2022 57th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST) | 978-1-6654-8500-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICEST55168.2022.9828625

A Survey of Three Types of Processing Units: CPU, GPU and TPU

Goran S. Nikolić[1], Bojan R. Dimitrijević[1], Tatjana R. Nikolić[1], Mile K. Stojčev[1]

Abstract – **The CPU, GPU, and TPU are three different types of processing units. For the overall performance of the computer, the CPU is responsible. For delivering high-end graphics and video quality, the GPU is responsible. Along with the CPU, the GPU is a piece of additional hardware. TPU is used in the field of Artificial Intelligence, Machine Learning, and Deep Learning. Each of the three processing units has its own set of functions. This article may be of help to a reader with aim to understand the distinctions between the CPU, GPU, and TPU processing units.**

Keywords – **CPU, GPU, TPU, hardware accelerator.**

I. INTRODUCTION

The present trend of usage of the computer system was significantly changed during the last two decades. From the user's perspective, there has been a huge change in the last twenty years. The user used the computer system not only for the processing of standard data (used for all kind of data accesses of the different forms such as text, image, video, gaming, in a form of structured, semi-structured and unstructured data) but also for solving problems in Artificial Intelligence (AI) [1]. In general, AI has revolutionized many application domains, defeating world champions in the game of Go, surpassing humans in image classification, and achieving competitive accuracy to humans in speech recognition and language translation, to name a few [2]. For AI computational problems, Central Processing Unit (CPU) - based algorithms are not fast enough to give a solution in a reasonable amount of time [3]. Furthermore, these problems can become even larger, to the point that not even a multi-core CPU-based algorithm is fast enough too. Heterogeneous processor systems with accelerators provide an opportunity to improve performance within a given power budget [4]. Many of these heterogeneous processor systems contain Graphics Processing Units (GPUs) and Tensor Processing Units (TPUs) [5] that can efficiently solve problems in graphics and problems used for Machine Learning (ML) [6] and Deep Learning (DL) [7] that characterize embarrassingly parallel computation orders of magnitude faster than a CPU while using less energy.

This paper will provide a short insight and elaborate the basic working principles and scope of applications of heterogeneous computer systems with accelerators, such as GPUs and TPUs, to researchers and motivate them to further harness the computational powers of CPUs, GPUs and TPUs with aim to achieve the goal of high performance.

[1]Goran S. Nikolić, Bojan R. Dimitrijević, Tatjana R. Nikolić and Mile K. Stojčev are with University of Niš, Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Niš, Serbia, E-mail: {goran.nikolic; bojan.dimitrijevic; tatjana.nikolic; mile.stojcev} @elfak.ni.ac.rs

II. INTERFACING COPROCESSOR AND ACCELERATOR

With the rapid development of technology, it is possible to integrate more than 7 billion transistors into one system. However, the more transistors are integrated in the system, the more challenges need to be addressed such as power consumption, thermal emission and memory bottleneck. Therefore, homogeneous and heterogeneous multi-cores are increasingly being used to overcome these issues. Hardware/software co-design is one of the important approaches for multi-core and many-core design. In such systems, there is usually one traditional general purpose processor (GPP) and one or more hardware specific chips that function as coprocessors or/and accelerators with aim to speed up the processing of special kernels of applications running on the GPP. The difference between an accelerator and coprocessor is the following. The coprocessor executes instructions dispatched by the CPU while the accelerator appears as a device on the bus. It is controlled by memory mapped I/O registers (see Fig. 1).

In the rest of this article we will point out to the properties of the more popular accelerator chips such as GPUs (Graphics Processing Units) and TPUs (Tensor Processing Units).

A. Hardware acceleration

In present days' semiconductor manufacturers develop high performance CPUs and hardware accelerators units with aim to efficiently solve complex application problems in computer graphics (CG), artificial intelligence (AI), and other numerous scientific fields of research. Their solutions are mainly oriented towards GPU and TPU accelerator based designs as an alternative to relatively slow only-CPU based ones. In the rest of the text, we will point to the main features that exist among CPU, GPU and TPU electronic devices.

