SOLUTION NOTE #1003: Accelerating Cyber Security Applications with MoSys GME (Graph Memory Engine)

Accelerating Cyber Security Applications

Cyber-attacks continue to increase in strength and frequency. This poses a new challenge for both the government and commercial entities across many industries - from finance and health care to the automotive and energy industries - to rapidly address these threats in a real-time manner.

The sheer volume of network traffic that needs to be tracked has increased substantially to the point where the bottleneck between:

- Memory access
- Bandwidth

is worsening and a new solution is needed. In the case of anomaly detection in cyber network traffic, keeping statistics on these anomalies is an important function needed to support detection, which in turn requires using a high random-access rate memory, in support of millions of reads from thousands of histograms in thousands of buckets.

The data-intensive nature of these types of applications becomes especially challenging in situations where real-time detection is paramount for making strategic decisions that could impact national security, energy infrastructure, or prevent the leakage of personal and financial data.

Traditionally security has been done with stateful fire walls in which the header is inspected, and the packet is classified with some sort of filtering data structure. The stateful aspect comes from tracking changes in the state of the connection requiring rapid updates to memory. When packets arrive every 6ns (100GE ➔150Mpps), memory like DRAM is challenged because it is limited to 50ns updates, thus the need for SRAM.

For traffic that is encrypted only a portion of the packet features can be inspected, thus other aspects like packet size, rate, spectral aspects can be analyzed. More recently, Random Forest of Trees (RFT) are being used a classification technique for aggregate traffic flows and in some cases even individual packet classes. In order to effectively identify and address attacks, it is important to perform detection in real-time and accurately flag specific data patterns. In this application, a monitor-detect-classify sequence is required. If available, it is crucial to monitor the distribution of packet features, including: sIP, dIP, sPort, dPort, package type, size, intervals, etc., and detect correlation in the distribution of the packet features by using classifiers (such as a random forest of trees) to identify and classify attack type. The goal is to raise awareness of a possible event in as short a time as possible between monitoring and detection such that the potential threat or intrusion can be eradicated before it infiltrates an organization's networks.

Random Forest of Trees (RFT) is a very powerful technique, but it is limited by the random performance nature of the underlying memory. DRAM for example can limit the performance of even the most powerful multi-core CPU because in the worst case the read bank cycle time is 50ns and the latency can be >100ns. When an algorithm like RFT is used, each read is dependent on the results of the previous read. Memory technology like MoSys 1T SRAM and its in-memory-compute can greatly reduce the time to process the serial nature of tree structures.

MoSys Solution

The MoSys (Graph Memory Engine) GME utilizes a proprietary high random-access rate memory with in-memory-compute functionality for searching and classifying in applications such as genomics, bioinformatics, and network security. The GME has been ported onto a MoSys monolithic IC. The Programmable Hyper-Speed Engine (PHE) has

1 Gb of internal memory, 32 RISC processing cores, and supports an on die bandwidth of ~1.5 Tb/s.

The 1T SRAM is exclusively provided by MoSys and offers many of the features of traditional 6T SRAM but with a much smaller unit cell, allowing for high memory density and lower latency.

Figure 1 illustrates the architecture of the PHE, which includes RISC cores, RMW logic and 1T SRAMs. Most of the additional speed-up occurs from the ability to have the RISC cores on the same die as the memory. The two are connected through a cross-point switch. The GME is a function that MoSys has ported onto the *in-memory* processors to function as an offload of the FPGA to free it up to execute other proprietary workload(s).

