Article

Prospects of GPU Tensor Core Correlation for the SMA and the ngEHT

Wei Yu [1,*], John W. Romein [2], L. Jonathan Dursi [3], Ru-Sen Lu [4], Adrian Pope [5], Gareth Callanan [6], Dominic W. Pesce [1,7], Lindy Blackburn [1,7], Bruce Merry [8], Ranjani Srinivasan [1], Jongsoo Kim [9] and Jonathan Weintroub [1]

1 Center for Astrophysics | Harvard & Smithsonian, 60 Garden Street, Cambridge, MA 02138, USA
2 ASTRON, Netherlands Institute for Radio Astronomy, Oude Hoogeveensedijk 4, 7991 PD Dwingeloo, The Netherlands
3 NVIDIA Canada, 431 King St W, Toronto, ON M5V 1K4, Canada
4 Shanghai Astronomical Observatory, Chinese Academy of Sciences, 80 Nandan Road, Shanghai 200030, China
5 Argonne National Laboratory, 9700 S. Cass Avenue, Lemont, IL 60439, USA
6 Department of Computer Science, Lund University, Ole Römers väg 3, 223 63 Lund, Sweden
7 Black Hole Initiative, Harvard University, 20 Garden Street, Cambridge, MA 02138, USA
8 South African Radio Astronomy Observatory, 2 Fir Street, Black River Park, Observatory 7925, South Africa
9 Korea Astronomy and Space Science Insitute, 776 Daedeok-daero, Yuseong-gu, Daejeon 34055, Republic of Korea
* Correspondence: wei.yu@cfa.harvard.edu

Abstract: Building on the base of the existing telescopes of the Event Horizon Telescope (EHT) and ALMA, the next-generation EHT (ngEHT) aspires to deploy ∼10 more stations. The ngEHT targets an angular resolution of ∼15 microarcseconds. This resolution is achieved using Very Long Baseline Interferometry (VLBI) at the shortest radio wavelengths ∼1 mm. The Submillimeter Array (SMA) is both a standalone radio interferometer and a station of the EHT and will conduct observations together with the new ngEHT stations. The future EHT + ngEHT array requires a dedicated correlator to process massive amounts of data. The current correlator-beamformer (CBF) of the SMA would also benefit from an upgrade, to expand the SMA's bandwidth and also match the EHT + ngEHT observations. The two correlators share the same basic architecture, so that the development time can be reduced using common technology for both applications. This paper explores the prospects of using Tensor Core Graphics Processing Units (TC GPU) as the primary digital signal processing (DSP) engine. This paper describes the architecture, aspects of the detailed design, and approaches to performance optimization of a CBF using the "FX" approach. We describe some of the benefits and challenges of the TC GPU approach.

Keywords: ngEHT; VLBI; SMA; correlation; GPU; Tensor Core

check for updates

Citation: Yu, W.; Romein, J.W.; Dursi, L.J.; Lu, R-S.; Pope, A.; Callanan, G.; Pesce, D.W.; Blackburn, L.; Merry, B.; Srinivasan, R.; et al. Prospects of GPU Tensor Core Correlation for the SMA and the ngEHT. *Galaxies* **2023**, *11*, 13. https://doi.org/10.3390/galaxies11010013

Academic Editor: Phil Edwards

Received: 15 November 2022
Revised: 5 January 2023
Accepted: 9 January 2023
Published: 11 January 2023

1. Introduction

The Event Horizon Telescope (EHT) is a globe-spanning Very Long Baseline Interferometry (VLBI) array that has captured images of the shadow region of black holes at the center of the galaxy M87 and the Milky Way [1,2], attracting worldwide attention. The next-generation EHT (ngEHT)[1] will push this scientific frontier even further, building ∼10 more stations and doubling the observation bandwidth to improve the imaging capabilities and capture the first black hole movies.

The future EHT + ngEHT array will include ∼21 stations. For each station, the sideband bandwidth will increase from the current 4 GHz to 8 GHz. Dual polarization, two sidebands, and simultaneous dual-frequency (230 GHz and 345 GHz) observations correspond to a bandwidth of 64 GHz and recording data rate of 256 Gbps for two-bit recording and Nyquist sampling [3]. Current EHT observations are 7 days per year, while ngEHT observations are expected to run 60 days per year. Thus, compared with the current EHT, the future array will have more stations, a wider bandwidth, and more observation days each year. The amount of observation data is anticipated to be tens of petabytes in size, which brings great challenges to the correlation. This is the motivation to build a dedicated high-performance correlator[2].

The Submillimeter Array (SMA) is a standalone radio interferometer consisting of eight antennas. A digital correlator-beamformer called SWARM [6] functions as two distinct instruments: an FX correlator that computes fringe visibilities across frequencies for every pair of antennas, and a beamformer that forms the coherent phased array sum of the eight antennas, aggregating the SMA collecting area to an equivalent single large-aperture telescope. The beamformer mode was designed in SWARM to enable its participation in EHT observations.

In order to match the observational capabilities of the ngEHT and the Wideband SMA Project (wSMA), it is desirable to upgrade the correlator-beamformer of the SMA. Likewise, the VLBI correlator now supporting EHT is limited in bandwidth, which can be processed in a reasonable computing time scale, and also the number of stations that it can handle. We propose a common GPU Tensor Core-based architecture for both of these wideband applications.

This paper reports on the prospects for a correlator-beamformer system having common features that can potentially benefit wSMA and ngEHT. The primary focus of the work described here is the design of an X-engine prototype based on an open-source Tensor Core library [7]. We note that we have already built a small prototype two-server four-GPU system, on which we have micro-benchmarked and tuned some of the codes and subsystems described in this work. We have made extensive use of the NVIDIA profiling tool Nsight to optimize subsystem performance. We have not yet built the full system described in this paper; however, we continue to actively develop the experimental counterpart to this work.

2. Related Work

The correlator-beamformer (CBF) is a key data processing system for radio interferometers and VLBI arrays. The correlator cross-correlates the baseband data of each antenna/station pair and outputs visibility data for imaging, while the beamformer aggregates the collecting area of the array for VLBI operations.

The correlator function of the CBF is divided into two broad classes. The XF type computes a direct cross-correlation between time series data from pairs of telescopes before transforming to frequency with Fast Fourier Transforms (FFT). The FX type computes the Fourier transform of time series data from each station first, followed by a cross-correlation. The two processing stages are called F-engine and X-engine, respectively. In the F-engine, data from each antenna/station are divided into multiple frequency channels. In the X-engine, a subset of frequency channels from all antenna/station pairs are bin-wise multiplied and the result integrated.

