A Scalable Packetised Radio Astronomy Imager

by

Jason Ryan Manley

Thesis presented for the degree of

DOCTOR OF PHILOSOPHY

in the Department of Electrical Engineering,
Faculty of Engineering and the Built Environment

at the

University of Cape Town

February 2014

Supervisor: Professor M Inggs

Abstract

Modern radio astronomy telescopes the world over require digital back-ends. The complexity of these systems depends on many site-specific factors, including the number of antennas, beams and frequency channels and the bandwidth to be processed. With the increasing popularity for ever larger interferometric arrays, the processing requirements for these back-ends have increased significantly. While the techniques for building these back-ends are well understood, every installation typically still takes many years to develop as the instruments use highly specialised, custom hardware in order to cope with the demanding engineering requirements.

Modern technology has enabled reprogrammable FPGA-based processing boards, together with packet-based switching techniques, to perform all the digital signal processing requirements of a modern radio telescope array. The various instruments used by radio telescopes are functionally very different, but the component operations remain remarkably similar and many share core functionalities. Generic processing platforms are thus able to share signal processing libraries and can acquire different personalities to perform different functions simply by reprogramming them and rerouting the data appropriately. Furthermore, Ethernet-based packet-switched networks are highly flexible and scalable, enabling the same instrument design to be scaled to larger installations simply by adding additional processing nodes and larger network switches. The ability of a packetised network to transfer data to arbitrary processing nodes, along with these nodes' reconfigurability, allows for unrestrained partitioning of designs and resource allocation.

This thesis describes the design and construction of the first working radio astronomy imaging instrument hosted on Ethernet-interconnected reprogrammable FPGA hardware. I attempt to establish an optimal packetised architecture for the most popular instruments with particular attention to the core array functions of correlation and beamforming. Emphasis is placed on requirements for South Africa's MeerKAT array. A demonstration system is constructed and deployed on the KAT-7 array, MeerKAT's prototype.

This research promises reduced instrument development time, lower costs, improved reliability and closer collaboration between telescope design teams.

Acknowledgements

I would like to thank the following people for their assistance and support:

Dan Werthimer for being a lighthouse in the fog of instrument specifications. Your insight and guidance is legendary. Also, thank you for the opportunity to work with the energetic CASPER team and to your family, Mary-Kate and Willy, for hosting me in Berkeley. The late Don Backer for always making time in his busy schedule to advise me and for providing me with the opportunity to participate in the PAPER experiment. I will forever remember him fondly. Aaron Parsons for the groundwork of the CASPER correlator and his insights and novel solutions to engineering problems. The rest of the CASPER, RAL and BWRC teams at UC Berkeley for providing much of the infrastructure that made this project possible and for keeping me entertained during my nearly two years at BWRC.

Thank you to the talented and dedicated DBE engineering team at SKA-SA: Francois Kapp for making the ROACH board a reality and for his encouraging leadership of the team. Andrew Martens and Paul Prozesky for their help with the KAT-7 F-engines and beamformer implementations and for assisting with software when I was under time pressure. Mark Welz for his unending patience with me and my Linux questions. Ruby van Rooyen for her help with quantisation and PFB analysis simulations and, together with Alec Rust, for performing the KAT-7 ATP. Sias Malan for his willingness to help with journal papers and ongoing enthusiasm for the project. Dave George for the underlying ROACH infrastructure and ROACH-2. Etienne Bauermeister for the KATADC and MeerKAT high speed digital samplers. Adam, Henno, Luzuko, Phil, Shanly, SimonR, SimonX, Wesley, Vaughn and the others for always lending a helping hand when it was needed. SKA-SA's System Engineers, Adriaan, Paul and Warnich, thank you for your help in specifying the requirements for the KAT-7 and MeerKAT systems. Alan Langman for introducing me to the SKA-SA project.

To my family: thank you for their unending support and encouragement and especially to my mother for all the proofreading!

To my supervisor, Professor Michael Inggs, for his faith in me and the free reign to complete this project in my own time.

Declaration

This document and all of its contents represent my own work unless otherwise stated. I acknowledge that all contributions made by others have been cited and referenced.

I have not, and will never allow this work to be copied by anyone with the intention of submitting it as their own work.

Furthermore, I acknowledge that plagiarism is wrong and declare that this project represents my own work.

Jason Manley
13^{th} of February 2014

Glossary

ADC Analogue to Digital Converter.

ALMA The Atacama Large Millimeter Array located in Chile.

AIPY Astronomical Interferometry in PYthon; An imaging package by Aaron Parsons, used primarily by PAPER.

ASIAA The Academia Sinica Institute of Astronomy and Astrophysics in Taiwan.

ASIC Application-Specific Integrated Circuit; a piece of electronic circuitry designed for a single purpose and contained within a single chip.

ASKAP The Australian Square Kilometer Array Pathfinder located in the Mid-west region of Western Australia.

ATA The Allen Telescope Array in Hat Creek, United States of America.

ATP Acceptance Test Procedure. In System Engineering, a list of instructions for executing an Acceptance Test, to verify the correct operation of a component, to ensure that it meets its specification. This test is typically run on each device off a production line before being placed into service.

Backplane A backplane interconnect is one that uses a printed circuit board (PCB) to connect multiple other PCBs (as opposed to cables). For example, multiple processing cards that need to communicate with one another can slot into a single larger backplane, rather than multiple cables connecting to- and from- each one.

ATP Acceptance Test Procedure. A System Engineering term for a document that describes test routines to be run during an acceptance test. See also *ATR*.

ATR Acceptance Test Report. A System Engineering term for a document containing the results of an acceptance test. See also *ATP*.

Baseline Refers to a pair of antennas in an array.

BEE2 The second-generation Berkeley Emulation Engine. An FPGA-based hardware computing platform.

BRAM Block Random Access Memory; the FPGA's onchip memory.

BWRC The Berkeley Wireless Research Centre, a UCB venture together with a number of industry partners.

CARMA The Combined Array for Research in Millimeter-wave Astronomy in Cedar Flat, California, United States of America.

CASPER The Center for Astronomy Signal Processing and Electronics Research, a research group at the University of California, Berkeley.

CfA Centre for Astrophysics, a collaboration of Harvard College Observatory (HCO) and Smithsonian Astrophysical Observatory (SAO) and home to Harvard's Department of Astronomy.

Corner-turn A matrix transpose operation.

CPLD A Complex Programmable Logic Device, a device typically used to interconnect other onboard devices with simple combinatorial and sequential logic. Typically less complex than an FPGA with reduced functionality. ROACH's CPLD interconnects the PPC, FPGA and SD-card busses amongst others.

CW Continuous wave signal (a sine or co-sine tone).

DBE Digital Back-End. The real-time digital processing system of a radio receiver.

DDC Digital Down Converter. The digital equivalent of an analogue heterodyne mixer and filter.

DDR Double Data Rate. The ability to transfer data on the leading and the falling edges of a clock cycle.

EVLA The Expanded Very Large Array, New Mexico, United States of America.

FASR The Frequency Agile Solar Radiotelescope at the Owens Valley Radio Observatory in California, United States of America.

FIR Finite Impulse Response; usually a reference to a digital filter implementation who's response to a signal of finite duration is also finite, before returning to zero. As opposed to IIR.

FPGA Field Programmable Gate Array; A reprogrammable device able to perform boolean logical and mathematical operations.

FX Correlator A correlator who's order of operation is to first perform data channellisation (typically through the use of an FFT) followed by the multiplication operation (see also *XF correlator*).

GMRT The Giant Metre-wave Radio Telescope north of Pune in India.

IBOB The Internet Break-out Board; An FPGA-based processing board designed by CASPER. It was originally designed to digitise data and retransmit it into an Ethernet network for processing by BEE2s and thus has limited compute ability.

IETF The Internet Engineering Task Force, responsible for a number of Internet protocol standards definitions.

IIR filter Infinite Impulse Response; a filter design that employs internal feedback such that its response to a finite input signal may continue to be non-zero infinitely. As opposed to FIR.

KAT-7 The first phase of the MeerKAT telescope. This initial deployment, comprising 7 dishes, was completed in 2010. See also *MeerKAT*

LEDA The Large Aperture Experiment to Detect the Dark Ages, a project currently utilising the LWA.

LOFAR The LOw Frequency ARray will consist of about 90 tiled stations throughout Europe. It is sensitive between 10MHz and 250 MHz.

LRU Line Replaceable Unit; a unit of measure describing a component device that can easily be replaced on-site in the event of a device failure.

LWA The Long Wavelength Array, located adjacent to the VLA in New Mexico.

MeerKAT The extended Karoo Array Telescope of 64 dishes. Construction around 2014. See also *KAT-7*.

MGT Multi-gigabit transceiver. This refers to the high-speed serialiser-deserialiser circuitry on special-purpose FPGA pins for high-speed serialised communications.

Multicast In networking, a term used to describe the process whereby a single copy of a message can be routed selectively to multiple destinations.

NRAO The National Radio Astronomy Observatory; a facility funded by the National Science Foundation (in USA) primarily charged with providing radio astronomy facilities.

NTP Network Time Protocol. A protocol for synchronising realtime clocks over the Internet.

PAPER The Precision Array for Probing the Epoch of Reionisation in Green Bank, West Virginia and the SKA-SA site in the Karoo, South Africa.

PCB Printed Circuit Board; typically a fibreglass board with copper tracks to interconnect circuit components.

PED The Phased Experimental Demonstrator, an early experimental array constructed from commercial C-band antennas during KAT's early design stages.

PPC PowerPC; a type of microprocessor.

QDR Quad Data Rate. When used to describe memories, a dual-ported memory that can perform two operations on each port per clock cycle (DDR).

QTP Qualification Test Procedure. In System Engineering, a list of instructions for executing a Qualification Test, to ensure that a design meets its requirements specification. This is typically run only one for a given design, before beginning production of a component.

RFC Request for Comments; typically a document describing technical or organisational ideas especially relating to Internet definitions (see IETF).

ROACH Reconfigurable Open Architecture Computing Hardware; The unified CASPER hardware platform designed to be able to perform the functions of both the BEE2 and the IBOB.

SERDES Serialiser-deserialiser; the process used by multigigabit transceivers (MGT) for high-speed serialised communication links.

SETI The Search for Extra-Terrestrial Intelligence; A global project utilising multiple telescopes and volunteers' compute resources to search for extra-terrestrial communications.

SPEAD Streaming Protocol for the Exchange of Astronomical Data; A unidirectional, self-describing data format used for sending complex data structures from a source to a destination and for on-disk storage.

SMA The Sub-Millimeter Array in Mauna Kua (Hawaii), jointly operated by the Harvard Smithsonian Astrophysical Observatory and the ASIAA. Alternatively, a common RF connector, then SubMiniature version A connector.

Tape-out A term used to describe the process of manufacturing an ASIC.

Unicast In networking, a term used to describe the process whereby a single message is sent from a single transmitter and routed to a single receiver.

UCB The University of California at Berkeley.

VACC Vector accumulator.

VLA The Very Large Array in New Mexico, United States of America.

VLBA The Very Long Baseline Array; A distributed collection of receivers throughout the western hemisphere used together to form very high resolution images (offline), managed by NRAO.

VLSI Very Large Scale Integration. The concept of creating integrated circuits with thousands of logic gates in a single chip package.

XF Correlator A correlator who's order of operation is to first perform the multiplication of time-domain data, followed by channellisation of the correlated signal (see also *FX correlator*).

Units and notation

In order to clearly differentiate binary multiples, I have adopted the IEC and NIST standard of referring to multiples of 1000 in the normal SI notation of Kilo (k), Mega (M), Giga (G) etc and multiples of 1024 as Kibi (Ki), Mebi (Mi), Gibi (Gi) etc. See `http://en.wikipedia.org/wiki/Binary_prefix` for an overview.

An uppercase 'B' denotes bytes, whereas a lowercase 'b' denotes bits.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

There are many telescope installations undergoing upgrades to their existing systems and new telescopes coming online in the next decade. This includes South Africa's MeerKAT, Australia's ASKAP, India's GMRT, Italy's Northern Cross, Chile's ALMA and America's CARMA, PAPER, FASR and the ATA with perhaps the most exciting example being the proposed Square Kilometer Array. These sites will all be requiring digital processing backends to perform the realtime signal processing operations.[1]

Thanks largely to the work of Dan Werthimer, Aaron Parsons and the other researchers of the Collaboration for Astronomy Signal Processing and Electronics Research (CASPER), it is now possible to use a single, reconfigurable FPGA-based hardware platform to perform almost all radio astronomy instrumentation tasks (Parsons *et al.*, 2005, 2006). When used together with these boards' standardised signal processing libraries, design time of typical radio astronomy instruments has been decreased to weeks and months from the more traditional years. Details of some implemented instruments have been publicised (for example, Mcwhirter (2010); Nagpal and Filiba (2007); Parsons *et al.* (2008); Ransom *et al.* (2009)). However, a general purpose, scalable interconnect and architecture suitable for use in radio astronomy instrumentation applications was still needed.

This work aims to show that high speed, commodity packetised commu-

[1]See http://casper.berkeley.edu/wiki/index.php?title=Correlator for a summary of some of their respective signal processing requirements.

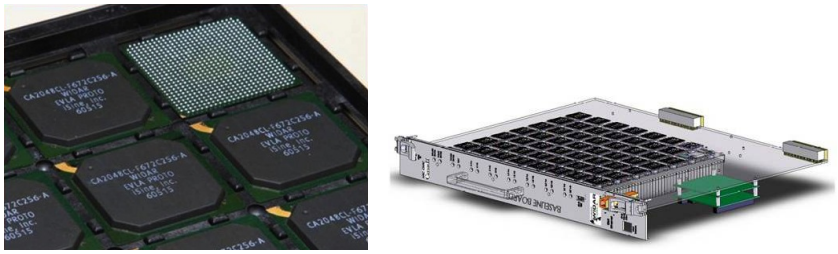
nication systems (such as Ethernet) are suitable for generic realtime radio astronomy applications. I aim to find a general-case method to construct large systems with multi-purpose boards using such an interconnect. This work represents the first general-purpose, reconfigurable, modularised, packetised, high-speed, FPGA-based design suitable for use in large, multi-purpose, next-generation telescopes.

During the course of this research work, versions of this system were deployed around the world and are currently in active service by PAPER, the INAF and SKA-SA. The architecture is also under evaluation by others including the GMRT, CARMA and LEDA.

1.1 The problem

Radio astronomy instruments can take many years to develop, at a cost of millions of dollars. This is often due to the fact that each instrument has been custom designed using the latest technologies to ensure the highest possible performance. The systems are designed to meet a specification and their function and abilities are fixed at design. They are usually not designed for upgradability and any improvements typically require significant re-engineering of components. The prohibitive redesign cost of the instruments resulted in existing processors being used far beyond their designed lifetimes. Perhaps the most recent example of this prolonged process is the upgrade of the 30-year old VLA instrumentation to the EVLA, whose WIDAR correlator, Carlson (2000b) shows, was expected to span 27 racks at a cost of 11 million US Dollars, excluding digitisers.

The early digital correlators utilised rooms full of discrete components. More recently, application-specific integrated circuits (ASICs) have been used. However, like their larger discrete counterparts, these highly customised designs still take a team of engineers years to develop and are very expensive. Furthermore, because tape-outs using newer processes is essentially a new design, the manufactured hardware must undergo extensive testing and verification before the system can be deployed. This means that by the time the instrument is used, Werthimer (2007) claims it is usually obsolete. In addition, traditional interconnects between the processing boards



(a) EVLA WIDAR ASICs. (b) EVLA WIDAR baseline card.

Figure 1.1: The EVLA employs a WIDAR correlator, based primarily on FPGAs and ASICs with a backplane interconnecting the cards.

have been complicated and difficult to set up correctly and often require a custom, once-off design. Integrating the design alone can take months to complete, as recently demonstrated with the ATA's FX backplane correlator.

To further complicate matters, because this development is such an expensive, time-consuming exercise, scientists often request that the instrument be able to perform diverse additional operations alongside its core function, the idea being that these might become useful later. These additional add complexity and are seldom actually used if they are ever made to work correctly. However, their inclusion in the initial system prolongs the design and development time.

These problems are eased in modern designs employing reconfigurable FPGAs. But the FPGAs are typically still statically connected and configured, limiting their general-purpose application by placing restrictions on the signal path through the processors.

Logistics and maintenance is also complicated by custom electronics. For example, if a hardware processing board were to malfunction, debugging can be a challenging affair requiring the expertise of an engineer to perform the fix. The repair could require a fresh batch of the custom PCBs or ASICs to be manufactured if the original stock of spare components has been depleted — a costly exercise when even possible.

This research aims to address these issues and to find a solution which enables digital back-ends in radio telescopes to more closely track the advancements in modern electronics while decreasing costs and development

times.

1.2 Initial guiding requirements

It is proposed that digital processing back-ends use a generic processing platform and a common, general purpose multicasting interconnect which can be reprogrammed at runtime to perform different functions and share processing resources.

The digital processor should use commercial, commodity components as far as possible and a minimum number of different types of processing boards or other custom hardware. This minimises the number of different platforms that must be supported and spare parts stocked and allows technological progress to be tracked more closely by limiting the redesign required with each subsequent generation. These processing nodes should operate independently and be replaceable in a piecemeal fashion so that the system can be incrementally upgraded and a single board failure does not result in a system failure.

Runtime reprogrammability would ensure that individual designs can be kept simple and that new designs can be loaded to support new requirements if and when they are needed, reducing initial development time. It is envisaged that instruments will be designed to perform a single function and that the back-end compute boards can be reprogrammed to enable different functions or modes of operation rather than employing a single design that attempts to do everything. These different functions can then be designed and debugged independently, rather than all-at-once during the initial design phase as is traditionally the case. Since the same hardware is being reused for the different instruments, deploying a new instrument at a facility does not require re-testing of the hardware, merely the correct operation of the new algorithms.

These reprogrammable boards should be programmed using a standardised programming interface from a standardised library. This will minimise the different runtime environments that need to be supported. Using a single library of known-good, tested, parameterised components means that verification of functional correctness needs only take place once. Designers can

have a certain amount of confidence in newly constructed instruments when using these pre-approved libraries. An overall acceptance test is sufficient for each completed instrument.

1.3 CASPER: towards an improved solution

CASPER, based at the University of California in Berkeley has also envisaged such an architecture: a generic digital back-end which allows the use of commercial, off-the-shelf computers to be used in conjunction with reconfigurable hardware interconnected by network switches (Parsons *et al.*, 2005). Two years were spent working closely with the CASPER team at the Berkeley Wireless Research Center (BWRC), UC Berkeley, to develop this architecture.

The proposed architecture is outlined in Figure 1.2. I have made significant progress towards this solution during the course of this research and have proven its feasibility and usefulness.

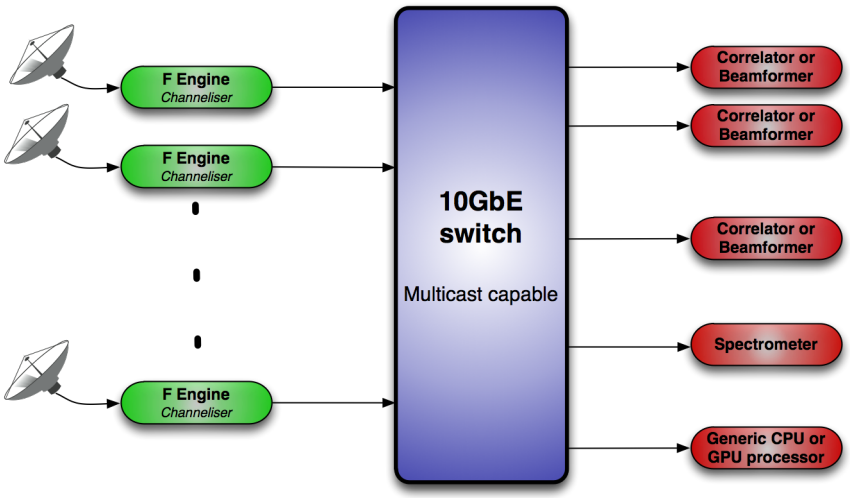


Figure 1.2: The proposed CASPER architecture.

The system allows for commensal operation of multiple instruments simultaneously by multicasting data digitally to compute engines. Reconfigurable hardware components can be reused for different purposes. It can be

expanded and upgraded as required simply by adding additional processing hardware to the central switch and changing the software. This work represents the first design and implementation of this architecture for a general purpose radio astronomy instrument.

This computer can assume different personalities by reprogramming the telescope's digital hardware and so the instrument can be tailored to the experiment which the scientist would like to run. I will design a system that accommodates this requirement but, for demonstration purposes, will focus the design on a single instrument which is representative of a typical telescope array facility.

In addition, I aim to show that by sharing hardware components, software libraries and other resources, additional instruments can be added simply and easily to an existing system without incurring significant costs.

This thesis will focus on implementing real instruments based on a switched CASPER architecture for use by KAT-7, MeerKAT and other installations. GMRT, the ATA, PAPER and CARMA (among others) have also expressed an interest in this design. The flexibility of the design makes it easily adaptable to their specific requirements.

A generic, unified design fosters collaboration as installations can share knowledge and intellectual property. It allows for all telescopes to share in the design advances of any one facility instantly without redesigning their own system or purchasing new hardware. This was not possible before CASPER as every installation operated different custom hardware and software designs.

This project will ultimately save time and money by sharing not only hardware but also instrument design IP and hence enable scientists to get larger volumes of higher quality data faster and more economically.

A recurring theme throughout this work is that of standardisation and re-use. Standardisation offers a number of benefits and the process of standardising various parts of the system guided this research. The following subsections outline the standardisations applied to the system architecture, processing hardware and interconnect along with the resultant benefits.

1.3.1 Standard hardware

I aim to limit the number of different types of hardware platforms with the goal of accommodating all the required functionality in a single generic hardware platform which could be used for any compute function. Ideally this should be a commercial, off-the-shelf commodity item but since a suitable platform was unavailable, one was designed in collaboration with SKA-SA, CASPER and NRAO. The resultant device, ROACH, was open-sourced and has been successfully adopted by the radio astronomy instrumentation community. Details of the hardware can be found in Chapter 3.4 of this work.

Using a single platform accelerates development and lowers system cost as only one type of board needs to be designed, debugged, maintained and supported. In addition, it lowers costs indirectly by reducing the required number of spare boards to be kept on-hand at the observatory in case of failure, thereby reducing storage requirements and investments in idle inventory.

Having fewer hardware designs makes it easier to track Moore's Law; designers only have to redevelop one hardware board in order to upgrade their entire system.

With a single multipurpose platform, the whole astronomy instrumentation community can make use of this single platform to perform all their required functions. By leveraging the increased volume of production, costs can be further lowered.

1.3.2 Standard architecture

Using a suitable standard architecture accelerates the development of new instruments. This prevents designers from having to invent new architectures for each instrument and falling into the all-too-common engineering trap of over-engineering and over-optimising the design. The digital back-end needs a generic, flexible architecture which allows for datapath re-ordering when device personalities are changed without requiring any recabling or physical modifications to the system.

It is noted that introducing a fixed architecture could result in an inefficient system design for a given instrument since this prevents optimisation through system layout and to some extent, design partitioning. But I aim to

show that the benefits of a standardised architecture outweigh the potential losses in efficiencies and that these losses can be kept to a minimum if the initial system architecture design is sufficiently generic.

To keep the design as generic as possible, I chose to interconnect all processing nodes with a full-crossbar switch, allowing any device to communicate with any other device at full speed.

1.3.3 Standard interconnect

The interconnect should outlive the hardware designs and enable incremental expansion. This is crucial for allowing mixed hardware generations and piecemeal upgrades of the system which, in turn, allows for cost-effective tracking of advancements in processing technologies. The application layer interface should also remain consistent to prevent having to redesign the interconnect protocol across different devices and hardware generations.

Additional processing nodes added into the system can be of any design or type suitable for the task to be performed. The interconnect should not dictate a specific technology for the processing elements. The ubiquitous Ethernet networking standard was ultimately chosen as the system interconnect at 10Gbps rates in this implementation, with expansion capabilities to 40Gbps and 100Gbps already available and higher speeds currently in development.

The communication protocols across these links, between processing elements and to the outside world, should be solved generically. This would allow for a standard software interface to all instruments, allowing for re-use of communication libraries and thus further accelerating the development of new instruments by reusing logic from previous designs.

No suitable existing protocol could be found and so the Streaming Protocol for the Exchange of Astronomical Data (SPEAD) was developed in collaboration with Aaron Parsons of PAPER and other members of the SKA-SA team.

SPEAD is used in this work and also forms the native protocol for exchanging intermediate products in SKA-SA's science processing pipelines. SPEAD is a self-describing format suitable for propagating data across net-

work links or for sequential on-disk storage of data. It is highly flexible, supporting complex structures and multidimensional arrays of such structures, including on-the-fly resizing and redefinition of data products along with machine and human readable descriptions. This allows the receiver to not only reassemble, but also interpret the data without any additional information. The protocol is able to receive data from multiple sources simultaneously. It was designed with high efficiencies in mind and so only *changes* in data products are propagated across data links. A subset of the protocol is also simple enough to implement on low-level logic devices such as FPGAs and 8-bit microprocessors. The protocol specification and links to reference implementations can be found at `https://casper.berkeley.edu/wiki/SPEAD`.

1.4 Context

This research is being funded by SKA-SA as part of their design studies. Implementation decisions throughout this work are guided by their requirements for an array processor able to perform the functions of the digital back-end (mainly correlation, beam formation and transient searching and timing) for both the KAT-7 and MeerKAT arrays, whilst remaining flexible enough to accommodate future requirements. A demonstration machine is constructed and deployed on KAT-7.

Instrument functionality and performance sufficient to produce data of scientific value is required. Instrument evaluation will proceed according to SKA-SA's Digital Back-End (DBE) Acceptance Test Procedures (ATP), (van Rooyen and Rust, 2012, 2011). In addition, multiple instruments are constructed and deployed on existing telescopes to process real datastreams. High-fidelity astronomical images have been produced using this work on KAT-7, GMRT, PAPER and Medicina's Northern Cross telescopes.

The end product will be a methodology for designing general-purpose radio astronomy instruments, along with a demonstration system on KAT-7 consisting of a combined correlator and beamformer – the primary array instruments. These also arguably represent the largest and most complex radio astronomy instruments and thus the most challenging engineering problem. While focus will lie with the development of these instruments, consideration

will also be given to the other typical instruments where they affect the architectural design so as to keep the architecture suitably generic to support all common radio astronomy instruments.

This research has a clear application. It shows potential for significant cost savings, while producing more versatile instruments. This work builds on the latest radio astronomy instrumentation research. It is of immediate benefit to the radio astronomy community.

1.5 Hypothesis, objectives and research questions

The overarching hypothesis for this research project is the following:

HYPOTHESIS:

It is possible to develop a cost-effective, scalable, flexible, radio astronomy imaging instrument for array signal processing using reprogrammable hardware and Ethernet interconnect.

Research Question 1: *Is it possible to design a radio astronomy signal processor using an Ethernet interconnect?* Smaller systems have already been built using an Ethernet interconnect, proving that this is possible for at least some instruments. In this work, I successfully design, construct and deploy a number of wider-band FPGA-based radio astronomy instrumentation machines that are interconnected using switched Ethernet. I focus on frequency-domain correlators and beamformers. The implementation is discussed in §4.

Research Question 2: *What does the system cost, and what is the cost penalty for an Ethernet interconnect?* The developed machine must be cost-effective when measured against machines that were designed using alternative construction techniques. I address this question in §5.2. Costs include design time, construction (hardware) costs as well as operational costs, such as power and cooling requirements.

I consider the costs of the PAPER-32 (Appendix A, Table A.3) and KAT-7 Appendix B systems and find that the initial cost of a switch represents a

barrier of entry for small systems, adding significant cost (18%) when only a fraction of the ports are needed on the smallest commercial high-speed switches. For larger systems where most ports on the switch are utilised, switching costs represent less than 10% of the instrumentation cost.

I opted to implement the reference design in this work using FPGAs. Analysing the lifetime cost of this technology, given assumptions of the cost of electricity and trending technological advances, I conclude that if the system is to be operated for over 20 years without a technology refresh, then FPGAs result in a lower cost of total ownership than GPUs. However, under these same conditions I also find that it can be more cost-effective to regularly replace the deployed hardware.

FPGA resource utilisation is also analysed. While resource requirements for implementing Ethernet on FPGAs are already low (approximately 5% on KAT-7), I conclude that as FPGA devices become larger and Ethernet links faster, the fractional cost of an Ethernet implementation will decrease and will soon be insignificant.

Research Question 3: *Is the system scalable?*

I define scalable in this context to be an instrument that is adaptable across the following three processing metrics of typical arrays: the number of antennas, processed bandwidth and spectral resolution.

This is tested by successfully constructing multiple machines that process various bandwidths, number of antennas and at different spectral resolutions. Scalability is an important aspect, especially in light of next-generation instruments such as the SKA which will require significantly larger processing systems than have been designed to date. This project is concerned with processing systems for the large next-generation array facilities and aims to meet their signal processing challenges, rather than smaller, targeted systems.

To demonstrate scalability, systems of varying sizes are constructed. It should be possible to scale the system by incrementally adding the necessary hardware without redesigning the existing architecture. Any changes to system architecture or resultant disturbances to the existing system would constitute a failure of the hypothesis. I analyse scalability aspects of the design in §5.3.

Research Question 4: *Is the system flexible?* I define a flexible machine as one that is able to cope with changing science requirements and is adaptable for installation at different telescope facilities (as opposed to a custom-designed instrument that is tailored for a specific telescope). This flexibility is demonstrated by deploying a system at different telescope facilities. Two examples of PAPER and KAT-7 are included in Appendices A and B. A third system was deployed at Medicina in Italy, and a fourth at GMRT for evaluation, but these are not described here.

Not only should design parameters of existing instrumentation be modifiable (such as the instrument's spectral resolution, bandwidth and number of inputs, for example) but new algorithms should be implementable through software, without requiring hardware architecture changes. This single machine should be able to host multiple instruments, to perform different science (wideband imaging, spectral line imaging, beamforming, pulsar timing, transient searching etc), rather than being single-functioned. An example of where such agility might be useful on existing instrumentation is the implementation of different RFI-excision algorithms depending on the EMI-culprits in the frequency band currently being observed. Adding such an additional processing stage into the DSP chain, or modifying an existing one, should not disrupt or require a rework of the architecture or physical intervention (hardware change or recabling).

To demonstrate this flexibility, I add a beamformer instrument to the existing correlator, and deploy it remotely. I leverage existing compute engines' intermediate compute products and free processing capacity in the X-engines to enable effective commensal observation and thus realise two instruments for the cost of one. Adding this new functionality is possible without requiring a redesign of the architecture. A discussion of this addition can be found in §4.6 and the results of this addition on KAT-7 can be found in Appendix B.5.

To allow the hosting of arbitrary algorithms, the system is based on reprogrammable devices which allow not only the processing elements' functions to be redefined, but also the flow of data between these elements. Various processing nodes and interconnect options are considered for this purpose and in §3 I ultimately choose FPGA-based processing nodes, interconnected

by an Ethernet network. To achieve the required flexibility, all processing nodes are interconnected by a full-crossbar, non-blocking Ethernet switch. Such switches allow for any node in the network to communicate with any other node at full line-rates. This allows for arbitrary DSP operations and also arbitrary data flows between processing nodes while ensuring no packet losses.

The architecture is processing-platform and instrument independent. This will be verified by constructing a heterogeneous processor consisting of multiple processing elements. It is demonstrated that the design is reusable across hardware generations, beginning with Virtex-II Pro based iBOB and BEE2 systems in PAPER (Appendix A) and migrating to Virtex-5 based ROACH hardware on KAT-7 (Appendix B). System designers are able to progress with technology developments without reengineering the whole system. This requires that the existing source code is able to be recompiled at a high-level, negating the need for instrument redesign across hardware generations. Naturally, this would exclude changes required in order to support new device-specific hardware features.

Although FPGAs were used in this work, the processing elements need not be FPGA-based as Ethernet allows interconnections across a wide variety of processors. This ensures a future-proof architecture whereby a yet-unforeseen platform might need to be added into the system to perform a highly specialised function.

In all cases, the design needs to be repeatable, reliable and have a deterministic effect on the data. This is significant when the instrument is to produce scientifically valuable data from telescope arrays.

By the end of this process, a methodology for designing scalable systems will be delivered along with a demonstration system suitable for use on KAT-7. At the least, this reference design should encompass the functions of a correlator and beamformer and should be scalable to meet MeerKAT requirements.

1.6 Methodology, experimental procedure and thesis overview

This project will proceed in a phased approach, beginning with research into existing instrumentation designs. This will be distilled to produce a design for a generic packetised processing machine. Correlation and beamforming functions will be implemented on this machine to demonstrate its functionality. This implementation will be described in detail before analysing it for shortcomings and limitations. The basic flow of the project, and this text, is presented in Figure 1.3.

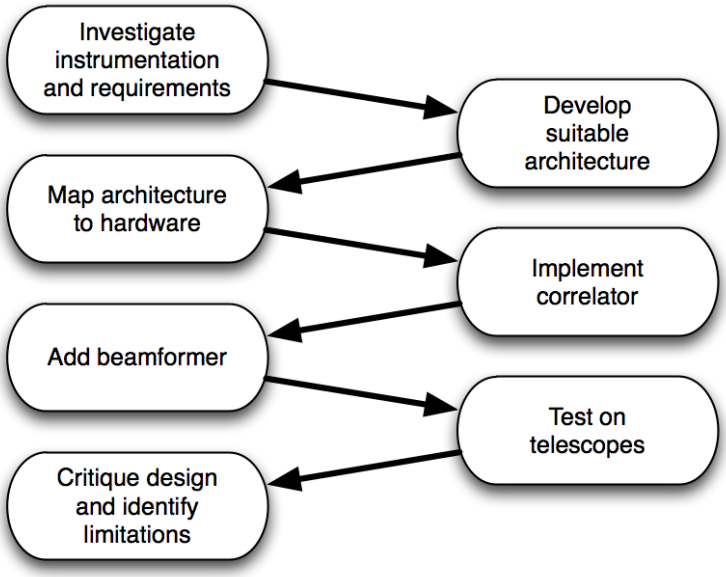


Figure 1.3: Project methodology flow diagram.

The project will proceed as follows:

- Investigate typical radio astronomy instrumentation system requirements and specifications. This will primarily be achieved by talking to scientists and experienced instrumentation designers and reviewing SKA-SA requirements specifications.
- Investigate potential solutions, including existing designs, architectures

and suitable hardware platforms. Expose the problems and limitations of existing designs and propose solutions.

- Develop a new – or implement an existing – generic system architecture that is able to scale dynamically with telescope requirements and is able to host multiple instruments using suitable hardware.
- Map this architecture onto a suitable hardware platform including the allocation of processing resources and system interconnect.
- Design a flexible, scalable correlator to be hosted on afore-mentioned architecture and hardware including the design of the filterbanks and other signal processing operations to meet the required specifications.
- Demonstrate the scalability of the design by recompiling for a different telescope facility of varying size.
- Add beamformer functionality to the existing design to demonstrate the upgradability and flexibility of the architecture.
- Analyse the design for shortcomings and limitations and consider the implications for SKA-SA's upcoming MeerKAT array.

The phases of the project are mapped to chapters in this report. What follows is a summarised description of these chapters.

Review of existing instrumentation (Ch. 2)

This chapter begins by outlining the operation, construction and the signal processing tasks of common radio astronomy instruments. Specifically, the basic single-dish radiometer, spectrometer and pulsar machines are outlined, as well as the array processors performing correlation and beamforming functions.

The radio astronomy instrumentation community has already done a significant amount work to find optimal architectures for single-purpose radio astronomy instruments. This chapter discusses the findings of existing solutions and their shortcomings.

Optimisations to one instrument or application usually have a small effect on this instrument itself, but may adversely affect another. Tradeoffs and limitations for commensal operation are discussed here.

A working overview of additional processing complications and requirements arising from earth rotation effects and Walsh switching operations (commonly applied in the analogue stages to lower crosstalk between inputs) is then provided.

After reviewing the instrumentation requirements, current implementations at existing facilities are considered. Practical implementations have typically suffered from high costs, inflexibility, long design times, poor serviceability and the inability to leverage design re-use in other similar instruments. Most of these existing designs are only deployed at the telescope for which they were designed. Re-use is limited to duplicate systems being installed on other telescopes (such as the SERENDIP spectrometer for SETI or the VLBI recorders). Tailoring designs to suit different facilities has proven difficult with a lot of rework required. The limited subsequent reapplication of the various existing designs demonstrates their shortcomings and clearly motivates for an improved solution.

Component selection (Ch. 3)

This chapter begins by comparing the processors and interconnect systems that are currently available as potential hosts for this work. It is concluded that FPGA-based platforms and UDP/IP over an Ethernet interconnect are best suited to these requirements.

The FPGA platforms available are then reviewed in order to find a suitable device. None were found to be directly applicable but the existing CASPER boards were found to be close to what are needed. Two prototype correlator systems were designed and deployed using this legacy hardware (for details of these systems, see Appendix A) before a new hardware platform, ROACH, was developed by SKA-SA in collaboration with CASPER and NRAO for use on KAT-7.

The ROACH board was designed to be compatible with the existing CASPER toolflow while including a superset of all functions and features

of the older boards but at a lower price point, using a later generation, larger FPGA device. This board became very popular with the radio astronomy instrumentation community; over 300 boards were servicing more than 30 institutions worldwide within two years of its introduction.

