advanced study as to the significance of this and other properties of the RNS fractional number system have yet to be exploited.

Highly adaptable for 3-D IC technology:

Because modular computation is carry-free, there exist opportunities to implement modular computation technology into 3D IC designs with very high efficiency. Many new applications exist for the future of arithmetic processors whose IC layout geometry is radically different than binary arithmetic.

About Maitrix, LLC

Maitrix, LLC is a soft IP solutions provider specializing in matrix processor IP for ASIC and FPGA implementation. Maitrix pioneered RNS for general purpose computation and is a world leader in this new field of non-binary arithmetic. Maitrix also offers its advanced forward error correction of arithmetic for critical deep space-based applications and high-performance matrix arithmetic. Maitrix error correcting technology is the only commercially available error correction for arithmetic in the world. Maitrix also provides contemporary floating-point solutions for scientific matrix computation and AI and ML training applications.

Maitrix, LLC is based in Henderson, NV and can be contacted at info@maitrix.com.