B. Design accelerator challenges

The crucial design accelerator challenges deal with [8]:

a) Debugging – how to properly test the accelerator separately, and then in conjunction with the rest of the system (hardware/software co-simulation);

978-1-6654-8500-5/22/$31.00 ©2022 IEEE

b) Coherency – how to safely share results between CPU and accelerator (the solutions which relate to cache design and shared memory looks similar to those for resource sharing in conventional operating systems); and

c) Analysis – determining the effects of any hardware parallelism on performance which must take into account: i) accelerator execution time, data transfer time, synchronization overhead; ii) heterogeneous multi-threading helps, but complicates design significantly; and iii) overlapping between I/O and computation (streaming).

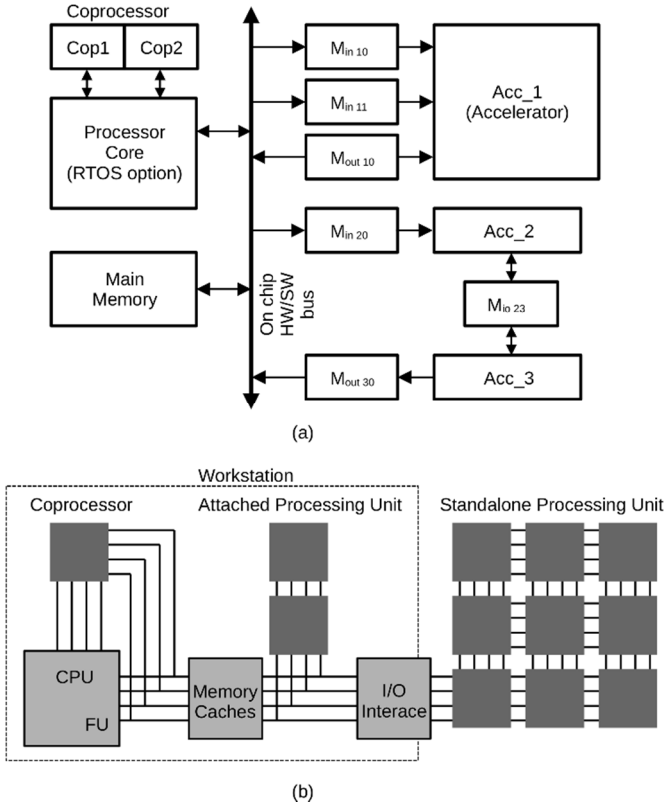

Fig. 1. Interfacing coprocessors and accelerators to a) on-chip bus; b) off-chip system bus

C. Benefits of harnessing accelerators

In general, accelerated systems are characterized by [8]:

1) Usage of additional computational unit - Usually implemented as hardwired logic in addition to the CPU logic, mainly dedicated to perform some specific functions. The hardware/software co-design of such system imposes joint design of hardware and software architectures.

2) Achievement of better real-time performance - Put time-critical functions on less-loaded processing elements, and attain better energy-delay tradeoffs.

3) High-speed operation - Accelerators are good for: a) I/O processing in real-time; b) Data streaming (audio, video, network traffic, real-time monitoring, etc.); c) Specific "complex" operations: FFT, DCT, EXP, LOG, …, d) Specific "complex" algorithms: Neuronal networks, …

4) Overlapping between I/O and accelerator computation - Usually, accelerated systems perform operations in batches, read in second batch of data while computing on first batch. In addition, they find other work to do on the CPU and may reschedule operations to move work after accelerator initiation.

5) Simplicity of the accelerator/CPU interface - Accelerator registers provide control registers for CPU. Data registers can be used for small data objects. Accelerator may include special purpose read/write logic (especially valuable for large data transfers).

6) Performance analysis - Critical parameter is speedup (S): How much faster is the system with the accelerator? This analysis must take into account: i) Accelerator execution time; ii) Data transfer time; iii) Synchronization with the master CPU. The total accelerator execution time is

$$t_{accel} = t_{in} + t_x + t_{out} \quad (1)$$

where t_{in} - data input, t_x - accelerated computation, and t_{out} - data output. Data input/output times include bus transactions such as: j) flushing register/cache values to main memory; jj) time required for CPU to set up transaction; and jjj) overhead of data transfers by bus packets, handshaking, etc. The accelerator speedup, *S*, can be determined if we assume that the program loop is executed n times. Under this assumption we compare the accelerated system to non-accelerated system (expressed as time difference):

$$S = n \cdot (t_{CPU} - t_{accel}) = n \cdot [t_{CPU} - (t_{in} + t_x + t_{out})] \quad (2)$$

where t_{CPU} - execution time on CPU.