Implementation (Ch. 4)

This chapter maps the system requirements into a design, including partitioning across boards. I begin by implementing an FX correlator. The operation of this system and the entire signal processing chain is scrutinised in detail.

While the developed machine is generalised to ensure scalability, the initial design target was PAPER's 8 antenna system deployed in 2008. Specifics for this setup are discussed in Appendix A.

I discuss opportunities for commensal operation by allowing multiple digital back-ends to share data and processing resources and the use of multicasting to support such distribution of data to multiple boards simultaneously without significantly increasing interconnect overhead. This is demonstrated by the addition of a beamformer.

Implementation analysis (Ch. 5)

Here I discuss the lessons learnt from the early designs and describe how the system may be improved for MeerKAT and subsequent arrays. I answer the research questions related to realisability, cost, scalability and flexibility.

The architecture is largely a success but there are a number of aspects of the design that need to be carefully considered. Possible future work is outlined, such as a multicast implementation. Further, I discuss the limitations of high-level design methods when attempting to port to new hardware platforms with different capabilities. The port of PAPER's iBOB F-engine to KAT-7's ROACH F-engine is an example of such a migration. Limitations encountered with the Simulink/System Generator development environment are outlined and I propose possible solutions.

Conclusion (Ch. 6)

I conclude with an overview of the design's achievements and shortcomings and thoughts on the future of radio astronomy instrumentation.

Naturally, there is a specific focus here on MeerKAT's requirements.

Appendix A: PAPER

The correlator design described in this work was prototyped on the PAPER array at NRAO's Green Bank site in 2007 using legacy CASPER (iBOB and BEE2) hardware and later combined with ROACH hardware. This chapter describes the early prototype deployments, how they differed from the concept design and how they subsequently evolved.

Appendix B: The KAT-7 digital back-end

This chapter describes the KAT7 system and how it differs from the PAPER and Medicina back-ends that preceded it. KAT-7's system was the first instrument constructed entirely on ROACH hardware. It follows the original design architecture more closely than the earlier systems.

This chapter introduces the KAT-7 array, with a particular focus on its digital back-end (DBE). I describe the hardware components, operating configurations and specifications and FPGA resource utilisation for the all-ROACH based system. Subsequent additions of narrowband (spectral-line) correlator and beamformer modes are discussed before I conclude with some early KAT-7 science results.

Appendix C: Looking ahead to MeerKAT

The KAT-7 system architecture will be modified to support MeerKAT and SKA-1, further generalising the prototype architecture to include time-domain data in the switch and full support for multicasting. Here I discuss the unique requirements of this arrays and implications for its digital back-end. A design is proposed to meet all of MeerKAT's requirements. This concept has since been accepted after a preliminary design review of MeerKAT's system-level design in June 2011 by an independent review panel.

1.7 Way forward

What follows this introduction is a review of existing instrumentation, including construction techniques and an overview of some existing telescope facilities.

Chapter 2

Existing instrumentation

The fundamental research of radio astronomy interferometric synthesis was conducted in the 1960s. The 1970s saw modern, large arrays such as the VLA being constructed. Since then, imaging and processing techniques have been refined, though rarely with significant changes to the fundamental operations. These instrumentation systems are large and progress is slow and incremental rather than revolutionary. This work continues to build on the successes of the early work.

Textbooks and reference works on radio astronomy interferometry began appearing in the 1980s and have continued to be updated into the 21st century. Perhaps the most well-known of these is Thompson, Moran and Swenson's Interferometry and Synthesis in Radio Astronomy, now in its second edition, which formed a core reference for this work. It covers topics from instrumentation to synthesis and the implications of system design on the produced images.

Modern research has been driven by scientific and engineering requirements of new arrays such as CARMA, ALMA, LOFAR and the ATA or upgrades to existing facilities such as Westerbork or the EVLA. Since these facilities typically take more than a decade to design and commission, the advances from each iteration can take a long time. A lot of this work is specific to the particular array being constructed and is typically published in memorandum or report form along with the rest of the telescope documentation. Only selected components are published in traditional scientific

journals, often making it necessary to refer back to the telescope's memo series or to presentations and discussions with the engineers involved in these projects to get up-to-date information. The citations in this work reflect this situation.

The following sections will outline the basic radio astronomy array instrumentation, with a specific focus on correlators, which is the standard instrument found on all interferometric imaging arrays. I consider the commonalities between the facility instruments before reviewing some existing machines in use at various facilities.

2.1 Radio astronomy instrumentation architectures

The astronomy instrumentation community has already undertaken a significant amount of research into potential architectures for radio astronomy digital back-ends. However, most of these papers have focused on optimising a particular design or instrument for a specific installation, in order to optimise the primary task for that array. Many novel architectures and interconnects have been proposed to solve a particular design problem or otherwise optimise a component of the system. This work brings together a number of these ideas and concepts in an attempt to construct a general purpose computing machine suitable for use on both general purpose and specialised telescopes.

The processing nodes themselves were traditionally custom built for specific portions of the required compute task. Until very recently, a new digital back-end would need to be redesigned from scratch for each installation. CASPER has made it possible to use a single hardware platform to perform multiple tasks, simply by reprogramming the devices in the system. This has decreased design time of these imaging instruments to weeks and months rather than years (Parsons *et al.*, 2005, 2006).

A next step is to develop a general purpose interconnect and architecture suitable for use in radio astronomy instrumentation applications. Perhaps an obvious option is to use the commercially highly successful Ethernet com-

puter networking standard. However, it has not yet been shown that high speed Ethernet communication (10Gbps and higher) techniques are suitable for wideband (hundreds of MHz and higher) realtime radio astronomy applications with its restrictive timing requirements and high data rates. The construction of large systems with such interconnected boards has not yet been generally achieved. This work represents the first reconfigurable, modularised, packetised, high-speed FPGA based design suitable for use in large, next-generation telescopes.

During the design of this system, I remain sensitive to the need to optimise power consumption (which significantly affects cooling and operating costs) and space constraints as shielded rooms are expensive and rack space is often limited in existing facilities.

2.2 Common instrumentation

This section gives an overview of instruments typically found at modern telescope facilities. Along with descriptions of their operation, I also describe typical construction techniques and outline complications and design tradeoffs that must be made. It provides a summary of popular radio astronomy instruments and is not meant to be an exhaustive review but rather provides a working overview of the designs and their challenges.

The scientific requirements for typical radio astronomy instruments need to be well understood if meaningful optimisations and tradeoffs are to be made. Many papers exist which concentrate on one or more optimisations of interest to this research. I will focus on the design of the correlator, a core requirement for all modern interferometric telescopes and arguably the most technically challenging. I will concern myself with the detailed design of CASPER packetised correlators (Parsons *et al.* (2008)). However, related works and other instrument types must also be considered in the interests of keeping the framework generic enough to be useful as an array facility. Specifically, spectrometers, pulsar machines and beamformers are all of interest. Moreover, high level system requirements must be considered carefully before implementation. The configuration of certain instrument parameters can affect other system properties and there is often a tradeoff

to be made when designing these systems.

This section will describe the basic operations of radio astronomy instruments along with overviews of existing implementations.

2.2.1 Radiometers

This is probably the simplest radio astronomy instrument and measures the total electromagnetic power received by the antenna. Digitally, this is implemented by squaring and integrating the incoming voltage signal. This is not used much on modern telescopes as a science instrument but is often still present for engineering purposes and for system set-up and calibration. A power meter cannot, for example, differentiate between the incoming sky signal and terrestrial interference, making this instrument largely useless on wideband streams for anything other than for preventing saturation at various stages through the signal chain.

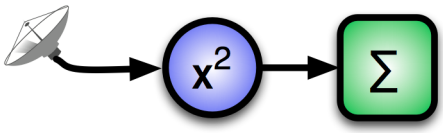


Figure 2.1: Block diagram depicting a basic power detector.

Digital implementations usually dispense with an integrator in favour of a simple accumulator and then scale the output appropriately afterwards (by dividing by the number of accumulations).

2.2.2 Spectrometers

Spectrometers (for example, Benz *et al.* (2005)) measures the power received in various frequency bands. Historically, this was done using an analogue bandpass filterbank to channelise the incoming signal into different frequency bins and then calculate the power in each of those bins using radiometers, as per §2.2.1. Modern digital designs achieve much higher spectral resolutions by employing an ADC and digital circuitry to perform the channelisation. FFTs can achieve high spectral resolutions efficiently.

Commercial bench-top instrumentation that performs this operation is commonly available in the form of spectrum analysers. Wideband spectrum analysers typically employ mixers and scan through the band, observing only a narrow band at any one time. This significantly increases the observation time, depending on the spectral resolution and noise floor required. Integrating units are significantly more expensive and are bandwidth limited but enable faster observations.

To construct such an instrument, the simplest approach might be to use an FFT directly for channelisation. But the FFT's sinc response (due to the inherent rectangular input window) has a poor out-of-channel rejection ratio with the first sidelobe only 13dB below the main lobe. This is normally considered insufficient for use in radio astronomy applications and digital spectrometers thus usually use a windowed FFT (if broadened channels can be accommodated) or a polyphase filterbank at its core. The accumulation period is usually limited by the digital data bandwidth available at the instrument output. Modern electronics are able to process and transport data ever more quickly so that this limitation is disappearing, but now storage and post processing is becoming a bottleneck.

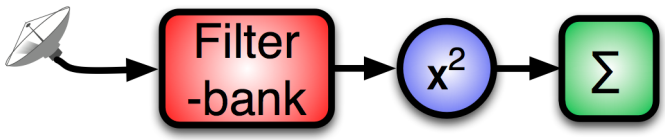


Figure 2.2: Block diagram depicting a basic spectrometer instrument.

The processed bandwidth, frequency resolution and accumulation periods differ significantly depending on the science driver. For example, the SETI spectrometer currently deployed at Arecibo calculates 128 million channels over a 200MHz bandwidth and incorporates thresholding logic at the output to only record channels which contain signals above the background noise level.

For continuum work, the largest processed bandwidths are required, to

maximise SNR, as shown by the standard radiometer equation

$$\sigma = \frac{T_{sys}}{\sqrt{BT}} \tag{2.1}$$

where B is the total receiver bandwidth and T is the integration time.

Spectral line observations typically require only a narrow band around the emission or absorption line of interest. The required bandwidth for a spiral galaxy observation, for example, is determined by the observed Doppler speed (such as when observing galaxy rotation), given by $\frac{\Delta f}{f} = \frac{\Delta v}{c}$, where Δf is the bandwidth in Hz, f is the rest frequency in Hz, Δv is the velocity span in m/s and c is the speed of light. This same formula can be used to calculate the required channel resolution given a desired velocity resolution. Molecular line surveys can require very high spectral resolutions, and bandwidth is often traded for this increased resolution. KAT-7's highest resolution mode, for example, requires channel resolutions below 580Hz for spectral line work, but over a bandwidth of only 580kHz.

In the case where the spectrometer forms the front-end for a pulsar machine, the accumulation period should be kept to a minimum to improve the time resolution and the widest possible bandwidth processed to increase sensitivity (see §2.2.3 for additional details). A complication for pulsar work stems from the PFB length when no output accumulator is used (maximum temporal resolution): for a given bandwidth, increasing the spectral resolution correspondingly decreases the time resolution possible as the filter bank's processing window lengthens.

Spectrometers are used with single dish signals. Their application to arrays is limited to individual dishes or to the output of the beamformer (see §2.2.5). Correlators, however, also typically produce spectral outputs, in which case these same arguments apply.

Some additional implementation considerations are:

- Bandwidth to be processed. For continuum or wideband observations, this is usually as much as possible because processed bandwidth maps directly to sensitivity. For spiral galaxy spectral line observations, this is determined by the size and speed of the galaxy's movement, which causes Doppler shifting of the rest frequency. This has a direct bearing on the amount of data that must be moved through the instrument.

- Frequency channel resolution. For spiral galaxy spectral line observations, for example, this is set by the desired speed resolution for each arm. The frequency resolution primarily determines the computation load.

- Out-of-channel rejection ratio, which determines number of taps required in the filterbank. This has a direct bearing on spectral line-to-line and line-to-noise selectivity.

- Choice of windowing function for polyphase filterbank. An informal survey of existing systems suggests that Hanning and Hamming windows appear to be the most popular.

- Complications arising due to bitgrowth in longer FFTs, and suitable management techniques.

- Dynamic range required, both instantaneous and averaged, and the permissible error term, which determines the use of floating point or fixed point arithmetic and the number of bits needed.

- Accumulation period and bitgrowth inside the final vector accumulator.

2.2.3 Pulsar timing and searching

Pulsar machines (for example, Demorest *et al.* (2004)) come in two forms: one designed to find pulsars and another to time them accurately. Both are essentially rapid-output, high time resolution spectrometer front-ends and dedispersion back-ends. The difference is in the channels resolution and dedispersion algorithm.

In the case where the instrument is searching for pulsars, it must search over pulse period (by folding at various periods) and dispersion measure (DM) (an operation known as dedispersion) variables. Searching instruments often employ fewer, wider frequency channels along with the computationally simpler but numerically less accurate incoherent dedispersion algorithm. This algorithm involves delaying and summing frequency channels. The search space can be vast, as hundreds or thousands of DMs are

trialled. This algorithm is well-suited to GPUs as a small dataset can require a lot of computation.

Once a pulsar is identified and the periods and dispersion measures determined to first order, a timing instrument is used to accurately determine the pulse period and profile as well as the dispersion measure using a coherent dedispersion algorithm.

Folding in time with a known period improves signal to noise ratio for detecting weak pulsars.

Modern trends are to place these algorithms on GPUs. The pulsar instruments have historically been connected to large, single-dish telescopes but they can equally be used on arrays by coupling them to the output stream of a beamformer.

The pulsar instrumentation concepts are well explained in Lorimer and Kramer (2005) and readers are referred to that text for further explanation of pulsars and algorithms.

2.2.4 The case for interferometers

The science of radio astronomy started by using single-receiver type telescopes. The simplest instruments (such as level detectors and spectrometers) are designed for single feeds from single receivers. Larger dishes have greater collecting areas (and are hence more sensitive, all else being equal) and have increased spacial resolution. Their narrower main beams, however, can hamper their use for certain sciences, such as when large objects are to be observed or when surveying large areas of the sky. While a smaller dish's wider beam is less sensitive, it is able to observe a larger part of the sky in the same amount of time.

Using many smaller dishes gains the observer the best of both scenarios: the ability to observe a large part of the sky simultaneously, while maintaining a high sensitivity. However, such a setup requires additional instrumentation and signal processing. Perhaps most obviously, it would be good to be able to emulate a large single dish from an array of smaller dishes (this is achieved by an instrument called a beamformer) in order to continue to use existing instrumentation.

2.2.5 Beamformers

A beamformer simulates the signal from a single-dish telescope by summing the signals from an array of dishes after passing through programmable delays. It forms an electronic beam which can be steered by changing the length of the delays.

These delays were historically implemented in an analogue delay-line using lengths of physical cables. Advances in digital systems have allowed for implementations using digital memory as the digital delay line. Beamformers require very fine adjustment of the system delays before summing the signals from each antenna. Errors in the delay coefficients result in reduced signal-to-noise ratio of the summed signal. For a critically-sampled signal, shifting by a single ADC sample provides a 0 – 180 degree phase delay slope across the band. Residual errors in the delay coefficients resulting in even a few degrees phase difference between input signals can significantly reduce the instrument's SNR. Delaying by whole ADC sample clocks is thus insufficient and I require fractional delay lines and sub-sample filtering. This can be done in the time domain (see (Laakso *et al.*, 1996) for a description of this process using FIR filters) or in the frequency domain using complex multipliers.

If the beamformer is to be mated to legacy instrumentation that requires an analogue input signal then the digital beamformer's output must be a time-domain signal and passed through a DAC. In all-digital systems, it is often preferred to have the output channelised so that it can be easily distributed across multiple processors on a sub-band basis for subsequent processing.

Additionally, beamformers usually provide weighting coefficients for each input. This can be used to correct poorly matched input levels or discard a faulty input entirely by setting its weighting to zero.

Figure 2.3 illustrates a simple beamformer architecture. In this case, a single beam is formed by daisy chaining multiple adders. Data from each antenna is processed on a local board in the time domain through integer delays of the sample clock and with fractional delay lines implemented in FIR or IIR filters. Each node simply sums its antenna's signal with the incoming signal and passes the output to the next processor. The beamformer output

is taken from the last board in the chain. If frequency-domain output is required then a PFB can be placed on the output stream.

Every subsequent antenna simply needs an associated processing node to be inserted into the processing chain, making for simple scaling as the number of inputs are increased. This is a simple, convenient system to form a single beam.

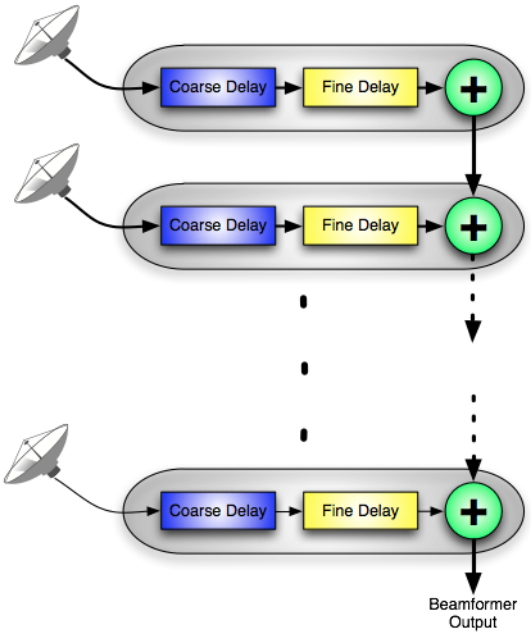


Figure 2.3: A simple architecture for a time-domain beamformer.

Time-domain, scalable beamformers of this sort have already been demonstrated on CASPER hardware by Nagpal and Filiba (2007), with some limitations. This paper also shows that for beamformer applications, DSP devices are IO limited rather than computationally bound.

Many modern observatories, including MeerKAT, require multiple beams to be formed simultaneously. In this way, multiple objects can be observed simultaneously which can significantly increase survey speeds. This time-domain architecture requires linear scaling of all components (filters, adders, interconnect and channelisers) when multiple beams are required. The computational requirements for beamforming are light and multibeam instru-

ments are usually IO limited rather than computationally bound. In this architecture, because each data link carries the traffic for a full beam, adding additional beams would thus require additional interconnect bandwidth over the single-beam solution. If the processing nodes are not able to unicast these increased data rates then there is no way to add additional beams to the system, short of duplicating the entire instrument.

For systems requiring many beams, it is more computationally efficient to perform beamforming in the frequency domain. In this case, each antenna's processor contains the same integer delay line along with a single polyphase filterbank that is common to all beams. Each beam is then steered by multiplying the PFB's output with a complex rotating phasor (usually derived from a sine/cosine lookup-table), thus introducing phase offsets across the band. In this case every subsequent beam simply needs a lookup table for these steering coefficients and an adder tree. While it has the advantage that incremental compute requirements for additional beams are small, the steering angle between beams in this case is limited by the shared coarse delay and the output of the beamformer is already channelised.

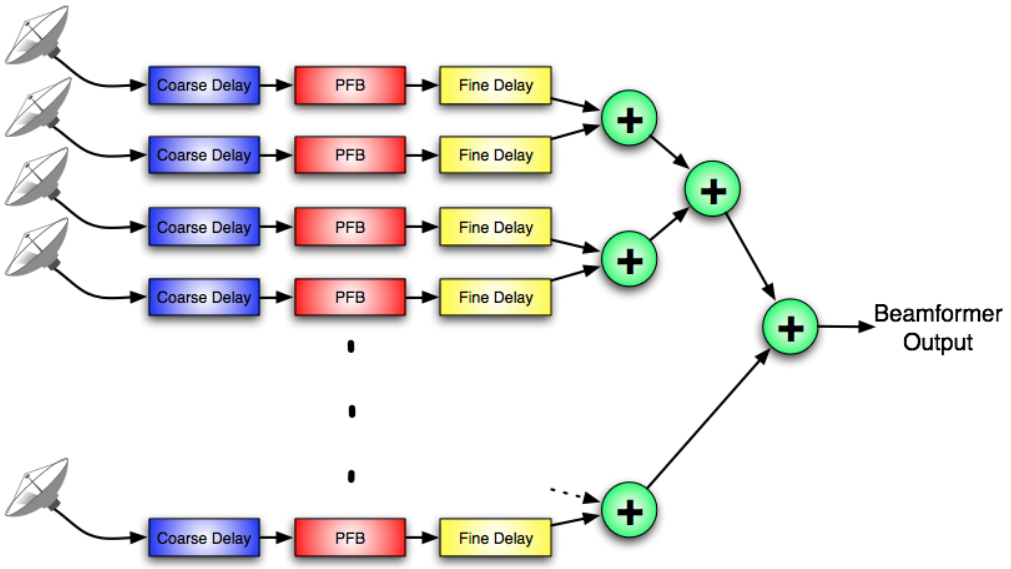


Figure 2.4: A frequency-domain beamformer architecture constructed around an adder tree.

One such implementation is shown in Figure 2.4 which uses an adder tree structure. This is the architecture employed by the ATA, for example. Each frequency channel from adjacent antennas are summed together and the result propagates to the next branch node where it is added to the sum from the neighbouring antennas and so forth until all inputs have been summed.

A problem not yet addressed is how to calculate the delays in the system. They can be calculated a-priori if the antenna and source positions are known but the precision is usually insufficient and closed-loop control is required. This is usually done using a small 1xN correlator (see 2.2.6 for details) to calculate the phase difference between input signals. The correlator requires access to the datastreams from each antenna. The adder-tree architecture allows a two-input correlator to be placed at every adder and thus phase-up the array in a piecewise fashion by first phasing adjacent antennas and then phasing that pair to the adjacent pair etc. In the aforementioned simpler time-domain system from Figure 2.3, each board only has the partially-summed incoming beam for a reference. In this case, a correlator on each board using the local antenna as one input and the summed data as the other input is also acceptable.

Phasing-up the array in this sequential fashion can take time. If a full correlator is available, the beamformer can be configured significantly faster.

The delays, or steering coefficients, must be continually updated to keep the beam on a given source. This is due to the relative motion of the observed object from the observation position (for example, due to the earth's rotation). Observation of geostationary satellites are the only objects that would not require changing coefficients. This ability is generically called delay tracking. The rate at which the steering coefficients need to be updated depends on the observation frequency, the speed of movement (and hence the rate of change of the system delays) and also the frequency channel bandwidth. With a large number of frequency channels, each channel represents a narrow bandwidth. Selecting the frequency channel width is important; it affects not only the tracking speed of the instrument but also has consequences for fast transient work. Very fast pulsar work requires a coarse channelisation or else a time-domain beamformer.

2.2.6 Correlators

The correlator is arguably the most complex instrument at a telescope array facility due to the computing and data transport requirements. Unlike typical signal processing operations where data rates through the signal chain reduce progressively, in the case of correlators they initially increase before decreasing in the final vector accumulator. This "explosion" of data within the system creates an interesting data routing and computation challenge.

A correlator calculates the amplitude and phase differences between the signals from antennas in the array, which can be used to determine the precise location of an object under observation. To do this, the complex signal from every antenna is multiplied by every other antenna's signal. The product of the signal from antenna A and B is called *baseline AB*. Typically, each signal is also multiplied by itself. These baselines are called *auto-correlations* or *self-correlations*, whereas baselines between different antennas are called *cross-correlations*.

XF vs FX

There are primarily two types of correlator implementations: *XF* and *FX*. The final outputs produced by both designs are equivalent.

In the case of an *XF* design, multiplication of the input signals takes place first (in the time domain), the results are accumulated and then a Fourier transform is performed on the accumulated output. A filterbank transform is required for every baseline in this design (ie quadratic scaling) but the filterbanks can run much slower as they process post-accumulation data. However, many more multiply-accumulate operations are required ($O(MN^2)$) than in the FX equivalent. In the case of an FX design, the Fourier transform takes place first, followed by multiplication and accumulation. It thus requires one filterbank for each input (ie linear scaling with number of antennas), running at full line speed and $O(N^2)$ MAC operations. For larger arrays, this architecture has been demonstrated to be more efficient by Bunton (2000); D'Addario (2001).

Earlier designs tended towards the XF architecture whereas most of the correlators currently in development are FX implementations.

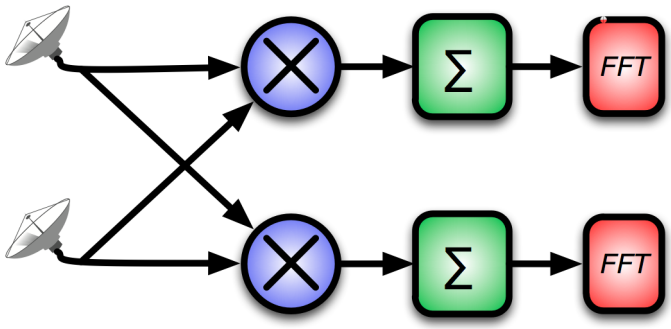


Figure 2.5: A simplified XF correlator architecture showing the order of the primary operations.

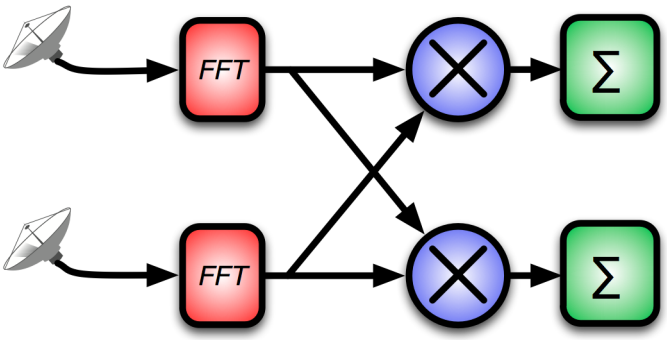


Figure 2.6: A simplified FX correlator architecture showing the order of the primary operations.

For a given implementation with a signal bandwidth of B, M frequency channels and N independent inputs, the computation requirements of an XF correlator scale as $O(BMN^2)$. D'Addario (2001) shows an FX correlator to scale as $O(BlogM + BN^2)$. Bunton (2000) shows that the simplicity of the XF architecture favours smaller arrays that do not require high frequency resolution whereas the FX architectures are better suited to larger designs requiring higher frequency resolution. The FFT window length which can be applied in an XF design is limited by the number of lags. Whereas Urry (2002) shows that the FX design's ability to easily use long-window PFBs allows it to offer better spectral isolation.

This paper will focus on FX designs which are current favourites for large, modern arrays. What follows is an overview of existing designs with particular attention drawn to their processors and interconnect architectures.

I concern myself here primarily with FX correlators.

Operation

If the signal from an antenna arrives at a different time from that of another antenna, then it will result in a time delay difference on that baseline. A time delay difference manifests as a phase ramp over frequency. When plotting a spectrum output, this can be seen as a phase slope. By measuring the angle of this slope, the difference in delay between the two inputs can be determined and thus the direction of the source.

For N antennas in the array, the compute requirements scale with $O(N^2)$, as a baseline for each pair is calculated. Also, since each signal has to be multiplied with every other signal, each baseline pair requires an interconnect path and thus the interconnect requirements between the multipliers also scales $O(N^2)$. It is this quadratic scaling that makes building large correlators so challenging. Scaling those parts of the design that are antenna-specific are easy as they are self-contained and can simply be replicated as many times as necessary. But it is the inter-antenna interconnect and cross-input multiplication that becomes difficult to manage for very large systems.

Correlators are typically used for wideband (continuum) imaging, or else for spectral line work.

In the case of wideband imaging, it is desirable to process as much bandwidth as possible to increase sensitivity of wideband sources. Here we are not so much interested in frequency resolution except to enable RFI mitigation (by ignoring frequency channels with interference), where a finer resolution enables better selectivity of typical man-made CW sources with a lower loss of total band and to minimise fringe smearing. It is not uncommon for FX correlators to support thousands of channels. XF correlators typically allow for tens to hundreds of lags.

In the case of spectral line work, a finer channel resolution is desirable but is only required over a narrow bandwidth. The same channelisation arguments and requirements apply here as for spectrometers, discussed in §2.2.2.

Consider the maximum measurable phase difference: For every sample clock period delay between two inputs, a linear phase ramp of 0 degrees at DC to 180 degrees at $\frac{f_s}{2}$ is introduced. As the delay increases, this ramp's slope increases and will eventually wrap. It can wrap a number of times across the band and will remain resolvable until there is a wrap within two individual frequency bins. This corresponds to a delay of N clock samples where N is the number of frequency bins. Longer FFTs with finer spectral resolution can thus resolve longer delays.

Also consider the accumulation period. If the system does not support delay compensation, the geometric rotation of the earth causes the source to move across the sky relative to the observer and this causes the measured phase between two antennas to change over time. This rotating phase over time is known as a fringe. Since the correlator averages over time, we calculate the average phase of the signal over the accumulation period. In order to be able to measure the sampled phase difference, Nyquist's theorem again applies and the sample rate must be at least double the rate of change of this phase. Thus, the integration time needs to be at least half of fringe rate. If a static image is to be formed by further integrating the output of the correlator (to further improve signal to noise ratio) then this fringe rotation must be stopped by rotating the phase back before long-term accumulation (ie compensating for the changing delays).

If the correlator is able to perform fringe stopping in hardware before

accumulation, it can essentially accumulate indefinitely, provided the rate of change can be set accurately. The primary cause of this changing phase is the Earth's rotation. Since the rate of rotation of the earth is a well-known constant, the rate of change can be determined by bandwidth, observation frequency and the location of the antennas and the signal source. These concepts are revisited in more detail in §2.4. Practically, accumulation periods are rarely longer than a few seconds and are commonly hundreds of milliseconds. This allows for accumulations to be excluded if they are corrupted by RFI, for example, at a finer resolution. There is interest in obtaining faster outputs, to allow for imaging of transient events at higher time resolution, if the increased data rates can be accommodated by the data capture and storage devices.

Most telescopes have dual polarisation feeds; either horizontal and vertical, or left- and right-circular. A full correlator will also calculate the products of the polarisations' signals, which enables the characterisation of the source's polarisation. Each baseline thus produces four products which are usually processed into Stokes parameters.

2.3 Similarities between instruments

The FX correlator, frequency domain beamformer and spectrometer all share per-antenna filterbanks. In facilities requiring all three of these instruments, the same station processor can perform all of these functions simultaneously where channellisation resolution and processed bandwidth requirements permit (determined by science use-cases).

The correlator's auto correlation output is mathematically equivalent to the spectrometer's output. If dynamic range and output data rate requirements can be met, the presence of a correlator provides a complete replacement of the basic spectrometer instrument.

An FX correlator and frequency domain beamformer can share not only their per-antenna filterbanks, but also interconnect. The full co-located correlator could then be used to calibrate the beamformer very quickly. A combined correlator/beamformer instrument is a good candidate to demonstrate efficient commensal observations by leveraging this proposed multicasting

architecture.

2.4 Delay compensation, fringe rotation and Doppler shifting

Delay compensation, fringe rotation and Doppler shifting are discussed in some detail in Chapter 6.1 of Thompson et al. and so they will not be re-explained here. Furthermore, the detailed application of these concepts as pertains to the KAT system, including the effect on sensitivity and residual phase errors can be found in §7 of Peens-Hough (2010). This section will provide a working explanation and summarise the issues as they apply to typical radio astronomy instruments.

Figure 2.7 shows the phase relationship between a common signal and a delayed copy of that same signal along with the effects of downconverting these signals to IF or baseband. A delay between two inputs is revealed as a phase slope across all frequencies. As the delay is increased, this slope becomes steeper. Static delays in the telescope system are typically caused by mismatched analogue cable lengths and changing slopes are typically due to geometric shifts during an observation as the source moves across the sky. Fringe rotation occurs as signals with different delays are down-converted before being sampled. A residual phase offset then remains across the entire band.

Hardware compensation for both delay compensation and fringe rotation can often be avoided if the integration periods are kept short or if the antennas are located close to each other or if the observation frequencies are low. This allows for fast sampling of the UV plane and subsequent rotation of the samples back to initial positions can be done during post processing. PAPER (see §A) is an example of an array employing this scheme. KAT-7 can also do without any form of hardware phase compensation but MeerKAT will need this functionality due to its longer baselines and higher operating frequencies.

Correlation efficiency drops as the signals decorrelate. It is thus desirable to align the incoming signals as closely as possible.

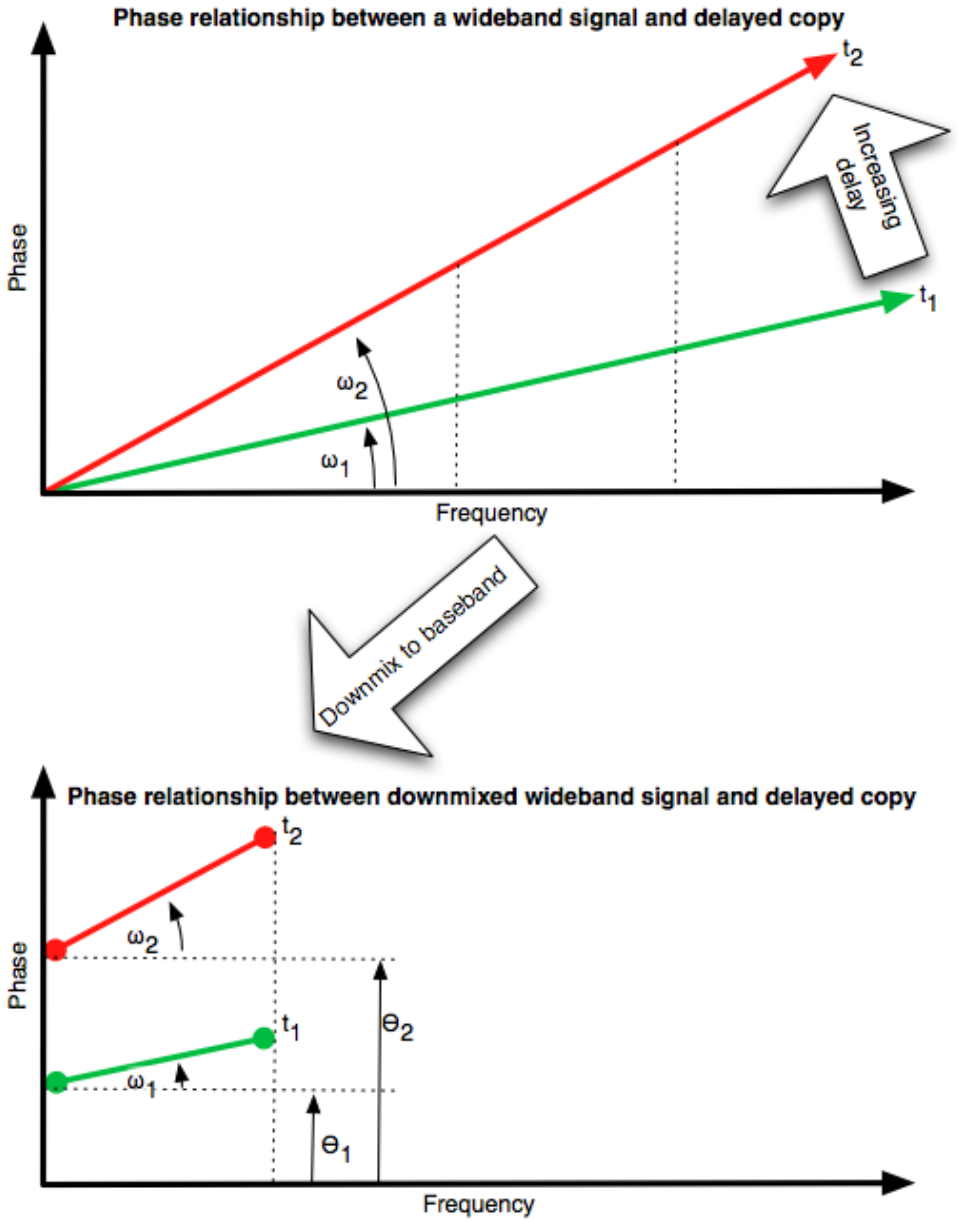


Figure 2.7: Correlating incoming wideband signals with different analogue delays produce a linear phase slope across frequency. Changing delays ($\omega = \omega_2 - \omega_1$) cause a proportional change to the phase slope which must be stabilised before accumulation, a process known as delay compensation. In addition, shifting in frequency introduces a phase error in the correlated signal, $\theta = \theta_2 - \theta_1$, known as fringe rotation, which changes with the delay. This effect must also be removed before accumulation.

2.4.1 Delay compensation

There are inherent static delay errors due to mismatched cable lengths throughout the system. It is too costly to length-match all cables, especially for large systems where baselines can be many tens of kilometers long. Naïve correlation between inputs is simply not possible in such dispersed systems as the signals are completely decorrelated by the cable and geometric delays. An extreme example of this is VLBI, where antennas can be thousands of kilometers apart.

A misalignment (delay difference) of one sample clock period produces a linear phase slope across the sampled band of $\frac{2\pi}{f_s}$ radians per Hz which can easily be removed in post-processing. In an N-point FFT correlator, signals become completely decorrelated when they are N sample clocks out of sync. At this point, phase winding takes place within a single spectral channel and relative phase can no longer be determined. At a minimum then, ignoring any efficiency implications, in an FX correlator, it is necessary to align the incoming signals to within N-clock cycles in order for the correlation operation to proceed.

Historically, these delay corrections were performed using analogue delay lines. Digitally implementing delays of integer multiples of the sampling clock are easily achieved using digital memory as a programmable delay line. Integer delays like this are sufficient for the operation of a basic correlator instrument, but as mentioned in §2.2.5, not for beamformers.

