ngVLA Electronics Memo #19
ngVLA CSP X-Engine Subarraying Concept

Nolan Denman
ndenman@nrao.edu

February 27, 2025

Contents

1 Overview

The operation of the ngVLA as a collection of independently-targeted subarrays without mutual interference is a key component of its observing strategy. The structure of the ngVLA's Central Signal Processor (CSP) has been informed by this requirement, and the CSP's structure (as per the conceptual design phase) is highly divisible as a result – once data enters the CSP Switched Fabric (CSF), as seen in Figure 1, the inherently parallel structure of the CSP will permit the subdivision of processing resources to support flexible and independent subarray observations. This document discusses the implementation of subarrays on the X-Engine at a high level, with the intent of identifying and discussing both the impact that support for subarray observation has on the X-Engine's design and the aspects of the resulting design which provide this capability.

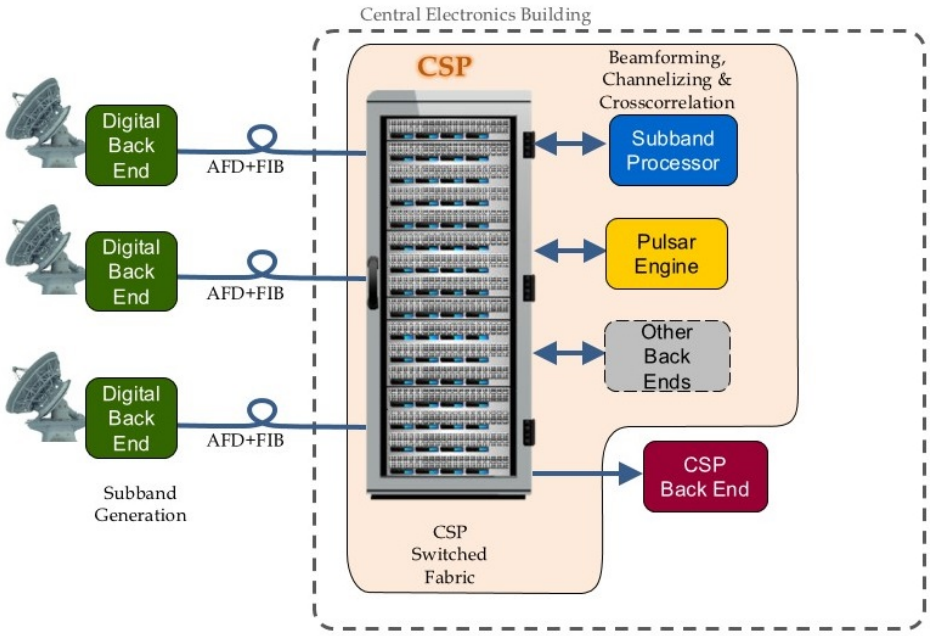

Figure 1: An overview of the ngVLA CSP, displaying its main components and their connections.

Section 2 presents relevant terminology, both pre-existing and newly-defined, which is used elsewhere in this document. Section 3 describes critical assumptions made about the ngVLA CSP and X-Engine in this context. The overall flow of commands to and from the CSP is summarized, and the internal operation of an X-Engine node is described; a modern tensor-core-equipped GPU is used as an illustrative example. Section 4 contains a selection of examples which display particular aspects of the desired subarray behavior. A few concluding remarks and notes about further development constitute Section 5.

2 Terminology

The formally-defined terminology available to describe observations with subarrays has some shortcomings; Table 1 contains relevant terms defined in the Project Lexicon (Doc # 020.10.10.10.00-0005-LIS), version B. For the purposes of this document, the definition supplied for "subarray" has been taken to refer specifically to the antennas, *not* including any of the processing resources.

There is also a set of concepts for which terms are not defined, or for which the terms in use are not consistent. Table 2 contains some of these terms and definitions; this is intended to be descriptive rather than normative, and feedback would be greatly appreciated. These terms are used freely elsewhere in this document.