7) Single- vs. multi-threaded execution - The most critical factor is the available parallelism. With single-threaded/blocking the CPU waits for accelerator, while with multithreaded/non-blocking the CPU continues to execute along with accelerator. Software must also support multithreading. Accelerator "usual" problems include: a) Memory consistency and coherency (especially if the CPU has caches); b) Partitioning the source code into accelerated chunks; c) Scheduling of the code chunks; and d) Allocation to accelerators (if many).

III. Definitions of Terms CPU, GPU and TPU

Let In the sequel we will define, in short, the meaning of the term CPU, GPU and TPU, respectively.

A. What is a CPU

The CPU is the abbreviation for Central Processor Unit, and it is the unit that carries out most processing inside a computer. It is the head and heart of the computer, i.e. the CPU works as a brains of the computer that perform the basic arithmetic, logical, control and input/output operations specified by the instructions of a computer program [3], [9]. CPU can handle tens of operation per cycle. The CPU comprises two typical components, which are the Control Unit and the Arithmetic Logical unit (ALU). We give instructions to the computer in the language we understand, but because the computer only understands binary, the instructions must be converted to binary. Now, the Control Unit plays a very important part here. The Control Unit extracts the instructions

and decodes/converts them to a language that the computer can understand, i.e., a binary language. After decoding the instruction, it executes it. It also instructs the ALU on what to do, and once the execution of the instruction is finished, it converts it into a human-readable language. The ALU performs all the mathematical and logical operations and follows the instructions of the control unit. The following three features clock frequency, power consumption and performance are crucial for massive usage of each microprocessor design. During the last five decades' numerous different microprocessor architectures have been developed. Many of these architectures are commercially available on the market even today, such as 16-/32-/64-bit microprocessors of Intel families x86, IA-32 and x86-64, respectively. Today these microprocessors are mainly used in IBM PC machines, process controllers and in embedded systems. Other well-known microarchitectures are Alpha, POWER, SPARC, PA-RISC, MIPS and IA-64. During the last 15 years 32- and 64-bit microprocessor cores developed by ARM Limited are very popular among VLSI IC designers. In general, two variants of CPUs exist. The first one is delivered to the market as socket version and is characterized as discrete chip with several thousands of contact points intended for power and data transfer between the CPU and the other constituents installed on the printed circuit board. The second variant is mainly intended for CPU integration within some complex System-on-Chip (SoC) design. SoC is an integrated circuit that shapes the CPU with other building blocks such as memory, digital signal processor, and other input-output peripherals into a single silicon chip. SoC based designs, at present days, are massively used in Internet of Thing, process industry, mobile, real time signal processing and other applications.

B. What is a GPU

The GPU is a special purpose IC which in cooperation with the CPU is primarily intended for fast execution of 2D and 3D graphic functions. Its pipeline organized architecture provides significantly faster processing of graphic information in respect to that of the CPU one. On the market, GPUs are delivered as constituents of video graphics card and used as accelerators for 3D graphics processing [10]. GPU can execute several thousands of operations per cycle what makes this chip to represent a very good design solution for performing rendering tasks in graphics. GPUs are produced in two variants: discrete GPUs and integrated GPUs (see Fig. 2). A discrete GPU (see Fig. 3) is an external graphics processor that is physically located at distance from the CPU, connected to the PCI Express x16 slot on the motherboard PCB. The GPU has its own memory separated from the CPU one. On the other hand, the integrated GPU together with the CPU and other building blocks (including memory, input-output ports, etc.) are built into the SoC integrated circuit. The power dissipation of graphics cards is in order of watts (W). Typical discrete GPUs consume within the range from 80W up to 250W. With aim to cope with the problem of high power dissipation graphics cards with discrete GPUs usually have heatsinks installed. Today graphics cards are very often used as accelerator for execution general-purpose calculations. By using CUDA or OpenCL library it is possible to load the executive code into card processors. Compared to the CPU, higher processing power of the GPU is a consequence to novelty in its architecture. Modern CMPs contain relatively small number of cores (usually no more than 20), while the GPUs were originally created as a multi-threaded structures with many cores (standardly more than 256). The difference in architecture also determines the difference in the principles of operation. Namely, within a SoC design the CPU is intended to perform sequential processing of information, while the GPU is designed for massively parallel processing which is inherent for computer graphics processing.