The trade-off between the XF and FX correlation architectures is multidimensional. For high spectral resolutions, the use of FFTs reduces the number of required multiplications in aggregate, although, because of bit growth in the FFTs, these multipliers need to be wider. There are other trade-offs in respect to memory utilization and the number of baselines. Reference [6] discusses multiplier utilization quantitatively and recommends the FX approach. Whereas XF correlators had been favored historically because wide multipliers were expensive, we observe that almost all new correlator designs (in the last decade, for example) use the FX architecture. We attribute this to the wider availability and lower resource cost of wide multipliers. In addition, FX correlators naturally provide more parallelism through early spectral decomposition. Modern correlator platforms mainly use Field-Programmable Gate Arrays (FPGAs), Graphics Processing Units (GPUs), or CPUs, which are both well equipped with sufficiently wide or even floating point multiplication and are designed at the root for a very high degree of parallelism.

FPGAs have the advantage of hardware programmability. Almost all of the existing hardware correlators are developed with FPGAs. Among them, the Collaboration for Astronomy Signal Processing and Electronics Research (CASPER)[3] open-source FPGA platform is the most popular solution; for example, the SWARM correlator for the SMA [6], the correlator for MeerKAT [8], and the correlator for the Arcminute Microkelvin Imager (AMI) array [9] are all developed with CASPER FPGA boards.

The powerful parallel computing resources of GPUs present an opportunity, and correlators are increasingly adopting GPUs for accelerated computing, especially in X-engines. Hybrid FPGA+GPU architectures have become a very popular solution for correlators. For example, the Canadian Hydrogen Intensity Mapping Experiment (CHIME) [10], the Large Aperture Experiment to Detect the Dark Ages (LEDA) project of the Long Wavelength Array (LWA) [11,12], the Donald C. Backer Precision Array for Probing the Epoch of Reionization (PAPER) [13], and the Murchison Widefield Array (MWA) [14] all adopt the hybrid FPGA+GPU architecture for their correlators, where FPGA boards (including ADCs) are used for data acquisition and channelization (F-engines), and GPU boards are used for cross-correlation computing (X-engines). There are also some correlators developed with pure GPUs, such as the early MWA 32-antenna prototype correlator [15] and the Cobalt correlator of the Low-Frequency Array (LOFAR) [16]. In these systems, the GPUs are used for the fine channelization of F-engines and cross-correlation of X-engines. ADCs and FPGAs are still required for data acquisition and coarse channelization.

For VLBI arrays, data acquisition and correlation are separated due to the remote station locations. Raw station data are recorded and shipped to a central location for correlation. VLBI correlators are typically CPU-based (software-based). Many VLBI institutions have developed a variety of software correlators. Among them, DiFX [4,5] is the most widely used one. Facilities such as the VLBI Global Observing System (VGOS), the Very Long Baseline Array (VLBA), the Australian Long Baseline Array (LBA), and the EHT [17] all use DiFX to correlate data. Recently, the EHT has been investigating a Cloud-based correlation scheme [18], which is also based on DiFX.

For VLBI and ngEHT, we sometimes use the term "near-real-time" to describe the desired performance of a wideband correlator. VLBI correlators use recorded data and therefore need not be strictly real-time. "Near-real-time" under these conditions signals that it is desirable to improve the efficiency of media recycling, reducing media costs, which can be very high for the wide bandwidths, high cadences, and increased number of stations of the ngEHT. As a side benefit, faster correlation reduces the time to science and improves fringe search iteration times and efficiency, and feedback on prior campaign results can improve the planning of future ones.

While various architectures have been introduced above, in this paper, we propose a GPU Tensor Core-based correlator for the wSMA and the ngEHT for the following reasons.

(1) GPUs have powerful computing resources, which are far more efficient than CPUs for parallel computing. In particular, the newly embedded Tensor Cores in NVIDIA GPUs are much more efficient than regular CUDA cores as well as FPGAs in terms of matrix multiplication [7].

(2) There is a wealth of open-source software libraries (both GPU-based and CPU-based) available for radio astronomical instruments. Although CASPER also has various open-source libraries, users still sometimes need to develop new firmware modules with the hardware description languages (HDL). Compared with CASPER libraries and HDL, the GPU source code is easier to upgrade and modify. The GPU development ecosystem includes sophisticated debugging and optimization tools and uses standard languages familiar to a wide swath of software engineers.

(3) The architecture of GPU-based correlators is flexible, maintainable, and scalable. High-performance hardware such as CPUs, GPUs, network interface cards (NICs), etc., can be integrated together easily.

3. System Design and Selection of Open-Source Libraries

We propose two schemes for the architecture of the correlator. In the first "expanded" scheme, the F-engines and X-engines are implemented in different GPU servers, which is shown in Figure 1. The corner-turner transpose between the two types of engines is implemented through the network switch. Each network packet from the F-engines contains data containing a subset of frequency channels. Different subsets of packets have different destination IP addresses and are transmitted to different X-engines through the switch. Thus, each X-engine processes a subset of frequency channels from all of the F-engines to form all of the array baselines.

In the second "compact" scheme, the core functions of the F-engine and the X-engine are arranged to execute in the same GPU server. This compact scheme reduces hardware resources such as GPUs, NICs, and CPUs. Thus, it is the preferred architecture for scaled deployment for smaller arrays. Since both wSMA and ngEHT have relatively few antennas and baselines, the compact scheme is potentially attractive. The expanded first scheme is simpler to build and debug, is the better starting point for development and laboratory debug, and is preferred for larger wideband arrays, such as ALMA.

We discuss the compact scheme in more detail in Section 7.

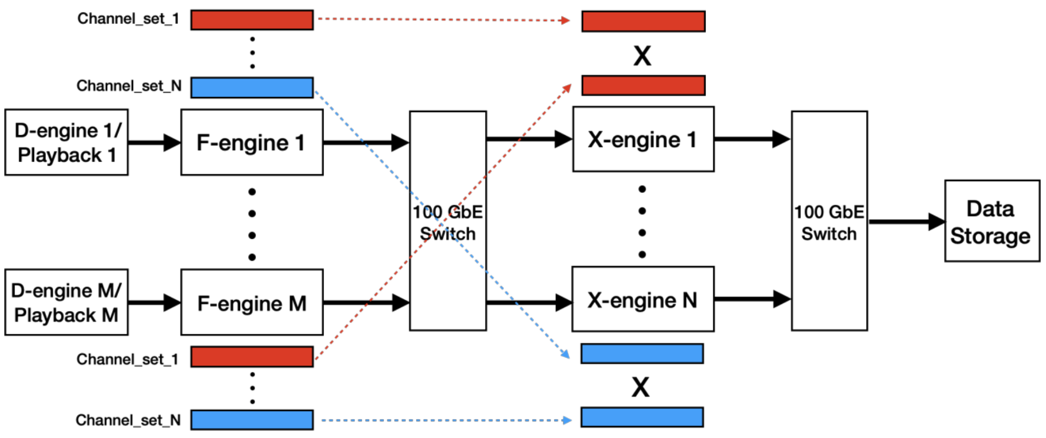

Figure 1. Level architecture of the proposed packetized FX-type correlator pioneered by CASPER. From left to right, note the D-engine samplers or VLBI playback recorders, the F-engines, a fast network switch implementing the corner-turner transpose, the X-engine, and archival visibility data storage. Not shown is the beamformer B-engine typically co-located with the X-engine.