3 Model

The structure of the ngVLA CSP in general, and the X-Engine in particular, is assumed to have certain features which both support and limit its ability to operate simultaneous subarrays. Notes:

- The CSF is assumed to be capable of routing any Subarray-Channel-Block pair to any X-Engine node, as required.

Term	Lexicon Definition
Subarray	A subarray is a set of resources (antennas) reserved for exclusive control that may be allocated to one or more independent sets of processing resources in order to produce one or more data product streams.
Functional Operating Mode	A Functional Operating Mode encapsulate a set of functional and performance requirements and produce corresponding data products and can be one of the following: Correlation modes: Interferometry, OTF Mapping Phased Array modes: VLBI, Pulsar Timing, Pulsar Search
Scan	An observation is divided in time intervals called scans, which have associated intents. The intent differentiates scans performed for the purpose of calibration from those performed over the astronomical objects of scientific interest
Schedule Block	"Scheduling block" is not intended to imply a specific structure to an observation, it is intended to mean the minimum unit of a planned observation that can be executed, completed, and gracefully exited.
Execution Block	When an iteration of a schedule block gets queued for observation, a time-specific instance is created known as an Execution Block.

Table 1: Relevant existing terminology taken from version B of the Project Lexicon.

- It is explicitly assumed that it is only reasonable to correlate antennas which are operating in the same spectral configuration (e.g. observing band selection, spectral zoom, channel width).
- Communication between the CMC and each node is described as one-to-one; while it is possible that multiple X-Engine nodes could share a management interface at the LRU level, this does not significantly affect the behavior described here.

3.1 Generalized Command Flow

Commands for the configuration of subarrays are issued hierarchically; Figure 2 shows an overview of this process focusing on the flow of instructions through the CSP and adjacent systems.

Observatory scheduling occurs at a much higher level than the CSP, with the primary interaction at this stage being queries between the observatory management system and the CMC regarding the capacity and utilization of CSP resources. In addition to hard limits on various parameters (channel counts, bit depths, integration lengths, etc.) there are at least three dimensions (input data bandwidth, output data bandwidth, and node processing power) in which the X-Engine resources have broadly cumulative limits. The viability of any specific ngVLA Configuration will therefore depend on the sum of the Scan Processing Resources for each of the subarrays in use. The specifics of the CMC interaction with observatory management are as yet undetermined, but the ngVLA Configuration Schedule will be arranged through this process.

Once the ngVLA Configuration Schedule is determined, observatory management systems will issue instructions to the CMC with the desired ngVLA Configuration and the set of Scan Timing Information, with a specific time of application (which may be immediate or in the future). The CMC must then determine which X-Engine hardware must be assigned or re-assigned in order to support this configuration and issue the appropriate commands (complete with any parameters required) to the X-Engine hardware.

Many details of the node hardware's behavior during the transition between configurations are not yet defined. Configuration changes which are scheduled for a future time and which take place at the boundary of an integration are uncontroversial, but interruptions of ongoing processing require further consideration. The details of how Scan Configurations are generated and transferred, and how the CSP State is informed

Term or Placeholder	Approximate Definition
Observation	[Implied by 'Scan' above but not defined]
Calibratable Scan Block (?)(*)	A term used elsewhere without definition other than as 'the smallest unit of observation', although one consisting of multiple scans.
[Scan](?)	A placeholder term, for a specific sort of atomic unit of observation like that of a schedule block. The definition of 'Scan' above specifically only relates to time intervals, so another term must be created.
[Scan] Processing Resources	The set of processing resources required to produce a desired data product stream for a particular Generic [Scan] Configuration.
Generic [Scan] Configuration	The functional operating mode of a generic subarray and its parameters (bandwidth, number of antennas, etc.) in sufficient detail to determine the [Scan] Processing Resources required.
Specific [Scan] Information	The timing, subarray, and spectral configuration information required to turn a Generic [Scan] Configuration into a [Scan] Configuration.
[Scan] Subarray Information	The specific set of antennas comprising a subarray, as a component of the Specific [Scan] Information.
[Scan] Timing Information	The time-of-execution information component of the Specific [Scan] Information.
[Scan] Spectral Information	The spectral configuration (e.g. band selection, spectral zoom, channel count) component of the Specific [Scan] Information.
[Scan] Configuration	A particular instance of a Generic CSP [Scan] Configuration with a specific subarray, frequency configuration, and timing information.
ngVLA System Configuration (?)(*)	The complete configuration of the ngVLA - antennas, DBE, CSP, etc. - at one specific moment; where the antennas are pointing, what receivers are live, what the CSP is doing, where the data is going, etc. up to the point where live products are recorded or transmitted for future use.
ngVLA System Configuration Schedule (?)(*)	The scheduled set of ngVLA Configurations over the forseeable future.
CSP State	The configuration of the entire ngVLA CSP at a specific moment.
Observation(?)(*)	The event during which the ngVLA observes some target(s) with a specific configuration and settings.
Channel Block	A contiguous portion of an ngVLA subband; in the context of subarray observations, that which assigned to a single X-Engine node.