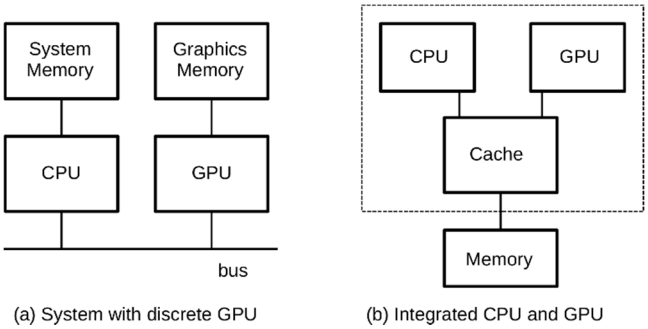

Fig. 2. GPU computing systems include CPUs

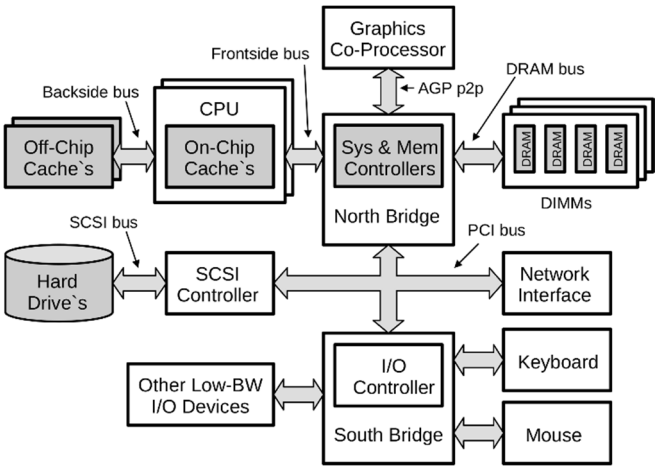

Fig. 3. Typical PC Organization: A System Overview with CPU and a Discrete GPU

C. What is a TPU

From mathematical point of view, a tensor is a geometric object that maps geometric vectors, scalars, and other similar objects in a multi-linear manner to a resulting tensor. In other words, a tensor is a generalized matrix that can be of order 1-D or vector, 2-D array or matrix, 3-D matrix of shape as cube numbers, 0-D matrix (single digit) or a structure of larger dimensions that is difficult to represent. The dimension of the tensor is called its rank. The TPU is an integrated circuit of ASIC (application-specific integrated circuit) type primarily intended to accelerate calculations in AI (Artificial Intelligence) and was designed and developed by Google, mainly used in ML (Machine Learning) and DL (Deep

Learning) applications by using TensorFlow as a programming framework that provides various tools and libraries. In essence a TPU processes geometric objects that describe linear relations between geometric vectors, scalars, and other tensors [5]. The TPU is able to perform 4 Trillion Operations Per Second (TOPS), running only on 0.5 watts of power for each TOPS. The TPU is owned by Google and is not commercially distributed.

D. When to use CPUs

The main feature of CPUs is reflected in the fact that these machines are very easy to program and that they support any programming framework. For example, you can program them in C / C ++, Java, Python or another programming language. In this manner it is possible in very short time to create relatively efficient small programs. But when we solve problems in ML applications and you want to run large models and large data sets then the total execution time for ML training will be inadmissible long.

E. When to use GPUs

GPUs are special function integrated circuits commonly intended for processing image and video information. The GPU is usually composed of numerous simpler processing units in comparison to CPU. These processing units are designed for fast parallel pixel processing of images and videos. Let note that for GPUs programming special languages such as CUDA and OpenCL are used, what in fact limit their flexibility in comparison to CPUs.