However, there are also dynamic changes in the delay compensation required as the antenna positions move (due, for example, to earth rotation) relative to the source being observed, resulting in changing geometries. As Earth rotates, delays can increase or decrease and can reach the point where the signals decorrelate.

It is thus necessary to adjust the delays while operating. During integration of the correlator output, these phases are averaged. If the rates of change of these phases are fast enough to allow a full wrap during an integration period then results are no longer useful. Considering the same scenario from an imaging perspective, pixels on the sky are smudged across the UV plane during an integration.

Provided the integration periods are kept short and the relative speed of the observed object is slow, this time-average smearing can be kept to a minimum and the slow changing delays can be corrected during post processing. However, if the baselines are long then the delays will change rapidly, necessitating the adjustment of the delay compensation circuits while integrating. These delays are computed and programmed into the delay compensation system periodically. The required update rate of these circuits can be calculated to meet a given time-average smearing specification. Provision for this runtime adjustment needs to be designed-in to the correlator.

The worst-case rate of change of these delays can be determined from the system bandwidth, baseline length and fastest object to be tracked, and hence the rate of updates required to meet a given maximum error specification can be calculated as $\Delta\phi_{g,max} \approx 2\pi f_{IF}\frac{\omega B}{c}\Delta t$, as shown in §7 of Peens-Hough (2010) which analyses these effects. It concludes that KAT-7 with its 200m baselines requires an update rate of 2.4Hz assuming a maximum phase error of 1.5 degrees and a tracking rate of 0.5deg/s with 10ms accumulations. However with longer 15km baselines and an increased bandwidth of 512MHz (similar to MeerKAT's specifications), this rate increases to 73Hz. Modern telescopes continue to push these boundaries, with higher-dynamic range imaging facilities requiring ever more precise measurement of visibility amplitudes and phases. MeerKAT further refines KAT-7 specifications to track phases to less than 1-degree, for example.

With continual adjustment of these delays, it is possible to compensate for the geometric effects and so accumulate for longer periods, reducing output bandwidth and easing post-processing.

Bandwidth smearing (chromatic aberration) errors occur in practical tracking correlators where the delay corrections are applied to frequency channels of finite bandwidth. The corrections are calculated and applied monochromatically, according to each channels' centre frequency. But these channels are not monochromatic and each has a finite bandwidth. Hence, the applied correction factors are incorrect for frequencies away from the centre frequency of each channel, resulting in an error term. This error can be minimised by reducing the channel width.

Bridle and Schwab (1999) provide a useful summary and analysis of both

the bandwidth-smearing and time-smearing effects for radio astronomy purposes.

2.4.2 Fringe rotation

Another effect, known as fringe rotation is due to the frequency conversion in typical heterodyne receivers as demonstrated in Figure 2.7. A phase offset is found across the entire band which changes over time as the delays change. Analogue systems would typically correct for this slowly changing phase by adjusting the local oscillators of the downconverters. This required a different mixing frequency for each antenna.

The rate of change of this phase is determined by the rate of change of the delays as well as the overall system mixing frequency.

It can be corrected digitally in the time domain by mixing with a low frequency (typically in the order of Hz for L-band systems) as is done in the analogue systems. But if the system is equipped with a sub-sample delay compensation function, it is often possible to combine the fringe rotation algorithm into that unit simply by adding a phase offset. This latter approach was adopted in my implementation and is outlined in §4.3.3.

2.5 Walsh switching

Walsh switching reduces crosstalk between inputs by mixing each analogue input signal with a sequence mathematically orthogonal to the sequences used on other inputs, and then reversing this operation once digitised in order to retrieve the original signal. It is typically performed using a binary sequence by inverting the phase in a local oscillator of a mixer in the RF front-end. The digital system can then simply invert the data stream using the same sequence to re-acquire the original signal before subsequently processing as normal.

Existing systems sometimes have separate hardware to generate these sequences and then require of the correlator system to synchronise to these units to reverse the effects. An alternative is to have the antenna-based digitisation hardware generate these signals internally and feed this to the

analogue front ends' mixers. In this case, the co-located hardware to generate the orthogonal sequences and recover the original signals can share resources.

Walsh switching will not be implemented in this work as it has been deemed unnecessary for MeerKAT. But these functions are conceptually simple to add should they be required for other telescopes.

2.6 Effects of quantisation and data representation

The incoming analogue signals are digitised and the effects of this discretisation need to be well understood. FPGAs are highly efficient when using integer based arithmetic, which is exclusively employed in FPGA-based radio astronomy instrumentation signal chains. The signal levels through this chain must be tightly controlled to ensure the best possible results are obtained from the instrument.

This system applies 4-bit quantisation before transmitting the data off-board to save on interconnect traffic. The non-linear transform due to this signal quantisation must be corrected. Van Vleck's study from 1966 (van Vleck and Middleton, 1966) shows how a noisy signal's measured power is affected when it is quantised. Fisher (2002) discusses some of the more practical aspects of this issue as applied to the Green Bank Telescope's spectrometer, which is also of interest to other telescope facilities. The average operating level affects various aspects of the system, including SNR, total power and correlation efficiencies. The chosen operating level can thus be set to optimise for specific metrics, as shown by Backer (2007). My requantisation implementation is discussed in §4.3.5.

Furthermore, the choice of data representation is also important. In the case of signed integer numbers, this is usually two's complement which does not allow for a symmetric range of values around zero. The effect of signal clipping is then significant: With the 16 levels provided by 4 bit quantisation, we have a valid representation range of -8 to +7. If the signal clips, data accumulates in the -8 bin and the +7 bin, resulting in an average negative DC offset on that channel.

To avoid this, some systems place the DC potential midway between the value of -1 and zero, thus averaging out the error term. However, in this case it is impossible to represent no signal (zero). §4.3.5 explains how I artificially limit the signal to -7 or +7 to cope with this problem while maintaining the expected representation levels (that is, zero represents no signal or DC). This wastes one of the 4 bit codes and reduces the number of useful levels to 15. This disused code could optionally be used for in-band signalling.

2.7 Existing implementations

This section will discuss the existing systems currently in use around the world with specific focus on correlators as the largest and most complex of array instruments. What follows is a review of some of the larger facilities and their choice of instrumentation.

2.7.1 VLA

Perhaps the most prominent interferometer in the world, the Very Large Array (VLA) in New Mexico consists of 27 dishes which are located on rail tracks in a Y layout. The antenna spacing and configuration of the array can thus be changed to suit different observations. The VLA is currently undergoing an upgrade, to be called the Expanded Very Large Array (EVLA) which includes a significant overhaul to the feed systems and a complete replacement of the digital electronics. Provision has also been made for the future incorporation of additional antennas.

The VLA employed the first digital radio astronomy correlator. The original system used discrete 74-series logic ICs with direct chip-to-chip wiring to form an XF architecture correlator. Brisken (2006) shows that the 1.5 bit (2 bits, 3 levels) quantisation resulted in low efficiencies ($\approx 81\%$). Bastian and Bridle (1995) outlines that in its final form the VLA correlator could process up to 50MHz of bandwidth (albeit with only eight lags in this configuration). This correlator was in active service for 30 years until retirement in 2010 when the VLA was upgraded to the EVLA using a WIDAR correlator, described in §2.7.6.

2.7.2 VLBI

Very Long Baseline Interferometery (VLBI) typically involves the processing of data from multiple, discrete telescope facilities, often across continents. Large distances between detectors allow for high spatial resolution imaging. This correlation is often performed offline, post-capture. Each facility records accurately timestamped data that is later processed at a central processing facility. This is mostly due to the difficulties and costs of linking remote sites over vast VLBI distances (typically hundreds of kilometers). Historically, data was recorded onto tape and played back at the central processor. Later, hard disk packs were used. Notable exceptions are realtime systems such as the UK e-MERLIN and the European e-VLBI efforts which now have high speed fibres linking component antennas to the central processing facility. This realtime linking is becoming increasingly popular in other facilities.

The Very Long Baseline Array (VLBA)'s website[1] claims the array of ten dishes located across the United States constitutes the world's largest dedicated, full-time astronomical instrument. Data was originally processed offline from tapes through an FX architecture correlator employing custom ASIC chips. More recently, this system has been replaced with a disk-based software correlator. The fact that most such VLBI systems process offline data or otherwise bandwidth-limited signals suitable for small software correlators means that they have limited applicability to this work which targets realtime processing of wideband signals.

2.7.3 GMRT

The Giant Metrewave Radio Telescope in India also employs an FX correlator. The original GMRT hardware correlator used the same ASIC multiply-accumulate chips as the hardware-based VLBA system but with custom F-engines. It processed 30 dual-pol inputs at up to 32MHz bandwidth. A new software-based correlator, described by Roy *et al.* (2010), has completely replaced this system.

Studies are currently underway at GMRT to develop a digital system able to process the upgraded 400MHz band provided by new wideband feeds.

[1] http://www.vlba.nrao.edu/whatis/

Options include the FPGA-based design described in this thesis, as well as a modified version using GPUs for the multiply-accumulate back-end.

2.7.4 CARMA

The Combined Array for Research in Millimeter-wave Astronomy hosts 15 antennas operating at up to 270GHz and an additional eight smaller receivers operating up to 115GHz. These two systems currently operate independently. A new XF design correlator using custom FPGA boards is being installed. It is capable of processing up to 4GHz bandwidth from the 15 larger antennas. The sampling clock is not distributed directly, but rather a reference frequency is provided to each sampler, each of which then employs a local PLL to generate the sampling clock. CARMA is investigating the use of high-speed samplers to increase the processed bandwidth.

2.7.5 Westerbork

Westerbork is currently undergoing a refit using focal plane arrays. They are developing a unified hardware platform called the Uniboard, similar in concept to the ROACH boards discussed in this work, to process data from the upgraded array. Uniboards have interfaces that could be used for networked interconnects but they are primarily designed to intercommunicate directly over a backplane. Each board contains multiple FPGAs and off-chip memory. Groups of boards are controlled by a local computer. At the time of writing, an effort was underway to create a unified software environment similar to CASPER's DSP library.

2.7.6 WIDAR

WIDAR is the upgraded back-end system used in the EVLA. It is the largest radio astronomy signal processor system currently deployed. Carlson (2000b) describes the WIDAR correlator as a 27 rack machine costing 11 million USD (excluding ADCs) and consuming 100kW from a 135kVA source.

It is a hybrid FXF design, processing 16GHz total bandwidth by down-converting (using analogue electronics) into 2GHz analogue bands which are

then further digitally subdivided into 16x 125MHz bands. Each 125MHz band is then processed independently. The EVLA is initially specified for 32 dual-polarisation antennas. Carlson (2007) shows WIDAR employing 3 bit sampling for the wide-band mode, where correlator efficiency is about 93%.

The processors use ASICs for the MAC operation and FPGAs for channelisation and interconnect. These FPGAs are not reprogrammed at runtime but weighting coefficients and routing tables are updated at runtime to change system parameters. It is essentially an XF architecture duplicated 64 times, each processing a portion of the band and can thus be considered a hybrid FXF correlator. A custom interconnect is employed and it allows almost continuous and seamless tradeoff between processed bandwidth and channel resolution. WIDAR supports full band processing or multiple narrowband spectral lines.

2.7.7 ATA

The Allen Telescope Array was to use an FX correlator based on various custom-designed FPGA boards. Inter-board communication would take place over a backplane with data from the F-engines entering the X-engines over the point-to-point 10Gbps XAUI protocol. CASPER iBOBs later replaced the intended F-boards and are still in service in the 42-antenna correlator.

The ATA beamformer is a stand-alone instrument constructed around CASPER BEE2s with an integral $1xN$ correlator for beamformer calibration, constructed using CASPER iBOBs and BEE2s hardware devices.

2.7.8 LOFAR

LOFAR, the LOw Frequency ARray in the Netherlands, uses an FX-type correlator based around a 4-rack IBM Blue Gene/P supercomputer (initially 6-rack Blue Gene/L) (Latour and v.d. Schaaf, 2007). It is a highly optimised hybrid realtime software correlator (Romein *et al.*, 2010). The system is reconfigurable and able to perform, amongst others, the standard functions of aperture synthesis correlation, tied-array beamforming and pulsar timing and searching modes.

LOFAR consists of 90 stations with nominally 32MHz bandwidth processed at approximately 600Hz channel resolution (Gunst, 2007). Each station consists of arrays of tiled antennas. These are first phased-up at individual stations before being fed to the central correlator. Some initial pre-processing takes place at the stations on FPGAs including initial coarse channellisation (v.d. Schaaf, 2005). Thus, an entire station appears to the central processor as a single antenna. By trading bandwidth, it is possible to form multiple beams at each station (with core stations supporting up to 24). Each beam is processed independently and not cross-correlated with other beams from the same station. The central processor then performs fine channelisation on each beam.

The LOFAR implementation required significant optimisation to obtain good performance figures from Blue Gene. Latour and v.d. Schaaf (2007) show that further investment in the form of input/output nodes were needed to get data buffered, formatted and fed into Blue Gene efficiently. These nodes consist of commercial PC-type servers which envelop the Blue Gene. Multiple networks for data interchange were also implemented, including a 3D torus inside the Blue Gene, Infiniband within the IO nodes and Ethernet from remote stations. LOFAR is thus not a homogeneous compute platform nor does it have a homogeneous interconnect.

However, LOFAR represents the closest existing architecture to the system proposed in this work: it is a well partitioned design using hybrid hardware, based around a general-purpose, reconfigurable processor and largely employs a general-purpose interconnect. Key differences include:

Scale Although processing 90 elements, only 32 MHz beams are processed. The system described here aims to target much larger systems on the scale of WIDAR and larger.

Interconnect The interconnect is not homogeneous. A mix of 3D torus data network on the Blue Gene, Infiniband into the IO nodes and Ethernet control and monitoring at the stations makes interconnect somewhat more complicated than it could be. As deployed, replacing nodes in the system with different technologies would be difficult without affecting the rest of the system. I aim to use an ubiquitous, homogeneous inter-

connect if possible.

Optimisation and portability The software design has been largely optimised for the Blue Gene. While this greatly improves efficiencies, it makes it non-portable. I aim to produce portable code, even if this results in diminished efficiency.

Custom hardware The IO nodes are commercial PC-type servers but the station processors use non-scalable custom hardware. And while the Blue Gene is off-the-shelf, it is expensive and proprietary. I aim for a low-cost, scalable solution that is easy to program and where all technical details of the machine can be shared between observatories. Open-sourced licensing for the entire system is proposed.

2.8 Conclusion

In this chapter, I reviewed common radio instrumentation and the typical signal processing operations in each. I find that while the instruments are different, there is a significant overlap in basic pipeline operations, and many components, such as filterbanks, power meters, accumulators, multipliers and adders are common to multiple instruments. I believe that this commonality lends itself well to using libraries of standard components to design the instruments.

Also discussed were some of the more common requirements in array processing for delay and fringe rate compensation. This, together with the considerations for data quantisation and representation, will feed into the implementation of a correlator and beamformer instrument that shares channelisers in §4.

I also considered a number of existing implementations at various telescopes around the world and found that while functions and basic operations are similar, each machine is uniquely implemented. I hope to design a flexible system that can alleviate some of the "wheel reinventing" taking place at each facility. Finally, I note that none of the machines were built using FPGAs with a packetised interconnect.

Chapter 3

Component selection

This chapter concerns itself with the selection of processing hardware and interconnects that are commercially available for the purposes of implementing the general-purpose machine that is the focus of this research. I conclude that FPGAs with an Ethernet interconnect make for a good construction platform and continue to evaluate and select an appropriate FPGA-based processing platform.

This chapter begins by outlining the requirements for this hardware, in the context of my DSP requirements.

3.1 Digital signal processing

Radio Astronomy instruments make extensive use of Fast-Fourier Transforms (FFTs), Polyphase Filter Banks (PFBs), Digital Down-converters (DDCs) and other signal processing operations. These algorithms are well-known and many textbooks and papers are available on the subject (for example, Crochiere and Rabiner (1983) and Vaidyanathan (1990)). What differentiates radio astronomy signal processing from many other disciplines is the volume of data and the processing speeds required. Operations are required to take place in realtime, continuously, on streaming data. Many of these operations must be performed at very high speeds, processing large bandwidths on specialist hardware. To save on expensive computing resources, efficient implementation is very important.

This thesis will not focus on the implementation of these well-known algorithms. Insofar as I need to select a suitable algorithm for my purposes, Chapter 4 will cover the motivation for the selected algorithm and discuss its effects. Wherever possible, existing libraries and code will be reused (much of which is already provided in the CASPER infrastructure described later in this chapter). Analysis of the critical signal processing components is also discussed in Chapter 4.

3.2 Suitable processing platforms

This section will give a high-level overview and summarise the current state of processing platforms for the purpose of selecting one for radio astronomy instrumentation use. No significant time will be spent evaluating these options ourselves but this section will report on the findings of other studies. It is included to motivate my decision to use FPGA platforms for the correlator construction. While this argument is valid at the time of writing, it should be noted that the computing market has become increasingly volatile, especially over the past decade, with many competing technologies emerging. It is expected that this section of the discussion will date quickly.

3.2.1 Requirements and ASICs

A core requirement of this work is that the design be flexible and agile. Custom designed ASICs are popular choices for radio astronomy signal processing engines but are not reprogrammable, are inflexible and are expensive to design both in terms of time and money. ASICs become financially viable in large quantities and so it is conceivable that ASICs may continue to play a significant role in radio astronomy instrumentation for large, next-generation systems. So while provision should be made for interfacing to these devices, my reconfigurability and agility requirements preclude their use as a development platform. Popular choices for reprogrammable devices in radio astronomy instrumentation include FPGAs, GPUS and CPUs or DSP processors.

3.2.2 Microprocessors and DSP processors

Microprocessor based systems are highly flexible, very competitively priced (due to their commodity nature and volume production) and are largely interchangeable thanks to industry-standard hardware interfaces and software support. They are easy to maintain and upgrade due to the lack of custom hardware. Code re-use is also simplified if common programming languages are used. But they generally are unable to process the large volumes of data produced by next generation instruments in realtime on modestly sized systems. Whilst smaller designs can make use of software-based instruments using standard microprocessor-based computers (such as the VLBA (Deller *et al.*, 2007) and GMRT (Joshi *et al.*, 2003)), most new installations have more challenging requirements of their realtime systems. Current technologies' microprocessor abilities would require inordinately large systems to process data from such telescopes. Perhaps the largest software-based system currently deployed is LOFAR, outlined in §2.7.8.

3.2.3 Accelerators and co-processors

PC-based hardware accelerator cards (such as GPUs, Cell or FPGA-based devices) are attractive but such systems typically have short lifespans and it is often not possible to reprogram such accelerators just a few years after their end-of-life due to the ageing and non-support of the programming tools. Radio astronomy instruments are often used for decades (for example, the 30 year old VLA digital correlator). While this thesis does not advocate an investment in hardware (indeed, the hardware should be considered disposable and part of the telescope facility operating expense; see §5.2) but rather an investment in reusable intellectual property that can be reused on future hardware platforms (ie software or gateware), it is unrealistic to expect that existing hardware would not be repurposed. Long-term support of hardware is thus advantageous.

It is most important that existing IP can be reused effectively. This requires that future hardware is able to execute existing code with minimal changes. The past decade has seen many different accelerators, with large variations in cost and processing power, including commercial offerings

like IBM's Cell blade, the FPGA-based systems like the Nallatech H101-PCIXM and solutions from EDT and academic solutions like CDAC's PCI option,Abhyankar *et al.* (2004) many of which have had no backwards-compatible successors or any successors at all. A telescope facility that had chosen one of these defunct technologies and invested into appropriate software for them would now be stranded with no other option than to start from scratch again at the next upgrade.

Perhaps the most promising of the accelerator technologies is the repurposing of the Graphics Processing Unit, or GPU, typically used in the computer gaming industry, for the purposes of scientific computing. They offer very dense compute power compared to typical CPUs. At the time of writing, it remains to be seen if this trend will become a mainstream one with longterm support. The ease of programmability and the portability of the source code for such accelerators is also still not clear with many competing languages existing, each with limitations and inter-language incompatibilities. The roadmap for continued growth is also uncertain, as they already seem to be encountering thermal limits for air-cooled packages[1], limiting chip growth to architectural and semiconductor node size improvements. Their high power consumption is an additional concern for large-scale use.

The recent uptake of these devices into the Top500 list of supercomputers is encouraging. GPUs perform best when the compute-to-IO ratio is high. For bandwidth-intensive applications, such as beamforming, GPUs would appear to be inefficient whereas they could work well for filterbanks, gridding and matrix operations.

3.2.4 FPGAs

FPGAs have the advantage of allowing the programmer to allocate, at his discretion, resources to different processing operations, thereby making optimal use of available resources. It is thus usually possible to make full use of the device's resources for most parallisable tasks (using over 90% of the FPGA's

[1]Per-package TDP has increased little over AMD and nVidia's 2010 capabilities (about 250W per package). Source: `http://en.wikipedia.org/wiki/Comparison_of_AMD_graphics_processing_units` and `http://en.wikipedia.org/wiki/Comparison_of_Nvidia_graphics_processing_units`

logic and DSP slices and BRAM is not uncommon). However, this flexibility comes at the cost of increased programming complexity as the programmer must then concern himself with logic-level operations.

Modern programming tools such as the various C-to-gates compilers are trying to alleviate this programming problem. However, the compiler-specific pragmas required to generate useful code are incompatible between tools and many such tools are short-lived, untried and with an unknown future. Fortunately, for the purposes of radio astronomy instrumentation, this FPGA programmability problem can be eased significantly and simply by using a small set of pre-implemented but highly configurable signal processing libraries using existing, well-supported tools.

The use of such libraries is made possible by the fact that many radio astronomy instruments re-use similar, common algorithms and regularly perform similar operations. Further details of typical instrument construction can be found in Chapter 2.2. A well-placed, once-off investment in designing such libraries, while initially expensive, can reap long-term benefits. This commonality and modularity eases FPGA implementations of radio astronomy instruments. CASPER has made significant headway toward providing an open source version of such a library for FPGAs.

The associated CASPER FPGA-based compute boards such as UCB's IBOBs, BEE2s, ROACH and ROACH-II platforms are also custom boards but are constructed with the aim of making them commodity, off-the-shelf items. They closely track Moore's law. The latter three boards are able to run the Linux-based BORPH operating system and allow for remote, runtime reprogrammability of their FPGAs (Chang *et al.*, 2005; So and Brodersen, 2006; So *et al.*, 2006) in a familiar software environment.

This does not completely solve the problem, however. Because without further optimisation of these libraries for individual FPGA architectures, it is often not possible to clock the chip at advertised maximum FPGA fabric speeds due to net latencies. This reduces the throughput from the theoretical maximum. To achieve high clock rates, restructuring of the code for a specific architecture is often required. However, even without any optimisation, it is usually possible to achieve useful speeds using well-designed generic libraries. I am happy to trade-off a reduced clock rate for a reduced time-to-instrument,

thereby allowing the use of later, more powerful compute platforms, which can offset the optimisation losses.

A further advantage that FPGAs offer over both microprocessor and accelerator-based solutions is a significantly reduced power consumption (see Table 3.1).

3.2.5 Performance comparison

Direct comparison of the various compute processor devices is difficult.

The efficiency of GPUs is highly dependant on massive parallelisation of the problem and careful management of memory. Not all use-cases map well to this architecture and it is rare to achieve the claimed performance figures for GPUs and CPUs in real-world applications without significant optimisation.

The performance is poor without code optimisation; Woods (2010) reports that he was only able to achieve a little over 20Gops/s with a Geforce 9800GT which has a theoretical throughput of 432Gops/s (ie less than 5% of claimed performance). To achieve better results, the code must be tailored to effectively utilise cache and onboard memory and care must be taken to avoid making unnecessary memory copies. This usually requires intimate knowledge of the memory and interconnect architecture. Requiring such optimisation undermines the very benefit of using these devices (namely that they are easy to program and use).

After this work was started, an efficient GPU correlator implementation has been demonstrated using the Geforce GTX 480 for a 512 input system. Clark *et al.* (2011) achieves very close to theoretical maximum throughput for the device (79%). GPUs perform particularly well for larger arrays where the $O(N^2)$ compute problem begins to dominate the $O(N)$ IO, as the GPU is no longer throttled by bandwidth limitations. However, this implementation also demonstrates that the system does not scale well to smaller arrays or very large arrays that are unable to fit into memory. In §C, I report that GPUs are being considered alongside FPGAs for some parts of MeerKAT's signal processing pipeline.

Generating a realistic *operations per second* figure for FPGAs is difficult

Table 3.1: **A comparison of real-world processor FPGA capabilities when configured for a CASPER packetised correlators (Virtex 5), compared to a pure-CPU implementation at the GMRT (Intel Xeon 2.3GHz) and manufacturer-claimed GPU capabilities and their power requirements, all as at 2009.**

Processor	Throughput Gops/s	Power Watts	Efficiency Gop/joule
CPU [a]	45	173	0.26
GPU [b]	432	338	1.3
FPGA [c]	250	70	3.6

[a]16-node Xeon cluster GMRT software processor at 260Mflops/watt as reported by Roy *et al.* (2010), averaged FFT and MAC use-case and excluding IO overheads.

[b]Nvidia Geforce 9800, power out the wall including host system with typical motherboard, HDD and Intel Core 2 Quad CPU.

[c]ROACH board with Xilinx Virtex 5 SX95T, power out the wall, including all ancillary devices and power supply inefficiencies

and implementation dependant. FPGAs have hardware multiplier slices, the number of which could be multiplied by the clock frequency to generate the maximum number of operations per second for the DSP slices. But it is also possible to construct multipliers from memory and also from logic slices. In this case, the bit-width of the operation makes a significant difference to the number of operations per second that one can perform on a single device. One could, for example, construct an 8 bit adder with given hardware resources but by reconfiguring those same resources you could build two 4 bit adders, thereby doubling one's OP/s figure-of-merit. It is thus practically easy to obtain better performance than naïvely anticipated, but equally easy to generate unrealistic peak performance benchmarks for these devices.

To illustrate this point, if I wanted optimistic values for CASPER's FPGA-based ROACH board for example, I could claim that it is possible to perform 7360 4-bit (a typical resolution for radio astronomy applications) additions or multiplications simultaneously from the 14720 6-bit slices on the V5 SX95T, resulting in a figure of 2.2TOP/s per ROACH board if clocked at 300MHz. This figure ignores the native DSP slices, the ability to construct

adders from lookup tables using onboard RAM and the fact that the chip is able to clock at well over 300MHz. These factors could easily push the figure of merit to over 4TOp/s. Compared to current GPU technologies (1TOp/s[2]) this would look very attractive. But even the 2.2Top/s figure is an unrealistic one for practical implementations as large portions of the device are consumed by retiming and routing requirements.

For this reason, the theoretical operations/s figure is a poor indicator for comparing FPGA and microprocessor platforms. A reliable metric would be to compare actual implementations of the same system on different platforms and the effort required to optimise the code for the given platform. However, the precise mapping of an application's operations onto a given FPGA architecture is usually not known a-priori and is estimated at design time and only concreted at compile time. It is thus difficult to compare solutions reliably without actually implementing the design on each platform (or a similar design from which results can be extrapolated). Even this would not be definitive in the general case, because a given system can remain fully functional but the choice of system parameters can favour one implementation over another (for example, 4 bit integer operations would favour an FPGA implementation whereas 32 bit floating point would favour GPUs, though both might be functionally sufficient for a given instrument). Thus, any such comparisons would need to be accompanied by a study of how the implementation favours one architecture over another and factor-in the invested effort in optimising for the given architectures.

With these limitations in mind, consider Table 3.1 which compares the performance and power consumption for typical processing platforms using 2009 technologies. The figures for the FPGA and CPU were taken from a real-world correlator deployment (the PAPER system detailed in Chapter A and the GMRT software backend, as detailed by Roy *et al.* (2010)), whereas the performance figures for the GPUs are theoretical claims. At the time of compilation and comparison, an efficient GPU implementation was not available.

FPGAs are significantly more power efficient than CPUs and GPUs. Even 3 year old FPGA technology is able to out-perform the latest generation

[2]ATI 3870 X2, fastest reported performing card as at 2009 by tomshardware.

Table 3.2: **A comparison of processor costs when implemented on different technologies. FPGAs are configured as CASPER packetised correlators, CPU cost is based on GMRT software backend and GPU capabilities are manufacturer-claimed. Pricing and capabilities are as at 2009.**

Processor	Throughput Gops/s	Cost USD	Cost per Gop/s USD / Gop/s
Excluding host system:			
CPU [a]	22	170	\$0.77
GPU [b]	432	140	\$0.03
FPGA [c]	250	2000	\$10
Including host system:			
CPU [d]	79	1750	\$22
GPU [e]	432	2000	\$4.6
FPGA [f]	250	4500	\$18

[a]Intel Core 2 Quad CPU, excluding host system. Prices from online retailer Newegg.
[b]Nvidia Geforce 9800, excluding host system. Pricing from online retailer Newegg.
[c]Xilinx Virtex5 SX95T. Cost from ROACH assembler, Digicom, CA.
[d]32-node GMRT Xeon cluster at 45MFLOPS/USD as reported by Roy *et al.* (2010).
[e]Theoretical throughput for Nvidia Geforce 9800, price including typical host system from online retailer Newegg.
[f]Xilinx Virtex5 SX95T. Cost from ROACH assembler, Digicom, CA.

microprocessor devices [3]. It should also be noted that multiple GPUs can be placed in a single host system, thereby improving the power efficiency figures quoted in Table 3.1. In such situations performance implications need to be carefully evaluated as many host systems would share memory and CPU resources under these conditions, creating potential bottlenecks. And in the same fashion, multiple FPGAs or CPUs can be placed in a single host system.

Considering the cost of processing, the picture looks considerably different depending on whether or not the host platform is considered as part of the processing cost. Table 3.2 illustrates this difference. A CPU-based system

[3]BEE2 board with five Xilinx Virtex 2 Pro 70 devices is able to do 500 GOP/s at 250W, resulting in a figure of merit of 2Gops/joule.

appears very cost effective when only the chip itself is considered. However, once in host system, the overall price rises significantly. In any scenario, as Table 3.2 shows, in terms of cost, as at 2009, GPUs clearly provide the fastest processing platform for a given budget.

3.2.6 Processing platform selection conclusion

The choice of hardware platform ultimately depends on the required performance and system flexibility as well as the system power, cooling and space budgets. And an intensive tradeoff study is beyond the scope of this work. So in order to remain competitive, this work should not dictate which platform to use but allow the designer to choose the most suitable processor for a given application and possibly allow for a mix of hardware types. A core requirement is then that the interconnect allow for such heterogeneous systems.

However, from this review, it seems the best balance between costs, longevity, programmability, flexibility and power, space and cooling requirements for my reference implementation is currently an FPGA-based platform. For this reason, the demonstration systems in this work will be constructed using FPGAs. Chapter 3.4 discusses the choice of FPGA board.

3.3 Suitable interconnect systems

Since most radio astronomy instruments require more processing power than a single processing node can supply, a suitable interconnect must be found to allow multiple nodes to share the processing load. This section will discuss some of the requirements and then consider existing options.

3.3.1 Requirements

In light of the conclusion in Section 3.2.6 (that multiple processor types may be required in the system), the interconnect of the processing nodes needs to be device independent and able to work with FPGAs, microprocessors, ASICs or any other device that may come along at a later date.

Ideally a single interconnect technology should be able to connect all the processing nodes across various technologies. This will provide a uniform hardware and software interface to processing nodes.

It needs to be simple (since it is costly to implement complicated communications stacks on hardware such as FPGAs and ASICs) while remaining flexible to accommodate future processors.

The interconnect should be highly scalable. This should allow for scalable systems to be constructed using a common architecture.

It should use the latest technologies, allowing high throughputs over links. Low performance systems would require additional cables and connections, thereby reducing system reliability and increasing physical size.

Long life is important, as is forwards and backwards compatibility. One of the major drawbacks of traditional systems is the short-lived interconnect (usually in the form of a backplane) that prevents interfacing to future technologies.

In order to support commensal operation efficiently by sharing of intermediate data products, the interconnect needs to be able to multicast data.

I prefer packet-switched networks. These are more flexible and allow for simple multicasting. Even the traditionally circuit-switched telecoms networks are moving to fully packetised systems (eg 4G GSM spec requires fully packetised network infrastructure). The desire for a packetised network, and the general trend of industry towards this technology eliminated SONET and other conceptually circuit-switched systems from consideration in this work.

As with the processing nodes, a commercially available commodity system would lower costs and reduce implementation efforts significantly. For this reason, the following popular high speed interconnects are considered:

3.3.2 Infiniband

Infiniband originated in 1999 from an industry consortium. It is generally faster to adopt newer technologies than Ethernet, enabling faster line speeds earlier. As of 2011, Infiniband is already shipping 120Gbps links. Perhaps most significantly, Infiniband has a much lower cost when compared to competing (Ethernet) solutions. However, Infiniband is not an open standard

and requires buy-in for implementors. This results in few manufacturers of Infiniband equipment.

Physical connections use CX4 plugs on SDR and DDR 4x interfaces (for up to 16Gbps wire speeds). These copper cables are inflexible and short-run only. Fibre is used for longer-haul links.

Later QDR implementations use Quad SFP and Quad-SFP+ (as used by FibreChannel, SAS and 40GbE) connectors. This raised a concern of incompatibility with existing CX4 equipment. Adaptor cables are available to physically interconnect QSFP+ and CX4 systems but DDR and QDR Infiniband systems cannot simply be connected together in this way.

Infiniband has seen large-scale adoption in datacenters, as can be seen from the uptake on the Top500[4] list, with five of the top 10 systems using Infiniband as at August 2011[5]. Infiniband offers significantly lower latencies than Ethernet so it is especially effective for processing tasks requiring inter-process communication where latency between nodes is a concern. For radio astronomy applications which are trivially parallelisable, independent parallel nodes do not need to communicate and so latency is not a significant concern.

A significant drawback is the lack of commercially available Infiniband PHYs. Native line speed MGT support of 10 Gbps for QDR, 14.0625 Gbps for FDR and 25.78125 Gbps for EDR per lane is required of the end-nodes. This effectively limits processing nodes to high-end FPGAs and compute nodes and eliminates low-power DSP and embedded processors as options at the time of writing.

3.3.3 Ethernet

Ethernet is a long-living packetised network standard with widespread use. It is layer-1 backwards compatible since the IEEE 802.3 10Mbps twisted-pair 10BASE-T standard of 1990 and nodes can still be interfaced to non-twisted pair 802.3 systems from 1983 with media converters. 100Mbps, 1Gbps and 10Gbps implementations are now popular. 40Gbps and 100Gbps implementations were standardised in June 2010 showing that Ethernet continues to

[4]http://www.top500.org/

[5]http://www.top500.org/connfam/8

scale.

A very significant advantage of Ethernet is it's ubiquity. Due to the long-running manufacturing mass-market nature of cables, switches and other interconnect hardware, consumer-grade Ethernet equipment is available cheaply. Open-source implementations (both hardware and software) are also readily available for most processing platforms.

For the purposes of this work, I am mostly interested in the latest, fastest generation of Ethernet which is commercially available at reasonable prices. At the time of writing, this was 10GbE over copper with CX4 connectors. However, the industry is already moving towards SFP+ which is a more reliable form-factor based on the existing SFP standard for 1GbE. This connector is smaller and more robust. It has the added advantage of supporting both copper and optical connections using the same interface (through plug-in modules). The ever-popular and ubiquitous copper BASE-T "RJ-45" 10Gbps standard was ratified by the IEEE in 2006 and is expected to popularise the usage of 10GbE once appropriate low-power PHYs become available.[6]

At the time of writing (2012), 40GbE and 100GbE direct-attach copper and optical links are available but at very high prices and without much industry support. No specifications currently exist to enable 40GbE or 100GbE over the BASE-T connectors, though potential for 40GbE transmission over Cat-7a copper cable has already been demonstrated by Enteshari (2009).

Ethernet commonly provides both a best-effort and a guaranteed delivery service in the form of UDP and TCP, respectively, on top of an IP stack.

Historical Ethernet

As of 2012, the commodity Ethernet standard is 1000BASE-T and is the current building wiring standard for small offices and home use, using Category 5e, 6 or 7 shielded or unshielded twisted pairs (four per link) and 8P8C ("RJ45") connectors. This low-cost copper cabling is ubiquitous and can be terminated on site with low-cost crimping equipment. 1000BASE-T can drive

[6]Aquantia, for example, have recently announced 2W per port parts for short links which should speed-up adoption. Intel has also recently announced a LAN-on-motherboard (LOM) solution, which is required for mass-market adoption.

(a) MPO connector (b) An LC fibre connector

Figure 3.1: A multifibre push-on (MPO) connector, as used for 40GbE MMF connections, contains a single row of 12 fibres, 8 of which are used per 40Gbps link (four in each direction). Looking forward, this connector will not provide a sufficient number of fibres to establish a 100GBASE-SR10 link, and two rows of fibres are needed. An LC fibre connector is already common in SMF links, which uses two uni-directional fibres to establish full duplex communication. The same LC connector is used for 100/1000/10G/40GBASE-LR standards, and will likely be used for the future 100GBASE-LR4 standard, providing simple forwards compatibility.

network segment links of up to 100m. Most terminal equipment (computers) today have support for 1000BASE-T using onboard Network Interface Cards (NICs). This technology has a very low total solution cost.

