02/14/2022

Discussion revolved around the Architecture and throughput and latency related to I/O operations. Back in 1995, GPGPUs are used for the video cards for the sake of computation where 3dfx invented #D video card for the playing games. The cost to build the hardware was expensive in 80s and SIMD used indexing and created the required computation units running on 4x4 matrix. Currently, we have multiple units to achieve the concurrency – 3080ti.

In 2005, several languages are tried to present the computation and a sophisticated and general language is used to express the same using the videogame cards for it. Nvidia launched Tesla cards with no video output rather computation output. Games require the precision and Tesla doubled the precision for scientific computation. Later, with the involvement of AI, FP16 and FP8 managed to avoid most of the computations and need less precision. Upon comparing GPGPU and Processor, interesting facts like GPGPU has a very restricted instruction set and requires kernels which are difficult to prepare. Performance was badly affected with branching and has fast access to the memory. With the new generations, some part of the external memory is reserved for caching mechanism. Restrictions around memory: GPGPU can have 80GB and whereas the processor can extend to 4TB with the extra necessity to transfer of memory to main memory to GPGPU.

Another such discussion around PlayStation 3, used a cell processor with 1 general purpose processor and 8 specialized processors with more explicit memory transfers. Intel Phi produced the idea to cluster all the components into a single chi and used X86 processor in turn reduced the out of order executions and saved the costs by saving silicon and invested the same on adding more cores to around 58.

02/16/2022

Session uncovers the concept of transfer of the memory from the memory to GPGPU with an external device that prefetches the job and uses it for complex data structures and gets the data explicitly transferred. Back in 80s, the speed of processor is way too high, and we used to stop the processor to send the IO Instructions to the peripherals via IO ports. Also, allowed the peripherals to insert the data directly to memory using DMA which is controlled by a controller. CPU manages the DMA to make necessary data transfers to the GPGPU based on the necessary requirements in 4k multiples.

DMA Transfers the data concurrently and OS takes care of the scheduling of the transfers, and the changes made are directly on the real memory not on the virtual ones requiring the pinned pages. Controller which plays a key role has the direct connection to the memory and PCI bus. Numa nodes have this feature as an option. Can experience the fast data transfers if the devices are directly connected to the PCI bus and results to a conclusion that large transfers require less setup and are faster.

PCI bus which are connected to the DMA has individual wires, and GPGPU – 16 PCI and NVME has 4 and 50 Gbit Network card has 8 and High-end server has 160 lanes and 10 GPGPUs contribute to 288GB/s. Coming to the disks we have SAS and SATA protocols contribute to 6-12 GBits/s, whereas the SSDs outperform disks with much faster speeds and is now evolved to NVMe.

02/18/2022

In the previous session, we discussed on several protocols required for the disks to be able to connect to host bus adapters like SATA, SAS. Today's discussion is all about Serial AT Attachment (SATA) – which is bus interface, used to connect storage devices like external disks, SSDs, etc. It requires a dedicates SATA controller which does all the communication on PCI. It matters in the process of communication, there is a delay in less than 100ms with the SATA disks, also the bandwidth of versions varies and ranging from 1.5 Gb/sec - 6 Gb/sec. The only difference with the evolution of different versions is the bandwidth speeds with the higher versions having a feature of backwards- compatibility. It is a serial connector and parallelism is not obtained with the SATA we must rely on PATA for the same. I/O operations, sequential are better than the random.

Nextly, NVME, which is a host controller interface and a storage protocol used to accelerate the rate at which the transfer speeds to increase. There are several usages of it: disks, cache, backup memory, etc. Considering the I/O speeds are blazingly fast ranging 300,000/sec, with a throughput of 2.8 – 7.2 GB/s. It has 4 lances of PCI which can throttle to get the throughput. Although, PCI has the bottleneck, it uses multi chip parallelism to evade the same. We can find the asymmetry between the read and write I/O operations, this leads to no failure recovery when and cannot proceed to write anymore. There are two main ways to obtain the peak performance of it namely, making use of the parallel requests (64) and use of the large sequential reads and 4K request at a time to get almost 64X loss in the performance. When 48 of such NVME are connected and it creates 3072 parallel requests obtains 128-1 GB of pages instead of the same 4K page size. Mechanical drives achieve single digit <10 ms latency, can achieve about 200 MB/sec sequential reads and 100-500 reads or writes / sec. The same goes with it too, cannot perform the read or write upon failure. It has around 32-64 requests queues to hold the pending requests.