F. When to use TPUs

TPUs as integrated circuits (ICs) have been intentionally designed to provide very fast execution of calculations that we meet in AI applications. These ICs perform very fast dense vector and matrix computations and are intended for running very fast program based on TensorFlow. As specialized ICs their flexibility in comparison to CPUs and GPUs is lower when some control- and data-oriented applications are considered. In other words, it only makes sense to use TPUs when the user models applications based on TensorFlow operations.

G. Memory Subsystem Architecture of CPU, GPU and TPU and Compute Primitive

The organization of the memory subsystem and compute primitive for all three previously mentioned processing units is presented in Fig. 4.

H. Typical Technical Characteristics of CPU, GPU and TPU

CPU:

- 1x1 data unit (dimension).
- Instruction execution is characterized by low latency.
- Arithmetic operations are executed in sequential order (for instance, the multiplication [5,4,6] * [3,7,2] = [15,28,12]) =>

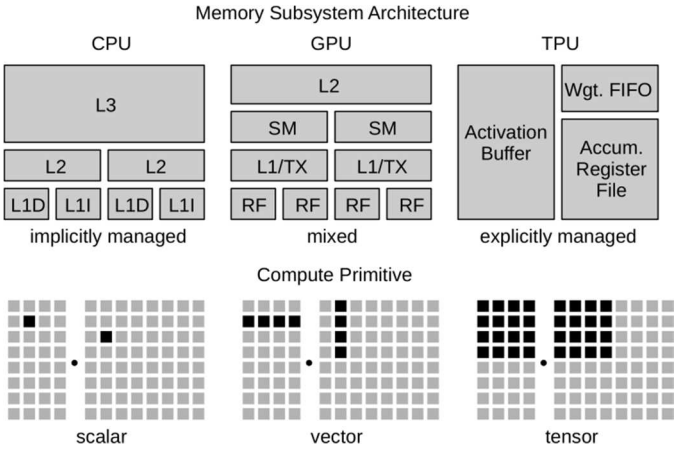

Fig. 4. Memory subsystem architecture for CPU, GPU and TPU and compute primitive

If for one multiplication is needed 2ns then the total execution time for three multiplications is 2ns+2ns+2ns = 6ns.

- CPU (superscalar and super-pipeline) can issue and execute several operations per cycle (performance)
- A CPU as multifunctional unit is designed to solve very large kind of computational problems. The memory subsystem (including primary and secondary storage) is designed to be near optimal for any general programming problem (purpose).
- As general purpose machine the CPU is intended to solve any programming problem (usage).

GPU:

- 1xN data units (dimensions).
- Program execution characterizes high throughput.
- Arithmetic operations are executed in sequential order (for instance the multiplication [5,4,6] * [3,7,2] = [15,28,12]) => If for one multiplication is needed 2ns then the total execution time for all three multiplications is 2ns only, what means that 3 multiplication happens at the same time in parallel.
- Depending of its hardware structure the GPU can execute several thousands of operations per cycle.
- As special designed processor the GPU is primarily intended to accelerate rendering tasks in graphics, ML models and inheritance, and programming problems that characterize parallelism of SIMD type.

TPU:

- N X N data units (dimensions).
- TPU can execute up to $1.28*10^5$ operations per cycle.
- As a coprocessor unit, the TPU is intended to accelerate execution of DL tasks and uses TensorFlow framework. Up to nowadays, compilers for general purpose programming have not been developed yet, so significant errors are needed to perform general programming on TPU.
- The TPU is suitable for ML model training and inheritance (using TensorFlow model).

- The TPU was designed and developed by Google with intent to accelerate computations in neural network computed oriented applications. As a machine constituent the TPU is connected to the host CPU via PCIe bus. During initiation of its operation, the TPU does not fetch the data from the register file for every operation instead the host CPU sends instructions to the TPU.