The main difference between the wSMA real-time correlator and the ngEHT VLBI correlator is that the former uses D-engines to generate data, while the latter uses playback servers. A D-engine is a hardware system with ADCs and FPGAs to digitize the analog signals from each antenna, and it then transmits the data to the corresponding F-engine. The D-engine FPGA firmware optionally includes data processing modules that implement coarse channelization, slope and ripple equalization, and requantization[4].

Based on this same ADC-FPGA hardware, we are developing a wideband digital backend for the ngEHT, as shown in Figure 2, which includes a Xilinx VCU128 FPGA evaluation motherboard and a sub-board with 4 $\times$ 16 Gsps ADCs. This backend can be used as D-engines for the wSMA correlator with only slight modifications of the FPGA firmware. For the ngEHT correlator, the playback servers play back the recorded data from the digital backends to the corresponding F-engines.

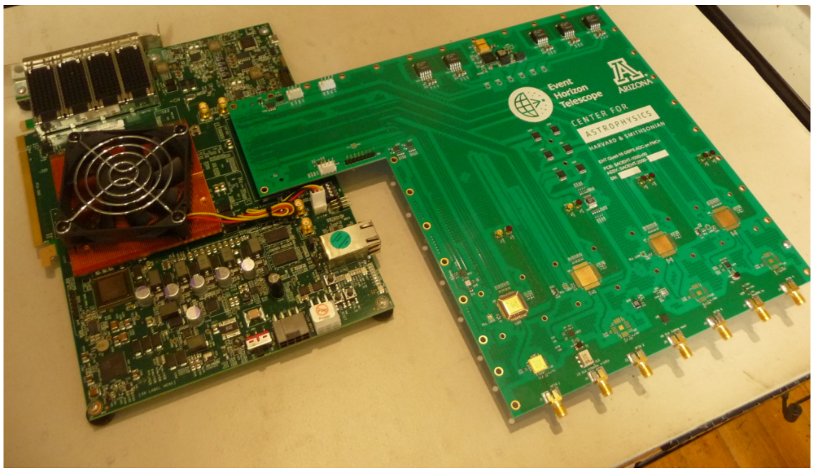

Figure 2. The FPGA evaluation and custom ADC board set proof-of-concept platform. This is a general-purpose open-source platform, which, at SAO, will find application as the ngEHT digital backend and wSMA D-engine. The left board in the set is a commercial-off-the-shelf (COTS) Xilinx VCU128 FPGA evaluation motherboard. The right board is a custom plug-in to the VCU128's "FPGA Mezzanine Card +" (FMC+ or VITA 57.4) standard high-speed connector. This custom ADC board has four analog inputs leading to four Adsantec ASNT7123A ADC chips, each proven to run at 16.384 gigasamples-per-second (GS/s). The 4 × 100 GbE QSFP28 network ports on the FPGA motherboard can connect to the F-engines of the wSMA correlator or ngEHT recorders.

For the F-engine, a GPU-based library called katgpucbf[5] developed by the MeerKAT team is being considered. We have installed and tested the F-engine of the library on our experimental server. The next step is to modify it according to the specific requirements. A geometric delay model is required by both applications. The ngEHT VLBI correlator will have much more rapid fringe (or delay rate) correction. It also needs fringe search features, an across delay and delay rate using a geographic model of station placement, and a VLBI Data Interchange Format (VDIF) deformatter. The present focus of this work is on the correlation engine with delay and fringe correction development, and VLBI fringe search features, planned for future development.

Another library under consideration is the new cuFFTDx library developed by NVIDIA. As it generates FFTs that are GPU-callable, it is possible to merge FIR filters, FFTs, and phase corrections in a single GPU kernel instead of three. As a result, only one pass over the data is made, and as each of these functions is limited by the GPU memory bandwidth, one can expect an almost three-fold speedup.

For the X-engine, we developed a pipeline based on open-source libraries. Key components include a network transmission/receiving module, a format conversion module, a GPU-based correlation module, a long-term accumulation (LTA) module, and a pipeline framework to manage the above modules. We review and evaluate the existing open-source libraries and select the appropriate ones in the following sections.

The open-source libraries for network transmission/receiving that were under consideration include PF_ring[6], jive5ab[7], and SPEAD2[8]. PF_ring has been adopted by Mark6 equipments to capture network packets from VLBI digital backends. jive5ab is mainly used for data transfer and recording in e-VLBI systems, which has been used in Mark5/Mark6 and FlexBuffer by the European VLBI network. SPEAD is a data format for radio astronomy. SPEAD2 is a python/C++ library with the functions of the SPEAD formatter/deformatter and the network system. It supports both the traditional networking stack and the verbs API [19]. In order to match the MeerKAT katgpucbf F-engines, we select SPEAD2 as the basic library for the network transmission/receiving module.

The original and most popular GPU correlation library is xGPU [20], which has been adopted by many radio arrays, such as the LWA, MWA, PAPER, etc. When xGPU was developed 10 years ago, Tensor Cores had not appeared yet, so regular CUDA cores were used for correlation computing, which are not as efficient as Tensor Cores. The CHIME team developed a correlation library based on the AMD GPU. The AMD offerings feature higher computational throughput per unit cost than the comparable NVIDIA GPUs [21]; however, the former is more difficult to program than the latter. The recently developed Tensor Core Correlator (TCC) library adopts Tensor Cores as computing resources; the library resolves the complexity of using Tensor Cores and addresses several optimization challenges, such as the missing hardware support for complex numbers [7], which makes it much more efficient than xGPU. Considering the computational efficiency and development threshold, we select TCC as the correlation library.

The format conversion module is used to implement data conversion between F-engines and X-engines. For the implementation platform, we have two options, CPU-based and GPU-based. The input data of the xGPU library must be in the host memory, so the format conversion module must be implemented on the CPU. TCC does not have this restriction, so we implement this module on the GPU, which is much faster than on the CPU. Some observations require long-time integration, and the GPU memory may not be able to cache the pre-integrated data for such a long time. Thus, the TCC library is only used for short-term integration, and an LTA module is required after TCC, which can also be executed on the CPU or GPU. Currently, we implement this module on the CPU.

In order to improve the throughput, each of the above modules should be executed on an individual CPU thread under a framework[9]. Some open-source frameworks for radio astronomy include kotekan [22], PSRDADA[10], bifrost [23], and HASHPIPE[11]. Among them, kotekan, PSRDADA, and bifrost already include the network function inside; kotekan even includes the GPU correlation function. Our computing and network requirements are different from these functions. If we use them in our X-engine, these extra functions will need to be removed, which adds difficulty to the development. The HASHPIPE framework is a very general and convenient multi-thread management pipeline, which does not contain any specific function other than a framework. Since we have already selected modules with specific functions, HASHPIPE is an ideal management framework.