Because of this low cost, this solution is popular in smaller datacenters that need runs of less than 100m and low bandwidths. One popular arrangement is to place a large network switch at the end of a row of racks and directly run cables to each server in the row. End-of-row switches are then interconnected to a central switch. Another popular alternative is to use small top-of-rack switches, with backhaul links to a large, centrally-located switch. This latter tree'd model works well when full-crossbar switching is not needed; 4:1 over-subscriptions of these backhauls are common (40x 1GbE ports to one 10GbE uplink, though some top-of-rack 1GbE switches support four 10GbE "uplink" ports).

To drive longer ranges, an aftermarket NIC or a switch that accepts SFP or GBIC modules is required. This provides the communication node with a port that can be populated with various optical transceiver modules that can drive up to 300m on OM3 Multi-Mode Fibre (MMF) or up to 80km on Single Mode Fibre (SMF) without any repeaters. The cost of these transceivers is significantly higher than the low cost 1000BASE-T ports.

Faster link technologies

Modern data centres are now deploying 10GbE en-masse, with 40GbE essentially reserved for backhauls. MeerKAT at L-band requires approximately 40Gbps from each antenna and so the 40Gbps Ethernet standards are well matched here.

10GBASE-T interfaces are soon to be standard on server-class computers and they are already offered as an option on some Dell and HP servers, for example. While already offering the lowest overall cost of all the 10GbE links, the commodity adoption of this standard will significantly further reduce the cost of this equipment and make 10GBASE-T links more attractive than 10GBASE-CR SFP+ direct-attach copper 'twinax' links. This is because 10GBASE-T cabling is backwards compatible with 10BASE-T, 100BASE-T and 1000BASE-T and also offers increased drive distances over a -CR

solution. However, SFP+ connectors offer the most flexibility as they allow for copper or fibre links (albeit at increased cost), whereas BASE-T will always be limited to 100m.

While 1000BASE-T SFP modules are available, 10GBASE-T modules are not technically feasible due to increased 10GBASE-T PHY power requirements, which exceed the SFP+ specification. A mixed-deployment of SFP+ and BASE-T is thus unlikely as nodes would not be able to communicate without adaptors. 10GbE SFP+ copper (-CR or twinax) interfaces consume less power than the 10GBASE-T ports, as they do not contain the power-hungry BASE-T PHYs.

Scalability of the twisted-pair BASE-T standard beyond 10Gbps speeds is also in question. While 40Gbps speeds have been demonstrated, it remains to be seen if this solution can be produced cost-effectively for consumer deployment and no standard has been ratified or even proposed as of 2011.

For 40GbE deployments, QSFP+ is thus required. QSFP+ allows for backwards-compatibility with 10GbE using "spider" or "octopus" cables, which break-out a single 40Gbps QSFP+ port into four 10GbE SFP+ ports. Such breakout cables are also available on -SR4 MMF optical links.

3.3.4 Layer 1 and 2 selection

The two protocols with the largest market share for high performance computing are Ethernet and Infiniband.

From a hardware perspective, the 10Gbps copper varieties of both Infiniband and Ethernet over CX-4 connectors use similar physical layers with the major difference being the clock speed (Ethernet: 156.25MHz and Infiniband 125.00MHz). FPGAs can thus be constructed to support both standards by simply changing the local oscillators and reprogramming the devices. The CASPER iBOB hardware (see section 3.4.5) for example, supports both standards.

The ubiquity of Ethernet at all speeds and for multiple purposes, however, makes it more attractive for this project and 10GBASE-CX4 was selected as the initial interconnect.

Table 3.3: Differences between various common Ethernet links, ordered by link speed and range. Types available in 2012 are in black, with red denoting anticipated standards. Costs are indicative for a typical end-to-end node connection to a datacentre-grade switch port and includes a NIC, switch port and, where applicable, associated transceivers. Cabling itself is excluded and listed in the following column as a per-meter cost. In the case of fibres, this is for a single duplex link (fibre is cheaper in bundles, so this represents a worst-case cost).

Link	Medium	TNC Connector	Max Range	Fixed Cost (USD)	Cable cost (USD/m)
1000BASE-T	Cat5e 4pair Copper	8P8C	100m	165†	0.32
1000BASE-SX	50μ MMF pair	SFP	550m	388	1.62
1000BASE-LX	9μ SMF pair	SFP	10km	644	1.20
10GBASE-T	Copper Cat5e/6 U/UTP	8P8C	55m	742‡	0.32
10GBASE-T	Copper Cat6/6A/7 U/UTP or F/UTP or S/FTP	8P8C	100m	742‡	1.20
10GBASE-CR	Twinax copper	SFP+	7m	796	17.94
10GBASE-SRL	50μ MMF pair	SFP+	100m	1786	1.62
10GBASE-LRM	62.5μ MMF pair	SFP+	220m	3106	1.15
10GBASE-SR	50μ MMF pair	SFP+	300m	2050	1.62
10GBASE-LRL	9μ SMF 1310nm pair	SFP+	1km	2178	1.19
10GBASE-LR	9μ SMF 1310nm pair	SFP+	10km	3370	1.19
10GBASE-ER	9μ SMF 1310nm pair	SFP+	40km	17530	1.19
10GBASE-DWDM	9μ SMF 1540nm pair	SFP+	40km	15732	1.19
10GBASE-ER	9μ SMF 1310nm pair	SFP+	80km	20732	1.19
10GBASE-DWDM	9μ SMF 1540nm pair	SFP+	80km	20732	1.19
40GBASE-CR4	Twinax copper	QSFP+	7m	1416	55.38
40GBASE-AR4	Active optic cable	QSFP+	50m	1390	3.19
40GBASE-SR4	Parallel (8x) 50μ MMF	QSFP+	100m	5948	6.10
40GBASE-LR4	9μ SMF 1310nm pair	QSFP+	10km	21150	1.19
100GBASE-CR4	Twinax copper		3m		
100GBASE-AR4	Active optical cable		30m		
100GBASE-SR4	Parallel (8x) 50μ MMF		100m		6.10
100GBASE-SR10	Parallel (20x) 50μ MMF		100m		30.60
100GBASE-MR4	Parallel (8x) 9μ SMF 1310nm		1km		2.20
100GBASE-NR4	9μ SMF 1310nm pair		1km		1.19
100GBASE-LR1	9μ SMF 1310nm pair		1km		1.19
100GBASE-LR4	9μ SMF 1310nm pair		10km		1.19
100GBASE-ZR1	9μ SMF 1310nm pair		100km		1.19

†Assuming onboard NICs and native BASE-T switches. SFP 1000GBASE-T transceivers cost USD160ea.
‡Assuming native BASE-T switches and NICs. 10GBASE-T SFP+ transceivers do not exist due to power constraints of the SFP+ standard.

3.3.5 Layer 3 selection

The Internet Protocol (IP) is the obvious choice for use with Ethernet. IP supports multiple addressing modes, with unicast, broadcast or multicast options and is the most widely deployed Ethernet layer-3 standard.

Unicasting IP addresses will deliver the packet to one recipient. This is optimal if only one instrument is running concurrently. However, with unicasting, sending the same data to multiple destinations requires the sender to issue a duplicate copy to each recipient, increasing the on-the-wire data rates from the sender. Broadcasting allows a single transmitter to issue one copy of the data and have it received by all connected recipients. Multicasting allows for selected recipients to receive a copy of the data, which is replicated inside the switching fabric when hopping across switched channels. Multicasting is thus ideal for commensal observations where multiple instruments must operate concurrently on the same dataset.

Multicasting

Ethernet multicasting and broadcasting is achieved by addressing packets to special, pre-defined IP addresses which have the broadcast bit set. Multicast addresses map to special MAC addresses. Multicast-capable switches then map these MAC addresses to physical ports if a connected device has subscribed to the multicast address.

Since multicasting is a special case of broadcasting, any connected nodes on the passive channel will receive copies of these packets and individual nodes are then responsible for filtering-out non-subscribed multicast groups. Multicast-aware ("layer-2+"), active switches will intelligently route packets to all nodes in the case of a general broadcast, or only to subscribed nodes in the case of multicast. In the event of non-multicast aware (pure layer-2) switches, multicast packets will be replicated over all links like ordinary broadcast packets.

There is a significant address space limitation with IPv4 multicasting. The 224.0.0.1/8 through 239.255.255.254/8 is reserved for multicasting, allowing a total of 28 bits. IPv6 increases this limit to 120 bits with the ff00::/8 range reserved for multicasting.

However, in both IPv4 and IPv6 cases, only the lower 23 bits map to unique MAC addresses, with 5 bits being ignored. This means that multiple IP addresses map to the same physical layer addresses, which are used by layer 2 switches for routing. Thus, we are limited to only 23 bits of unique address space with layer 2/2+ switches, irrespective of IP implementation.

Commercial switch memory and processing limitations often further restrict the number of simultaneous multicast groups. This limitation is often unpublished. Networks (2011) is an example of a larger modern switch supporting 4096 concurrent groups while some other, smaller units only support 32.

VLANs

An alternative to multicasting might be to make use of Virtual-LANs, which can be implemented to achieve a similar result to multicasting. Inbound packets contain VLAN identifiers which are compared against the switch's routing tables in order to determine to which outbound ports the packet should be routed. Multiple outgoing ports can be configured. Static routes such as this can be configured on modern switches using management interfaces. Switches are designed for such routes to be configured at setup time and not reconfigured during runtime (though many support this facility).

Inbound packets can have VLAN headers appended to them on entry at the switch, based on the inbound port or source IP address. Alternatively, transmitters can tag outgoing packets with a VLAN header themselves.

There is again a limit as to how many VLANs can be operated simultaneously. Modern switches typically support thousands of such groups, a number which is frequently quoted in manufacturer's datasheets.

Custom multicasting schemes

Special-order, custom switches can be constructed to support larger numbers of simultaneous multicast channels. Verbal communications with at least one vendor representative has indicated that their ASICs have the ability to route packets selectively depending on headers in the packet stream in a similar manner to VLANs. This could conceivably be used to implement a suitable

multicast solution with more concurrent groups.

3.4 Hardware selection

After deciding on a stand-alone FPGA-based technology in §3.2.6, I started evaluating suitable processing platforms. The following list of requirements for the processing elements is derived from the research in Chapter 2:

- Digitise analogue streams
- Allow high speed data exchange
- Offer a relatively low heat load (ie low power consumption)
- Be reprogrammable
- Modular, allowing for different ADCs and DACs to be used and the most appropriate selected for a given telescope
- Offer high processing density
- Feature high speed memory for buffering of data and performing matrix transpose operations
- Be easy to program and able to host a library of basic functions
- Low cost

3.4.1 Commercial test equipment

Commercial test equipment such as oscilloscopes and spectrum analysers have been used in some radio astronomy applications. Many modern digital units have interfaces for extracting data through a GPIB interface, via a serial port or over an Ethernet network which can be used to automate their functions and integrate the unit into a larger system. Such machines generally work well when the integration periods can be long (to lower data rates from the bandwidth-limited digital output interfaces), the number of inputs are few (because these systems are costly) and the required operations of the

telescope system are not specialised. But these limitations have precluded their widespread use in radio astronomy instrumentation.

Spectrum analysers are useful in single-dish systems used for spectral work. Spectrum analysers are capable of wide bandwidths but this is historically achieved by sweeping an analogue LO over the frequency range of interest, creating slow instrumentation but allowing a convenient tradeoff of speed vs bandwidth (and usually spectral resolution too). Multiple units can be synchronised allowing slow correlation and imaging work. As of 2010, real-time integrating spectrum analysers are available, but only for narrow bands ($\sim$100MHz) and users generally only have access to data post integration, preventing their use in correlators and beamformers.

Digital (or sampling) oscilloscopes capture time-domain data, making them much more flexible. However, they do not capture continuous data and cannot stream data at full rates. Like scanning spectrum analysers, sampling oscilloscopes make for slow radio astronomy instruments and in general, snapshot data is insufficient for most astronomy tasks; it takes too long to do an observation with millions of accumulations. It can be used for some transient work, however, if a detection system can be devised for triggering the oscilloscope when an event is detected.

Commercial test equipment such as National Instruments solutions also exist and include a comprehensive suite of hardware and software components for radio astronomy use, including analogue data acquisition and processing solutions in the form of FPGA, CPU and GPUs, accessible from a common programming environment (LabVIEW). The flexibility and shrink-wrapped nature of such solutions are attractive. Unfortunately, they are very expensive and are unable to process the wide bands and high data rates typically encountered in larger, modern radio astronomy systems.

3.4.2 Reference platforms

Most of the larger FPGA manufacturers provide reference platforms to aid designers in using their products. These boards are usually very competitively priced to attract new customers to the manufacturer's silicon devices. They generally feature a number of different interface types, including

10/100/1000/10G Ethernet, LVDS ports and also include multiple memory types such as SRAM and DRAM. However, they are designed to demonstrate functionality and are not optimised for high bandwidth signal processing, nor designed for deployment. They typically suffer from a lack of high speed IO and memory interfaces, preventing them from streaming large volumes of data.

While these platforms historically did not support standardised interfaces for ADCs and DACs (for example, Xilinx's ML-501), they now often include FMC or HSMC headers for ADCs. Unfortunately, reference designs for both FMC and HSMC implement both CMOS (LVDS) and SERDES interfaces. Only one of these standards are typically used for a given ADC technology, wasting the remaining lines on the FPGA which could otherwise be repurposed for Ethernet IO or memory connections. FMC commercial products are also very costly for radio astronomy use.

A further limitation for typical reference boards is the lack of an onboard co-processor. This makes remote management clumsy as the platform needs to be connected to a computer for reprogramming. Some are daughter cards that are designed to be hosted on PCI-E or PCI-X busses and this PC integration suffers from short-lived interfaces and high power consumption and makes the processing platform bulky.

Table 3.4 shows two popular reference boards as at 2011 from the two largest FPGA manufacturers, compared to CASPER's ROACH platform (see §3.4.5). Both boards are poor for high bandwidth DSP stream processing due to mis-matched IO bandwidth through all components. Ultimately, commercial reference platforms are poor matches for the radio astronomy instrumentation requirements.

3.4.3 Computer add-on cards

The various commercially-available computer expansion cards are attractive in that they are commercially available, tested parts, often with community support. There are two flavours of cards that might be interesting for radio astronomy use, namely digitiser boards and processor cards. Some smaller radio astronomy systems have been constructed using commercial add-in

Table 3.4: A comparison of popular 2011 FPGA reference boards against CASPER's 2009 ROACH platform, which still provides significantly more IO and memory bandwidth, even though it is two years older.

Metric	Xilinx ML605	Altera DEV-4SGX230N	ROACH
FPGA	XC6VLX240T	EP4SGX230K	XC5VSX95T
IO	2x FMC 1x SFP 1x PCIe x8	2x HSMC 1x 1GBASE-T 1x PCIe x8	2x ZDOK 4x 10GBASE-CX4 1x 1000BASE-T
Memory	1x DDR3 SODIMM	1x 512MBx64b DDR3 1x 128MBx16b DDR3 2x 4MBx18b QDR2	1x 1GBx72b DDR2 2x 36Mbx18b QDR2
Cost	USD 1800	USD 4500	USD 4500

card digitisers that perform all processing on the computer's CPU. GMRT's software correlator (32 input, 32MHz, on 48 1U-servers) is an example of such a system. But for larger applications, the CPU is not able to cope with the high data rates and an accelerator is required.

It is rare to find a single card designed to both digitise large swathes of analogue bandwidth and also provide an onboard processing accelerator. One such commercial offering is EDT's PCIe-8LX hardware accelerator, which is a PCIe add-on card that can host daughter cards such as their DRX16 digitiser which is able to digitise two streams of up to 300MHz. This is sufficient for many small telescope facilities. Unfortunately many commercial accelerator offerings have small target markets, making them prohibitively expensive.

An exception to the high cost add-in card problem is the GPU which is a commodity, competitively priced processing accelerator, however, after Chapter 3.2, I concern myself only with FPGA-based devices.

An example of a promising processing-only card is the NetFPGA[7] project, started at Stanford, which uses open-source hardware. It is a sponsored project and boards are available at low cost. NetFPGA has an established

[7]http://netfpga.org/

toolflow and software environment with a primary focus on processing and network acceleration and so features a number of high performance network interfaces. Unfortunately, it also suffers from a lack of analogue-capable interfaces.

An additional concern with any add-on card is the short-lived nature of the PC bus interface, and the cards themselves, relative to an average telescope operating lifetime. Cards must thus be considered disposable, like rest of the commodity PC. This makes piecemeal upgrades difficult.

Also, the uncertainty and volatility around the different programming environments for each of these cards complicated the question of future support and code portability.

The costs and uncertain futures of the various add-on cards disqualified them for use in this project.

3.4.4 USRP

The Universal Software Radio Peripheral (USRP)[8] offers a low-cost software defined radio platform. The hardware ecosystem consists of ADCs, DACs and network or USB-connected FPGA processing platforms. There is an open-source software ecosystem in the form of the GNU Radio Project[9]. USRP boards were used by SKA-SA for initial array prototyping with PED, the Phased Experimental Demonstrator. The USRP project offers an attractive programming environment and excellent value for money. The modular concept, allowing the motherboard to mate with complete RF front-ends is also attractive as the system can be tailored to suit. Furthermore, the USRP is a popular device with existing community support.

The original board used four 12-bit ADCs operating at 64Msps. This large dynamic range, but low bandwidth is ideally suited for typical SDR applications, but the restricted bandwidth precludes its application on large radio astronomy telescopes. The Altera Cyclone EP1C12 FPGA was primarily used for upfront signal selection and decimation using a digital downconverter. Further signal processing operations are designed to take place on a

[8]https://www.ettus.com/

[9]http://gnuradio.org

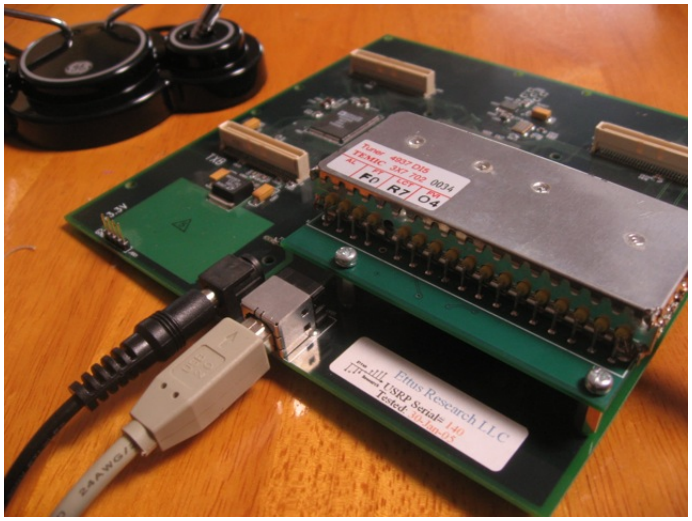


Figure 3.2: A revision 3 Universal Software Radio Peripheral (USRP) 1, serial #140, with an attached TVRX daughterboard.

USB or Ethernet-connected control computer. There are various analogue front-ends available, with frequency ranges from DC to over 6GHz.

Later models have faster ADCs and larger FPGAs[10], but even the largest Xilinx Spartan 3A-DSP 3400 FPGA is still too small to perform typical radio astronomy instrumentation tasks and the 100 MS/s dual ADCs are still too slow for typical modern radio astronomy bandwidths.

Ultimately, the lack of processing capacity and speed led to the USRP processing option being dismissed.

3.4.5 CASPER Hardware

The Collaboration for Astronomy Signal Processing and Electronics Research provides open-source hardware and software designs for use with their Mathworks Matlab Simulink digital signal processing libraries. These boards and libraries are specifically designed for radio astronomy instrumentation. They are ideal for multipurpose back-end processing solutions and, together with the existing CASPER DSP libraries, provide a good starting point for this work.

[10]https://www.ettus.com/product/details/UN210-KIT

At the start of this project, two CASPER boards were available, both using Xilinx Virtex 2 Pro FPGAs. The Internet Breakout Board (or iBOB, §3.4.5) was designed to be a cheap platform whose sole function was to digitise and packetise data. The packetised data would then be fed over high speed serial links into a second-generation Berkeley Emulation Engine (BEE2, §3.4.5) board for processing. This model works well and was ultimately adopted for use in this project.

After becoming acquainted with the iBOB and BEE2 boards during prototyping of the PAPER system (see Chapter A), the concept was improved upon and a new board was designed by SKA-SA in collaboration with CASPER and NRAO for use in KAT-7. The resulting Reconfigurable Open Architecture Computing Hardware (ROACH) board kept many of the key features of the iBOB and BEE2 platforms and was able to fulfil the role of both iBOB and BEE2. Existing iBOB and BEE2 designs are largely compatible with the ROACH, barring some physical interface changes, such as to the QDR memory. The ROACH board proved to be very successful and for two years the boards were back-ordered as production was unable to supply the demand.

SKA-SA's KAT-7 digital back-end (and the focus of this project) is using the ROACH hardware platform. However, technologies are ever improving and the successor, ROACH-2 is already in production with further improvements and optimisations over the original ROACH board. At the time of writing, ROACH-3 was in early design stages.

In addition to these processing platforms, CASPER has a number of ADCs and some DACs available too, with sampling rate solutions ranging from 64Msps to 6Gsps at 8 to 14 bit resolutions.

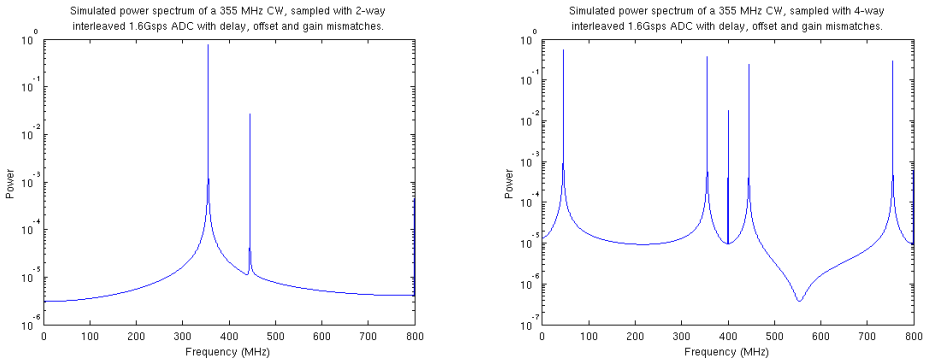
Details of the latest hardware platforms can be found on their wiki at `https://casper.berkeley.edu/wiki/Hardware`. What follows is a brief description and a comparison of the various CASPER boards. A table comparing the compute resources can be found in Table 5.1.

ADCs and DACs

The ROACH and iBOBs have two 80pin ZDOK connectors for LVDS interfacing to carry data, clock and control pins to and from ADCs and DACs.

While some DACs have been built and are available, these are mostly used for interfacing new designs to existing analogue processors which expect analogue inputs. For the purposes of this all-digital design, I am most concerned with selecting the appropriate digitisers for these Z-DOK ports.

Selecting a suitable ADC is important as it affects system performance and cost. Since the sky signal is mostly noise of known power level, the required resolution and dynamic range is primarily driven by the RFI environment. The sampling rate is determined by the analogue bandwidth and front-end analogue characteristics (will the signal be downconverted or sampled directly, filter rolloff slopes determining guard bands etc). For this reason, studies of the RFI environment were undertaken to determine the required dynamic range and select an appropriate ADC for KAT-7.



(a) 2-way interleaved ADC.

(b) 4-way interleaved ADC.

Figure 3.3: Simulated spectra from interleaved ADCs with gain, offset and phase mismatches for a CW input tone at 355MHz. This illustrates the output that can be expected from an iADC (left) and a KATADC (right) if used in interleaved mode.

CASPER's first high speed sampler, the iADC[11] was based on an Atmel (now E2V) AT84AD001B dual-input, 8-bit, 1000Msps (also usable as an

[11]https://casper.berkeley.edu/wiki/ADC2x1000-8

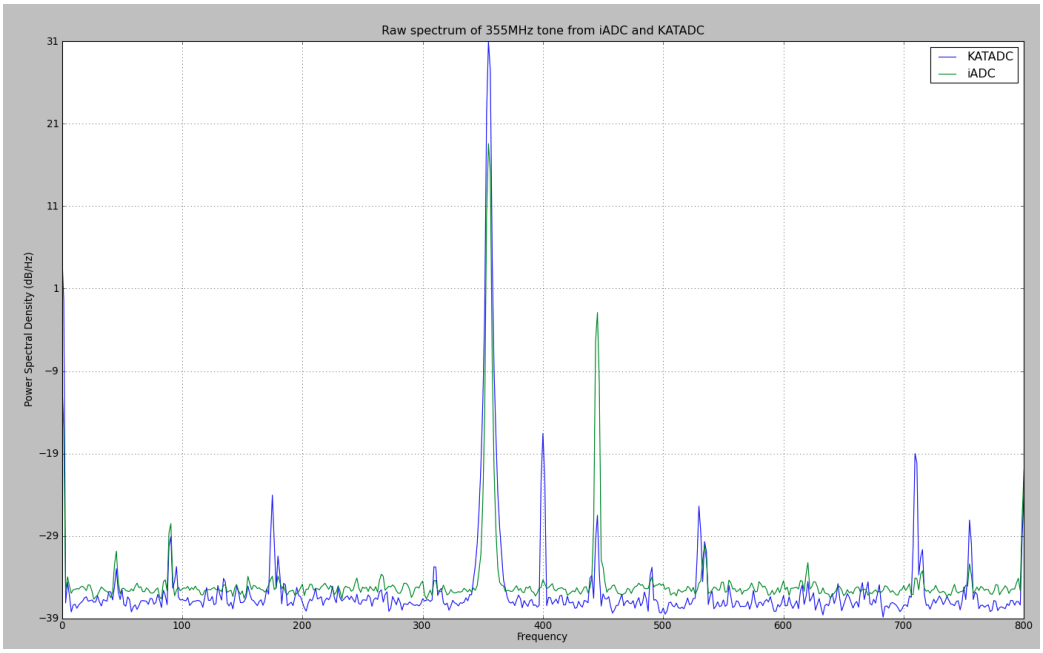


Figure 3.4: Measured spectral output when using the iADC and KATADC in interleaved mode (single input) at 2x800Msps when sampling a 355MHz CW tone. The dual-interleaved iADC produces artefacts at 0Hz and $\frac{F_s}{2}$ due to DC offsets and $\frac{F_s}{2} - F_{tone}$ due to gain mismatches and misaligned sampling clocks. The KATADC, however, produces many more artefacts, with the additional $\frac{F_s}{4}$ DC offset revealing a quad interleaved ADC.

interleaved single-input 2000Msps) unit. This is still the most popular ADC in the CASPER range and two of them can be seen mated to an iBOB in figure 3.5. The iADC board was later revised with PCB-edge mounted SMA connectors and different baluns for improved signal propagation.

A number of additional samplers have been developed to suit specific projects. The PAPER project, discussed in §A uses a quad-input, 8-bit, 250Msps board[12] which traded the high bandwidth of the iADC for additional inputs as their band of interest is only 100MHz wide.

KAT-7, on the other hand, wanted additional features and better analogue matching and so developed the KATADC[13] based on the National Semiconductor ADC08D1520, a dual-input 8-bit 1.5Gsps (or single 3.0Gsps) part. It has programmable analogue attenuation, a 20dB amplifier and RF switch to self-protect in the event of a large over-range input signal. It was later discovered while analysing data from this part that the ADC08D1520 internally interleaves four 750Msps ADCs to create two 1.5Gsps or a single 3Gsps sampler. Figure 3.4 shows the effects of this interleaving. Chen and Wagner (2010) analysed these effects for the dual-interleaved iADC case. We can extend this trivially to the N-way interleaved case and show that spectral artefacts are expected periodically every $\frac{F_s}{N}$ due to mismatched DC offsets (see simulations in Figure 3.3), in addition to mirror copies of any incoming signal at the supplementary angle around these DC lines. For the 1.6GSa/s case of Figure 3.4, the presence of a static 400MHz component in the KATADC spectral plot reveals it to be a four-way interleaved unit, which is not disclosed in the manufacturer's datasheet.

The use of interleaved ADCs can create scientific complications, especially for spectral line work when in the presence of narrowband RFI. In this case, scientists are searching for spectral lines in known frequency bands but the mixing effects of interleaved ADCs could produce false positives due to the presence of out-of-band narrowband signals that get mirrored into these bands. The noisefloor of the system is also raised as the noise-like input signal is mixed and added throughout the band. SKA-SA tries to avoid interleaved ADCs when possible.

[12]https://casper.berkeley.edu/wiki/ADC4x250-8
[13]https://casper.berkeley.edu/wiki/KatADC

iBOB

The Internet Break-out Boards (or iBOBs; Figure 3.5) are constructed around a Xilinx Virtex-II Pro FPGA, along with two external 36bit wide SRAM modules of 18Mbit capacity each. iBOBs have two Z-DOK ports for interfacing to ADCs and two high speed CX4 ports for 10Gbps communications to other devices.

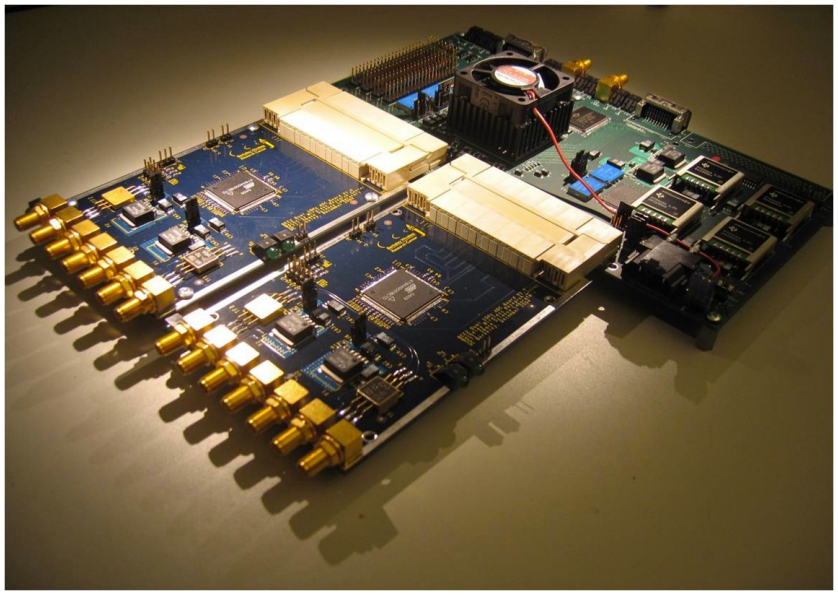


Figure 3.5: An early revision of the Internet Break-out Board (iBOB) populated with two first-generation dual-input iADCs.

The iBOBs were designed to host a smaller Virtex-II Pro FPGA device, as its sole purpose was to packetise the data from high speed digitisers and transmit these packets over high speed serial links for processing on other boards, either over an Ethernet network or by direct connection. However, the platform became popular as a full signal processing board and was later revised with larger power supplies to host a larger Virtex-II Pro 50 device.

These boards are small, require only a single 5V supply to operate and can be mounted in a modified Compact-PCI (cPCI) chassis for larger deployments of multiple boards. This creates a very dense computer. Due to the depth of the device when populated with ADCs, the standard backplane

in any cPCI chassis must be removed. The non-standard power supply and enclosure requirements created complications for larger deployments but did not prevent the board from becoming very popular with the radio astronomy instrumentation community.

The iBOBs were successful because they did not require a significant investment by adopters and were simple to understand and operate. The boards are relatively cheap to purchase and the software support package, apart from Matlab/Simulink, is open source. Adopters are able to understand the boards quickly due to their simplicity and so are able to use them quickly and easily.

However, there are ultimately two major difficulties when using iBOBs as part of a deployed reconfigurable system.

Firstly, the FPGA configuration is stored in onboard EEPROM which must be programmed via JTAG. This makes it difficult to reprogram remotely. Some finite number of boards can be chained together on a single JTAG chain but this has proven to be a slow and unreliable means to reconfigure the devices at runtime.

Secondly, users are only able to interact with their designs through TinySH, a basic interactive shell that runs on one of the FPGA's onboard Power PC cores. It is a simple command-line interface for reading and writing registers on the FPGA. Without an Ethernet stack, TinySH is accessible only over a direct serial (RS232) connection. And the "Lightweight IP" component, which allows for remote Ethernet communication, requires a significant fraction of the FPGA's resources. For this reason it is often not used, or must be hand-tuned for each implementation to fit into available resources. The lack of a unified, simple and flexible operating system with a networked control interface is the iBOB's Achilles heel.

BEE2

The second generation Berkeley Emulation Engine (or BEE2; Figure 3.6) is based around five Xilinx Virtex-II Pro70 FPGAs. Memory is available in the form of 18 1GB DRAM DIMMs. It uses 18 CX4 ports exclusively for IO, allowing for up to 180Gbps of high-speed data. Without any significant IO

expansion ports, BEE2s are unable to host ADCs directly. This was never a design concern as the boards were to be used for simulating ASIC designs, for which ADCs are not required. For this reason, CASPER users must use iBOBs to digitise analogue signals which can then be linked to the BEE2's FPGAs, using point-to-point XAUI communication over the CX4 ports at 10Gbps per link, for further processing.

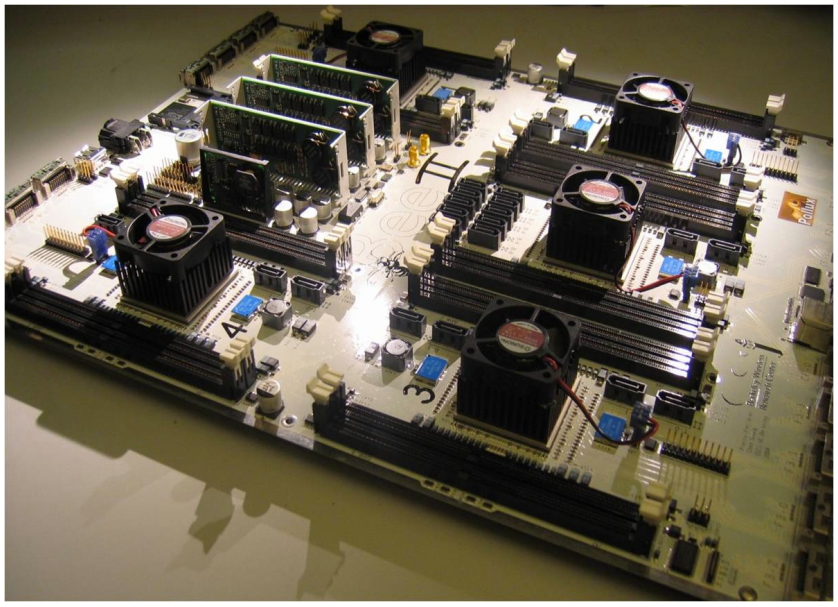


Figure 3.6: A second generation Berkeley Emulation Engine (BEE2) processing board.

The BEE2 was a popular radio astronomy instrumentation research platform and has been deployed for scientific use at NRAO's Greenbank site and at the ATA, for example. The BEE2 offers high compute densities and a lot of fast IO, making it popular for bandwidth-intensive applications. Unfortunately, the BEE2 is an expensive platform that is sometimes difficult to deploy reliably in production environments.

The BEE2's fifth FPGA is often used solely for control and monitoring. It runs BORPH (a Linux-based operating system which allows for the programming of and interaction with FPGAs as if they were ordinary software processes; see (So *et al.*, 2006)) on one of the FPGA's Power PC (PPC) cores. BORPH provides a convenient programming abstraction of the FPGA

processes but reconfiguring this central FPGA at runtime for another DSP task is cumbersome, because loading a new design at runtime also resets the onchip PPC cores; the very cores running BORPH. Since the BORPH operating system is used to reprogram the FPGAs, in order to make use of the fifth FPGA, partial reconfiguration would be required. This is not supported by the CASPER toolflow at the time of writing. For this reason, in many cases, the central FPGA does not perform any DSP function itself and is configured at power-up from an onboard compact-flash card (which also hosts the Linux root filesystem). This unfortunately makes the BEE2 expensive in this implementation as the entire fifth FPGA's DSP capabilities are wasted.

The 100Mbps Ethernet control port off the fifth FPGA provided a very convenient interface to the board. However, the onboard PPC is not a fast CPU and only has an 8bit, 10MHz bus link to the FPGAs. This created a significant bottleneck for exchanging data between the network and the users' FPGA designs. Populating large memory lookup tables is a slow exercise and quickly becomes a limitation for radio astronomy instrumentation, especially for operations like beamforming which require regular, timed loading of steering coefficients.

The BEE2 is a very large, complicated board. As a result, many production runs suffered from low yields. In one batch, only one third of the boards were functional off the production line. This made the boards expensive and unreliable. The BEE2 also requires a large, custom 5V power supply and chassis, both costly products.

The BEE2's asynchronous memory is more difficult to program for many astronomy uses, complicated by burst read/write requirements to obtain high performance and variable latencies depending on bank, rank and row operations. It is simpler to use synchronous memories which the BEE2 does not have. However, the large memory capacity coupled with the high memory bandwidth is useful for operations such as circular buffers and gated or triggered captures of high speed samples.

Interconnecting a BEE2's FPGAs is an on-board point-to-point LVDS bus in a torus configuration. These busses could be clocked synchronously with all the FPGAs and so for many single-board applications it provided a simple

to use, convenient interconnect. However, for multi-board applications, these busses were often eschewed in favour of a unified interconnect architecture, both between boards and between FPGAs on a single board. This was in the form of the asynchronous CX4 ports which could host 10Gbps Ethernet or XAUI protocols.