Table 2: Terminology *not* taken from the Project Lexicon. Those terms marked with (*) are outside the CSP's scope, and consultation with other groups will be required to arrive at consensus definitions. Those terms marked with (?) are especially uncertain.

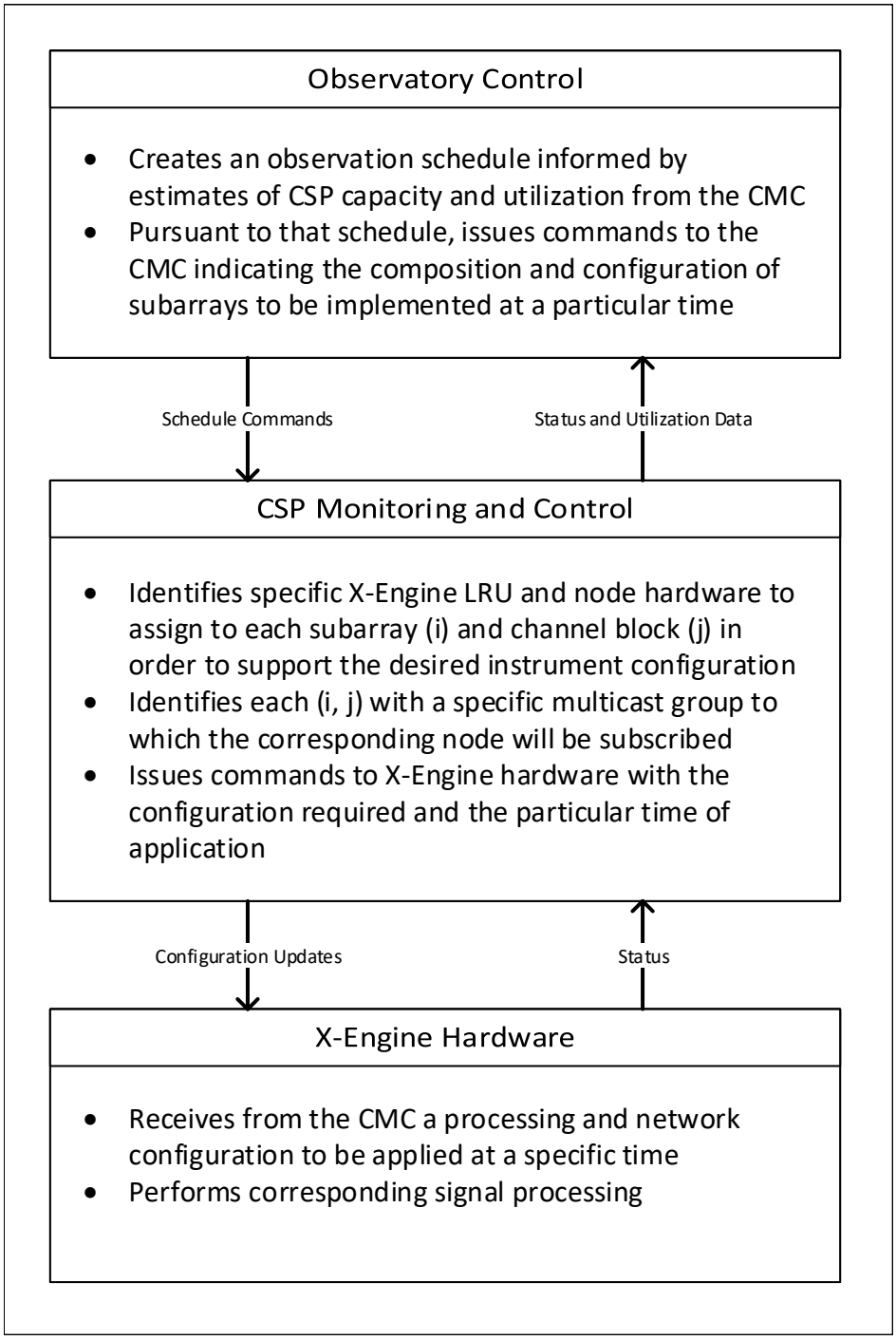

Figure 2: An overview of the command flow for X-Engine configuration.

moment-to-moment, will be part of the greater set of Interface Control Documents (ICDs) which have not yet been created.

This interrelates with the definition of Scan Timing Information arrived at by the system; the X-Engine does not have a high-precision timing source, but incoming data will be associated with precise timestamps. As such, the time at which the X-Engine node needs to change configurations can be defined in terms of either the low-precision time (correct to within a few seconds) available to the node itself or the high-precision time (sub-ns) present alongside the data – only available once the new data arrives for processing, necessarily causing congestion while the node changes configuration. Further work on this issue is required.

3.2 X-Engine Node

Each node within the CSP X-Engine will be responsible for computing the time-integrated outer product of all members of a specific subarray for each frequency channel within a particular portion of the observing band (the 'channel block'). More formally, within a given integration each node will be responsible for the computation of the output visibilities $\mathcal{V}$:

$$\mathcal{V}_{i,j,T,m,l} = \sum_{t} \mathcal{E}_{i,j,T,m,k,t} * \mathcal{E}_{i,j,T,m,k',t} \tag{1}$$

where the variables and indices are as defined in Table 3.

For the purposes of performance modelling, it is assumed that each X-Engine node has some maximum input data bandwidth, processing capability, and output data bandwidth; it also has onboard memory with some limited size and access bandwidth. Within each X-Engine node are processing resources organized into complex multiply-and-accumulate units, each of which is capable of performing the complex outer product of n_{unit} input values and appropriately accumulating the result. Each node contains N_{units} of these CMAC units.