4. Key Hardware Technologies

As the correlator will be a cluster composed of multiple GPU servers, high-throughput network connectivity is key to harnessing the power of CPUs and GPUs. For this reason, we choose the NVIDIA ConnectX-5 2 $\times$ 100 Gbps NICs for the network system. Other key hardware includes the NVIDIA A5000 GPUs with Tensor Cores inside and the Intel Xeon Silver 4314 CPUs, shown in Figure 3. In this section, we provide a description of the advantages of the key hardware technologies of the proposed correlator.

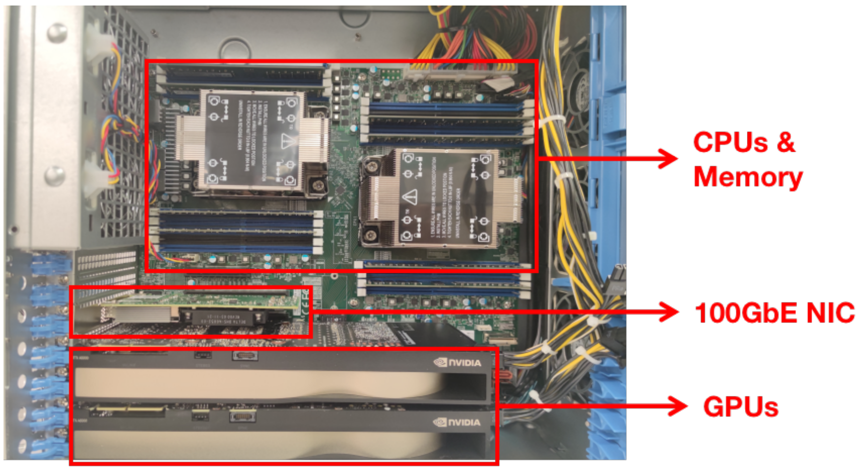

Figure 3. Key hardware inside the GPU server.

4.1. The NVIDIA ConnectX NICs

In the traditional socket-based network scheme, the CPU copies network packets from the NIC to the networking stack; after processing, the payload data are copied to the user space. This approach results in many memory copies and lots of processing for the CPUs, which limits the throughput. The SPEAD2 library is able to use hardware support in the NVIDIA ConnectX NICs to bypass the kernel's networking stack and directly access Ethernet frames with minimal copying[12]. The wire protocol is standard UDP. A potential alternative would be to use an RDMA protocol (such as RoCE) to have the NIC place data exactly where they are needed, but this is not currently supported by SPEAD2. In addition to being used for data communications between F-engines and X-engines, the NVIDIA ConnectX NICs will also be used between the D-engines and F-engines of the wSMA correlator, and between the digital backends and recorders of the ngEHT stations.

4.2. GPU Tensor Cores

Tensor Cores are mixed-precision computing units of NVIDIA GPUs. A Tensor Core can perform the matrix-multiply-and-accumulate operation ($D = A \times B + C$) in one GPU clock cycle, where A, B, C, and D are fixed-size matrices (typically 16×16) [7]. The A5000 GPU that we are using contains 256 third-generation Tensor Cores. Currently, NVIDIA provides three different ways to program Tensor Cores: the WMMA API, CUTLASS, and cuBLAS GEMM [24]. Here, TCC uses the lowest-level interface, the WMMA API, which mainly includes 3 functions. The first function is $load\_matrix\_sync()$, which loads matrices from the GPU memory to the registers of GPU threads. The second function is $mma\_sync()$, which implements the actual matrix-multiply-and-accumulate operation. The third function is $store\_matrix\_sync()$, which copies the calculated results from the GPU registers to GPU memory. Currently, Tensor Cores are only used in the X-engines, and the F-engines use the regular CUDA cores.

5. Introduction of the Katgpucbf F-Engine

As mentioned previously, we are considering to use and modify the F-engine of the katgpucbf library, so we describe it briefly here. A detailed description can be found in the online documentation of https://katgpucbf.readthedocs.io/en/latest/index.html, accessed on 20 December 2022. The framework of the F-engine is developed in Python and mainly includes three functions: the $run\_receive()$ function, which receives network packets from a D-engine or a playback server; the $run\_processing()$ function, which calls the GPU to process the received network packets; and the $run\_transmit()$ function, which transmits the processed data to the X-engines through the network. The three functions run in parallel with the framework of Asyncio, which is an asynchronous programming framework of Python.

The $run\_processing()$ function is the core of this library. After a chunk of data is received, coarse delay compensation is first performed, and then the GPU kernel functions process the data to achieve channelization. The main signal processing algorithm is the polyphase filter bank (PFB), which consists of an FIR filter and an FFT. The FIR filter has branches equal to the FFT size, and each branch is executed on an individual GPU thread. The output of PFB is a frequency-domain spectrum. However, the X-engines expect time-domain samples of each channel. Thus, a transpose operation is required to convert the data from the frequency domain to time domain. Other functions, such as fine delay compensation, fringe rotation, and quantization, are also required; these functions are integrated into one GPU kernel function called the PostProc function. All of the above kernel functions (FIR Filter, FFT, and PostProc) are implemented with floating point arithmetic.

6. The GPU Tensor Core X-Engine

6.1. The X-Engine Pipeline and Key Modules

The diagram of the proposed X-engine pipeline is shown in Figure 4, which includes 4 main modules working in parallel (each module is an independent thread) within the framework of HASHPIPE.

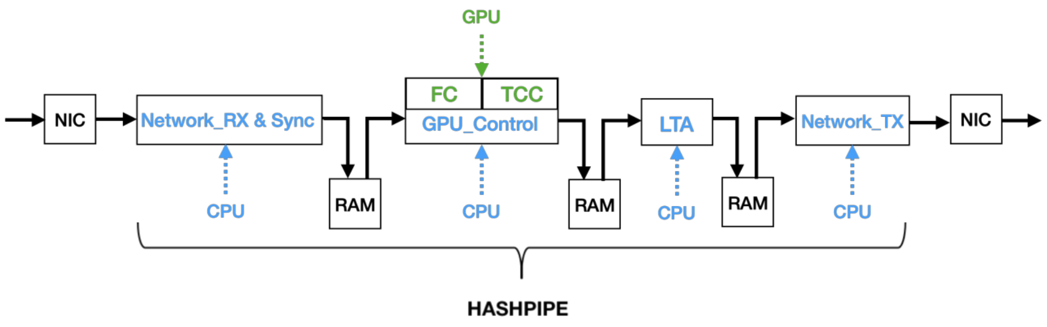

Figure 4. Diagram of the proposed X-engine pipeline.