- A crucial building block of the TPU is a systolic array, SA, (see Fig. 5). The SA is implemented as a network of specialized processing units (PUs) that rhythmically compute and pass data through the system. They derived their name from drawing an analogy to how blood rhythmically flows through a biological heart as the data flows from memory in a rhythmic fashion passing through many elements before it returns to memory.

- The hardware structure of the PU is simple and optimized for low power consumption, whereas the total silicon area occupied by the SA is efficiently used for performing fast matrix multiplication. A major benefit of using SA deals with the fact that all operand data and partial results during matrix operation are stored within PUs (passing through) of the SA. This means that there is no need to access external buses, main memory or internal caches during each matrix operation as is the case with Von Neumann or Harvard sequential machines.

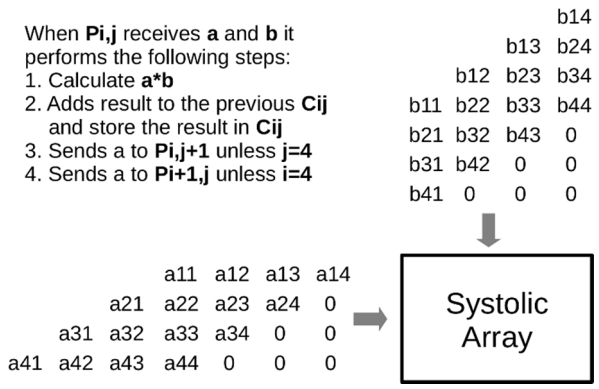

Fig. 5. Systolic Array implementation

Notice: During the execution of matrix multiplication presented in Figure 5, all intermediate results are passed directly between PUs without any memory access. In this manner the TPU's power consumption is significantly reduced whereas its throughput is drastically increased. For SA which consists of 256 × 256 processing elements, there are in total 65,536 PUs. This implies that a TPU can process 65,536 multiply-and-add (MAC) operations (for 8-bit integers) every cycle. When a TPU runs at 700 MHz, it can compute 65,536 × 700*106 = 46 × 1012 MAC operations, or 92 TOPS (92 × 1012) in TPU.

I. Advantages of TPU

Numerous advantages are offered by using TPUs. They relate to increased efficiency and computational speed. As a consequence, the following benefits are obtained: i) Accelerated performance of linear algebra computation (typical for matrix x matrix and matrix x vector multiplication) commonly used in ML applications; ii) The time-to-accuracy during training large and complex neural networks (usually of order several hours) is drastically reduced; iii) Scalable operations of computing machines with installed TPU can be achieved.

J. TPU Limitations

TPUs as special function integrated circuits are primarily designed to perform fast matrix multiplication operations. But, from the other hand, Cloud TPU based machines (mainly realized as TPU servers) are also used in applications that are not dominated by matrix multiplication, including: j) Linear algebra programs that require frequent branching; jj) In programs when memory is rarely accessed; jjj) In programs where high precision arithmetic is used; ivj) In programs that cover neural network applications and contain custom TensorFlow operations written in C++.

The combination of TensorFlow and TPU is very suitable for usage in the fields of medicine, image processing and ML. Since this combination allows the model training time and computation time to be drastically reduced (from weeks to hours), the ML becomes more competitive and attractive for use by a wide range of machine users. This fact causes researchers from various scientific fields (medicine, biotechnology, e.tc), computer and electrical engineers, those who are engaged in business, and even students to start working in the field of ML and successfully implement their ideas and projects.

K. CPUs/GPUs/TPUs Contrasted

Let note that CPUs and GPUs can be used, without limitation, for executing almost all ML oriented tasks, but at a price of an unacceptable long execution time. Some experts believe that TPUs are essentially implemented by using NVidia's GPUs installed in Cloud computer system. In general, having in mind that the semiconductor technologies and hardware architecture are very similar, in the rest of the text we will give a short review, in form of two tables, contrasting the CPU, GPU, and TPU chips. Table I deals with definition and type of their applications, whereas in Table II some technical and implementation characteristics are presented.

At the end of the Subsection 2 we can summarize the following: CPU, GPU and TPU are very different from each other. The variation between the CPU, GPU, and TPU is that the CPU handles all of the computer logic, calculations, and input/output. In comparison to the GPU built into the CPU, the GPU is an additional processor for improving the graphics interface and running high-end operations. TPUs are powerful custom processors for executing a project in a TensorFlow framework.