Each node is responsible for computing the full outer product of all $n_{sub,i}$ antennas (from 1 to 526) in a single subarray, over some specific bandwidth $\Delta\nu_{i,j}$ (up to[1] 20 GHz) which has been divided into $N_{chan,i,j}$ frequency channels (between about 2^4 and 2^{14}). This outer product must occur once every Δt_i, generating $\frac{1}{2}(n_{sub,i})(n_{sub,i}+1)$ output values per accumulation period ΔT_i.

The number of CMAC units N_{CU} required to correlate a subarray with a total number of receivers n_{sub} is

$$N_{CU} = \frac{1}{2}\left(\left\lfloor \frac{n_{sub}}{n_{unit}} \right\rfloor\right)\left(\left\lfloor \frac{n_{sub}}{n_{unit}} \right\rfloor + 1\right) \tag{2}$$

For subarray sizes where N_{CU} is less than or equal to N_{units}, the entire subarray may be correlated at once; in the alternative case, some or all CMAC units will need to correlate multiple portions of the subarray and track their accumulants separately. Figure 3 illustrates the redundant computations which result from this model treating auto- and cross-correlation products identically; it is possible to eliminate this at the cost of increased complexity in managing the computations.

3.2.1 Accumulation

Accumulation of the results is a notable complication in all cases where $N_{CU} \neq N_{units}$, as each CMAC unit is responsible for a varying selection of receivers. The effect is compounded as the number of frequency channels increases; each CMAC unit must handle an increasing number of distinct accumulants, including readouts at the correct integration times. A solution to general accumulation is desired which would also permit conditional accumulation based on either flags (e.g. RFI detection) or a time-dependent function (e.g. phase-binned visibilities); additional comments on the latter use case may be found in Subsection 4.2.