The first module of the pipeline is called *Network_RX* & *Sync*, which is based on the SPEAD2 library. The network packets output from the F-engines are captured and reconstructed into heaps by the SPEAD2 library. A heap consists of a header containing a timestamp and an array of time-domain samples of one F-engine. Timestamps of multiple heaps may be out of order and thus reordered. Then, heaps with the same timestamp from different F-engines are synchronized into a heap array. Some heaps may be lost due to various reasons. If this happens, the software replaces them with zero values. After synchronization, all K^{13} heap arrays are assembled into one large data block, which will be transmitted to the subsequent *GPU_Control* module.

The throughput of the entire X-engine pipeline depends on the module that consumes the longest time—that is, the *GPU_Control* module. This module includes 4 processes: memory copy from CPU to GPU, the format conversion (FC) kernel function, the TCC kernel function, and memory copy from GPU to CPU. In order to improve the throughput of this module, the four processes are executed with 3 streams to form a pipeline; that is, the two kernel functions use a common stream, and each memory copy uses an individual stream. Through profiling with the software of Nsight, the throughput of the pipeline is around 140 Gbps, and the main time is spent on the memory copy from CPU to GPU. After TCC operation, the data have been integrated greatly, so the time consumption of the memory copy from GPU to CPU is not significant.

The input data format of TCC and output data format of katgpucbf F-engines are as follows:

$$\begin{aligned}&Input\_TCC[Channel\_ID][Block\_ID][Fengine\_ID][Pol\_ID][TimePerBlock\_ID].\\&Output\_Fengine[Fengine\_ID][Channel\_ID][Time\_ID][Pol\_ID].\end{aligned} \tag{1}$$

The dimensions from left to right of the above equation change from slow to fast. The format conversion (FC) kernel function is used to rearrange the high-dimensional matrices between these two formats.

Before using the WMMA API to implement the matrix-multiply-and-accumulate operation ($D = A \times B + C$) for correlation, TCC will construct two matrices of A and B at first. For matrix A, the first axis represents time-domain samples of each channel for integration, and the second axis represents antennas/stations. Matrix B is the transpose of matrix A. Values of matrix C should be set to be all zero. After the multiplication of A and B, we can obtain the visibility data of each channel between any two antennas/stations.

After TCC, the data rate has been greatly reduced, and the LTA module will further reduce the rate by integrating the data over a long period of time. The *Network_TX* module is also based on the SPEAD2 library to transmit the visibility data to the data storage servers. Since the visibility data have been integrated by the TCC kernel function and the LTA module, the pressure of data transmission is far less than the *Network_RX* & *Sync* module.

6.2. Hardware Requirements

As mentioned previously, the key hardware of the proposed correlator includes GPUs, NICs, and CPUs. For the F-engines, the required hardware resources depend on the number of antennas/stations. For the X-engines, the required hardware resources depend on the computing capability and IO bandwidth. We analyze the hardware requirements and try to find the bottleneck in the following content.

For the correlation computing of X-engines, the required performance of multiply-and-accumulate operations per second (OPS) can be calculated with the following equation:

$$N\_corr = B \times 2N(N+1) \times 8. \tag{2}$$

where B is the bandwidth of each sideband, which is 8 GHz. N is the number of antennas/stations, which is 8/21 for the wSMA/ngEHT. We consider full-stokes correlation with dual polarization—for each antenna pair, the vertical polarization and horizontal polarization must be correlated [25]. Auto-correlation of each antenna/station per polarization is also considered, so the number of correlation operations is $2N(N+1)$. The factor 8 arises from the complex-valued multiply–accumulate operation [20]. The result of Equation (2) is 9.216/59.136 TOPS for the wSMA/ngEHT correlator. We evaluated the the performance of TCC on an A5000 GPU at different bit widths; the results are shown in Figure 5. For the 4-bit situation, when the number of antennas is 8, the performance is approximately 6.52 TOPS, which means that two A5000 GPUs can meet the computing requirements of all X-engines of the wSMA correlator. When the number of stations is 21, the performance is approximately 31.36 TOPS, which means that two A5000 GPUs are also sufficient for the computing requirements of all X-engines of the ngEHT correlator. Through the above analysis, we can see that due to the extremely high performance of Tensor Cores, the correlation computation is no longer a bottleneck in the correlator.

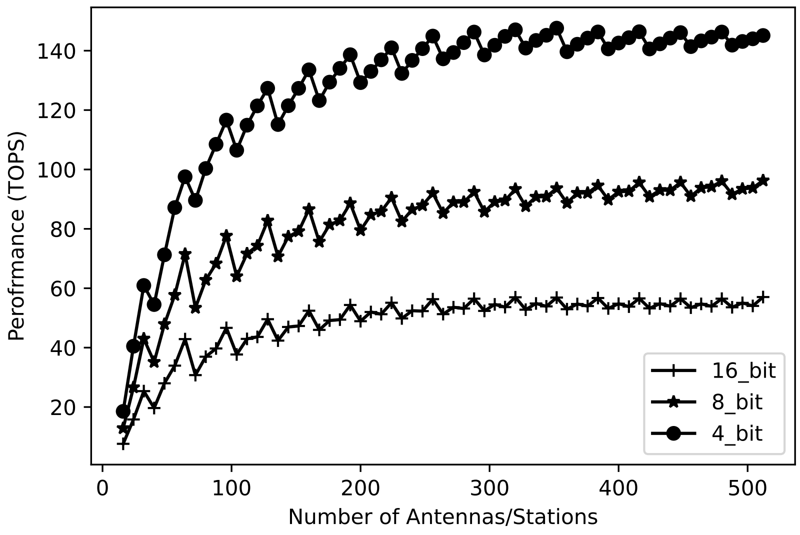

Figure 5. Performance of TCC on an A5000 GPU. The sawtooth shapes are caused by work distribution imbalances and redundant computations for non-multiples of 48 or 64 antennas/stations [7]. For smaller numbers of antennas or stations, the TCC is memory-I/O-bound, so it cannot achieve peak performance. For larger numbers of stations, the TCC converges to a plateau as it becomes compute-bound, even though the memory bandwidth use remains high. The GPU is eventually limited by the power use, as the driver slows down the clock to keep the GPU within its power limit.

For the IO bandwidth, the total data rate between the F-engines and X-engines can be calculated with the following equation:

$$B \times 2N \times 2 \times 4bits. \tag{3}$$

The result of the above equation is 1024/2688 Gbps for the wSMA/ngEHT correlator. The A5000 GPU uses $PCIe4.0 \times 16$ with an actual throughput of around 200 Gbps as the data path. As mentioned previously, the throughput of the *GPU_Control* module is around 140 Gbps, and the main time is spent on memory copying, so the utilization of $PCIe4.0$ effective bandwidth is approximately 70%. Therefore, due to the bandwidth limitation, theoretically, the number of GPUs required by the X-engines of the wSMA correlator and ngEHT correlator is $\lceil \frac{1024Gbps}{140Gbps} \rceil = 8$ and $\lceil \frac{2688Gbps}{140Gbps} \rceil = 20$, respectively. The NVIDIA ConnectX-5 NIC also uses $PCIe4.0 \times 16$ to connect to the host server, and the throughput of the network interface is 200 Gbps, which is almost the same as the throughput of PCIe, so the required number of NVIDIA ConnectX-5 NICs is the same as the number of GPUs.