ROACH

The second generation hardware platform, ROACH (*Reconfigurable Open Architecture Computing Hardware*), is a Virtex5-based upgrade which addresses some of the issues with the iBOB and BEE2 solution. It merges aspects from the iBOB and BEE2 platforms into a single board with a superset of all their features. ROACH is a single-FPGA board favouring the use of 10GbE interfaces for all inter-FPGA communications.

The iBOBs' CX4 interfaces are kept along with the same Z-DOK daughter board interface, maintaining backwards compatibility with existing ADCs, DACs and other interface boards. A high bandwidth, high capacity DRAM memory is available along with an upgrade to the iBOB's SRAM in the form of QDR memory which supports simultaneous reads and writes on every clock cycle. The stand-alone BORPH usage model of the BEE2 was also kept, with Ethernet remote control. It also features a standard ATX form-factor, allowing the use of off-the-shelf enclosures, rack-mounting equipment and power supplies.

The ROACH developers opted to retain the BEE2's BORPH (So *et al.*, 2006) as the operating system of choice. When selecting the CPU to host BORPH, various approaches were considered including the use of FPGA-softcores, FPGA-hardcores and external processors. On-chip CPUs make it difficult to reprogram the FPGA as discovered with the BEE2; you lose the CPU during a full reconfigure and would thus need to do a partial reconfiguration if you wanted to use the CPU to configure the FPGA. Softcore processors have this same problem and also wastes potential DSP capacity. In either case, the FPGA needs a bootloader to start the boards initially. ROACH opted for an external processor for simplicity and independence from the FPGA operations. A PowerPC (PPC) was originally chosen for its

Figure 3.7: A Reconfigurable Open Architecture Computing Hardware (ROACH) processing board.

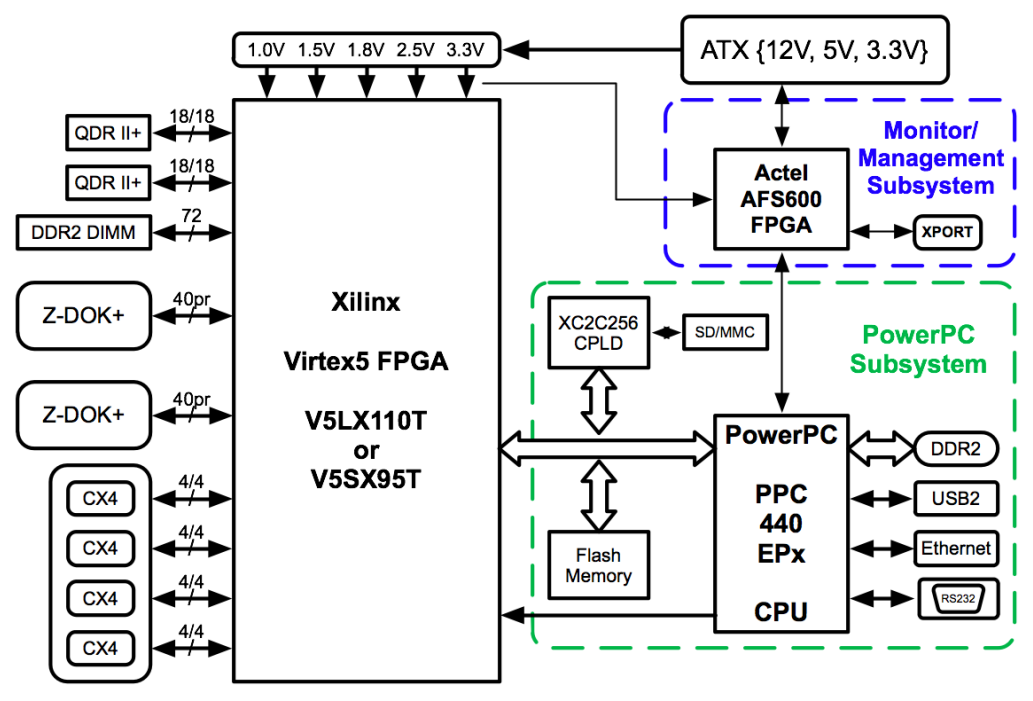


Figure 3.8: Simplified ROACH block diagram.

IBM CoreConnect architecture and hence simplified FPGA-PPC integration and migration from BEE2 systems with onboard PPC cores. Each ROACH has an onboard AMCC PowerPC 440EPx coupled to a National Semiconductor 10/100/1000 Ethernet PHY for remote control and communication. This processor runs BORPH and is the primary means of programming and communicating with the FPGA.

ROACH is able to host either a Virtex 5 LX110 or SX95T FPGA. These FPGAs have different ratios of on-chip memory, multipliers and logic resources, with the LX range favouring logic and the SX range favouring signal processing applications (DSP and BRAM resources).

Working with the DRAM and SRAM on the iBOBs and BEE2s clearly demonstrated that the synchronous SRAM was simpler and more efficient to use, resulting in more streamlined vector accumulators and matrix transposes, albeit at a higher board cost and reduced memory capacity than a DRAM solution. Compared to the iBOB's SRAM, ROACH was populated with larger capacity and faster QDR synchronous memories. Early boards contained two fully independent 36Mbit parts that could do two 18 bit transactions per clock (DDR) on separate read/write ports. Later boards were populated with 72Mbit parts. A single DRAM module slot was kept for those applications requiring large memory capacities.

The Karoo Array Telescope Control Protocol (KATCP)[14] over Ethernet is the preferred communications protocol for interacting with devices at the Karoo Array Telescope facility. In the ROACH implementation, it uses TCP/IP as its underlying communications layer. It is a human readable, but machine-parsable line-based text control protocol not dissimilar to FTP or SMTP. ROACHs are supplied with a pre-installed KATCP server for configuring the FPGA and transferring low-speed, high reliability data across a network. ROACH's KATCP server runs on top of BORPH on the PPCs.

The iBOB and BEEs' FPGA-CPU interface was designed for configuration and control of the FPGA only and was often a limiting factor in designs which repurposed this bus for larger data transfers. ROACH's bus is more than 12 times faster with a theoretical data rate chosen to match the 1GbE link. This is fast enough to stream output products for smaller instruments.

[14]https://casper.berkeley.edu/wiki/KATCP

PAPER (§A), for example, makes use of this link to output data products, for system configuration and for realtime monitoring of system health.

The ROACH was CASPER's first general-purpose instrumentation platform and was chosen as the hardware base for this project. ROACH-2 has since succeeded ROACH-1. Details of ROACH-2 can be found in §5.5.5

3.5 Conclusion

It was ultimately decided to base this project on FPGA processing elements. After evaluating popular platforms, the CASPER iBOB and BEE2 devices were selected. Limitations with these boards were identified and a replacement device (ROACH) was jointly developed by SKA-SA, NRAO and CASPER to address these limitations. ROACH development was undertaken in parallel, while the early stages of this work continued on iBOBs and BEE2s.

A commercial 10Gbps Ethernet switch was also selected as the basis for the interconnect of the processing elements. A UDP/IP protocol is layered on top of Ethernet for data exchange between nodes.

Chapter 4

Implementation

This chapter describes the construction of an FX correlator and the later addition of beamformer functionality. Both the design and implementation of this reference system were undertaken while visiting CASPER at the University of California at Berkeley, in collaboration with Aaron Parsons and Dan Werthimer, initially during the 11 month period of May 2007 to April 2008, with further visits in July to Nov 2008 and June to August 2009.

The resultant design is modular, allowing for components to be implemented using various processing technologies. This chapter describes the design and operation of these components for the reference, FPGA-based platforms.

The essential processing elements and DSP operations are shown in Figure 4.1. Each of these operations will now be discussed in this chapter.

In my FX design, channelisation takes place first in *F-engines* and the cross-multiply-accumulate operation occurs afterwards in *X-engines*. Connecting these engines is an Ethernet switching fabric. Figure 4.2 shows the proposed system architecture.

It was successfully implemented on iBOBs, BEE2s and ROACH boards, demonstrating not only that the packetised concept works but that the design is also portable across hardware generations with minimal modifications.

The initial design target was the PAPER array and later KAT-7. This section will focus on the implementation of a generic correlator with telescope-specific parts for these two systems discussed in §A and §B respectively. The

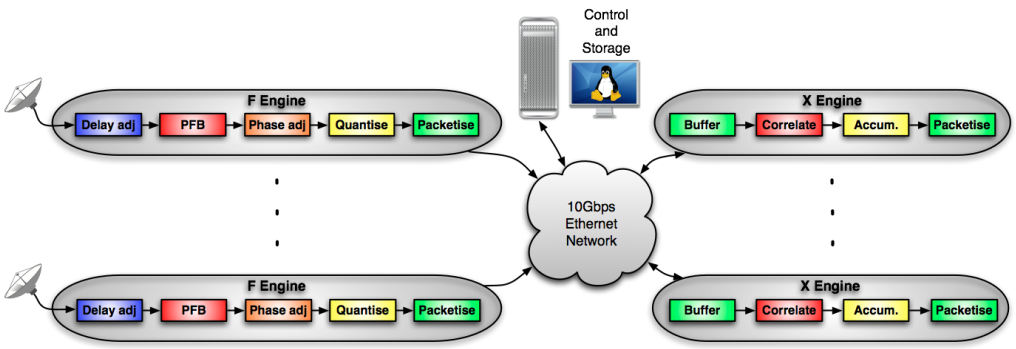


Figure 4.1: Block diagram depicting the operations and signal flow of the implemented FX correlator. This was deployed on KAT-7 for wideband imaging.

design has changed subtly over the course of this project with working ideas refined as lessons were learnt. An analysis of the design can be found in §5. Appendix C provides a forward-view of the direction of this work as it pertains to the design for the future MeerKAT telescope.

The implemented design is open-source, with all source code, firmware and software available on SKA-SA's public github account.[1]

4.1 Mapping to Architecture

There exists a natural break in operations between the per-input channelisation operation and the per baseline multiply-accumulate operation. These can easily be placed on separate compute boards. Furthermore, each spectral channel can be processed independently from all others, forming another natural dimension to split the multiply-accumulate operations across boards. Each X-engine then processes a subset of the spectral channels.

These X-engines now require a subset of the data from every F-engine. Connecting these engines is a common routing problem for radio astronomy correlators. To overcome this, I use an Ethernet switch. Data from F-engines can then be routed to different X-engines by altering the destination

[1] https://github.com/ska-sa

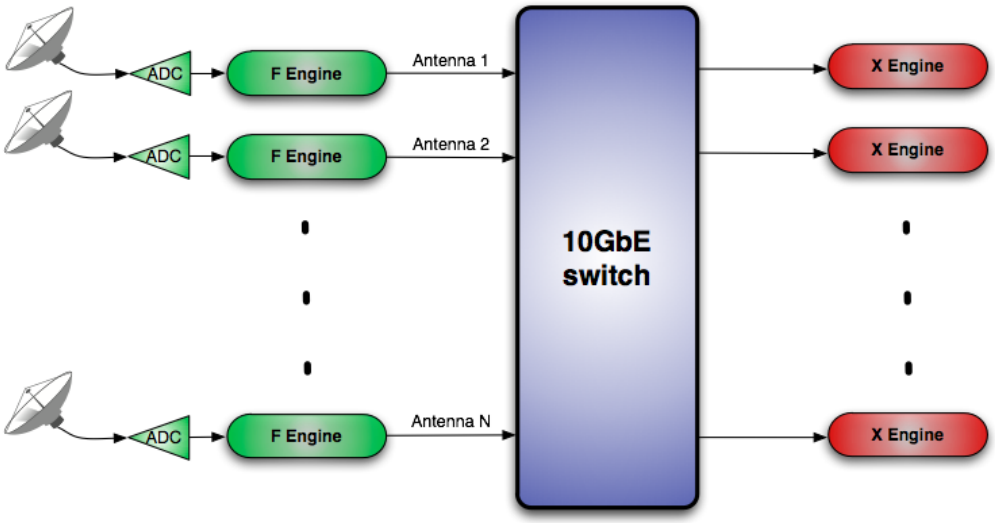


Figure 4.2: The proposed CASPER system architecture for the prototype correlator, showing the F-engines, X-engines and packetised interconnect.

IP address of a packet. F-engines are then required to packetise their data streams into groups suitable for processing by the X-engines. Unfortunately, Polyphase Filterbanks typically produce a time-series of complete spectra. Packing this output into sensibly-sized packets thus requires a matrix transpose operation to group together a time-series from each spectral channel before dispatching these spectral channel groups to X-engines for processing.

In order to maintain processing efficiency, hardware compute power must be fully utilised. Coarsely, this can be achieved by hosting multiple engines on a single compute node when capacities allow. This sets a lower limit of hardware utilisation of 50%, which is deemed sufficient for initial implementations. To allow for efficient scaling to larger systems, in the event that a single engine cannot be contained on one board, provision should be made to split an engine across processing nodes.

Since all F-engines and all X-engines perform the same operations, the system should only need two compiled FPGA bitstreams: one for the F-engines and another for the X-engines. Once programmed, all elements of a given type behave identically and boards gain identification by numbering

them at runtime.

4.2 Interconnect

It is only with the emergence of 10Gb Ethernet that a switched interconnect has become a viable option for high-bandwidth radio astronomy instruments. The data rates required by these systems were too high for earlier, slower implementations of Ethernet. However, now that we have fast packetised, switched interconnects available commercially, many new instrument features become possible including dynamic re-use of shared hardware.

4.2.1 Switching latency

Network latency is a concern for the interconnect design of modern supercomputers and large data-centres as many applications require two-way interprocess communication between processing nodes. However, radio astronomy processing is not generally sensitive to interconnect latencies, provided these are constant, because processing nodes operate independently on different antennas or different frequency bands. Nodes can thus operate in a streaming fashion.

Accumulators in imaging pipelines typically run for hundreds of milliseconds or tens of seconds, whereas modern Ethernet provides latencies of significantly less than a millisecond across links. Processing delays thus dominate the signal path latency. Pulsar observations require shorter integrations but the processing time for the de-dispersion algorithm is again typically longer than the network latencies.

Latency (and jitter) is significant only insofar as it causes mismatches through non-homogeneous switches when packets from multiple sources take different routes (and thus are subject to different delays) which must be reassembled by the receiver before processing can take place. Empirical evidence from this work suggests that this is not a significant concern in modern switches and variations in latency are absorbed by the re-ordering logic already in place to handle out-of-order packets.

Details of constructing large, scalable switches can be found in §5.3.3 and the operation of this re-ordering logic is described in §4.5.3.

4.2.2 Commensal observations

We would like to be able to route data from antennas to multiple instruments digitally. Data multicasting will promote commensal processing by multiple back-ends, sharing digitisers and front-end processors. A modern interconnect, such as an Ethernet network, is a cornerstone enabler for such a system.

Ideally these processing back-ends would be able to subscribe to any data from any individual antennas and any frequency channels. For a medium sized array, 64 antennas with 2048 frequency channels, this would require 17 bits of address space. Looking ahead to large, SKA-sized arrays, we'd want more like 4096 antennas and 1M channels or 32 bits of address space. This is equivalent to the entire IPv4 space and is considered unreasonable for IP technologies.

§3.3.5, §5.3.3 and §3.3.3 detail the current state of affairs of IP multicasting, scalable switches and Ethernet technology in general.

Fortunately, most astronomy processes can be made to be independent over some frequency band, and it is unlikely that a single processing node will want access to a single frequency channel only. This makes bands of frequencies a natural addressable unit. Determining the size of this band is a more difficult decision and depends on the science driver and instrument processing engines' capabilities.

For the correlator and beamformer application, I wish to send some frequency channels for all antennas to a single processing node and initially I will concentrate on this one-to-one source to destination (unicast) mapping, with multicasting being considered later.

4.2.3 Signal routing

Packets are re-ordered by the switch and frequency slices are distributed to X-engine boards in a round-robin fashion. Each X-engine then processes a subset of the band for all antennas.

X-engines of initial systems processed a comb of frequencies as the sequential output of the F-engines were distributed in a round-robin fashion directly to the X-engines. Each X-engine thus did not process a contiguous band of frequency channels. See §A and Parsons *et al.* (2008) for further details of traffic flow for the initial PAPER deployment. In these systems, should an X-engine board fail, channels throughout the spectrum are lost, potentially disrupting subsequent imaging processors which expect complete spectra. Also, with this distribution scheme, if multicasting of individual channels is not possible (typically due to the switch address table limitations as outlined in §3.3.5), grouping adjacent channels into a single multicast group may become difficult. Implemented naïvely, there is increased potential for receive buffers to overflow as all F-engines would send consecutive, sequential packets to be funnelled into a single X-engine board.

The KAT-7 designs improved on this naïve implementation to reduce network buffer requirements by re-ordering the output of the F-engines and routing a number of adjacent channels to the same X-engines. Grouping these outputs together into a single multicast group is then sensible as it represents a continuous sub-band of the frequency spectrum. Subsequent processors can then subscribe to bands of multicast groups. Should an X-engine fail, only that sub-band is lost and the rest of the system processing other bands of the spectrum is unaffected. Network buffering requirements are reduced as consecutive packets are routed to different ports. §4.5.6 details these two routing schemes more thoroughly.

4.2.4 Interconnect optimisation

The most generic solution would see every processing engine in the system physically connected directly to the network switch. However, an optimisation proposed by Aaron Parsons, leveraging the predictable nature of the dataflow in the correlator system, can halve the number of switch ports required. This optimisation was originally discussed in Parsons *et al.* (2008) and was initially applied to the PAPER system described in §A.

In this scenario, packets are not fed directly into the network switch from the F-engines. Notice in Figure 4.2 that although Ethernet provides full

bi-directional links, connections are inefficiently being used unidirectionally, with F-engines sourcing data but not sinking any. Figure 4.3 shows how packets can be sent from F-engine boards directly to X-engine boards over point-to-point XAUI links. X-engines can then loop this F-engine data out over a bi-directional exchange port. The data returning from the switch is the same as before the optimisation. This arrangement halves the number of required switch ports, as each switch port link is now used bi-directionally.

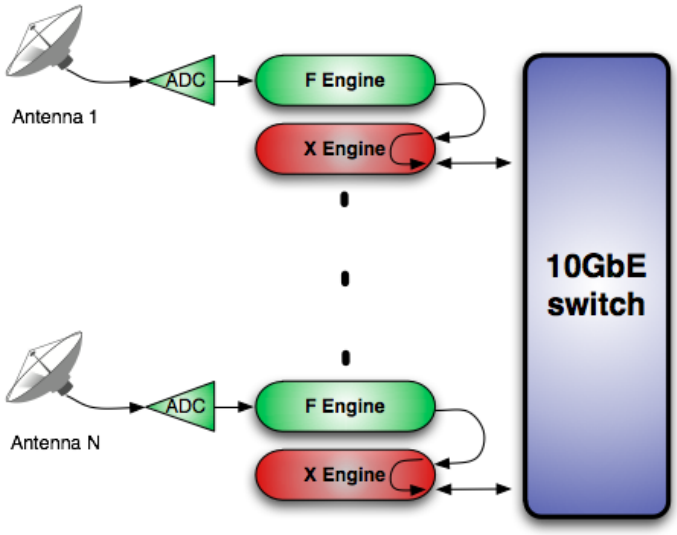


Figure 4.3: Packets can be routed through the X-engines to save switch ports.

There are a number of drawbacks to this optimisation.

- There is added complexity within the X-engines which now need to include point-to-point (in PAPER's case, XAUI) cores for receiving data from F-engines on a separate port. Fortunately, the XUAI cores do not consume significant device resources and so this is a low cost option.
- Switches do not forward packets back to a destination located on the same physical port as the source. So if a node sends a packet destined for itself, the switch will not return that packet. In these cases it is

expected that the node will keep a local copy of the packet. In ordinary computers, this is handled by a local loopback interface. Because a fraction of the frequency channels from each F-engine must be returned to the X-engine to which it is connected, each X-engine must be outfitted with a loopback function. Table A.2 shows that this can be a costly consumer of FPGA resources (9% of the logic in PAPER's implementation) due to switching latency and buffering complications.

- In the case of an X-engine board failing under these optimised conditions, not only is the subset of band lost that the X-engine was processing, but also the associated F-engine and its connected antenna (ie the entire band for that antenna).

While PAPER leveraged this optimisation, the KAT-7 system does not and has elected for the simpler system shown in Figure 4.2 at the cost of an extra switch port for each connected antenna.

4.3 F-engine operation

As shown in Figure 4.4, the F-Engines perform the following main functions:

- Sampling (digitising) the RF signal from an antenna
- Channelisation of the sampled signal
- Apply delay compensation (both coarse and fine) and fringe rotation
- Apply complex gain correction.
- Frequency band selection
- Re-quantisation
- Re-ordering of data (packetisation preparation)
- Packetisation of data for transport over 10Ge

Each of these functions will be discussed in this section.

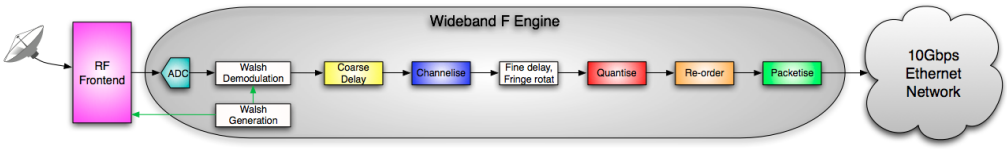


Figure 4.4: A simplified block diagram showing the primary F-engine operations. Walsh switching was not implemented in this design.

4.3.1 Digitisation

To allow the correlation operation, digitisation of the analogue signals must take place synchronously. A clock distribution scheme is required in order to supply each of the samplers with a coherent sample clock. In the case of the centralised processing systems, where the analogue signals are brought back to a central processor to be digitised, this is a simple matter of amplifying and splitting the clock signal to each ADC.

In the case of a distributed digitisation system (such as the EVLA or MeerKAT), the sampling clock is affected by the distribution network which can induce significant phase errors on the sampling clocks. Fibre optics are often used, which can be buried for improved performance. On high frequency systems, a round-trip phase measurement system can be implemented to correct for these effects.

A complication arises when digitising wide bandwidths. FPGAs' maximum clock rates are often low (hundreds of MHz) relative to sampling clocks (GHz). This forces us to process the signal in parallel on the FPGA. I clock the F-engine FPGAs synchronously, by deriving the FPGA clock from the sampling clock: $f_{FPGA} = \frac{f_s}{P}$ for P parallel streams. The design caters for the use of many suitable ADCs by allowing P to be altered at compile time. Values for $P < 1$ (sampling at rates below the FPGA clock rate) are currently not supported as most modern telescope systems are already processing bandwidths in excess of 100MHz (already requiring real sampling clocks of over 200MHz, a reasonable clock rate for modern FPGAS).

Complex sampling is avoided when possible as it introduces analogue sine/cosine mixing errors. Fast samplers in excess of 30Gsps have been an-

nounced commercially, reducing the historic need for complex mixing and sampling to digitise wideband signals with slow samplers. However, should this be desired, the CASPER libraries do support complex DDCs, FIR filters and FFTs.

4.3.2 Digital down-conversion

Some systems (such as PAPER, discussed in §A), do not require the entire digitised band to be processed. In these cases, a sub-band must be selected. This is done using a digital downconverter (DDC) which mimics the analogue operations of mixing and filtering to baseband before decimating the sampling rate.

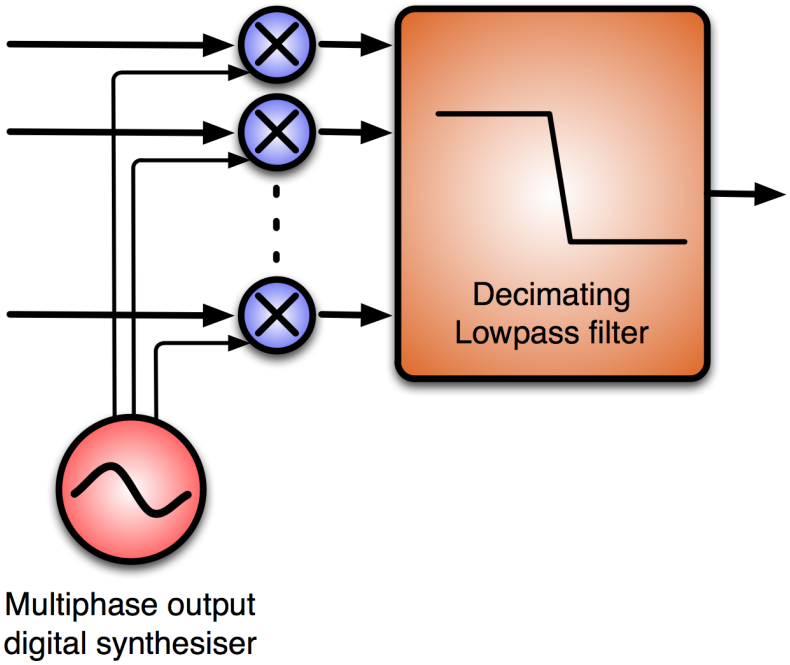


Figure 4.5: A digital downconverter digitally mixes a sin and cos generator with the signal before filtering. Once the band of interest has been selected, the data can be decimated to lower the data rate.

While downconverting digitally (as opposed to in analogue, and digitised with slower samplers) can increase the cost of the samplers, it does offer a

number of advantages over the more traditional analogue approaches. Digital filters are reliable and repeatable with predictable effects on the signal. They also do not suffer from temperature or ageing effects. Analogue systems performing complex quadrature mixing and sampling also suffer from imperfect alignment of the sine/cosine mixing oscillators. Digital sine/cosine mixers can be constructed with near-perfect phase and amplitude alignment (barring quantisation effects).

The CASPER DSP library includes a DDC, though it is currently not runtime configurable. Mixing frequencies can be set to $F_s(\frac{N}{M})$ where M and N are integers. In PAPER's early correlators, which drove the development of this processing element, the selected band of interest was fixed at 150MHz, centred at 150MHz, derived from a 600Msps digitiser.

This library component could be modified to support runtime configuration of the selected band by manipulating the filter coefficients and mixing frequencies. Configuring the decimated bandwidth at runtime is more difficult when using CASPER's current polyphase filterbanks as these are configured at compile time to process a given bandwidth and are designed to accept synchronous data. The preceding DDC must be matched to these filters. This effectively means that the processed bandwidth is fixed at compile time. Different FPGA personalities could be loaded at runtime to support alternate processed bandwidths, as has been done for KAT-7 (see §B).

4.3.3 Delay and phase compensation

As outlined in §2.4.1, delay adjustment is required to compensate for varying cable lengths in the system. Fringe correction is also needed for systems with analogue LOs. Peens-Hough (2010) analyses the effects as they pertain to KAT-7 and MeerKAT.

The delay compensation in my FX correlator is constructed in two parts, consisting of a coarse component (in units of ADC samples) and a fine component (less than one sample clock) as shown in Figure 4.6.

The coarse delay, illustrated in Figure 4.7 is simply implemented using an addressable memory configured as a programmable shift register. In designs where the signal bandwidth is larger than the FPGA clock rate, this compo-

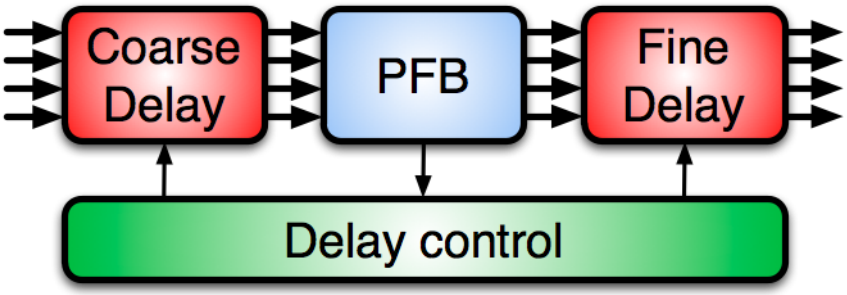


Figure 4.6: The delay infrastructure is split into two parts, consisting of a coarse delay able to delay the signal in increments of the sampling clock and a fine delay for sub-sample, per channel phase adjustment post PFB. A common control interface co-ordinates these operations.

nent is complicated by the fact that samples arrive in parallel. Data cannot, thus, just be written-to and read-from a single, wide, monolithic memory. A barrel-shifter is needed to re-order the parallel streams, in the event that the number of samples to be delayed is not a multiple of the number of parallel streams.

By the Fourier identity $x(t-t_0) \Longleftrightarrow a_k e^{-jk(\frac{2\pi}{T_0})t_0}$, a time shift is just multiplication by a rotating phasor in the frequency domain. By limiting the shift to one clock cycle (positive or negative), it can be implemented in the frequency-domain[2] by multiplying with a phase ramp over all frequencies. While fine delays are also sometimes implemented in the time-domain using a phase-shifting FIR filter, this requires an entire additional wideband filter. In this FX correlator, however, a DFT is already being performed as part of the F-engine process and so this this fine-delay functionality can be implemented simply by multiplying the DFT output with an indexed sine/cosine lookup table. The practical implementation of the fine delay component for a single datastream is illustrated in Figure 4.8.

The process is complicated because the delay is split into coarse and fine sections. Recall from §2.4.1 that there is a need to automatically update

[2]A small, finite error is incurred due to the finite channel width. It would be perfect for a monochromatic signal.

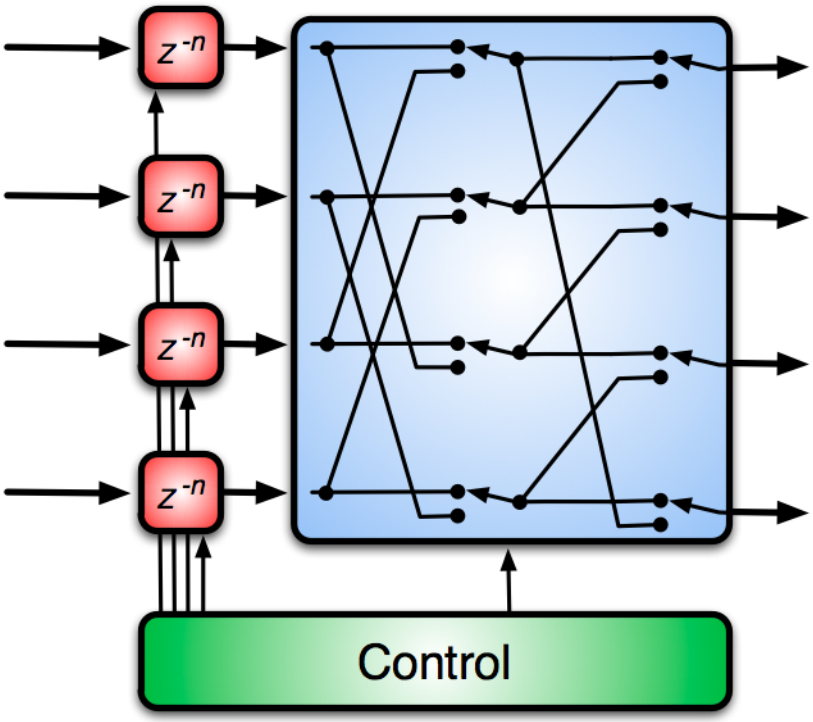


Figure 4.7: The coarse delay block architecture as used in the KAT-7 system, which accepts four samples in parallel. The coarse delay consists of an addressable memory configured as a programmable shift register along with a barrel shifter to re-order the four incoming parallel samples arbitrarily, to allow for delays of less than four sample clocks.

these delays while the system is operating as geometric delays change. The updates of the coarse and fine components must now be synchronised. That is, when incrementing or decrementing the fine delay shift to the point of an ADC sample boundary, the fine delay component must wrap, and the coarse delay must be incremented or decremented by one sample. But the coarse delay takes place before the FFT, which incurs a processing latency of L clock cycles, before the signal reaches the fine delay. So the coarse delay must be pre-emptively adjusted L cycles before the fine delay wraps.

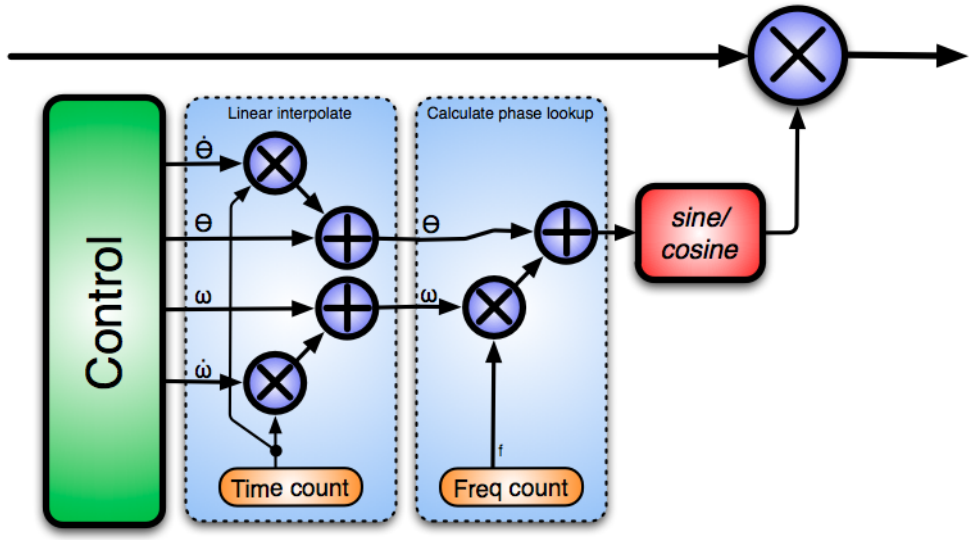


Figure 4.8: The fine delay block is implemented as a phase lookup into a sine and cosine table which is then mixed with the datastream from the polyphase filterbank. It also implements the fringe rotation function and has hardware linear interpolation for both delay and fringe offset. The control system allows for timed loading of new values, to synchronise it with the coarse delay block. One signal path is shown; this path may need to be replicated with offsets for systems processing high sample rate parallel data streams.

Furthermore, I need to account for fringe rotation due to possible down-conversion in the analogue receiver stages. The combined effect of both fringe stopping and fine delay compensation is implemented as an incrementing

phase lookup in a sine/cosine table on the FPGA in the form $phase = \omega f + \theta$ for a given spectral channel, f, where ω is the delay rate and θ is the instantaneous phase offset due to downconversion. Hardware linear interpolation is supported so that software can periodically update the phase and delay values along with update rates for each, and then leave the hardware to continue interpolating independently, thereby reducing the software processing load. This greatly reduces the control and monitoring bandwidth required for each input during runtime.

4.3.4 Channelisation

My channelisation operation employs a polyphase filterbank (PFB) constructed from a sliding-window FIR filter followed by an FFT. The CASPER library contains various versions of these blocks, which I have used directly with minimal modifications. They boast a number of optimisations to make best use of the various Xilinx FPGA architectures and are highly efficient. For example, Parsons (2009) describes the in-place re-order algorithm employed by the FFT to reduce memory requirements.

Channelisation of both complex and purely real data is possible. In the case of the latter, an N-point real CASPER FFT produces $\frac{N}{2}$ complex output channels per spectrum; only the real half is output (since the other half is a mirror image) in order to save logic resources.

What differentiates this FFT from traditional FFT implementations is its streaming nature. Whereas the Xilinx-supplied FFT, for example, buffers a block of data and then processes it serially before generating the output, the CASPER FFTs implement the butterflies in parallel. This consumes additional multiplier and adder resources but requires less buffering (thereby reducing memory requirements). Wideband versions of the cores are also provided which can accept samples from the ADCs in parallel, allowing for processing of very wideband signals on low-clockrate FPGAs, provided sufficient resources exist.

The PFB signal path uses fixed-point arithmetic to make best use of the FPGA's onboard DSP multipliers (18x18 bit on Virtex-2 Pro and 18x25 bit on Virtex-5). Bit growth is limited through the butterfly stages by an

optional downshift at each stage. Received sky signals are noise-like and so a growth factor of $\sqrt{2}$ is expected per stage. In these cases the shift schedule can be relaxed to allow the noise to occupy a significant fraction of the available dynamic range. However, in the presence of narrowband RFI, such as received from satellites and terrestrial broadcast towers, significant power can accumulate in individual bins and so in RFI-rich environments it might be necessary to be more aggressive with this shifting schedule.

Care must be taken not to be too aggressive with the shifting schedule, especially with longer FFTs, and risk dividing-down the signal so much that significant bits are lost and hence degrading SNR. At the extreme, the signal of interest could be lost entirely. An overflow in any one channel ruins the results in other channels too, due to the employed algorithm's subsequent re-use of intermediate results in later stages. Overflows and underflows inside the FFT must thus be avoided and spectra discarded if these do occur. A status bit is provided to indicate if an overflow has occurred so that the shift schedule appropriately re-adjusted. A proposed addition to the CASPER FFT is to allow bitgrowth at each stage.

Figure 4.9 shows the simulated response for a single channel from a multi-tap, Hamming windowed PFB as used in the PAPER (Chapter A) and KAT-7 (Chapter B) systems. The channel widths can be adjusted and are normally chosen to allow channels to intersect at their half-power points, as shown in Figure 4.10. The core is parameterised, allowing the choice of filter and number of channels to be changed at compile time. Figure 4.9 also shows the channel response as additional taps are added. The acceptance test report for the KAT-7 system has verified the channelisation core's operation and found the response to analogue signals to match this expectation. Relevant excerpts can be found in B.6, and a full copy of this report is available from SKA-SA upon request.