IV. CONCLUSION

Nowadays, A CPU is like the brain of your computer. The CPU controls all of the other parts of your computer. The GPU can only handle graphics, but some special scientific

applications use the GPU to calculate stuff. A TPU is a special chip for machine learning. In general, all three processing units are designed for different applications.

TABLE I
DEFINITIONS AND APPLICATIONS OF CPU, GPU AND TPU

	Definition	Application
CPU	A CPU controls instructions and data flow to and from parts of the computer, relying on a chipset - a group of microchips located on motherboard	- Quick prototyping - Simple and quickly trained models - Models with small effective batch sizes - Models limited by available I/O or networking bandwidth of the host system
GPU	A GPU is a computer chip developed by NVIDIA that performs rapid mathematical calculations, primarily for rendering images	- Medium-to-large models with larger effective batch sizes - Models used for image processing - Models for which source does not exist or is too onerous to change
TPU	A TPU is an AI accelerator ASIC developed by Google for neural network ML algorithms and, in particular, to work with TensorFlow	- Large and very large models with very large effective batch sizes - Models that train for a long period of time Models dominated by matrix computations

TABLE II
COMPARATIVE ANALYSIS OF CPU, GPU AND TPU

Parameter	CPU	GPU	TPU
Performance	10's operations per cycle	10-103 operations per cycle	Up to 128000 operations
Dimension of data	Unit of 1x1	Unit of 1xN	Unit of NxN
Usage	Normal Programming	Graphical Programming	Machine Learning
Manufact-urers	Intel, AMD, IBM, Samsung	NVIDIA, AMD	Google
Cost of Machine	10-15 $	150-200 $	350-450 $

1. A CPU is the main processor that handles basic math, logic and I/O (input output). Its job is to handle your operating system and whatever you do in your PC. The CPU is designed for general purpose applications, used in all embedded systems or desktops, laptops, mobile phones to run applications apps, or software do normal number crunching at the end result, 32bit or 64bit max at a time, it used windows, Linux or RTOS for micro controller as a OS. It doesn't process graphics.

2. GPU is specifically used for Graphics processing where lot of DSP operations happen like multiplications and addition and custom hardware is built to do this operation. A GPU handles rendering everything you see on your monitor or TV whatever you use

3. A TPU on the other hand or a Tensor Processing Unit processes tensors, or geometric objects that describe linear relations between geometric vectors, scalars, and other tensors. TPU is used for machine learning.

ACKNOWLEDGEMENT

This work was supported by the Serbian Ministry of Education and Science, Project No TR-32009 – "Low power reconfigurable fault-tolerant platforms".

REFERENCES

[1] R. Kuhn, and D. Padua, Parallel Processing, 1980 to 2020, Morgan & Claypool, 2021
[2] M. Negnevitsky, Artificial Intelligence: A Guide to Intelligent Systems, 2nd Ed., Pearson Education Limited, 2005
[3] D. A. Patterson and J. L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, RISC – V Edition, Morgan Kaufmann, 2018
[4] M. Zahran, Heterogeneous Computing: Hardware and Software Perspectives, ACM Books series, #26, 2019
[5] P S. Raj and Ch. Sekhar, Comparative Study on CPU, GPU and TPU, International Journal of Computer Science and Information Technology for Education Vol.5, No.1 (2020), pp.31-38
[6] A. Smola and S.V.N. Vishwanathan, Introduction to Machine Learning, Cambridge University Press, 2008
[7] E. Charniak, Introduction to Deep Learning, The MIT Press, 2019
[8] Y.S. Shao and D. Brooks, Research Infrastructures for Hardware Accelerators, Morgan & Claypool, 2015
[9] A. Gonzalez, F. Latorre, and G. Magklis, Processor Microarchitecture: An Implementation Perspective, Morgan & Claypool, 2010
[10] T.M. Aamodt, W.W.L. Fung, and T.G. Rogers, General-Purpose Graphics Processor Architectures, Morgan & Claypool, 2018