Although we use GPUs to implement correlation computing, CPU resources are also required. The reason is that the HASHPIPE framework and the multiple modules in Figure 4 require multiple CPU cores. The specific number required cannot be calculated theoretically, but depends on the actual situation during the development and testing stage. Currently, each server has 2 powerful Intel Xeon Silver 4314 CPUs with a total of 32 cores, which are sufficient for our proof-of-concept.

7. The Full Compact Architecture

The full compact architecture is shown as Figure 6. Compared with the previous architecture in Figure 1, the changes are as follows. First, the D-engines of the wSMA correlator need to support the functions of coarse channelization and corner turning. For the ngEHT correlator, since the recorded data have been coarsely channelized into sub-bands by the digital backends, the playback servers simply need to corner turn these sub-bands to different GPU servers for further processing. Second, the *Network_RX & Sync* module needs to perform coarse delay compensation for each sub-band. Third, the main functions of the F-engine and X-engine are integrated into the same GPU to form a new engine called the FX-engine.

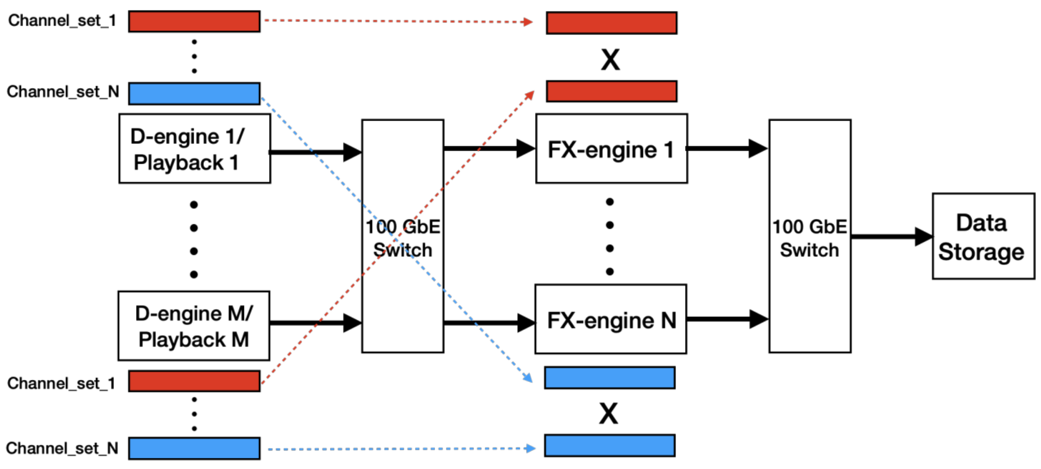

Figure 6. Full compact architecture of the proposed correlator.

The pipeline of the FX-engine is shown in Figure 7, which is based on the previous X-engine pipeline in Figure 4, but with the addition of new kernel functions of FIR filters (FIR_Pol0/FIR_Pol1[14]), FFTs (FFT_Pol0/FFT_Pol1), and PostProc to achieve fine channelization. These new kernel functions are very similar to that of the katgpucbf F-engine library, but the difference is that the latter only processes data from one antenna/station, while the former needs to process data from all antennas/stations. The other two kernel functions, Format Conversion (FC) and TCC, are the same as in Figure 7. At present, we have developed an initial design with a throughput of around 86 Gbps for the whole pipeline, but there is still much room for optimization in the future.

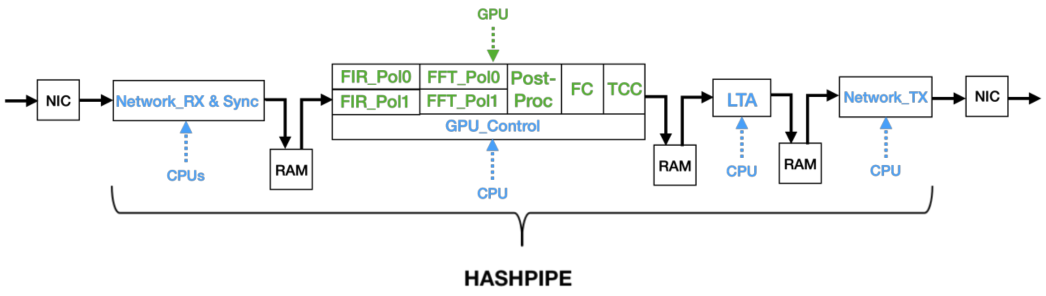

Figure 7. Diagram of the proposed full compact FX-engine pipeline.

8. Acceleration with New Hardware

The data path from the NIC to the GPU in Figures 4 and 7 can be divided into 3 steps. At first, data are transmitted from the NIC to the host server's memory via PCIe. Then, the host server caches and synchronizes these data. Finally, the synchronized data are sent to the GPU via PCIe for correlation. Multiple passes over bandwidth-limited PCIe are very inefficient and represent the bottleneck of the entire pipeline.

The NVIDIA ConnectX NIC that we are using can accelerate the first step through bypassing the kernel's networking stack. Going a step further, NVIDIA's new product of BlueField Data Processing Units (DPUs) can provide an acceleration of the first and second steps of the data path. By combining a ConnectX NIC with an array of ARM CPU cores, BlueField DPUs can be fully software-programmable [26], which can improve the performance of the *Network_RX* & *Sync* module in Figures 4 and 7. The embedded NIC captures network packets at first, and then forwards them to the subsequent embedded ARM CPUs, which can replace the host's CPUs to maintain network connections and execute data crossover and time synchronization operations in parallel multi-threads. Since the data receiving, caching, and synchronization are combined and implemented on the same DPU, the whole system can be more efficient and has a lower cost due to the reduced demand on the host CPUs [27].

The NVIDIA converged accelerator is an upcoming product, which integrates a Bluefiled DPU and a GPU on one board[15]. If the X-engine is equipped with this board, the entire data path from the NIC to the GPU no longer needs to go through the host server's memory. Data can be sent from the DPU to the GPU directly, so all three steps can be accelerated, and the end-to-end throughput can be further improved.

9. Conclusions

We have reviewed the architectures of existing correlator-beamformers. With the benefit of this context, and using codes shared by collaborating institutions including SARAO and ASTRON, we built a proof-of-concept GPU Tensor Core-based correlator for the ngEHT and the wSMA. The requirement is to provide a solution to the transmission and computing challenges brought about by the massive data rates required by wideband instruments.