[1]The maximum channel block bandwidth per node may be limited by the implementation, to an extent which has yet to be determined. This would create a minimum node count per subarray, but not otherwise affect system performance.

Quantity	Name	Description
i	Subarray Index	Identifies the subarray in question
j	Channel Block Index	Identifies the portion of each subband assigned to a specific node
(i, j)	Node Index	Identifies the specific node to which the unique (i, j) is assigned.
k_i	Antenna Index	Identifies a specific antenna within subarray i
$l_i = L(k_i, k_i')$	Product Index	Identifies the specific product (k_i, k_i') within subarray i
$m_{i,j}$	Channel Index	Identifies a specific channel within the subarray i and channel block j
T_i	Inter-Integration Time Index	Identifies a specific X-Engine integration for subarray i
t_i	Intra-Integration Time Index	Identifies a specific time-sample within an integration
$p_i = P_i(t_i)$	Phase Bin Index	Identifies the time-dependent accumulation bin for subarray i
$\mathcal{E}_{i,j,T,m,k,t}$	Channelized Input Data	Post-channelization antenna data; input to the X-Engine
$\mathcal{V}_{i,j,T,m,l}$	Time-Integrated Visibilities	The time-integrated visibilities as computed by the X-Engine
$\mathcal{V}_{i,j,T,m,p,l}$	Phase-Integrated Visibilities	Visibilities accumulated into phase bins by the X-Engine
n_{sub_i}	Receivers per Subarray	The distinct receivers in subarray i which must be correlated
$\Delta\nu_{i,j}$	Channel Block Bandwidth	The total signal bandwidth assigned to a specific node (i, j)
$N_{chan,i,j}$	Channel Block Channel Count	The number of channels into which a channel block is divided
ΔT_i	Integration Time	The total integration time required for visibilities in subarray i
Δt_i	Sample Cadence	The time interval between successive in-channel samples
n_{unit}	Inputs per CMAC Unit	The number of inputs each CMAC unit may compute
N_{units}	CMAC Units per Node	The effective number of CMAC units in each node
$N_{CU_{i,j}}$	Compute Units Required	The number of notional CMAC units required to correlate (i, j)

Table 3: Variables and indices used in the model from Subsection 3.2 and in subsequent discussion.

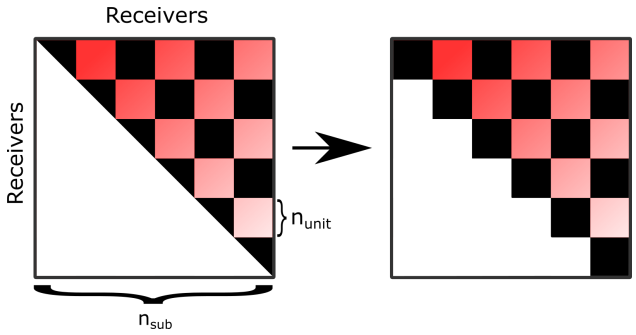

Figure 3: Redundant computations required by the subdivision of the computational task into homogeneous units; the portions of the outer product containing auto-correlations are computed twice.

3.2.2 Example Node Structure

Figure 4 presents the internal structure of an NVIDIA 'Hopper' H100 GPU as an illustrative example of the internal structure model assumed for an X-Engine Node. The node has one or more external interfaces (here, PCIe v5.0 at top and NVLink at bottom), onboard memory (HBM, at the sides), and a number of processing units ('streaming multiprocessors' or SMs) each of which can perform independent computation. In this example, the external PCIe interface would be paired with a PCIe Ethernet Network Interface Card (NIC) to provide Ethernet connectivity for the node.