4.3.5 Quantisation

Although configurable, the PFB normally produces complex data at 36 bits per sample (optimised for an 18x18 multiplier). Radio astronomy correlators have historically been constructed using significantly fewer quantisation lev-

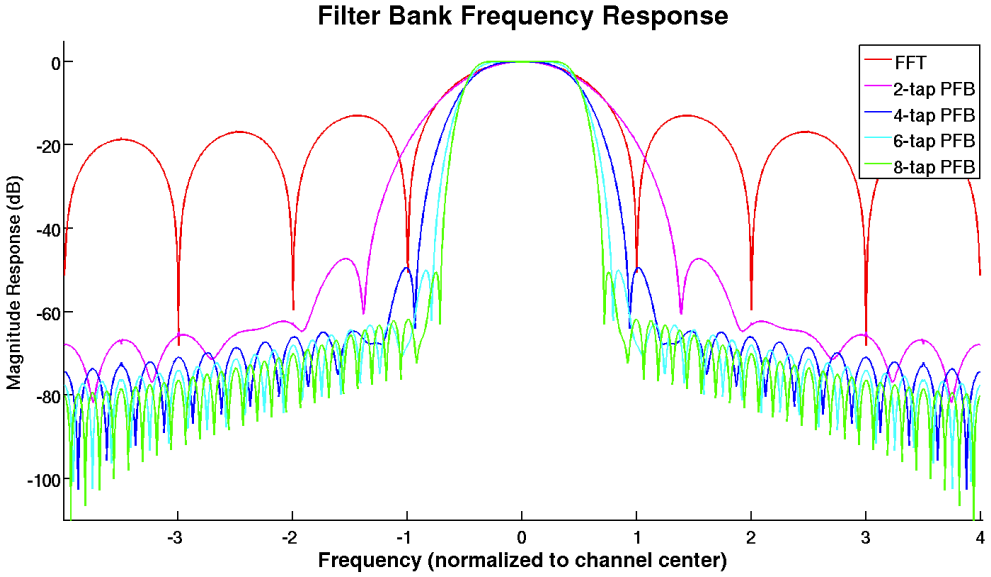


Figure 4.9: The Polyphase Filterbank channel response for various tap settings compared to a standard FFT. PAPER uses a four-tap, and KAT-7 an eight-tap, Hamming window FIR filter together with a 2048 point (1024 channel) FFT to create their PFBs. Out-of-channel rejection is improved significantly with an increased number of taps.

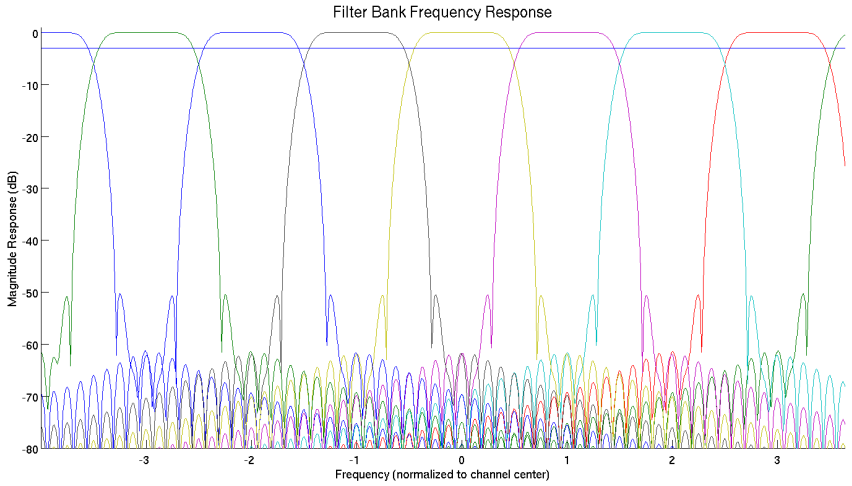


Figure 4.10: The PFB's channel widths are configurable. This figure shows the channel intersections for a 4-tap PFB. Normally, the PFB is configured to allow adjacent channels to intersect at their half power points, thereby preserving total power in the system.

els; the VLA has demonstrated good science with just 1.5 bits. This works by leveraging the statistical properties of a typical wideband astronomical signal, which is noise-like, and so signals are dithered. Temporal averaging then allows much higher spectral dynamic ranges to be achieved. Such systems are able to perform reasonably well in noisy conditions but it does reduce correlation efficiency, suffers from a highly non-linear power scale and a narrow usable instantaneous dynamic range for the signal. Van Vleck and Middleton (1966) showed that the non-linearity errors can be reduced during post processing by applying a correction curve to the measured power levels, a process called *Van Vleck correction*.

RFI is a significant concern when the system's dynamic range is constrained. A filterbank allows for individual single saturated channels to be discarded if they contain narrowband RFI (such as when a satellite passes through the antenna's main beam during an observation) after requantisation. The rest of the band is still usable as the PFB offers high levels of out-of-channel rejection. While it is still advantageous to use as many

quantisation levels as possible to increase linearity and improve the dynamic range (and high dynamic range is required if channels containing RFI are to be used), this must be traded against increased system costs.

Thompson *et al.* (1994) shows that a correlator for typical noise-like sky signals can operate successfully and efficiently with very few levels. With only 4 bits (16 levels), the correlation operation remains 98.8% efficient, provided that the signal occupies all of these levels. Figure 4.11 shows the power response of a 4-bit quantiser to gaussian noise.

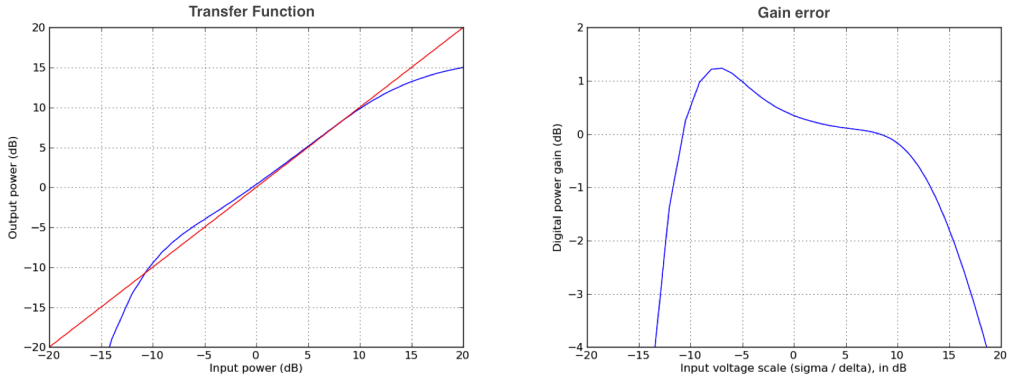


Figure 4.11: The theoretical van Vleck transfer function of a 16-level quantiser to gaussian noise. Reproduced from Herselman (2010).

While the system is parameterised and flexible to allow any number of bits to be processed at compile time, all existing implementations have chosen to requantise the PFB output to a complex representation of 4 bits real + 4 bits imaginary. This saves interconnect bandwidth between boards, reducing interconnect costs by a factor of 4.5 over the 18-bit complex PFB output, while maintaining a very effective system.

This quantisation operation must be performed carefully. Simply upshifting or downshifting the data from the PFB provides a very coarse control of the signal levels, which is insufficient if the average level is to be controlled to remain in the narrow optimum range of 4-bit values. I chose to implement this as a scale factor using a complex multiplier and then always selecting the four most significant bits. This scale factor can be fixed for the entire band, or specified separately for each frequency channel at runtime. In this

way, imperfections in the analogue passbands' amplitude responses can be corrected digitally. Furthermore, if the scale factor is made to be complex, phases can also be corrected.

An additional complication arises from the use of 2's compliment representation. If no input sets a level of 0, then the number of levels to either side of this baseline is not symmetric, ranging from -8 to +7. Should one of these channels be driven into saturation, greater negative numbers would accumulate than positive ones resulting in a negative DC offset. This will significantly skew the correlation results. For this reason, I limit the signal to 15 levels, symmetric ± 7 about zero. The slight reduction in correlator efficiency with 15 levels vs 16 levels is a fraction of a percent and not significant.

4.3.6 Network preparation

The datastream is prepared for transport by the network prior to packetisation for distribution to X-engines. This is primarily a matrix transpose operation to group a selection of time series samples from a single FFT channel into each packet.

The number of time samples in a packet is chosen to match the internal accumulation in the X-engine core, typically 128 (PAPER, KAT-7) or 256 (anticipated for MeerKAT). As will be shown in §4.5, this accumulation length must increase for systems with larger numbers of antennas in order to keep the data rates from the X-engine cores manageable, which in turn increases memory requirements within the F-engines.

This re-ordering requires a matrix-transpose operation, which is typically double buffered, writing into one memory while reading from another in a different order. This requires a memory capacity of $2MP$ for each input, where M is the number of frequency channels and P is the packet size. A new, generic re-order component implemented in the CASPER library by Aaron Parsons allows for in-place corner-turns, halving the required memory capacity. In-place re-ordering requires simultaneous read and write from the hardware which is not supported on all platforms (notably the iBOB's SRAM). My current implementation still double-buffers all data.

The matrix transpose operation on ROACH is complicated by two aspects

when using QDR:

- To achieve maximum throughput on ROACH's QDR, burst operations are required of back-to-back reads and writes to adjacent memory locations.
- Multiple frequency channels must enter and exit this corner-turner in parallel in wideband designs. In the case of KAT-7, this is two frequencies at a time.

To deal with these complications, a three stage transpose operation is required: A large, primary corner-turn with dimensions of $S \times M$, hosted in QDR, followed by a second smaller transpose in BRAM of $2 \times S$ (to account for the 2-operation burst) and a third barrel-shifting stage, also in BRAM, of $P \times S$ (to account for the parallel frequency channel output), where S is the packet size, M is the number of frequency channels and P is the number of parallel frequency outputs.

A difficulty was encountered during the implementation of the generic corner-turn block. The primary transpose operation is typically performed in off-chip memory due to the capacity and speeds required. Off-chip hardware can differ across hardware platforms. The ROACH QDR, for example, supports simultaneous reading and writing while the iBOB's SRAM does not. The interfaces to these hardware devices thus change, making it difficult to construct a single, generic hardware abstracted corner-turn function without compromising performance by adopting the lowest-common-denominator. With this as an example, it seems likely that modifications to existing code will always be necessary in order to take advantage of new hardware features.

4.4 F-X packetisation

I wish to keep all packet headers as small as possible to reduce network overhead while still fully describing the F-engine packet contents for processing by the X-engines. In contrast to the F-engines, the X-engines run asynchronously. They must buffer and re-order packets received from multiple

F-engines before processing can take place. To achieve this, at a minimum, I need to label the packet with a timestamp, source antenna and which frequency(ies) it represents.

UDP over IPv4 is used to transport these packets and so some of this could be done using the network addressing (for example, send each frequency to a different UDP port, as is done at the ATA). A similar scheme could be used with the source or destination IP addresses. However, this imposes significant limitations and as in the example of the ATA, only 16 bits are available for port addressing with some of these reserved for core network services. It is highly likely that this will be insufficient for future systems with more than 65384 channels. It is also sometimes necessary to send multiple antennas' data from a single IP/port combination and so this cannot be used as a flexible identification system.

I would like the packets themselves to be able to identify their contents, irrespective of routing, so that they can be multicast, broadcast or be routed without losing the identification.

So instead, for the demonstration system, I accomplish this with a very simple packet format: a single 64 bit header on each packet (which contains some data for a single frequency channel). Originally, F-engines produced output packets with sequentially incrementing frequency channels. This motivated for the following header scheme: A packet counter formed the first 48 bits of the 64 bit header. Because each packet represents a different frequency channel, the lower M bits of this counter effectively identify the frequency channel in an M channel system. The upper $48 - M$ bits form a timestamp. The remaining 16 header bits are used as an antenna identifier. This single 64-bit header allows the X-engines to identify the data source, sample time and frequency channel. When the output format was changed, as described in §4.5.6, this header format was retained.

It is acknowledged that this hard-coded packetisation scheme as used in PAPER and KAT-7 is inflexible. But it is highly efficient. I envisage that SPEAD, or a similarly flexible protocol, will be adopted for all packet data exchanges in future deployments.

With the packet format decided, I must then choose suitable packet size. Recall from §4.3.6 that I do not wish to individually packetise each timeslice

for a single frequency channel as this would make inefficient use of network resources and so multiple data words are packaged together. I choose to match the number of data words in the packets to the number of accumulations that take place within the X-engine. In this way, one packet from each F-engine is required to assemble a complete window for processing.

Payload sizes can be increased to improve network efficiencies at a cost of increased memory requirements, both in the F-engines' corner-turn and on the X-engines for reassembling a processing window. This would also imply an increased accumulation time within the X-engines, which in turn raises the minimum system-wide accumulation period.

I have chosen a payload size of 128 timesliced data words of 16 bits each (dual polarisation, complex 4 bit data) for all existing deployed systems. Each packet contains a single frequency channel for two inputs (for dual-polarised feeds).

In general, the use of jumbo Ethernet frames (over 1500 bytes long) are desirable for interfacing to ordinary computers. Without interrupt coalescing on the network interface card (NIC), the arrival of each packet creates an interrupt which the CPU must service. At high packet rates, this rate can be high enough to prevent correct operation of the computer as the machine is constantly doing context switches. Jumbo frames allow for high throughput with less frequent interrupts. Modern NICs support a feature called interrupt coalescing which buffers a number of packets before issuing a single interrupt. While this increases latency (not significant for most radio astronomy applications), it also allows for higher processor efficiencies. However, the desire for large packets must be traded against F-engine memory capacity needed for the corner-turner operation described in §4.3.6.

Reducing the header size and increasing the packet's payload size for a given data rate also improves on-wire efficiency and reduces loads on switches and routers (which operate on data on a per-frame or per-packet basis).

Table 4.1 shows the distribution of interconnect resources in the existing 128-word packetised systems.

Choosing a payload size of 2048 bits (128 $\times$ 16 bit words), 64-bit bounded words and a bare-minimum 64-bit header size (to label time, frequency and antenna) results in 99.6% packet efficiency and keeps the packets small

Table 4.1: **Packet utilisation of existing, deployed systems with 128 4-bit complex, dual-polarisation payload sizes and single 64 bit header. Link is used at 76% wire-speed efficiency. With 4 bit quantisation, this allows for a maximum total analogue bandwidth of 950MHz per 10GbE link, or 475MHz of dual-pol data.**

	Bytes	%pkt
Layer1 and Ethernet header and footer	44	13%
IPv4 header	20	6%
UDP header	8	2%
Application header	8	2%
Data payload	256	76%
Total	336	

enough to implement the corner-turn in iBOB or ROACH SRAM. This packet size is a good match for the F-engine hardware capabilities on both the iBOB and the ROACH. The ROACHs' increased compute power can generate finer spectral resolution, which still fully utilises its extra QDR capacity for a larger corner-turner, than the iBOB.

4.5 X-engine operation

Each X-engine in the system operates independently of all other engines in the system and processes a subset of the frequency band. All X-engines are identical and are differentiated only by their logical addresses. The X-engine's primary functions are to:

- receive the packets from the network, re-order and account for missing packets,
- cross-multiply each antenna's data with every other antenna's and
- accumulate the results

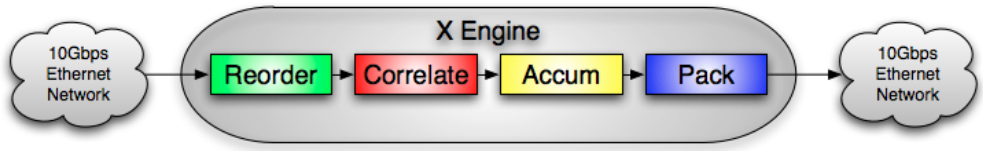


Figure 4.12: A block diagram showing the basic X-engine operations.

4.5.1 The X-engine core

Since every antenna must be multiplied by every other antenna (cross-correlations) and also by themselves (auto-correlations), the number of products (or baselines) for N inputs is $n_{bls} = \frac{N(N+1)}{2}$ and it is an $O(N^2)$ compute problem. A neat algorithm was developed by Lynn Urry to feed the antenna data into a core sequentially and have it calculate all the baselines, with internal accumulation to reduce the output data rates and prevent the N^2 data rate explosion. The core of my X-engine is based on Urry's ATA design, the detail of which can be found in ATA memo 73 Urry *et al.* (2007). Figure 4.13 illustrates its operation.

The operation of this engine is discussed in the 2008 PAPER instrumentation paper by Parsons *et al.* (2008), which detailed the first working packetised system as described here. It can be read for further details regarding the X-engine cores' operation. What will follow in this section is an overview and a discussion around the implementation details not covered in that paper along with some minor extensions that were made to that initial design.

The X-engine cores run at the native clock rate of the FPGA, which is asynchronous to other boards in the system. Since the free-running serialised X-engine cores keep state internally, it is not possible to clock individual data samples into the cores without using more expensive gated logic.

X-engines are instead allowed to freewheel faster than the data rate; they will process data irrespective of the data buffer state or its validity. These cores are purposefully faster than the incoming data rate to ensure that buffers will always underrun. But this implies that the cores' output will not

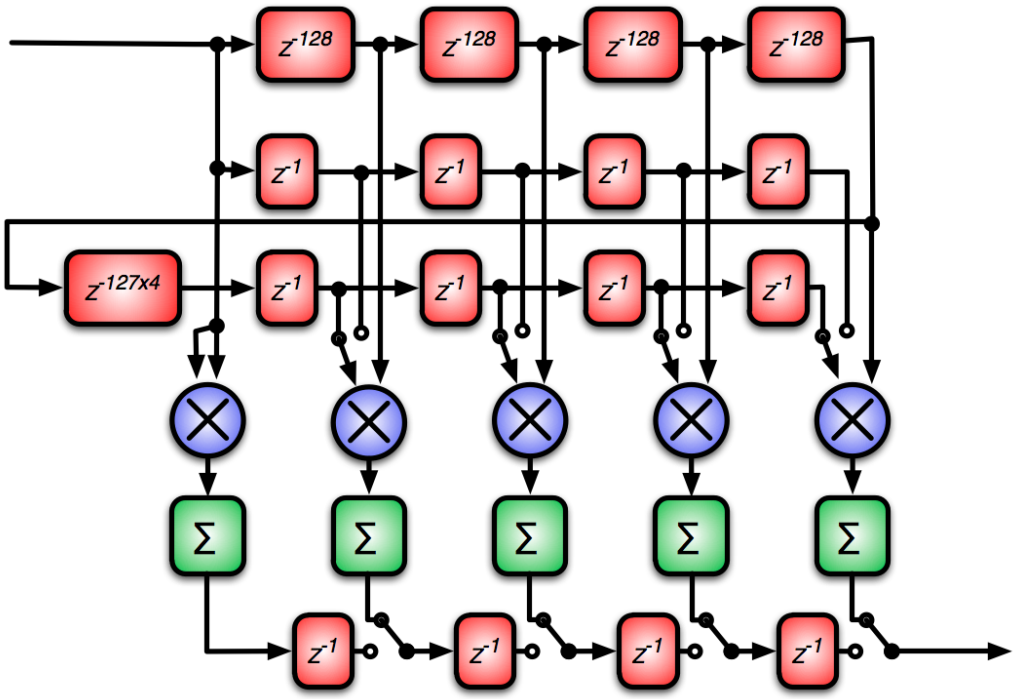


Figure 4.13: An X-engine core showing the $\frac{N}{2}+1$ compute stages for an N antenna system. This example is for an 8 antenna system. Windows of adjacent time-series data for each antenna enter the core and propagate through a series of delays. Each stage in this pipeline calculates a different antenna separation, with a switch to select calculating separations of S or $N-S$, in order to prevent calculation of products across two different windows. Products are accumulated internally to reduce output data rates. The resultant output ordering can be found in Table 4.2. Each multiplier produces valid data on every clock cycle, resulting in a highly efficient design. Further details of this core's operation can be found in Parsons *et al.* (2008) and Urry *et al.* (2007).

always be valid and a control system is required to manage the cores and track valid and invalid data.

The concept of *windowed data* is used, where incoming packetised antenna data is buffered and assembled on the FPGA into blocks of data, or windows, which are aligned with the X-engine core's next window boundary before being processed. This is a much coarser form of gated logic. The X-engines remain free-running, processing windows back-to-back and will process the window whether it's valid or not. The X-engines thus periodically process null, or invalid windows.

Windows are constructed to include a single frequency slice from all antennas. The rate at which the windows are emitted from the buffer for processing is controlled by timestamps of arriving packets. As a packet with a new timestamp enters the ring buffer, a previous window is marked as ready to be shipped. The buffer waits for the X-engine core to start a new window before emitting all the data for the oldest buffered timestamp to be processed. This process is outlined in Figure 4.14 and discussed in more detail in §4.5.3.

In addition to flagging a window as valid or invalid, each window is also tagged as being good or bad depending on whether or not all the packetised data for that frequency channel was actually received. In this way, should an F-engine fail and one antenna's data is not available, the data from all other antennas can still be processed. The goal is to propagate this tag through the entire system and store it as metadata along with the actual data. This has not yet been implemented, and I currently simply monitor the buffers to ensure that no packets have been lost so that I am confident that all valid windows exiting the X-engine are calculated using reliable data.

4.5.2 X-engine core input and output

Each X-engine core calculates all complex baselines, so a core's output data rate would scale as $O(inputdatarate^2)$. However, the X-engines perform a small accumulation internally to reduce the output data rates.

The size of the incoming network packets are chosen to contain the same number of samples as is accumulated within an X-engine core such that only

Table 4.2: This table shows the output ordering for the example 8-antenna core described in Figure 4.13, as used by PAPER-8 and KAT-7. Baselines are shown as numbered antenna pairs. Data is read out from right to left, top to bottom. The core outputs data in a parallelogram shape for a given window. One example window has been highlighted in green. Duplicate data is also produced in the case where the number of antennas is even, highlighted here in red. The lower triangle of data also has the baseline ordering reversed (eg $7,0$ instead of $0,7$). These values must be conjugated to conform to normal astronomy practices of specifying baselines as (j,k) where $j < k$. To correct the output ordering, avoid duplications and ensure correct conjugation, the output data is buffered and corrected before being propagated out of the core.

0,0	7,0	6,0	5,0	4,0
1,1	0,1	7,1	6,1	5,1
2,2	1,2	0,2	7,2	6,1
3,3	2,3	1,3	0,3	7,3
4,4	3,4	2,4	1,4	0,4
5,5	4,5	3,5	2,5	1,5
6,6	5,6	4,6	3,6	2,6
7,7	6,7	5,7	4,7	3,7
0,0	7,0	6,0	5,0	4,0
1,1	0,1	7,1	6,1	5,1
2,2	1,2	0,2	7,2	6,1
3,3	2,3	1,3	0,3	7,3
4,4	3,4	2,4	1,4	0,4
5,5	4,5	3,5	2,5	1,5
6,6	5,6	4,6	3,6	2,6
7,7	6,7	5,7	4,7	3,7

a single packet from each antenna is required to form a single input window. A window is then $N_{ant} \times T_{acc}$ clock cycles long. All output products must be produced and streamed out within this number of cycles to prevent overflows. This places a lower bound on the number of accumulations that must take place within the core.

In general, I aim to keep the output data rates less than or equal to the input data rates. The minimum number of accumulations that must take place to ensure that all data can be read out of the X-engines is $acc_{min} = \frac{n_{bls}}{n_{ants}}$ where $n_{bls} = \frac{n_{ants}(n_{ants}-1)}{2}$. So this simplifies to $acc_{min} = \frac{(n_{ants}-1)}{2}$. Additional accumulations must thus take place inside the X-engine cores as more antennas are added. This increases buffer requirements and also decreases the available output time resolution because the minimum integration time is increased.

The current core is hard coded for dual polarisation data. The two polarisations are input in parallel and each multiplier shown in figure 4.13 is actually a quad, complex unit. Four cross-pol products are thus produced in parallel for every baseline pair. Bit growth also occurs within the X-engine as data is accumulated. The output bit-width for a typical n-bit implementation is then $(2n_{bits} + 1 + b_{growth}) \times 2 \times 4 = 16n_{bits} + 8b_{growth} + 8$.

The output bus quickly grows large; in the case of a 128-word accumulation, b_{growth} accounts for 7 bits of each number, resulting in a 128 bit wide output bus for a 4 bit X-engine. This would impose a requirement on the subsequent long-term vector accumulator for a wide input bus width that practically limits the technology to DRAM (DDR2 or DDR3 typically have busses of 64 bits, allowing for 128 bit words to transfer on every clock cycle) or else requiring the use of multiple, smaller memories in parallel.

Initial VACC implementations (as deployed on PAPER-8) used the BEE2 hardware which featured four 144-bit wide DDR DIMM modules. With such a wide bus available, the X-engine output bus width was not initially a significant design consideration. Each asynchronous X-engine core could be assigned its own vector accumulator which used separate DRAM memory modules. ROACH-2, however, features only one DRAM DIMM. The X-engine cores would need to be synchronised in order to share an accumulator based on this single module.

An alternative approach takes advantage of the two QDR parts, with their narrower bus interfaces, by accumulating for longer than necessary within the X-engine. This further reduces the data rates, allowing the output to be serialised into narrower, but longer, vectors based on the QDR memories. Two asynchronous X-engine cores can then be accommodated on the ROACH board with a separate QDR memory assigned to each. A further advantage is that the QDR interface allows for true local synchronous operation (synchronous within one streaming processing thread). The design of the vector accumulator is greatly simplified over the BEE2 implementation because the QDR interface operates synchronously with the FPGA fabric which results in a simpler vector accumulator design that reduces fabric logic requirements. This is the approach adopted for KAT-7, PAPER-32 and PAPER-64. The four complex cross-pol terms can be demuxed by a factor of up to eight times if the real and imaginary components are also serialised. This approach has the drawback of raising the minimum accumulation period by the same factor. However, this is usually negligible in imaging applications as an accumulation period of many seconds is often specified.

Transient searching and timing applications require careful consideration of these limitations. In these situations, it can be beneficial to leverage the accumulation within the X-engine cores and forfeit a long term accumulator entirely.

The X-engine core requires some tweaking of the output products:

Firstly, some efficiency is lost for cores operating on an even number of antennas, where $\frac{N_{ants}}{2}$ results from the last stage are redundant as shown in figure 4.13. Otherwise, multipliers are 100% utilised and produce valid results on every clock cycle. These extra baselines should be removed from the output products. Secondly, and also shown in figure 4.13, the lower triangle of products produced for a given window are incorrectly conjugated according to astronomy convention, which requires that a given baseline AB^* has $A < B$.

To achieve this, I buffer the output of each window from the X-engine, blank the excess data and conjugate the reversed baselines. This unfortunately raises memory requirements and requires some control logic. An alternative is to just output all data and correct for these effects during post

processing. The imager could discard the excess baselines and conjugate the reversed ones as part of the preprocessing pipeline. This requires a slightly longer vector for the accumulator and a preprocessor that is able to recognise and correct these products. All currently deployed systems have opted to correct these items in the FPGA.

Future work will be to parameterise the core so that any number of parallel inputs can be accommodated, not just fixed at two polarisations. Some additional duplicates occur for auto-correlations in the dual-polarisation case, due to the two-input parallel architecture. For each correlation baseline, with polarisations x and y, the cross-polarisation terms $A_x B_x$, $A_y B_y$, $A_x B_y^*$ and $A_y B_x^*$ are calculated. In the case of auto correlation baselines, where $A == B$, conjugated duplicates occur: $A_x B_y^* = (A_y B_x^*)^*$.

The output of the natively dual-polarised engine can be remapped so that the correlator may be used as a single-polarisation correlator with twice as many inputs. In this case, the aforementioned cross-polarisation duplicates create duplicate baselines in the single-polarisation remapping.

4.5.3 Buffering and re-ordering of packets received by X-engines

Since UDP is not a guaranteed service, data scrambling and non-delivery can cause problems for X-engines. The system is designed to be tolerant of dropped packets and performance degrades gracefully under these conditions, though none are typically observed during normal operation. When packets are received from the network, they are re-ordered and missing packets' data are flagged before being processed.

The re-ordering and buffering mechanism is implemented as a gated circular buffer in on-chip BRAM. The buffer is divided into windows, each of which contains the packets from each antenna for a given frequency, ready to ship to the connected X-engine core.

Arriving packets' headers are interrogated to determine its allocated position within the buffer. Because the number of antennas, number of X-engines, number of frequency channels and number of time samples within a packet are all multiples of 2^N, these addresses can simply be assembled by extract-

ing bits from the timestamp and antenna counter in each packet and these used to generate a memory address within the buffer.

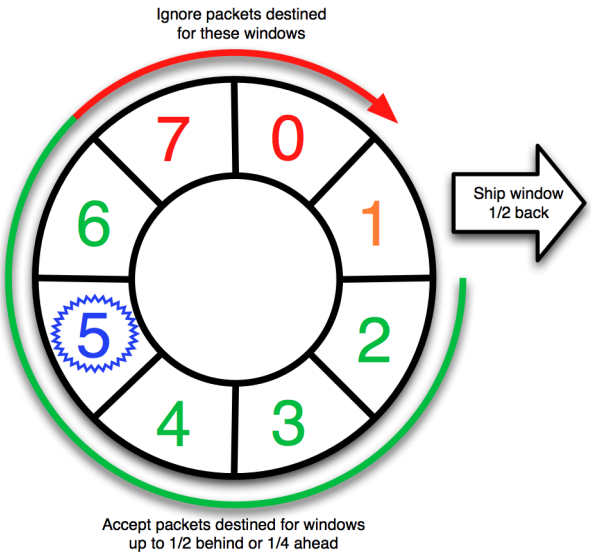


Figure 4.14: This diagram illustrates the operation of the KAT-7 packet re-order mechanism. Eight windows are buffered. When the first packet destined for window five arrives, window one begins readout and the circular buffer rotates clockwise to advance by one window, in order to accept packets destined for windows two, three, four, five and six (half the buffer space back and a quarter of the buffer space ahead, including the current window). Arriving packets destined for other windows are dropped.

To prevent out-of-sync or faulty F-engines from corrupting the buffer, only packets around the current valid timestamp are accepted. This is achieved by forcing incremental advances of the buffer. Packets are accepted for windows 1/4 of the buffer-space ahead of the latest accepted timestamp, and for 1/2 of the buffer-space behind the latest accepted timestamp, including the window of the latest accepted timestamp itself. This algorithm is demonstrated in Figure 4.14.

Every time the buffer advances, the opposite window is shipped, regardless of whether or not all packets were received for that window. This prevents the system waiting indefinitely for packets not received. An extra bit in the

buffer for each packet records if the data was successfully received or not. This is propagated along with the data and to error registers for monitoring data integrity. PAPER and KAT-7 have both demonstrated that systems are able to operate continuously for months without experiencing any lost packets.

Should the system latch on to a spurious timestamp, a timeout ensures that the system resynchronises to a valid packet, thereby preventing lockups.

KAT-7 and PAPER use the 20 port Fujitsu XG2000C 10GbE network switch. Empirical data from KAT-7 and PAPER systems suggests that reorder buffer capacity for only one window is needed when employing such switches based on single ASICs. While packets for a given timestamp arrive out-of-order from each antenna, all packets arrive before the first packet for the next timestamp arrives.

4.5.4 Long term accumulation

The accumulation that takes place within the X-engine cores (and hence minimum output period for the correlator system) is given by $t_{min} = \frac{PM}{B}$ where P is the packet size in words, M is the number of PFB channels and B is the system bandwidth in Hz. With a jumbo frame size of 4096 bytes and 512 channels over 1GHz bandwidth, it is possible to achieve accumulation times of 1.048576ms; fast enough for some millisecond transient detectors to be coupled to the correlator output.

However, this is typically too fast for imaging applications. In these cases, data rates are further reduced by accumulating in a long-term vector accumulator (VACC). External memories are required for this VACC as vector lengths for medium-N and large-N systems are too long to host in internal BRAM resources.

In addition to the issues outlined in §4.5.2, bitgrowth must be considered within this long-term accumulator. The accumulation periods are typically hundreds of milliseconds to seconds. The fact that the input vector is mostly noise-like power data can be leveraged to reduce the number of bits practically required by the long term VACC. Outputting a complex data type of 32 bits real + 32 bits imaginary, allows for accumulation periods in excess of

60 seconds for medium bandwidth designs such as KAT-7.

Early designs, based on the BEE2 hardware, used DRAM for the long-term VACC, however, the design of this logic is complicated by the fact that DRAM must be read and written in asynchronous sequential blocks to obtain reasonable performance. This necessitated the use of input and output buffers with feedback loops which adversely affected FPGA signal routing and timing. ROACH designs opted to use QDR for the longterm VACCs, with fixed timing and synchronous clocking which simplified the VACC logic significantly. While this is possible for smaller designs like KAT-7, the lower-capacity QDR memory is fully-utilised for medium-N applications such as PAPER-64. PAPER-64's increased antenna count, and associated additional baselines, generates much longer vectors.

Since the X-engine cores on each board process received packets as they are assembled on arrival, temporal alignment across engine pipelines cannot be guaranteed, even within a board. That is, while one core is processing a given timestamp, another core might be processing a different timestamp. For this reason, each X-engine core is assigned a separately addressable VACC to store its output. This is simply done on ROACHes (where multiple QDR SRAMs are available) or on BEE2s (where multiple DRAM DIMMs are available).

Should the engine cores need to share a VACC, they can be synchronised simply by aligning their processing windows with a simultaneous reset. Each re-order/descramble block must then hold-back its valid window of data until all re-order blocks have valid data, at which point they all emit them simultaneously. While feasible, it has not, as yet, been necessary to implement this architecture as no practical need has arisen to share a single VACC across multiple engine cores.

4.5.5 Number of X-engine cores required

Each hardware processing board is able to host multiple engine cores; as many as the platform's capacity allows. The engines can share network interfaces to maximise wire speeds. In PAPER and KAT-7's case, each ROACH board is populated with two X-engines which share a single network interface.

The number of X-engine cores required for a given system depends primarily on the bandwidth and the number of antennas in the system, given by $n_X = N\frac{B}{f_X}$ where N is the number of antennas in the system, B is the analogue bandwidth per antenna and f_X is the clock rate of the X-engine FPGAs.

The number of cores required thus scale linearly with the number of antennas. This accounts for one dimension of the $O(n^2)$ compute operation. The other dimension is contained in each core, whose length also grows linearly with the number of antennas (see §4.5.1 and Urry *et al.* (2007); Parsons *et al.* (2008)).

In the case of PAPER and KAT-7, each X-engine processes 200MHz with the X-engines clocked at 210MHz, making the X-engines temporally 95% efficient. KAT-7 uses 16 X-engines on 8 ROACH boards to process 400MHz of analogue bandwidth from 16 inputs. KAT-7's FPGAs are intentionally under-utilised (see §B to reserve processing capacity for future instruments, such as the beamformer, which may be co-located on these boards. PAPER-16 uses eight X-engines on four ROACH boards to process the 100MHz analogue bandwidth from the 32 inputs.

4.5.6 Allocation of data to X-engines

Each engine processes a subset of the frequency band. These sub-bands can be arbitrarily allocated to the available processing engine cores. Two allocation systems are currently in use, both based on a round-robin scheme but with a different channel ordering:

- Interleaved X-engine channels, as illustrated in Figure 4.15. This was used in PAPER and early GMRT designs. Frequencies are distributed to the next available X-engine in the order that they are emitted from the FFT core. Each X-engine thus processes a comb of channels throughout the spectrum.

- Contiguous X-engine channels, as illustrated in Figure 4.16. This was used in KAT-7 and later GMRT designs. Frequencies are re-ordered

out of the F-engines so that each X-engine processes a contiguous band of the spectrum.

In both cases, all F-engines simultaneously emit packets for a single frequency channel and all those packets are queued by the switch for output to the desired X-engine. The switch's buffer needs to be large enough to cope with all these packets. The average data rates must also be kept below the linerate so that the switch's output queue is emptied in time for the next allocated frequency channel to be buffered for that X-engine. This process is illustrated in Figures 4.15 and 4.16.

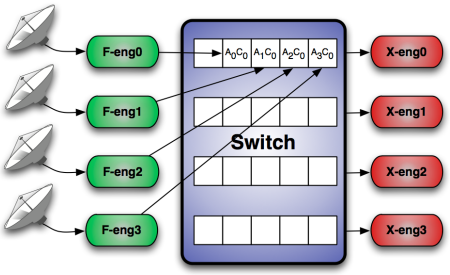
Should the switch have insufficient buffer capacity to simultaneously hold a packets for all X-engines' allocated frequencies, the F-engine outputs can be staggered. This has not yet been necessary in any existing deployments.

The round-robin technique relies on the fact that there are always more frequency channels than there are X-engine cores. My implementation further imposes a limitation that the number of frequency channels is cleanly divisible by the number of X-engine cores, with both the number of channels and the number of X-engines be multiples of 2^N to allow the destination address to be calculated simply by bit-slicing an integer representing the current frequency channel. This restriction does not adversely affect the F-engine as the PFB benefits from efficient implementation with a 2^N number of channels. However, the other restrictions are unfortunate for many telescope facilities as this effectively restricts the number of supported antennas to a multiple of 2^N.

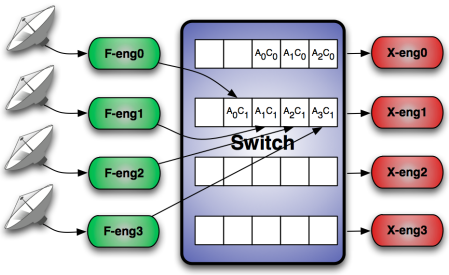
The system can be made to operate with non-2^N multiples and non-divisible number of X-engines at the cost of some efficiency. In the case of the X-engines, some would not be operating at full capacity. All current implementations have 2^N multiples for all elements of the design and there is no plan to implement an alternative until such a need arises.