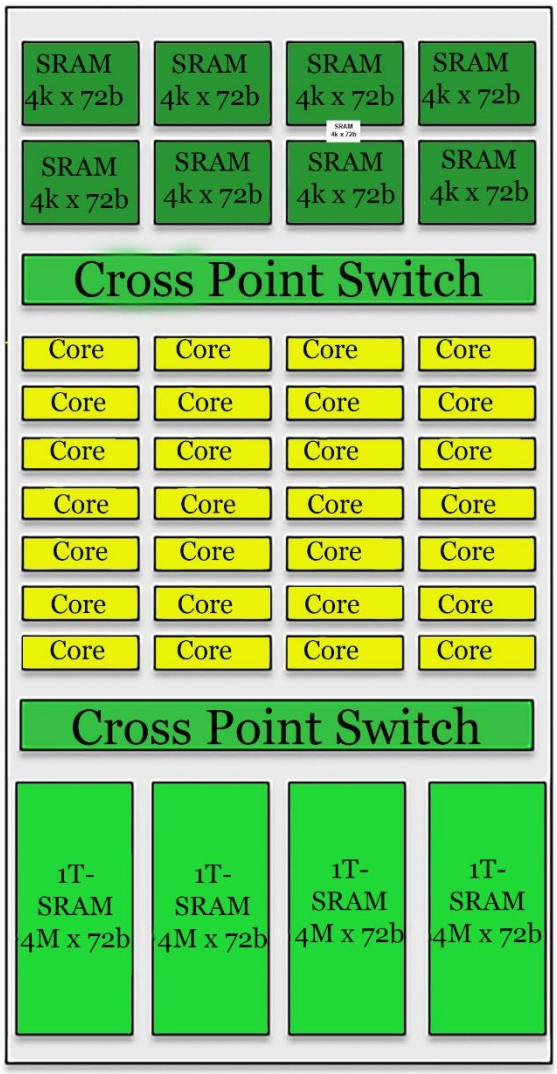

Figure 1. Block Diagram of a PHE

Unlike traditional classification functions running on a processor, the GME is not limited by DRAM access rates and latency; instead it leverages the PHE's ability to achieve very high random-access rates to the on die 1Gb of high-speed memory. The architecture of the PHE enables graphs to be processed at 3 to 6 Billion nodes per second with an average latency of ~30-50 ns per node.

Figure 2 illustrates an example graph structure, which can be implemented to support functions like network security and genomics. Performance of algorithms used in hierarchical data structures, like random forest of trees, decision trees, and graphs is directly dependent on the number and depth of the graphs/trees required to perform a classification. In addition, the speed of traversing the graph is directly linked to the memory access rate of the processor.

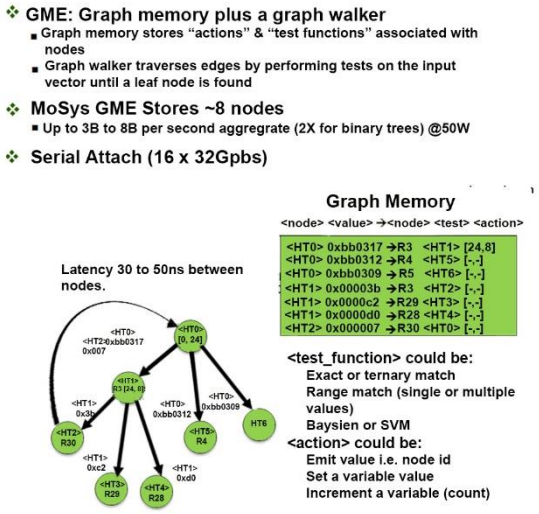

Figure 2. Demonstration of GME used for Graph Walking.

Processing these algorithms is quite frequently memory-bound and not compute-bound because each decision in the simplest form is a numeric comparison. The performance of GPUs and CPUs will improve as memory access rates improve. Rates of DRAM have marginally improved over many years, while the volume of data, processor speed, and algorithm efficiency have increased significantly. Simple monolithic memory solutions will not fix this bottleneck. A new approach can bring the memory and processing

elements into a tightly coupled package that can enable random access memory acceleration.

Going beyond GME, which is a generalized flexible solution, the PHE can used to execute code and data structures specifically optimized for Random Forest of Trees. All told, the PHE is capable of up to 18B (6B + 12B) read/sec utilizing the 1T-SRAM and SRAM local to the processor cores. Compressing 2 to 3 levels of a tree in each double word fetch, further acceleration can be achieved yielding 36B nodes/sec. If a classification task requires, 32 trees with depth of 6, then PHE could perform the task on the order of >100M per second. With memory access latencies 25ns and 2 levels per access, room for instruction execution and the parallel processing with 32 cores, the aggregate latency can be around 400ns for a complete classification task (32 trees x 6 levels). With optimized code mainly limited by the random-access rate of compressed tree structures in the integrated memory, the projected performance/Watt will be between 20-100x over GPUs, CPUS and pure FPGAS, as shown in Figure 3 below.

Figure 3. Random Forest of Trees Performance Comparison PHE vs. FPGA, GPU and CPU.

Applications:

- Network Security
- Machine Learning
- Genomics
- Neural Networks

MoSys is always interested in how you found the ideas presented in this solution note, so any feedback would be greatly appreciated and will support us in what future topics and data will be addressed in future solution notes.

If you are looking for more technical information or need to discuss your technical challenges with an expert, we are happy to help! Email us and we will arrange to have one of our technical specialists speak with you. You can also sign up for our newsletter. Already convinced? You can request a quote from sales. Finally, please follow us on social media so we can keep in touch.