The preceding descriptions have presented correlation as a sample-by-sample procedure iterating over time as the fastest-varying axis; in the case of tensor-core-equipped GPU nodes, this may instead be expressed as a tensor contraction operation on the channelized voltage data tensors $\mathcal{E}$. The NVIDIA-supported cuTENSOR library provides a simplified interface via the `cutensorCreateContraction` and `cutensorContract` methods to a general tensor contraction $D = \alpha\mathcal{A}\mathcal{B} + \beta\mathcal{C}$; experiments based on this formalism will begin in the near future, although John Romein's results with the Tensor-Core Correlator[2] suggest that computational performance will be significantly improved relative to the non-tensor methods.

3.3 Resource Requirement Scaling

Consider an X-Engine capable of receiving data from and correlating the entire 263-antenna array across a full 20 GHz of bandwidth. Reallocating any antenna from the main subarray to a new subarray does not change the total amount of data entering the system (which scales as the bandwidth multiplied by the number of antennas across all subarrays) but reduces the total compute load (which scales as the sum of the squares of the number of antennas in each subarray). The same argument applies for any change which moves an antenna from one subarray to a smaller one - the total input data rate is unchanged but the total compute load reduced.

It follows that resource allocation sufficient for any set of subarrays should always be possible provided that the original resources are sufficient for the full array and the resources can be arbitrarily subdivided. However, if it is not possible to split a single node between subarrays, there is a potential case in which the processing or data transport requirements for an enlarged subarray just narrowly do not fit within the number of nodes previously allocated. In this case, one node per possible excess subarray would be sufficient to allow for continued operation.

[2] A&A v. 656 a. 52, 2021

Figure 4: The internal structure of an NVIDIA H100 GPU, shown with the maximum complement of 144 Streaming Multiprocessors (SMs). The PCIe v5.0 x16 interface is shown at the top, HBM interfaces at each side, and the NVLink interface at the bottom. The detailed structure of the SM is shown in detail in Figure 5.

4 Example Scenarios

In the absence of preliminary ICDs and an appropriate testing suite, it is difficult to have confidence in the X-Engine's ability to handle the span of all possible subarray configurations and the transitions between configurations which may be required for the observation programme. The following examples are intended to illustrate broad categories of subarray configurations and behaviors envisioned, so as to capture the general behaviors of interest. Comments from the broader ngVLA community would be welcomed.

4.1 Adding or Reallocating a Subarray

It is desired that, wherever possible, reassigning subarrays should not interrupt the processing of other subarrays. Once the CMC has determined the required node allocation and configuration, commands will be issued to the nodes whose operational state will be updated. The nodes responsible for the new subarray, which may have been processing other data previously, will apply their new network and processing configurations at the time specified in the command.

Given the scaling properties described in Subsection 3.3, the existence of a sufficiently large and divisible pool of X-Engine nodes should satisfy the processing and input data bandwidth needs of any subarray configuration – provided it is possible to assign and distribute data to X-Engine nodes arbitrarily. An unavoidable departure from the goal of full subarray independence is introduced by system constraint CON104's hard limit on CSP output data bandwidth; subarray observation scheduling must reflect this, and subarrays may not be freely reassigned if this would violate this constraint.

The precise details of how the CSF routes data to the X-Engine nodes, and particularly the mechanisms by which the nodes determine and implement their multicast address selection, are still to be determined.

4.1.1 Interruptions

Although the typical operations of the ngVLA are expected to give notice sufficient to gracefully end observations before a change of configuration is applied, there may be cases in which a interruption of an