4.5.7 Loopback implications

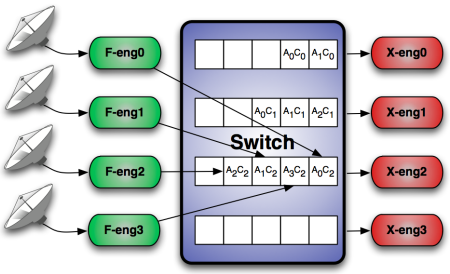
Systems optimising for lowest switch port counts, by routing F-engine data through the X-engine boards (§4.2.4), require some fraction of this data to



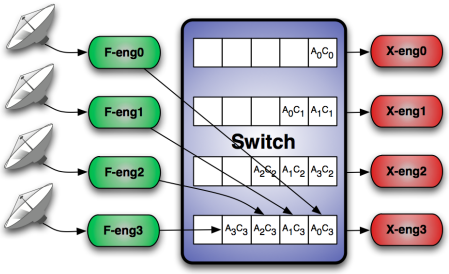
(a) The first packet from each antenna, containing channel 0, is routed to the first X-engine. The switch buffers all packets and queues them for output.



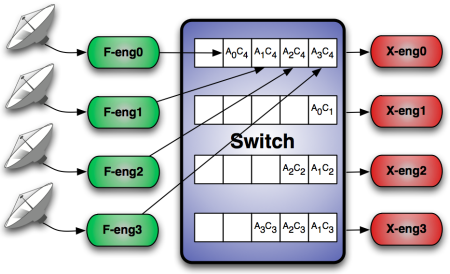
(b) The second packet from each F-engine is routed to the second X-engine. Meanwhile, the first X-engine's buffer begins to empty.



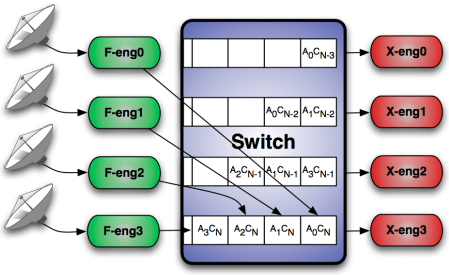
(c) Packets are queued by the switch in arbitrary order.



(d) The routing process continues with the fourth packet from each F-engine.

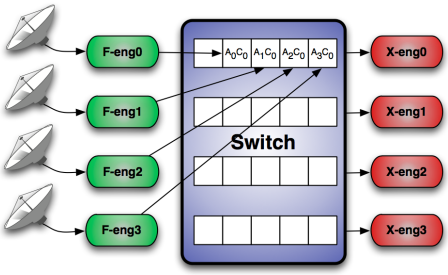


(e) By the fifth channel's packets, the first output queue will be empty, ready for the next channel's packets.

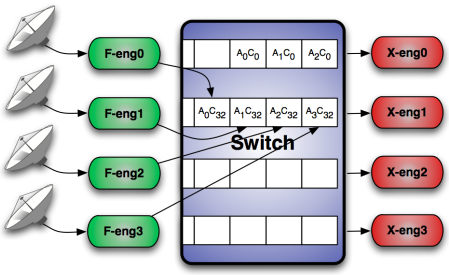


(f) The last channel in the spectrum is distributed to the last X-engine.

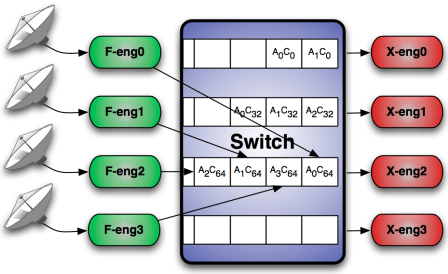
Figure 4.15: Round-robin packet routing for a four X-engine, N-channel interleaved system. Each processes a comb of channels, eg. X-eng0 processes channels 0, 4, 8 ... N-3, while X-eng1 processes channels 1, 5, 9 ... N-2.



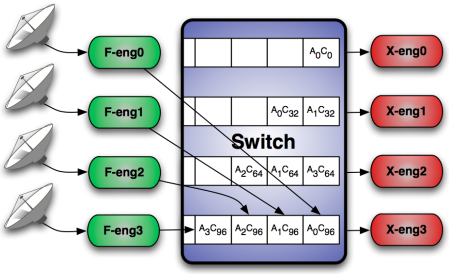
(a) The first packet from each antenna, containing channel 0, is routed to the first X-engine (just as in the interleaved case).



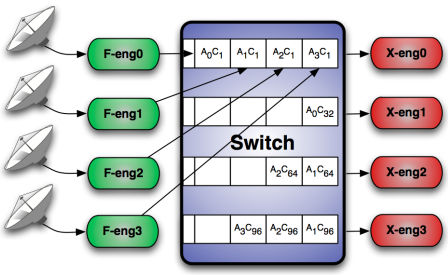
(b) The F-engines re-order their output to comb through the spectral channels; channel 32 follows channel 0. It is routed to the second X-engine.



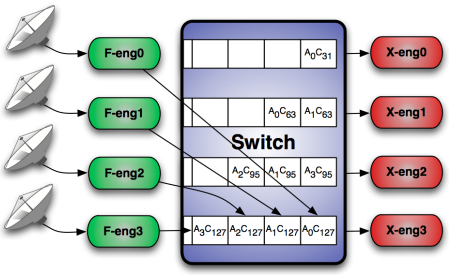
(c) The third X-engine processes spectral channels 64 through 95.



(d) The fourth X-engine processes spectral channels 96 through 127.



(e) The first X-engine is issued with its second channel, channel 1.



(f) The last channel in the spectrum is distributed to the last X-engine.

Figure 4.16: Six snapshots in time for a 128-channel contiguous routing system with four X-engines. X-engines process a contiguous band; eg. X-eng0 processes channels 0, 1, 2 ... 31 while X-eng1 processes channels 32, 33, 34 ... 63. Compare with Figure 4.15.

be processed on the local X-engine board. This data must be retained locally as Ethernet switches do not forward packets destined for the source.

This functionality is implemented as a small FIFO buffer on each X-engine which keeps copies of all packets destined for its own IP address. Naturally, this data will be available sooner than packets from other F-engines which must be routed through the network switch. For this reason, packets are only read out of these loopback FIFOs when similarly timestamped packets are received from the network from other antennas. This wait-before-readout ensures that large volumes of memory are not required by the general-purpose packet re-order and buffer process, as all antennas' packets now arrive at similar times.

A simple mechanism for achieving this is to compare the timestamp of the next packet in the local FIFO buffer with the incoming stream from all other F-engines. As soon as a later timestamp is noted in the incoming stream, the packet is released from the local buffer. This also allows for automatic resynchronisation should the buffer contain obsolete data as it will be flushed as soon as the newer (later) timestamps arrive. However, should the X-engine receive a bad, future-dated timestamp from the locally connected F-engine, it will block the FIFO. This is simply worked-around by adding a timeout that flushes a packet from the buffer if no packet has been emitted in a reasonable period. This period is determined by the speed of the network and number of antennas in the system (since this affects how long the cycle time is between timestamps). Network testing determined that latencies through 10GbE switches are typically significantly less than 10 microseconds, and so for modestly sized arrays, a $\sim$1ms timeout is more than sufficient to prevent unwanted flushing while still allowing for rapid resynchronisation in the event of a failure.

4.5.8 X-engine FPGA output

Early designs transmitted their accumulated output products by reading shared memory from the PPC using BORPH, packetising them on the PPC using a non-documented protocol and transmitting these through the boards' control and monitoring Ethernet ports. This proved to be a slow mechanism,

unsuitable for arrays requiring short accumulation periods (such as KAT-7), or with long output vectors due to many baselines, or large number of frequency channels (such as MeerKAT).

For this reason, a hardware output block was developed using the standardised SPEAD protocol to transmit the accumulated products over the existing 10GbE links on each X-engine. The output bandwidth is generally a small fraction of the data rates exchanged between F- and X-engines. Recall from §4.5.4 that the accumulation period's lower limit is determined by the internal accumulation inside the X-engine cores. For KAT-7, the correlator itself is capable of 0.33ms accumulations. But accumulations at such short periods are not practically realisable because the receiving computer is unable to capture and store the data at the increased data rates. The fastest practical accumulation period possible on KAT-7 is in the order of 20ms.

4.6 Adding a beamformer: the B-engine

While a single time-domain beamformer could be added by daisy-chaining spare data ports from the F-engines (as shown in Figure 2.3), constructing multiple beams in this way is expensive as each beam would need to be propagated to every board, and each beam requires a separate delay line filter.

But multiple frequency-domain beamformers can be added easily and cheaply to the X-engines, since it can be made to operate on the same, existing incoming datastream. This involves duplicating the X-engine core input data to be processed by a B-engine core. Since this data is already serialised, the hardware cost of the beamformer is a single complex multiply and accumulate (CMAC) along with a lookup table for the beam steering coefficients. Naturally, supporting logic is also required to manage the lookup addresses and control synchronisation etc.

By leveraging the existing correlator infrastructure in this way, multiple coherent beams can be generated cheaply. Fringe rotation and delay tracking of the primary beam is already performed in the F-engines for the correlator, requiring additional beamformer beams to simply be steered at an offset from this primary beam. A further advantage is that the full correlator

is operating in parallel allowing for fast beamformer calibration and rapid source tracking.

4.6.1 The B-engine core

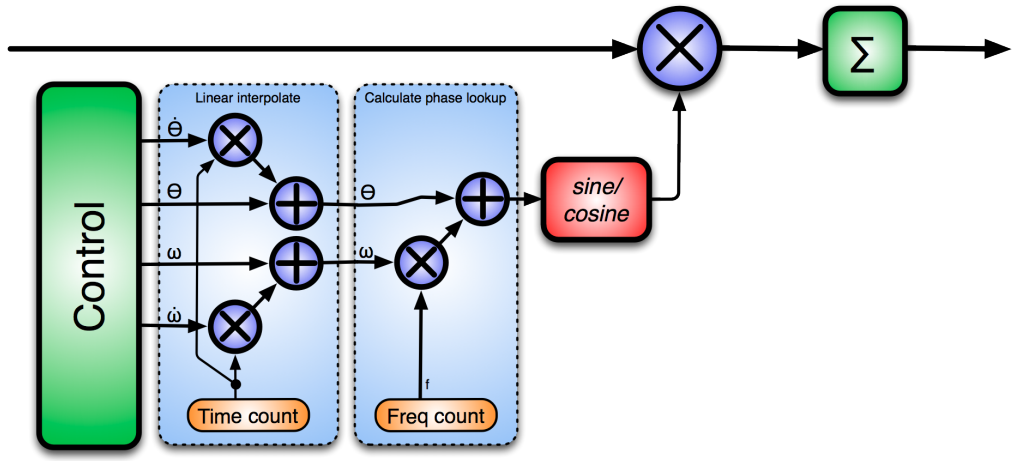


Figure 4.17: The B-engine core is very similar to the fine delay block. The incoming serialised datastream is multiplied by a CW tone before each frequency channel is summed across all antennas. As in the fine delay/fringe rate operation, the control logic allows for programming initial phases and frequencies as well as the rate of change of these two variables. This linear interpolation in hardware reduces the rate at which coefficients must be updated from software.

For every B-engine core placed alongside the X-engines, an additional beam can be formed.

A boresight beamformer has been successfully implemented to demonstrate this functionality. While the beam is not independently electronically steerable in the X-engines, it can be steered using existing delay logic in the F-engines. Should independent control be desired, the same logic described in §4.3.3 can be copied to the B-engine and placed before the summation, as shown in Figure 4.17.

The summed beam is transmitted by merging the stream with the X-

engine vector accumulator output (which is at a much lower data rate) and sent together, over the existing 10GbE link. Thus, no changes to the hardware, architecture or interconnect are required.

4.6.2 Limitations

The primary limitation to this shared, commensal approach is that the beamformer is forced to use the correlator's requantised input datastream. This is typically only 4 bits from each antenna, which is quite sufficient for correlation operations where the primary interest is not to distort the signals' phase, which is less sensitive to quantisation. The instantaneous dynamic range in the 4-bit case is significantly limited to 24dB. This must be considered, especially in light of transient events and terrestrial narrowband RFI sources.

The 4-bit, 24dB range is post-PFB, where narrowband RFI signals will already have been confined to polluted frequency channels by the PFB, and can be ignored (assuming a PFB with good channel isolation). In general, incoming signals from natural astronomical sources are wideband and noise-dominated. Provided the observer is not interested in observing narrowband objects with large dynamic range, the available (limited) dynamic range can be carefully adjusted to allow for linear observation of the sky signal while allowing the ignored, terrestrial narrowband RFI channels to saturate. This is typically sufficient even for transient pulsar work: while the dedispersed pulses are bright, the dispersion spreads the impulse power temporally, resulting in smaller overall instantaneous power level changes.

Furthermore, if a reduction in temporal resolution can be accommodated (such as for slow transient work), the dithering effect of this noise can be leveraged, by accumulating the output, to generate greater averaged dynamic ranges, albeit with non-linear transfer functions.

There is also bit growth through the B-engine core's summation operation as antennas are added to the beam, which can increase the summed beam's dynamic range. Even for small arrays such as KAT-7, bitgrowth due to summation of the beam results in a reasonable output bit-width ($4 + 3 = 7$ bits for the 8 input KAT-7). For larger arrays, such as MeerKAT, this

growth is significant and the beamformer output is requantised back down to 8 or 16 bits. Practically, this does not necessarily improve the system's dynamic range as all antennas are observing the same signal(s). A large burst of wideband RFI is likely to saturate all antennas' signals, each of which only has the aforementioned 24dB instantaneous dynamic range.

A secondary consideration is for beam steering in the frequency domain. Recall that, with the fine delay, it is only possible to correct over a range of $\pm\pi$ radians. Figure 4.18 shows that a frequency-domain beamformer that employs only fine delay phasing will lose efficiency as it is steered far away from boresight. Small steering angles, even beyond 1 sample period, can be accommodated.

4.7 Timestamping

A distributed synchronisation pulse is used to synchronise F-engines across multiple boards. This is usually supplied by a GPS-disciplined source (eg. crystal, rubidium or hydrogen maser) in the form of a one pulse per second (1PPS) output. Upon initialisation, the system is armed and all F-engines then wait for a 1PPS pulse. When the next 1PPS pulse is received, master counters (64-bits in current implementations) across all F-engine boards are reset. A control computer keeps track of when this initialisation occurred, and hence the actual time represented by a counter value of zero. The computer's system time is synchronised by NTP or some other external mechanism. Care must be taken to arm the F-engines well away from a second boundary to prevent F-engines triggering on adjacent 1PPS pulses.

Because F-engines are running synchronously, this counter increments simultaneously across all boards. They are used to timestamp the data out of the F-engines and provide a means for reassembling data in the correct order in the X-engines. The 48-bit packet counter (see §4.4) is derived from this master counter. The packet counter will eventually wrap if the machine is left running for extended periods, and so this wrap must be tracked and anticipated by software using the equation $t_{wrap} = \frac{2^{48}}{f_{fpga} \times n_{pkt}}$. Processing 100MHz bandwidth, with packet sizes of 128 values, allows a system employing a 48-bit counter to run for 11.4 years before the counter wraps. In the case

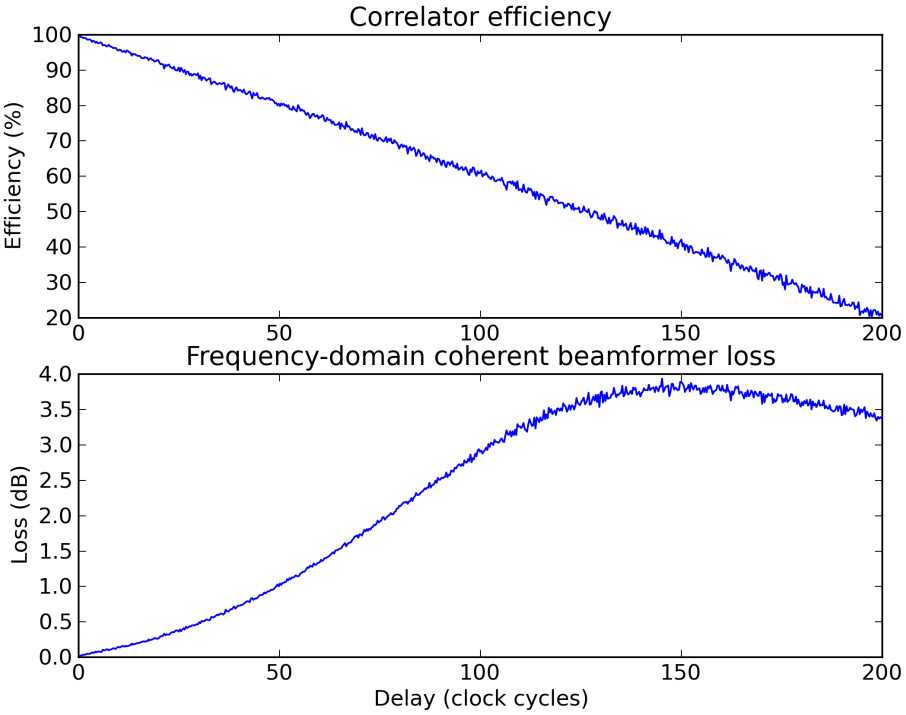


Figure 4.18: A frequency-domain beamformer that only steers beams by adjusting the phase of the PFB's output (no coarse delay adjustment) will lose effectiveness the farther the beam is steered, as the signal begins to decorrelate. In this simulation, a 128-channel (256 point) real PFB channeliser is employed in a noisy two-input system. The upper plot shows the correlator instrument's response where the signal decorrelates linearly down towards 0% when the delay difference increases to 1 PFB period (256 clock cycles). The lower plot illustrates the output beam power for the frequency-domain beamformer, described here, against a perfect beamformer. Small steering angles can easily be accommodated, with little signal loss. But losses increase until, by half a PFB period (128 clock cycles), the beamformer is completely ineffective and the 3dB gain, that was to be had by using two receivers, is completely lost. For delays of 1/2 to 1 PFB period (128 to 256 clock cycles), the signals are summed out of phase and results are worse than just summing pure noise. At delays of 256 clock cycles, the signals are completely decorrelated and so the beamformer has no gain and the loss tends to 3dB.

of a very wideband system, such as the SMA with 5GHz sample rates, this could wrap in months. Since it is unheard of for single observations to run continuously for such extended periods, this is not considered a significant limitation.

Locating these counters early in the signal chain, in the F-engines, ensures that the system remains synchronised and can recover from faulty or lossy data links (though not from bad sampling clock distribution). The first prototype system employed the loopback mechanism (described in §4.2.4) and point-to-point XAUI links to the X-engines. The timestamping operation was performed, not on the F-engines but on the X-engines' incoming XAUI interfaces. If these XAUI data links dropped data words then the X-engine would no longer be synchronised with the rest of the system and would need to be re-armed to lock onto a 1PPS pulse. It would correlate misaligned data. This was experienced when too-long data cables were used and was observed in recorded data as variable delays in some baselines that periodically stepped by a few clock cycles. Timestamping in the F-engines avoids this problem.

Timestamping should occur as soon as possible, as far up the signal processing chain as possible. Details of these early systems and the lessons learnt are discussed in §A.2.

4.8 SPEAD

A need existed for a standard data format to allow exchanges between components in the processing pipelines. This format needed to be implementation agnostic to allow for use between FPGAs, conventional compute nodes and software processes.

The Streaming Protocol for the Exchange of Astronomy Data (SPEAD) was developed in collaboration with the SKA-SA and PAPER teams as a standard interface for such exchanges. SPEAD can be used for on-the-wire exchange, for on-disk storage or for piping data between application processes within a single compute node.

It is a one-way, best-effort, self-describing protocol, containing both machine and human-readable descriptions. This allows receivers to automatically unpack data so that application designers can concern themselves with

their algorithms rather than the tedium of data exchange. It also allows for data sources to comment their datastreams so that stored data can be reinterpreted again at a later date or to easily debug data exchanges.

SPEAD complements KAT's existing guaranteed service protocol, KATCP, which is used for all control and monitoring aspects of the telescope facility, from building sensors such as HVAC and door locks to weather stations, analogue signal chain components and DSP devices.

SPEAD leverages a number of concepts from FITs, MIRIAD and other existing astronomy standards. It is designed to exchange arbitrary data structures and aims to keep receivers' copies fresh by propagating dataset changes to the receiver as these changes occur on the transmitter. Receivers are required to keep state so that the values of all variables within the structure are always available, as only the changed subset is subsequently received.

As a one-way transport, and because it operates over UDP on Ethernet networks, multicasting is supported transparently, allowing multiple devices to subscribe to datastreams.

While being highly flexible and supporting features such as runtime alteration of variable sizes and shapes, these can lead to inefficient implementation on the receiver as memory must be reallocated on-the-fly for new structure formats. Practical deployments have dispensed with this feature so that a given implementation has a static, predictable structure. And so, lightweight implementations are possible when all features are not required. This can ease implementation on FPGAs and other simple hardware devices.

SPEAD's primary drawback is due to its streaming nature and arbitrary, potentially changing, datastructure. Since it is not indexed, when used as on-disk storage, the dataset must be scrubbed when attempting to jump to specific timestamps or other indices. For this reason, SKA-SA has adopted the HDF-5 file format for on-disk storage and archiving, with SPEAD being used for inter-device data exchange only.

An open-source reference design implementation is available in Python along with a simple unpacker plug-in for Wireshark[3] to aid in debugging network exchanges. A limited-functionality, GPU accelerated subset has also been implemented in C and in 2011 demonstrated successful streaming of

[3]http://www.wireshark.org/

over 12Gbps into a single compute node. Two different FPGA transmitters (synchronous and asynchronous with flow control) also exist for the CASPER DSP library. Links to the reference implementations, the specification document and a full description can be found on the CASPER wiki site at `https://casper.berkeley.edu/wiki/SPEAD`.

4.9 Verification and testing

The evaluation of system performance is critical for gaining confidence in the system's ability to produce high quality data. The system deployed on KAT-7 was most rigorously tested to asses functional and performance envelope metrics. These included such aspects as accurate timestamping, channelisation characteristics, fringe stopping and delay compensation accuracy and internal interpolation stability, digital efficiency (related to quantisation noise) and repeatability.

In keeping with System Engineering principles, the Acceptance Test Procedure (ATP) document describes the procedures used to bench-test the correlator systems prior to deployment, many of which are applicable to testing correlators in general, and the Acceptance Test Report (ATR) contains the results obtained during the test. The tests were passed and the system was successfully deployed on KAT-7 in 2010. The ATR was revised and regenerated in 2012 after beamforming and narrowband capabilities were added, and the system again passed.

Both documents as pertaining to KAT-7's Digital Back-End are available from SKA-SA on request (van Rooyen and Rust, 2012, 2011).

4.10 Conclusion

In this chapter, a correlator instrument was successfully implemented on FPGAs and interconnected using a commercial 10Gbps Ethernet switch. Later, a beamformer was added that ran concurrently with the correlator, and shared its channelisers. Existing CASPER DSP libraries and processing platforms were leveraged for the primary PFB and multiply-accumulate

operations.

A number of Ethernet idiosyncrasies were dealt with in order to obtain a practical and useful implementation, such as Ethernet not returning packets to senders. Mechanisms were also developed for resynchronising from packet headers that cleanly handled missing or incorrectly-ordered packets.

This correlator instrument was initially deployed on PAPER (Appendix A) and later on KAT-7 (Appendix B). It was also trialled at GMRT and at Medicina. The design has been successfully compiled for iBOBs, BEE2s and ROACH platforms, and deployed for various numbers of inputs (16 to 128) and bandwidths (32MHz to 400MHz) and spectral channels (512 to 8192).

Chapter 5

Design analysis

This chapter discusses some of the lessons learnt from PAPER and KAT-7, while identifying limitations of the architecture and implementation. Possible solutions are proposed where problems are identified. The architectural, hardware and software lessons learnt here have been fed into the design of MeerKAT (Appendix C) and the next generation hardware platform, ROACH-2 (§5.5.5).

5.1 Architecture and proof of packetised concept

Packetised locally synchronous, globally asynchronous (GALS) designs are flexible, reliable and inherently fault tolerant. By keeping local (on-chip) operations synchronous, hardware feedback loops are kept short and control logic simple. This allows for fast, efficient operation.

By allowing processing engines to operate asynchronously, hardware connections and required clocking infrastructure is reduced. It also allows processing nodes to be easily exchanged for alternate technologies without requiring specialist clocking interfaces, as is being considered with GPU X-engines (see §A.4.2 and Appendix C).

However, this modularity and scalability does come at a cost, especially on FPGA platforms which are quite capable of coping with simpler synchronous systems. Synchronous systems can avoid packet buffering, re-ordering and

flow control logic. GPUs and other microprocessor architectures are inherently asynchronous and include hardware for flow control functions.

There are also concerns for network security and consequences of link errors. This section will consider the implications of packetised interconnects on the system.

5.1.1 Asynchronous receivers

Asynchronous, faster-running, free-wheeling data-consumer cores (as the X-engines are implemented) need flow control. This requires additional control logic over a purely synchronous system. To minimise timing impact, feedback loops must be kept short and localised as closely as possible around the function in question. In general, realtime radio astronomy signal processing operations are streaming in nature and have few such feedback loops. So this architecture works well.

This model fits well with the modularised approach adopted, with each operational component on the FPGAs considered as separate, tightly-packed cores, avoiding feedback loops (backpressure) between these modules wherever possible. Latencies are easily inserted between cores, wherever needed, in order to meet timing closure requirements. Where back pressure is needed, it is loosely coupled so that the system can cope with inserted delays into the feedback path.

While the inter-board connections are asynchronous, within the FPGAs, muxing asynchronous dataflows is expensive due to handshaking overhead and required feedback loops. For this reason, I try to avoid muxing of asynchronous streams and to keep linear, synchronous dataflow, wherever possible, within the FPGA.

5.1.2 Error rates

Since the employed UDP protocol is a best-effort service and no provision is made for retransmission of lost data, there is an understandable concern for corrupted, dropped or undelivered packets. The 10GbE links themselves implement a coding scheme (8/10 encoding) that allows for a limited error correction by the receiver.

The Ethernet frames, IP headers and UDP datagrams incorporate simple checksum verification for detecting corruption. While these are not infallible, I believe this mechanism offers sufficient error detection capabilities for my purposes. Laboratory testing of PAPER's 32-input correlator over a continuous 3 day period revealed no UDP checksum errors or dropped packets during this time. PAPER has further evidence of operating their system in the field for over 3 months without any detected corruptions or losses.

This places a limit on the achieved post-coder bit error rate to be better than 1 in 1×10^{-18}.

It is questionable if such low error rates are even required. Since the processed sky signal is mostly noise-like, the system can tolerate additive noise, which can be modelled as an increase in system temperature or decrease in system sensitivity, provided this added noise does not statistically affect the signal over time (for example, skew the phase or change the Gaussian nature of the received noise). A budget for bit errors can then be calculated based on system design parameters.

5.1.3 Network security

Ethernet is natively robust and a downed network segment mostly does not affect other segments. Certain failures, can, however, affect the rest of the system depending on how the network has been architected. These can become security risks.

Consider this example of a localised failure that could affect the entire instrument: If one ROACH board loses its network configuration and so begins to broadcast packets to the default broadcast MAC address (FF:FF:FF:FF:FF:FF) at a high rate, a Layer-2 switch will flood the entire network with high speed data which could overwhelm receivers, especially those systems connected via slower Ethernet links (1Gbps is common for control and monitoring). This failure mode has been demonstrated in the laboratory. Since UDP offers no flow control, it is unable to throttle the faulty receiver and all connected nodes' buffers will overflow, losing valid data coming from operating nodes. One solution to this problem is to employ a layer 2+ switch that offers broadcast control and will throttle misbehaving inbound ports. Denial-of-service

type attacks are also possible by flooding simple FPGA devices.

Coupled with possibilities for IP spoofing and the general lack of authentication, it is my opinion that any packetised radio astronomy instrument should be firewalled and isolated from the public internet and placed in a well-controlled environment with due diligence paid to the network configuration of any additional machines introduced into the network. This is especially significant for attached ROACH devices, which do not feature robust security mechanisms or the large network receive buffers of typical PC systems, and so are easily overwhelmed if incoming traffic exceeds local processing capacities.

5.1.4 Debugging and monitoring packet exchanges

Debugging tools for interrogating Ethernet packet exchanges are already commonplace in network engineering. This is a significant advantage for the packetised construction approach as standard computers can be used to sample the network traffic and so debug board data exchanges. Data can also be easily copied to hard disk for offline analysis, without needing any specialist interface hardware beyond a common computer with an Ethernet port.

5.1.5 Simplified interfacing to adjacent systems

With Ethernet ports available as standard on most processing systems, including laptops and commercial rack-mount servers, interfacing this machine with other telescope elements (drive controls, weather stations, RFI monitoring systems etc) for data exchange and control is greatly simplified.

As KAT-7 has demonstrated, a single protocol (in this case KATCP) can be used to control many systems, and provide a unified interface across multiple subsystems. An example of such an exchange can be found between the correlator and subsequent imaging pipeline, where a feedback loop is employed to configure fringe and delay compensation once the desired astronomical source has been selected.

SPEAD has also simplified data exchange throughout the entire telescope-wide pipeline, with all subsystems using a standardised data format, allowing

for modular insertion of signal displays throughout the signal chain.

5.2 Cost

A key research question in this work is *What does the system cost, and what is the cost penalty for an Ethernet interconnect?*

The use of mass-produced hardware, such as commodity network switches and off-the-shelf CASPER hardware, allowed for economies of scale to be leveraged. Even small systems are able to benefit and leverage the low cost-per-element of much larger systems.

The hardware costs of the network switch itself, the FPGA resource utilisation and the development time spent on developing the Ethernet subsystem must all be considered. Let us now consider these costs more closely.

5.2.1 Switch costs

Analysing the cost of the PAPER-32 system presented in Appendix A, Table A.3, I find that the instrumentation cost is USD41631 + USD20400 if FPGAs were purchased. USD11300, or 18%, of this total cost is for the switch.

But not all ports on the switch are used. This skews cost statistics for small systems, such as PAPER-32 which uses only four of the 20 ports on the switch. This same switch is able to host up to 128 PAPER antennas (16 ports used), whose instrumentation is projected to cost USD133000. On such a system, the switch contribution to cost reduces to less than 9% of the overall instrumentation cost. On KAT-7, which uses 16 of the 20 switch ports, the switch represents 11% of the system cost.

Taken to the opposite extreme, for very small systems where only two ports are used on the switch, the fractional cost of the switch would be 38% in PAPER's case. But in this case where only two ports are in use, it would make more sense to plug the ethernet cable directly from one unit to the other unit, without a switch, and so this extreme can safely be ignored. Given 2^n scaling limitations, the lowest number of ports in use on a switch is then likely to be four. PAPER-32, thus, represents a worst-case costing scenario.

I conclude that the switch represents between 9% and 18% of the system

cost, depending on the fraction of the switch ports in use. In all large systems such as MeerKAT and SKA, where the switch module size is small in relation to the total number of required ports, the switch contribution should be below 10%.

In general, as with the correlator compute cost, the interconnect cost scales as N^2 for N ports (antennas). The fraction of purchase cost that the switch contributes thus remains constant at under 10% as the system scales to larger numbers of antennas.

Note that this is assuming copper interconnect; the cost of any optical transceivers that the implementer elects to install would need to be added. Fortunately, the number of cables (and/or transceivers) scales linearly (§5.3) and so this cost remains small. The N^2 interconnect scaling is contained within the switch.

5.2.2 Switch power consumption

Switch power requirements are small when compared to the rest of the digital back-end's power consumption. In the smaller PAPER and KAT-7 systems, power consumption is dominated by the single control computer and the switch represents less than 10% of system power. For larger systems, the compute processors' power requirements begin to dominate as the same control computer is able to cope with a much larger system.

1-RU 10GbE switches rated at 4W per port are now common, whereas these FPGA compute nodes typically consume 70W (it should be noted that this could be lowered with more efficient power supplies). The switch thus represents 5.7% of the digital signal processing system's power consumption. Since the scaling requirements for compute and switching are both $O(N^2)$, this 5.7% figure should remain constant for larger systems.

5.2.3 Ethernet packetisation resource costs on FPGAs

The on-chip costs for implementing Ethernet on an FPGA are also important when costing the system. Let us again consider PAPER-32 and KAT-7 from Appendix A and B.

BRAM and logic requirements for Ethernet and packetisation operations remain mostly constant for different sized arrays, but later generation devices offer ever increasing capacities. Additional bandwidth is also typically available over faster links on later devices, but it requires the same logic to use those links, irrespective of the line speed. So, the relative size of the resources to support Ethernet on FPGAs becomes ever smaller and will eventually be insignificant compared to the DSP operations on larger devices.

Let us now consider the resource requirements for the pack and unpack logic components, as well as the Ethernet cores themselves, in more detail.

Transmitter packetisation logic

The streaming nature of the design makes the packetisation operation simple; a header is inserted periodically into the data stream to demarkate packets and add timestamps and other metadata. Thus, the overhead for implementing packetised data exchanges is small. On KAT-7's F engines, this represented less than 2% logic and needed no BRAM or DSP slices.

Asynchronous receiver packet re-ordering

Resource requirements for unpacking, buffering and re-ordering Ethernet UDP packets are much higher than the transmitting cores, as receivers must cope with potentially dropped or re-ordered packets. Buffer space is allocated for this task, and FPGA logic is applied to re-order any out-of-order packets.

On PAPER-32, this represented 3% logic and 6% BRAM. On KAT-7, this represented 5% logic and 3% BRAM. Both of these systems were implemented on ROACH-1 boards, but they use slightly different unpack logic, making a direct comparison impossible. PAPER elected to use a larger buffer window than KAT-7 and explains the discrepancy in BRAM resource requirements.

Ethernet transceiver core

Each connected node requires an Ethernet and UDP core to handle the OSI layer 1, 2 and 3 header generation, packetisation and error handling.

A possible system-level optimisation could be to co-locate the F and X-engines on a single FPGA. In this case, only a single 10GbE core and buffers would be required for an F-X combination (rather than one on the F-engine and another on the X-engine) and so some logic could be saved. In the general case, however, and especially for larger systems, the number of boards required for F and X operations are unlikely to be equal.

On ROACH-1, PAPER-32 used 8% logic, 7% BRAM for the Ethernet transceiver core. KAT-7 needed 8% logic, 3% BRAM. KAT-7's core was optimised to reduce BRAM memory requirements. Neither transceiver core used the hardened CRC cores on the Virtex-5 FPGA, but rather, each performed this operation in user logic, thereby unnecessarily increasing the transceiver's logic requirements.

Scaling and conclusion

As with packetisation logic, the size of the Ethernet core remains constant even as device and application size increases. I expect that later generation devices will use smaller fractions of the FPGA resources for packetisation and communication, and that this will soon dwindle into insignificance as future devices continue to increase in size.

The 10GbE core, together with all associated packetisation and re-ordering logic, account for less than 8% of KAT-7's F-engine (transmit only) and 13% of the X-engine (transceiver) logic resources. For MeerKAT, the Ethernet overhead will shrink to less than 2%.

Ultimately, I consider this to be a small cost to pay for the flexibility afforded by the Ethernet interconnect and expect these fractions to be even smaller on later generation devices.

5.2.4 Reduced development time and cost

Reducing development time is highly desirable. For many one-off systems (such as radio astronomy instrumentation), the total contract cost can be dominated by NRE. By spending fewer engineering hours developing the system, total cost can be directly lowered.

Perhaps more significantly, it allows for the latest technology to be used (by freezing technology selection as late as possible), which brings all the lower cost and power benefits of more modern devices and lengthens the effective lifetime of the deployed system.

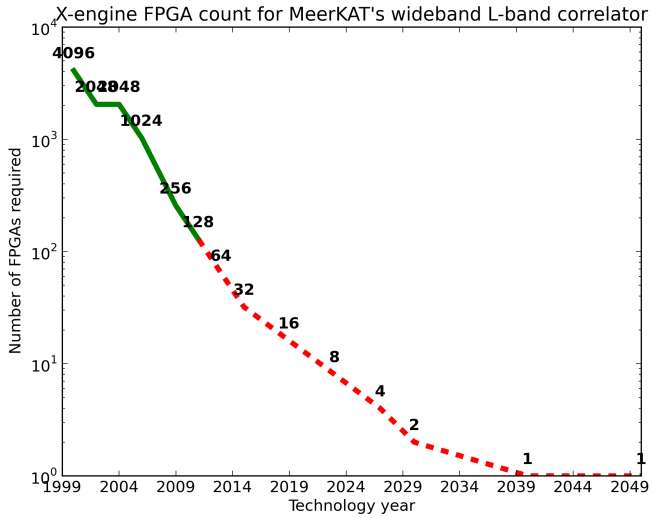
The system described in this work was developed and deployed rapidly. By leveraging the existing CASPER DSP libraries, correlator reference designs, hardware and other IP, this parameterised, packetised correlator was designed and deployed for use by PAPER in less than 12 months, during 2007. Once the design was complete and design difficulties overcome, modifications and later deployments happened more quickly. It was adapted for the Northern Cross and deployed in Medicina in 10 days. For KAT-7, the design was ported to ROACH and the required features added in less than 3 months, though it took a further 3 months to complete the acceptance tests before the system was in general use on KAT-7.