It is notable that this architecture is a purely software- as opposed to firmware-driven design. Compared to the currently favored approach using FPGAs, the development and ongoing maintenance of a software machine is much easier compared to the very difficult hardware description language (HDL) FPGA firmware design. The GPU's floating point arithmetic yields improved digital efficiency with lower quantization and clipping losses.[16] In summary, GPUs are an order of magnitude faster to code for complex applications, while being both flexible and maintainable after deployment.

To start the proof-of-concept, we evaluated open-source libraries and selected the appropriate ones according to the requirements. Then, we described the proposed X-engine pipeline and analyzed the hardware resource requirements. We also introduced a full compact architecture, which integrates the main functions of the F-engine and X-engine into one GPU. Finally, we discussed how new hardware can accelerate future correlator-beamformers.

The paper presents proof-of-concept design work and associated benchmarks, which lead us to an optimistic assessment of the prospects for true- or near-real-time computation for connected and VLBI interferometry. We anticipate that this new development will support the improved imaging capability of the ngEHT and wSMA upgrades, thus allowing both instruments to achieve their respective transformative scientific goals.

Author Contributions: Conceptualization, J.W., J.W.R. and W.Y.; methodology, J.W., W.Y., J.W.R., L.J.D., R.-S.L., A.P., G.C., D.W.P., L.B., B.M., R.S. and J.K.; software, W.Y., J.W.R., L.J.D., G.C. and B.M.; writing and editing, W.Y., J.W., J.W.R., G.C., D.W.P., L.B. and B.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Black Hole Initiative at Harvard University, which is funded by grants from the John Templeton Foundation and the Gordon and Betty Moore Foundation to Harvard University. This work was also supported by National Science Foundation grants AST-1726637, AST-1935980, AST-2034306; the Gordon and Betty Moore Foundation (GBMF-10423); the European Commission (DEEP-EST grant agreement 754304); the Netherlands eScience Center (PADRE, RECRUIT), and the European Southern Observatory. We acknowledge donations of hardware and software from NVIDIA and from AMD/Xilinx. The SMA is a joint project between the SAO and ASIAA and is funded by the Smithsonian Institution and the Academia Sinica.

Data Availability Statement: Not applicable.

Acknowledgments: This research has made use of the NASA Astrophysics Data System (ADS). The authors have benefited from technology shared under open-source license by the Collaboration for Astronomy Signal Processing and Electronics Research (CASPER). We thank Sheperd Doeleman, Robert Wilson, Garrett Karto Keating and Ram Rao for their helpful conversations. We acknowledge the significance that Maunakea, where the SMA is located, has for the indigenous Hawaiian people.

Conflicts of Interest: L. Jonathan Dursi discloses that he is an employee of NVIDIA. The authors declare no other conflicts of interest.

Notes

1 https://www.ngeht.org/ (accessed on 20 December 2022).
2 The current EHT data are correlated by two DiFX [4,5] clusters located at the Bonn and MIT Haystack observatories. These two correlators are not specifically deployed for the EHT; they have other routine correlation tasks.
3 https://casper.berkeley.edu/ (accessed on 20 December 2022).
4 These functions are also available in the CBF. In a full system, it is not decided yet whether they will be implemented in the D-engine, CBF, or split across both.
5 https://katgpucbf.readthedocs.io/en/latest/index.html (accessed on 20 December 2022).
6 https://www.ntop.org/guides/pf_ring/ (accessed on 20 December 2022).
7 https://github.com/jive-vlbi/jive5ab (accessed on 20 December 2022).
8 https://spead2.readthedocs.io/en/latest/ (accessed on 20 December 2022).
9 The format conversion module and the TCC module are two kernel functions executed on the same GPU, so they will be executed on the same CPU thread.
10 http://psrdada.sourceforge.net/ (accessed on 20 December 2022).
11 https://github.com/david-macmahon/hashpipe (accessed on 20 December 2022).
12 It uses the ibverbs library with IBV_QPT_RAW_PACKET queue pairs.
13 At present, we fix K to be 128.
14 The two polarization data paths are processed separately.
15 https://www.nvidia.com/en-us/data-center/products/converged-accelerator/ (accessed on 20 December 2022).
16 VLBI correlation is limited in most practical applications to 88% digital efficiency because samples are typically quantized to 2-bit width for recording. Greater efficiency using floating point arithmetic is achievable for real-time tied array correlators, such as

for wSMA. For the VLBI case, starting with 2-bit samples, data widths grow in the correlation processing, so the floating point arithmetic can still be beneficial to actually achieve the 88% efficiency, which is possible in principle.