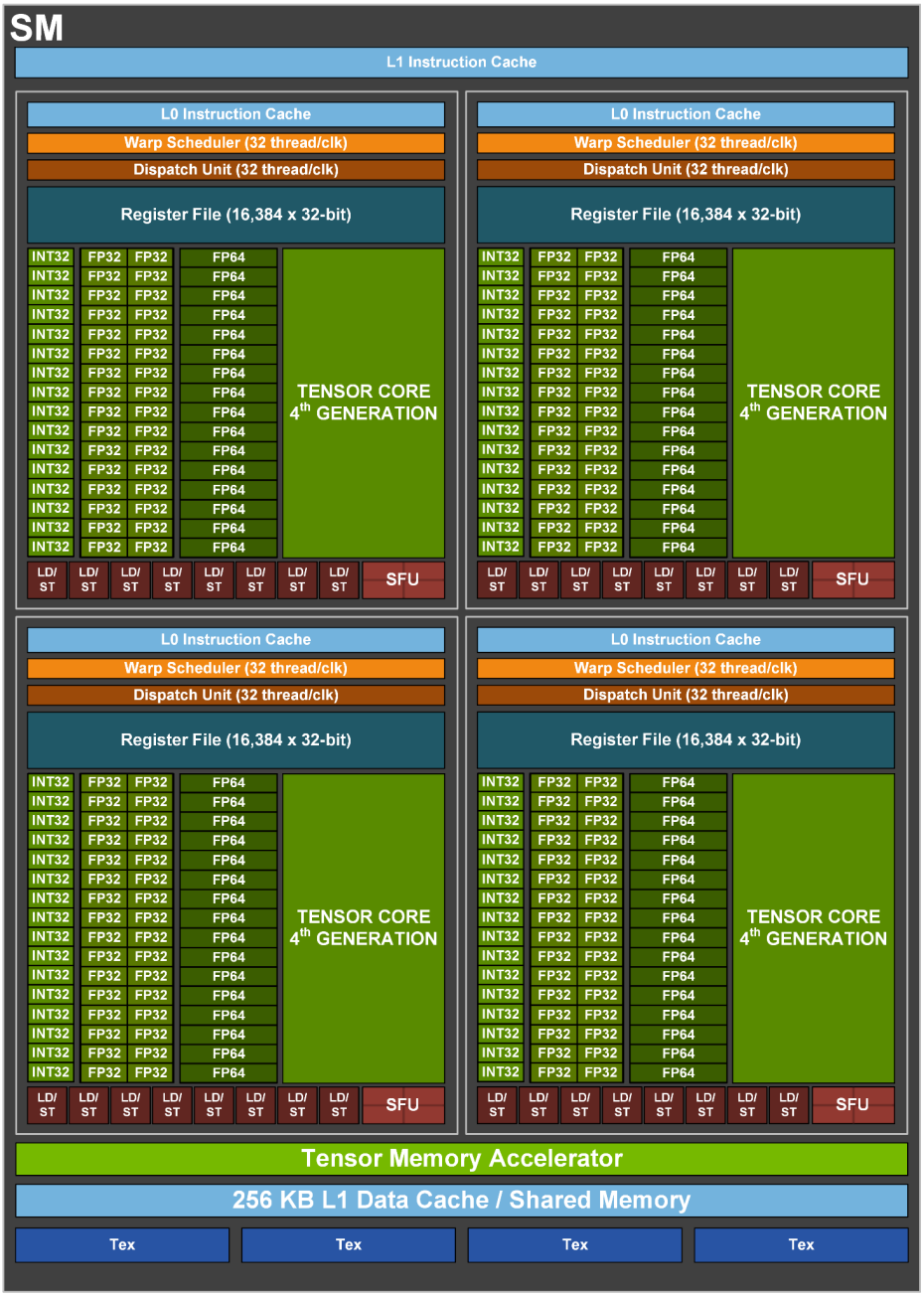

Figure 5: The internal structure of an NVIDIA Hopper Streaming Multiprocessor (SM); each contains four tensor cores, 128 32-bit floating-point (FP32) processing cores, 32 64-bit floating point (FP64) processing cores, and 64 32-bit integer (INT32) processing cores as well as 64k 32-bit registers and a 256 kB L1 data cache.

ongoing observation is required. Handling of the data flow and change of configuration will require care to avoid the introduction of invalid but non-flagged data.

4.1.2 Insufficient Resources

In the event that more subarray resources are allocated than are available, it is presumed that an error will be raised at the level of the CMC's response to observation management systems. Failing this, the most immediate consequence would be the issuing of new commands to X-Engine nodes which are required for observations which were previously scheduled and may already be in progress, essentially acting as an unintentional interrupt. It seems undesirable that the X-Engine should reject interrupts, but permitting arbitrary reassignment of in-use resources would risk a significant operational error; further consideration of this case is required.

4.2 Binned Integration

In the event that phase- or time-binned integrations are called for, for example to align with pulsar observations or noise diode tests, the summation formalism expressed in Equation 1 no longer holds. Instead of a single sum along t, there will be multiple 'phase bins' into which the output visibilities will be accumulated:

$$\mathcal{V}_{i,j,T,m,p,l} = \sum_{t\,|P_i(t)=p} \mathcal{E}_{i,j,T,m,k,t} * \mathcal{E}_{i,j,T,m,k',t} \tag{3}$$

This explicitly assumes that, within a single subarray, the mapping of time to phase bins is the same for each channel block and frequency channel – that is, the period and desired phase resolution are constant across the observing band.

4.3 Repeated Fast Switching: Paired Antenna Calibration

Based on ngVLA-internal discussions, one area in which subarrays may be applied is for paired antenna calibration, in which a set of antennas is periodically all aimed at a calibrator and then split between that calibrator and a science target. In this case, there would be two sets of subarrays comprised of the antennas observing the same source. These subarrays would need to change whenever the antennas are re-targeted, emphasizing that the subarray switching would need to be quick and minimally disruptive, and preferably that the overhead involved in repeatedly changing between two subarray configurations be minimized.

4.4 Subarrays in Reverse: Partition

Another case to consider is one in which many antennas with the same timing and frequency configuration are pointed at many different locations on the sky. Contingent on support from the downstream software, the most effective way to handle this in the X-Engine would be to keep them all in a single subarray during correlation and then re-partition the data at a later stage of processing. Although this would involve calculating unnecessary inter-target correlations and consume limited output bandwidth, it would only require the allocation of a single subarray and so could support many more simultaneous observations than if each were required to form its own subarray.

This plan breaks down if frequency channelization and integration times need to differ between the antenna sets; in this case, under the current X-Engine configuration paradigm, each group of identically-configured antennas would have to be in a separate subarray.

4.5 Multiple Phase Centers in One Beam

Although outside of the formal requirements for the CSP, a use case of significant interest involves correlating multiple phase centers within a single primary beam. For each observing antenna, each center location

would be separately phase-corrected and then form a separate data stream into the X-Engine – essentially an additional subarray.

Notably, this would invalidate the scaling principles described in Subsection 3.3 – it would be entirely possible for an alternative set of subarrays to require far more X-Engine input bandwidth and processing than the full-array full-bandwidth (but single-phase-center) case. This could be prevented if the multiple-phase-center observation accepted proportionately reduced observing bandwidth, or if the other simultaneous observations were such that there were sufficient unallocated X-Engine resources. Further consultation and collaboration will be required to determine the range of desired parameters; at the very least, the X-Engine can support this to the greatest extent possible within the existing performance envelope.

4.6 Synchronous Subarrays

Another desirable extension of capabilities involves coordinated observation by multiple sets of antennas aimed at a single target – each set would have different spectral configurations but are additionally required to have approximately aligned integration time boundaries. This requires a high degree of predictability in how the X-Engine begins observations, and implies the CMC is able to precisely coordinate between subarrays; it does not otherwise constrain the X-Engine's behavior.

5 Conclusion

Subarray operation of the ngVLA is a key observational capability, one which permits efficient use of the instrument's immense potential. The ngVLA CSP as a whole, and the X-Engine in particular, benefits greatly from the granularity inherent to its design. This allows easy subdivision and reallocation of resources to support flexible subarray configurations while limiting the time and resource costs associated with each transition.

Understandably for the preliminary design phase, there are a number of technical aspects of the X-Engine design which are not yet determined. The details of the generation and application of new Scan Configurations are of particular relevance to the subarray case, as are the interfaces between the X-Engine, CMC, and observation management systems. The X-Engine, as described here and in its conceptual design, is expected to be capable of supporting the full set of subarray functions required; further consideration of example scenarios will take place alongside ongoing technical development to ensure that the breadth of subarray applications is reflected in the final system.