References

1. Event Horizon Telescope Collaboration; Akiyama, K.; Alberdi, A.; Alef, W.; Asada, K.; Azulay, R.; Baczko, A.K.; Ball, D.; Baloković, M.; Barrett, J.; et al. First M87 Event Horizon Telescope Results. I. The Shadow of the Supermassive Black Hole. *ApJL* **2019**, *875*, L1. [CrossRef]
2. Event Horizon Telescope Collaboration; Akiyama, K.; Alberdi, A.; Alef, W.; Algaba, J.C.; Anantua, R.; Asada, K.; Azulay, R.; Bach, U.; Baczko, A.K.; et al. First Sagittarius A* Event Horizon Telescope Results. I. The Shadow of the Supermassive Black Hole in the Center of the Milky Way. *ApJL* **2022**, *930*, L12. [CrossRef]
3. Doeleman, S.; Blackburn, L.; Dexter, J.; Gomez, J.L.; Johnson, M.D.; Palumbo, D.C.; Weintroub, J.; Farah, J.R.; Fish, V.; Loinard, L.; et al. Studying Black Holes on Horizon Scales with VLBI Ground Arrays. *arXiv* **2019**, arXiv:1909.01411.
4. Deller, A.T.; Tingay, S.J.; Bailes, M.; West, C. DiFX: A Software Correlator for Very Long Baseline Interferometry Using Multiprocessor Computing Environments. *Publ. Astron. Soc. Pac.* **2007**, *119*, 318–336.
5. Deller, A.T.; Brisken, W.F.; Phillips, C.J.; Morgan, J.; Alef, W.; Cappallo, R.; Middelberg, E.; Romney, J.; Rottmann, H.; Tingay, S.J.; et al. DiFX-2: A More Flexible, Efficient, Robust, and Powerful Software Correlator. *Publ. Astron. Soc. Pac.* **2011**, *123*, 275–287. [CrossRef]
6. Primiani, R.A.; Young, K.H.; Young, A.; Patel, N.; Wilson, R.W.; Vertatschitsch, L.; Chitwood, B.B.; Srinivasan, R.; MacMahon, D.; Weintroub, J. SWARM: A 32 GHz Correlator and VLBI Beamformer for the Submillimeter Array. *J. Astron. Instrum.* **2016**, *5*, 1641006. [CrossRef]
7. Romein, J.W. The Tensor-Core Correlator. *Astron. Astrophys.* **2021**, *656*, A52. [CrossRef]
8. van der Byl, A.; Smith, J.; Martens, A.; Manley, J.; van Balla, T.; Rust, A.; Patel, A.; Callanan, G.; Isaacson, A.; New, W.; et al. MeerKAT correlator-beamformer: A real-time processing back-end for astronomical observations. *J. Astron. Telesc. Instruments Syst.* **2022**, *8*, 011006. [CrossRef]
9. Hickish, J.; Razavi-Ghods, N.; Perrott, Y.C.; Titterington, D.J.; Carey, S.H.; Scott, P.F.; Grainge, K.J.B.; Scaife, A.M.M.; Alexander, P.; Saunders, R.D.E.; et al. A digital correlator upgrade for the Arcminute MicroKelvin Imager. *Mon. Not. R. Astron. Soc.* **2018**, *475*, 5677–5687. [CrossRef]
10. Denman, N.; Renard, A.; Vanderlinde, K.; Berger, P.; Masui, K.; Tretyakov, I. A GPU Spatial Processing System for CHIME. *J. Astron. Instrum.* **2020**, *9*, 2050014. [CrossRef]
11. Kocz, J.; Greenhill, L.J.; Barsdell, B.R.; Bernardi, G.; Jameson, A.; Clark, M.A.; Craig, J.; Price, D.; Taylor, G.B.; Schinzel, F.; et al. A Scalable Hybrid Fpga/gpu FX Correlator. *J. Astron. Instrum.* **2014**, *3*, 1450002. [CrossRef]
12. Kocz, J.; Greenhill, L.J.; Barsdell, B.R.; Price, D.; Bernardi, G.; Bourke, S.; Clark, M.A.; Craig, J.; Dexter, M.; Dowell, J.; et al. Digital Signal Processing Using Stream High Performance Computing: A 512-Input Broadband Correlator for Radio Astronomy. *J. Astron. Instrum.* **2015**, *4*, 1550003. [CrossRef]
13. Ali, Z.S.; Parsons, A.R.; Zheng, H.; Pober, J.C.; Liu, A.; Aguirre, J.E.; Bradley, R.F.; Bernardi, G.; Carilli, C.L.; Cheng, C.; et al. PAPER-64 Constraints on Reionization: The 21 cm Power Spectrum at z = 8.4. *Astrophys. J.* **2015**, *809*, 61. [CrossRef]
14. Ord, S.M.; Crosse, B.; Emrich, D.; Pallot, D.; Wayth, R.B.; Clark, M.A.; Tremblay, S.E.; Arcus, W.; Barnes, D.; Bell, M.; et al. The Murchison Widefield Array Correlator. *Publ. Astron. Soc. Aust.* **2015**, *32*, e006. [CrossRef]
15. Wayth, R.B.; Greenhill, L.J.; Briggs, F.H. A GPU-based Real-time Software Correlation System for the Murchison Widefield Array Prototype. *Publ. Astron. Soc. Pac.* **2009**, *121*, 857. [CrossRef]
16. Broekema, P.C.; Mol, J.J.D.; Nijboer, R.; van Amesfoort, A.S.; Brentjens, M.A.; Loose, G.M.; Klijn, W.F.A.; Romein, J.W. Cobalt: A GPU-based correlator and beamformer for LOFAR. *Astron. Comput.* **2018**, *23*, 180. [CrossRef]
17. Event Horizon Telescope Collaboration; Akiyama, K.; Alberdi, A.; Alef, W.; Asada, K.; Azulay, R.; Baczko, A.K.; Ball, D.; Baloković, M.; Barrett, J.; et al. First M87 Event Horizon Telescope Results. III. Data Processing and Calibration. *ApJL* **2019**, *875*, L3. [CrossRef]
18. Gill, A.; Blackburn, L.; Roshanineshat, A.; Chan, C.K.; Doeleman, S.S.; Johnson, M.D.; Raymond, A.W.; Weintroub, J. Prospects for Wideband VLBI Correlation in the Cloud. *Publ. Astron. Soc. Pac.* **2019**, *131*, 124501. [CrossRef]
19. Kalia, A.; Kaminsky, M.; Andersen, D.G. Design Guidelines for High Performance RDMA Systems. In Proceedings of the 2016 USENIX Annual Technical Conference (USENIX ATC 16), Denver, CO, USA, 22–24 June 2016.
20. Clark, M.A.; LaPlante, P.C.; Greenhill, L.J. Accelerating radio astronomy cross-correlation with graphics processing units. *Int. J. High Perform. Comput. Appl.* **2013**, *27*, 178–192. [CrossRef]
21. Denman, N.; Amiri, M.; Bandura, K.; Cliche, J.F.; Connor, L.; Dobbs, M.; Fandino, M.; Halpern, M.; Hincks, A.; Hinshaw, G.; et al. A GPU-based Correlator X-engine Implemented on the CHIME Pathfinder. *arXiv* **2015**, arXiv:1503.06202.
22. Recnik, A.; Bandura, K.; Denman, N.; Hincks, A.D.; Hinshaw, G.; Klages, P.; Pen, U.L.; Vanderlinde, K. An efficient real-time data pipeline for the CHIME Pathfinder radio telescope X-engine. In Proceedings of the 2015 IEEE 26th International Conference on Application-Specific Systems, Architectures and Processors (ASAP), Toronto, ON, Canada, 27–29 July 2015.
23. Cranmer, M.D.; Barsdell, B.R.; Price, D.C.; Dowell, J.; Garsden, H.; Dike, V.; Eftekhari, T.; Hegedus, A.M.; Malins, J.; Obenberger, K.S.; et al. Bifrost: A Python/C++ Framework for High-Throughput Stream Processing in Astronomy. *J. Astron. Instrum.* **2017**, *6*, 1750007. [CrossRef]

24. Markidis, S.; Chien, S.W.D.; Laure, E.; Peng, I.B.; Vetter, J.S. NVIDIA Tensor Core Programmability, Performance & Precision. In Proceedings of the 2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Vancouver, BC, Canada, 21–25 May 2018.
25. Callanan, G.M. A GPU based X-Engine for the MeerKAT Radio Telescope. Master's Thesis, University of Cape Town, Cape Town, South Africa, 2020.
26. NVIDIA BLUEFIELD-2 DPU Data Center Infrastructure on a Chip. Available online: https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/documents/datasheet-nvidia-bluefield-2-dpu.pdf (accessed on 20 December 2022).
27. Deierling, K. Achieving a Cloud-Scale Architecture with DPUs. Available online: https://developer.nvidia.com/blog/achieving-a-cloud-scale-architecture-with-dpus/ (accessed on 20 December 2022).