5.2.5 Total cost of ownership

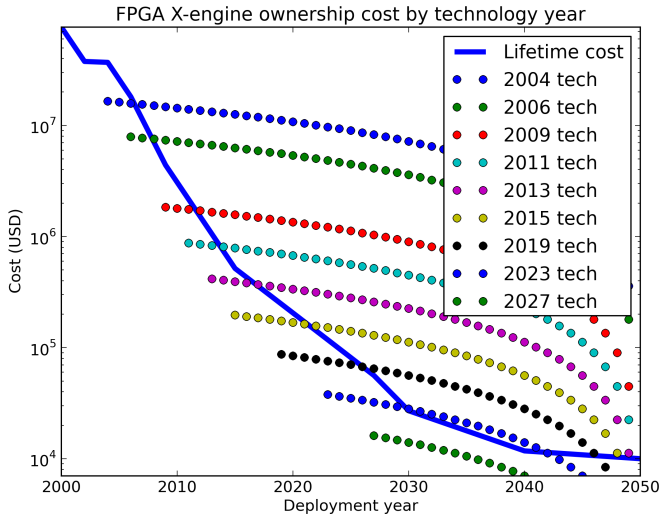
The purchasing cost does not reflect the full price of the system. It needs to be operated over the lifetime of the telescope. For long-lived systems, this operating cost may well dominate the total cost of the system.

Figure 5.1 shows MeerKAT's X-engine total cost of ownership for various years of technology, if based on FPGAs. To derive these curves, I assume the following simplistic model and representative values:

- Electricity costs are constant at USD0.15/kWh over the lifetime of the system.
- Cooling 100W requires 33W of power for the cooling system.
- The latest-generation FPGA hardware always costs USD10000 each.
- The latest-generation FPGA hardware always consumes 100W.



(a) The number of FPGAs needed for MeerKAT's X-engine, projected forward to future generation devices.



(b) The total cost of ownership for MeerKAT's X-engines, implemented on the latest available FPGA technology.

Figure 5.1: TThe total cost of ownership for MeerKAT's X-engines, when implemented on FPGAs. The left figure trends the FPGA technology curve to predict the number of FPGAs needed for each generation device. The right figure's solid line shows the total cost of ownership to purchase the latest generation technology, and operate it until 2050. The dotted lines represent the remaining operating costs for selected technology years to 2050. Each dot represents one year. The intersection points thus indicate where the remaining operating costs for a given technology generation equal the combined purchase and operating costs of a more modern technology.

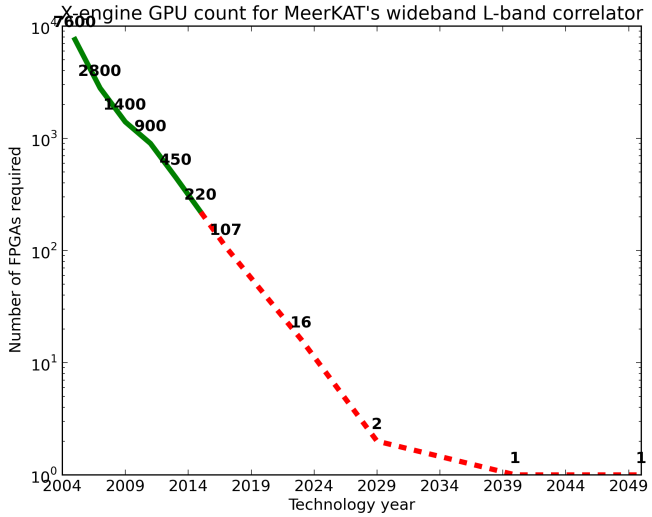
I observe the following:

- From the trending blue line, I can conclude that there is a significant decrease in total costs if later technology is deployed. Because fewer boards are needed when using later generation technology, the initial purchase cost is lower and operating expenses are much reduced.
- From the increasing number of dots to the left of the solid blue line with later generation devices, I conclude that later technology lasts longer. Because it uses less power and the telescope is nearing end-of-life, we need to think about upgrading less often. Or, considered from another angle, we need to operate a more efficient system longer in order to break even. The remaining ownership cost of old technology represents an ever smaller fraction of the purchase cost of newer technology.
- From the leftmost curve, it is predicted that just two FPGAs will be able to perform all of MeerKAT's X-engine signal processing needs by 2030.
- There is little point upgrading the system after 2025, as the purchase price of the new hardware exceeds the cost of operating 2025-model equipment to 2050.

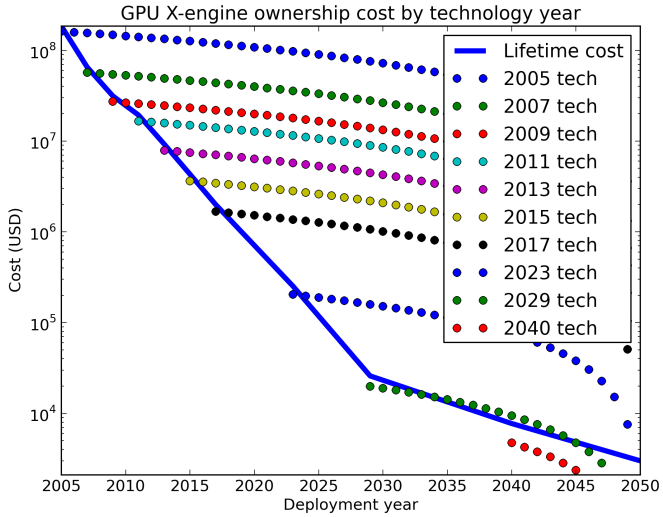
Given these cost assumptions, if MeerKAT were to deploy and operate 2013 equipment until 2050, the total ownership cost would be USD1 054 872. If a single technology refresh were to be deployed in 10 years' time (2023) and the old equipment simply discarded, total cost of ownership would be USD869 947, a saving of nearly 18%.

It is interesting to consider how this compares with GPUs. Figure 5.2 shows MeerKAT's X-engine total cost of ownership for various years of technology, if based on GPUs. I assume the same conditions as with the FPGA case, but with the following GPU costs:

- The latest-generation GPU hardware and host system always costs USD3000 (a dual GPU node costs USD6000).
- The latest-generation GPU hardware always consumes 270W per GPU node (a dual GPU node draws 540W).



(a) The number of GPUs needed for MeerKAT's X-engine, projected forward to future generation devices.



(b) The total cost of ownership for MeerKAT's X-engines, implemented on the latest available GPU technology.

Figure 5.2: The total cost of ownership for MeerKAT's X-engines, when implemented on GPUs. The left figure trends the GPU technology curve to predict the number of GPUs needed for each generation device. The right figure's solid line shows the total cost of ownership to purchase the latest generation technology and operate it until 2050. The dotted lines represent the remaining operating costs for selected technology years to 2050. Each dot represents one year. The points where the dotted lines intersect the solid line thus indicate where the remaining operating costs for a given technology generation equal the combined purchase and operating costs of a more modern technology.

I note that the GPUs in general have a shorter lifetime. This is to be expected since GPUs consume more power and cost less than FPGAs and are thus cheaper to replace regularly.

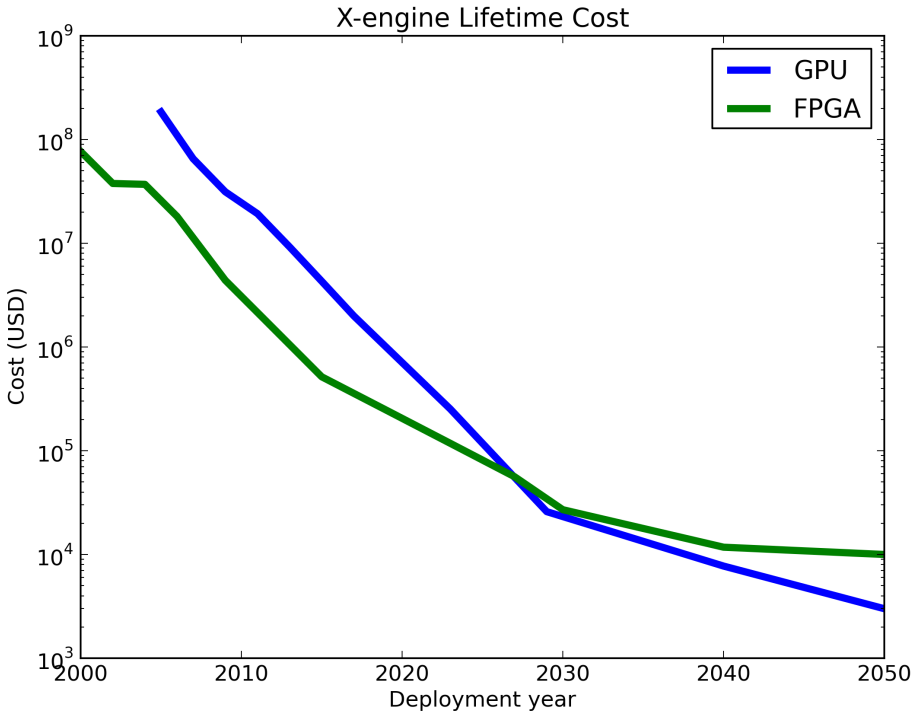


Figure 5.3: The lifetime cost (purchase + operating costs) for MeerKAT X-engines implemented on GPUs and FPGAs to operate a given technology generation to 2050.

From Table 5.3, it is clear that the total cost of ownership of an FPGA-based system is significantly lower over a typical telescope's lifetime than competing CPU or GPU-based solutions, if the system is deployed and operated as originally designed. However, if the system is periodically upgraded using the latest processing nodes available (as should be done to save costs), then the argument for FPGA-based processing elements may change in favour a GPU solution. It is difficult to predict which technology will be optimal over the lifetime of the telescope, as the technology curves cannot be accurately predicted.

None of these models factor-in practicalities, such as time taken to develop and install the new system or the cost of any associated downtime. Also, the models for cost, power and cooling could be refined, and the system then optimised for minimal total cost of ownership. Electricity costs are likely to rise, passive cooling technologies using circulated water could lower cooling costs significantly, and the picture is not yet clear how silicon technology will continue to scale with regards to power and cost, or if another semiconductor substrate technology will change the cost picture significantly.

However, I can safely conclude that, irrespective of the deployed technology, operators should budget for technology refreshes, as these will lower the total cost of the system over its lifetime.

5.3 Scalability

Scalability is a key requirement of this design. Although the architecture is conceivably scalable for any number of antennas, channels and bandwidth, implementation choices made throughout the design have placed practical limitations on various aspects of the scalability. That said, I believe that the current system is sufficiently scalable for all telescope systems on the horizon.

The scalability implications for the correlator implementation are discussed in this section, along with identified architectural limitations.

The linear scaling of each component of this FX correlator (F-engines, X-engines and the number of switch ports) makes the system highly scalable. Adding additional antennas simply requires additional processing boards and a larger switch to accommodate them. Crucially, the number of required switch ports scales linearly with the number of antennas (ie the $O(N^2)$ routing problem is constrained within the switch).

For the purposes of comparing digital correlators, four metrics are of primary importance:

1. the number of antennas,
2. the processed bandwidth,
3. the spectral resolution and

4. the bit-width used to represent the data.

Figure 5.4 illustrates the scaling of the system as each of the processed bandwidth, spectral resolution and number of inputs metrics are changed individually. Changing these metrics has scaling implications for each component in the system, which will be discussed in the following sections.

5.3.1 F-engines

F-engine scaling is generally linear with the number of antennas in the system. The F-engine pipeline is simply repeated, unchanged, for each additional input, except for possible growth within the corner-turn operation due to larger packet sizes (in order to accommodate additional accumulation within the X-engines; see §4.5.2). Each board operates completely independently of any other.

The data rate from each F-engine is determined by the processed bandwidth and quantisation (number of bits used to represent the data) and is independent of the number of spectral channels. While this may seem initially counter-intuitive, CASPER's streaming PFBs and FFTs produce an output sample for every input sample. Thus, for an increased spectral resolution, the PFB grows in length but output sample rates remain equal to input sample rates, as fewer spectra are output per second. Network switch requirements are thus not affected by spectral resolution (ie number of FFT channels) requirements. The bit-growth through the FFT due to multiply/add operations is ignored here, since this is requantised to a fixed, user-defined output resolution. The output data rates are, natually, directly proportional to this bit-width.

However, the compute scaling requirements for this increased spectral resolution are more complex as memory, adders and multipliers do not scale at the same rate. Primiani *et al.* (2011) has analysed the CASPER filterbank implementations and has shown that the scaling is nearly linear for all cases: for spectral resolution, memory is $O(N)$ and adders and multipliers are $O(logN)$, while adders and multipliers scale $O(NlogN)$ with bandwidth.

After the filterbanks, data is transposed before being distributed to the X-engines. The size of this memory is given by $capacity = 2MPQ$ where

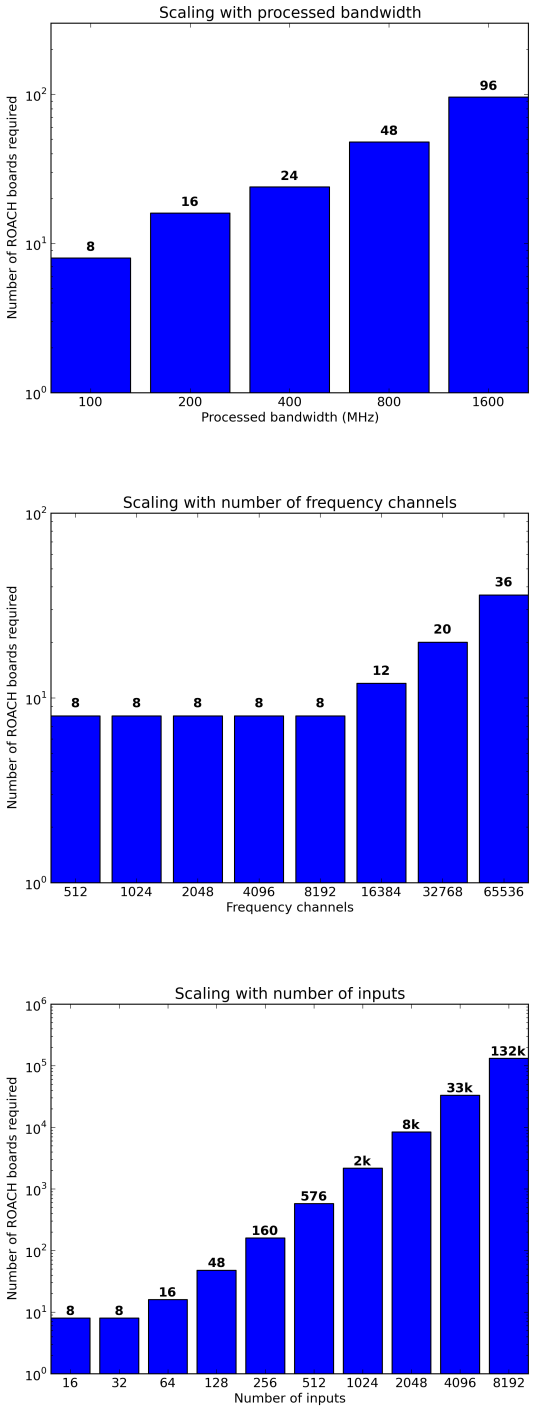


Figure 5.4: Hardware requirements for scaling of a baseline, 4-bit, 16-input, 100MHz, 512 channel system as each of the processed bandwidth, spectral resolution and number of inputs metrics are changed individually.

M is the number of spectral channels, P is the chosen packet size and Q is the number of bits per sample. This matrix transpose operation's size thus also scales linearly with the number of channels in the system and is independent of processed bandwidth. It is possible to optimise this memory usage by performing an in-place re-order of the data. But a given hardware platform will eventually reach a limit of how many channels an F-engine can process due to memory limitations. Beyond this point, the PFB operation must be partitioned across processing elements. No attempt has been made to enable such a split because current boards are already capable of higher spectral resolutions than are typically required. Indeed, it is often the case that sufficient resources exist to co-locate multiple F-engines on a board (for example, KAT-7 hosts two while PAPER hosts 16).

Although not directly dependent on the number of antennas in the system, the number of antennas does indirectly affect the required size of the transpose memory by dictating a minimum packet size, because the X-engine cores require a longer time (more internal accumulation) to process additional inputs in order to keep their $O(N^2)$ output sample rates below the $O(N)$ input sample rates (see §4.5.2 for details). In the case of PAPER, the iBOBs will no longer have sufficient SRAM in which to perform the required matrix transpose when PAPER reaches 128 antennas, at which point the packet size must be increased to allow for this additional accumulation within the X-engines. By this stage, PAPER will already be performing in-place re-ordering (as opposed to double-buffering) and no further memory optimisation will be possible. The number of inputs will then need to be halved, data further quantised, or a larger (eg ROACH) platform employed for the F-engines. This limit does not exist for the ROACH series of boards as they feature DRAM memory in addition to the QDR SRAM. This would support a corner turn of millions of frequency channels across jumbo packets, allowing for scaling well beyond any proposed arrays, including the SKA.

The interface speed required of this external memory, however, does scale directly with the digitised bandwidth. Data must be written-to and read-from this memory simultaneously. Naïvely, this is currently double-buffered so that while one portion of memory is being read, another is being written. Some hardware (such as the iBOB's off-chip SRAM) does not allow the mem-

ory to be simultaneously read and written and so at a minimum, two memory chips must be used. Others, such as the on-chip BRAM or ROACH's off-chip QDR are dual-ported and allow for simultaneous reading and writing and so a single chip can be used. DRAM allows for much wider data busses (64 bit DDR on ROACH 1 and 2), supporting wider bandwidths.

5.3.2 X-engines

The X-engine compute requirements scale with the number of baselines (given by $\frac{N(N+1)}{2}$ for N antennas, including all autocorrelations) and is thus an $O(N^2)$ problem. For larger numbers of antennas, the X component of the FX correlator quickly dominates the overall computation requirements. The CASPER X-engine core decomposes this computational problem into two linear dimensions: one increases the size of each X-engine core and the other increases the number of X-engine cores required. Thus, for every doubling of the array size, both the number of X-engines doubles and the length of each engine doubles. This would require ever larger FPGAs for bigger systems.

At some point the length of each X-engine and the onboard memory requirements will outstrip the capacity of a processing board's FPGA and peripherals. The streaming nature of the X-engine core architecture allows the engine to then be split across multiple FPGA boards, allowing increasingly large designs to be constructed. However, this has not yet been necessary. By closely tracking technology progress, larger FPGAs become available to coincide with array upgrades and antennas construction schedules. The same design can be recompiled for the latest hardware platforms which allows the array to leverage technology advances, thereby reducing costs and decreasing power consumption in the digital system without redeveloping the intellectual property (IP). The PAPER array, discussed in §A, is one of the fastest growing telescopes and is a driver for ever larger platforms. PAPER has already progressed through two hardware generations as PAPER-GreenBank migrated from BEE2 based X-engines (Parsons *et al.*, 2008) to ROACH-based X-engines, described in §A.2 and §A.3, respectively. As processing capacity increases with each generation, I have not yet needed to partition processing elements across multiple FPGAs. As of 2011, ROACH-II is already in pro-

duction, catering for arrays with up to 256 inputs even though no such need exists yet. I expect this trend to continue into the foreseeable future, where technology progression outstrips telescope construction.

While the length of each X-engine core increases with the number of antennas, the width of each engine scales directly with the number of bits employed, as this determines the width of the multipliers and accumulators required. However, this cost is generally cheap on modern FPGAs, with devices offering onboard multipliers wider than 18 bits while correlator systems typically require only two to eight bits. Thus, changing from 4 to 8 bits has a negligible effect on the X-engine cores but a significant impact on the buffering requirements and system interconnect, which would need to transport these higher data rates between boards.

The number of required X-engine cores is given by $N_x = ceil(\frac{NB}{f_x})$ Where N is number of antennas, B is the processed bandwidth from the F-engines and f_x is the clock frequency of the X-engines. The length and number of X-engines is independent of the number of frequency channels in the system. The number of X-engines scales linearly with the processed bandwidth (though it is a step-function, integer multiple of the FPGA clock rate). Additional X-engines must be introduced to accommodate an increased system bandwidth if the X-engine cores cannot be clocked faster.

A further limit is currently imposed that N_x and N be multiples of 2^n due to addressing schemas employed throughout the FPGAs to save logic resources. This restriction allows addresses into buffer memories and destination IP addresses, for example, to be generated simply by bit-slicing the binary timestamps. While the system is fundamentally capable of any integer number of inputs and X-engines, this would come at a cost of increased complexity and has not been implemented in current systems, primarily because there has not yet been a need for it.

5.3.3 Interconnect

The interconnect scales proportionally to the data rates to and from each engine (and thus proportional to the analogue bandwidth). While the switch complexity itself scales as $O(N^2)$ due to the full-crossbar routing required,

the number of ports required is linear as more antennas are added and is independent of the number of spectral channels.

The current implementation has all F-engines outputting a packet for a single X-engine destination simultaneously. The switch must then buffer all these packets and emit them from a single port to the appropriate X-engine. While that is happening, the F-engines emit their packets for the next X-engine and so forth until they come back around to the first engine, at which point the switch's output queue for this first port must have emptied to be ready for new packets and an egress link sufficiently fast to cope with the aggregate bandwidth. This requires large buffer sizes in switch. While no existing implementation has exhausted a switch's buffer capacity, it is foreseeable that this buffering could become a limitation. Should this become a problem for large designs, the readout order in the F-engine corner turner could simply be altered to offset outputs so that each F-engine is sending to a different X-engine at any given moment in time. This would ensure that only a single connection from any given ingress port would exist to any given egress port at any one moment in time.

On KAT-7, processing 400MHz on 16 inputs requires 16 switch ports. PAPER uses the same model switch for 128 inputs with a reduced bandwidth of 100MHz, together with the loopback optimisation described in §4.2.4. Single-ASIC, monolithic 1-U switches with 64 10GbE ports are already available, which is sufficient to host a small to medium sized array of modest bandwidth (for example, 64 antennas at 400MHz, with the loopback optimisation). For very large systems, it is unlikely that a single, monolithic switch will be able to host the entire instrument. It is thus important to be able to construct large, distributed switches from smaller units. §5.3.3 discusses this in more detail. Such scalable, modular switches with thousands of ports are already available commercially which would be capable of hosting even the largest envisaged instruments, such as the SKA.

Scalable switches

A key requirement of the scalable architecture proposed here is a central full-crossbar non-blocking Ethernet switch, ideally with multicast support.

The largest commercial chassis units available off-the-shelf as at 2012 (such as the Arista 7508E at 288 40GbE ports) host less than the estimated 300 ports required for MeerKAT. For this reason, it is important to consider mechanisms for constructing larger switches.

It is possible to build larger switches using smaller units as building blocks. Charles Clos formalised this structure in the 1950s for the purposes of constructing circuit-switched telephone networks(Clos, 1953). Figure 5.5 demonstrates this basic architecture. For circuit-switched telephone networks, it is important that existing circuits not be interrupted when new calls are made. To ensure full-crossbar, non-blocking switching under these conditions, $m \geq 2n - 1$ for a spine constructed from m switches of n ports.

For the purposes of bi-directional packetised computer networks, the Clos network can be folded to produce a *Fat Tree* architecture, as shown in Figure 5.6. In packetised networks, which can accommodate packets being rerouted along a different route, the non-blocking requirement relaxes to $m \geq n$.

The total number of switches in a folded 3-stage Clos network is then given by $\frac{2p}{s} + \frac{p}{s} = \frac{3p}{s}$, where $p \leq \frac{s^2}{2}$ for a total of p ports constructed from s port units.

When using practical 64-port switches as building blocks, a 3-stage Clos network will thus scale to 2048 ports, which is more than sufficient for MeerKAT's needs. Thereafter, 5, 7, 9 or any other odd number of stages (not detailed here) will allow further scaling to SKA sizes and beyond.

There are complications with the aforementioned scaling equation in keeping links to integer numbers. Consider trying to construct a 300 port switch from 32-port switches. The formulas above suggest that $\frac{3p}{s} = \frac{3\times300}{32} = 28$ switches are required. The ingress/egress layer would consist of $\frac{600}{32} = 18.75$ switches, so 19 are required. To interconnect these, $\frac{300}{32} = 9.375$ so 10 should be required. However, 10 doesn't divide into 19 cleanly. Whereas each ingress/egress switch should have 2 (1.9) links into the middle layer, only a single link to each of the 19 ingress/egress switches can be accommodated by each of the 10 middle-stage switches due to port count limitations (32, whereas $2 \times 19 = 38$ is required for two links). This means that 19 middle-layer switches are now also required, giving a total switch count of 38; a significant increase over the expected 28.

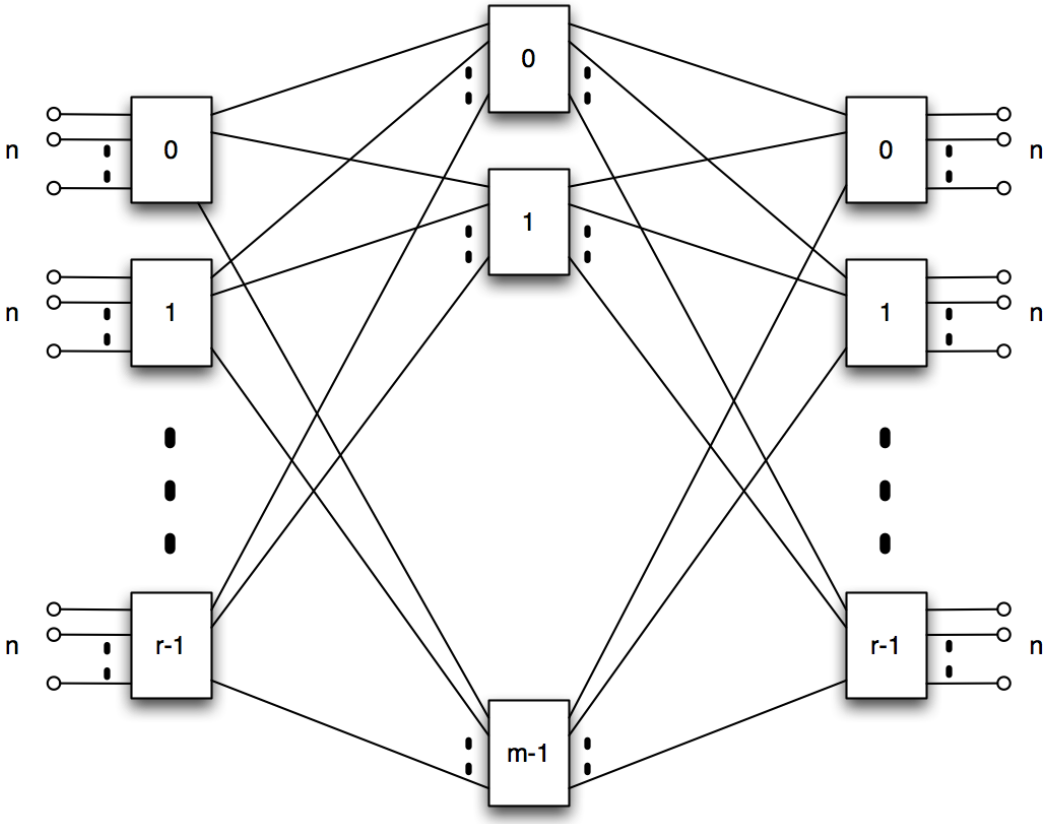


Figure 5.5: A Clos crossbar network switch, constructed from smaller crossbar switches. In a circuit-switched network, in order to ensure full crossbar, non-blocking capability, with the added proviso that existing connections not be rerouted, $m \geq 2n-1$. This figure shows a 3-stage Clos network, consisting of n ingress ports on each of the r ingress switches, m intermediate switches of $2r$ ports each, and n egress ports on each of the r switches in the egress stage.

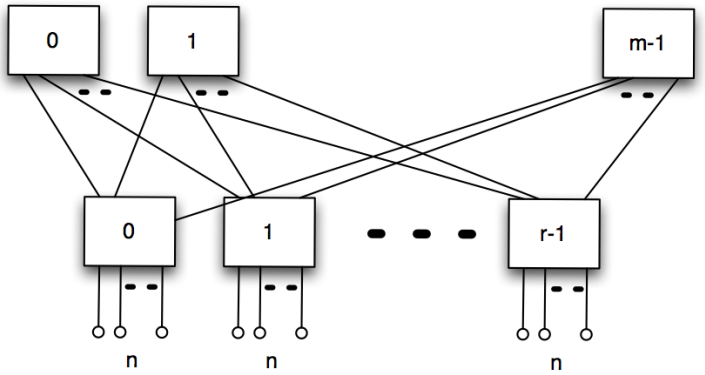


Figure 5.6: A folded Clos network, also known as a fat tree architecture.

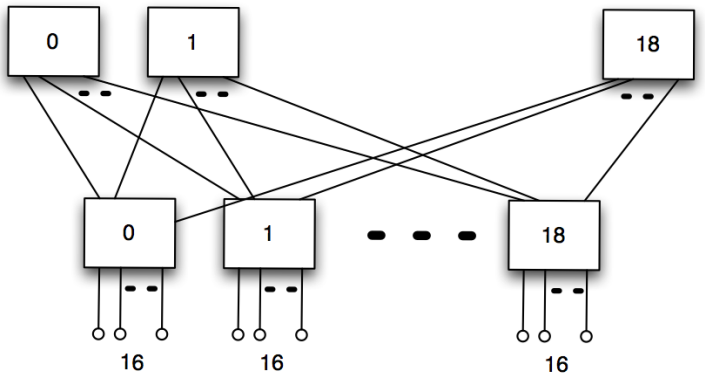


Figure 5.7: A 300 port Clos network constructed from 32 port units. The design does not achieve perfect utilisation: of the 32 ports on each middle layer switch, only 19 are used.

Routing traffic over multiple paths

In the Clos network, the ingress switches have multiple output ports through which packets destined for other egress switches can be routed. The non-blocking prediction (when $m \geq n$) only ensures that there is a possible non-blocking route for packets from any ingress port to any egress port, but does not attend to the issue of how this route is determined.

How do ingress switches decide on which of the multiple possible egress ports to forward a given packet?

Broadly, there are two options: randomly choose one, or deterministically decide which to use. In the case of simple layer-2 Ethernet switches, each switch only has localised knowledge of port buffer statuses and capacities. Some proprietary solutions have attempted to enable communication between switching nodes to exchange such congestion status data, but these solutions are not universally deployed. Without a wider view of the network, it is impossible for switches to intelligently select the end-to-end route with the lowest congestion.

Alternatively, if the traffic patterns are known, or can be predicted, then it should be possible to configure specific routes based on source and destination without needing to know dynamic information about subsequent switches. Generally, however, this is not possible as computer network traffic patterns are not always predictable.

Common implementations are thus based on a form of randomisation, such as the popular equal cost multi-path routing (ECMP) or the recently ratified Shortest Path Bridging (SPB) which use hashing functions to load balance over multiple links. Over the Internet, different routes can exhibit different latencies and maximum transmission unit (MTU) differences. This can cause rapidly changing latencies and packet re-ordering if packets belonging to a single stream are distributed across different route paths. These effects can disrupt the operation of many stateful Internet protocols, most notably TCP. These issues are discussed in IETF's RFC2991(Thaler and Hopps, 2000), *Multipath Issues in Unicast and Multicast Next-Hop Selection.*

It is unlikely that packetised radio astronomy instruments will suffer from these particular problems for two reasons:

1. The Clos network will use the same or similar makes and models of switches, interconnected in a uniform architecture. This means that the switching fabric will be uniform. MTUs will be the same and latencies through any intermediate switch (ie ports on different ingress/egress switches) should be similar if loads are properly balanced.

2. I use stateless UDP, not TCP. UDP has no flow control mechanism like TCP whose back-off engine can be upset by changing latencies. The end-node receiver's buffer simply needs to be able to accommodate the difference between the maximum and minimum latencies through the network, which, as we've mentioned, should be small.

In an attempt to solve the multipath problem in the general case, RFC 2992 analyses one particular multipath routing strategy involving the assignment of flows of data, rather than individual packets. The flows are identified by hashing flow-related data in the packet header (such as source and destination MAC addresses and IP addresses and ports), and this hash is used to select an egress port. This allows for all packets of any particular network flow to traverse the network on the same single path, while balancing different flows over multiple paths in general. This solution is commonly implemented on commercial switches and routers.

In the case of this work, the only significant consideration is that a routing path be chosen that will not result in the packet being dropped or significantly delayed (queued in an output buffer), which would adversely affect jitter and increase the buffer requirements in the processing nodes. Perhaps of greatest significance, then, is that the load balancing algorithm be effective. The system can tolerate per-packet load balancing rather than per-flow load balancing, and this would be preferable as it would provide greater entropy.

Using any randomised approach, it is possible that sub-optimal routes can be chosen, resulting in buffers filling due to micro-bursts on individual links. This is significant for MeerKAT, since each link is expected to be used at above 90% capacity, so individual links and buffers can easily become swamped. Since radio astronomy instrumentation traffic patterns are deterministic, it should be possible to calculate the minimum required port buffer size, given a certain routing algorithm and desired probability of overflow.

Links can also be *under-subscribed* to reduce the probability or altogether prevent such overflows.

Many switch vendors have proprietary solutions to load balancing, and some provide the option to manually configure routes. It is recommended that these issues be discussed with the selected switch vendor and a mechanism employed to allow even distribution of traffic over available fabric links, while minimising the chances of micro-bursts.

5.3.4 Maximum number of antennas

While it might be considered that the 16 bits allocated for antenna labelling in the packet format would allow for up to 65536 antennas in the system, there is an inherent problem with the chosen data exchange format preventing this limit being reached. Since each packet contains data for a single antenna for a given channel and most network equipment limits jumbo packets to a little over 8KiB, this means that the maximum number of 4 bit (complex) samples that can be contained in a single packet is 8192.

Assuming that the selected FPGAs have sufficient memory resources to buffer such large packets for all connected antennas, the maximum length of an X-engine is limited to 4096 antennas in order to ensure sufficient readout time for all calculated baselines before the next window is processed. This is because this packet size is currently used to set the number of accumulations within an X-engine (see §4.5).

In order to overcome this limitation and be able to process additional antennas:

- either the X-engines would need to process more data in parallel so that internal accumulation periods can be shortened, or,
- vector lengths can be increased beyond the size of a single packet by employing a more sophisticated data exchange protocol between FPGA nodes that supports such fragmentation, such as SPEAD, to create a longer serialised window for the X-engine core, or,
- the spectral processing order could be changed to sequentially emit a time series of a single spectral channel, and the logic feeding the

X-engines could then be modified to accumulate multiple sequential windows internally.

New X-engine cores are already planned that would enable arbitrarily configurable parallelisation. MeerKAT is also planning to implement a subset of SPEAD for all inter-FPGA communications within the DBE. Either of these solutions will overcome this limit.

5.3.5 Processed bandwidth

The architecture fundamentally does not limit the processed bandwidth. As many network links can be added as necessary to carry the bandwidth. However, the processing boards selected impose practical limitations. The existing CASPER ZDOK standard (used for ADCs) limits bandwidth of data into the F-engines to approximately 40Gbps (40 LVDS pairs at approximately 1Gbps per pair) on ROACH-1 or 50Gsps (40 pairs at 1.25Gbps) on ROACH-2. This is 5GSa/s, or 2.5GHz bandwidth, at 8 bits per ZDOK. For faster ADCs, I envisage using serialised devices conforming to a serialised standard, such as JEDEC JESD204 [1], in the ROACH-2 mezzanine slots. It is likely that ROACH-3 will deprecate the ZDOK interfaces and add additional mezzanine slots.

After the data is ingested, the pipelined FFT and PFB-FIR algorithms can be split across processing boards if resources on a single board are exhausted. However, ROACH-2 is already capable of processing in excess of 5GHz bandwidth with up to 18-bit datapaths, outstripping all current telescope requirements, so I do not envisage this becoming necessary.

Care must be taken when designing the FPGA platforms to ensure that the processing capabilities of the FPGA, peripheral IO bandwidths, external memory capacities and bandwidths, as well as off-board interconnect bandwidths, are balanced to allow all resources to be fully and efficiently utilised. This is discussed further in §5.5.3.

[1] `http://www.jedec.org/sites/default/files/docs/JESD204B-01.pdf`

5.4 Flexibility

The designed machine has proven to be very flexible. It has been deployed at multiple telescope facilities, each requiring a different digitised and processed bandwidth, number of inputs and spectral resolutions. The design is agile over these parameters. KAT-7 utilises this capability to change the machine's personality for wideband imaging or spectral line work, for example.

Additionally, the FPGA design has been deployed across three generations of FPGA hardware, Virtex II Pro, Virtex-5 and Virtex-6 on four different processing platforms, as well as in heterogeneous systems together with with CPUs and GPUs.

The modular CASPER hardware offers various ADCs from within the community. The appropriate digitiser can be selected for a given installation with tradeoffs between the number of inputs per processing node, the bandwidth and data resolution. Changing ADCs requires a recompile of the F-engine code to accommodate the new digitised bandwidth and data representation format.

MeerKAT and PAPER are investigating the use of remote digitisers at the antennas (as opposed to embedded in the compute platforms) with an Ethernet-based backhaul, further demonstrating the flexibility of the architecture.

5.4.1 Parameterised designs

The use of parameterised components throughout the system allows the system to be tailored for a particular installation's needs. The basic design parameters, the numerical representation, bandwidth, number of antennas and spectral channels, are simply adjusted at compile time. In addition, features such as the delay compensation and fringe correction's range and resolution are configurable.

In the event of limited FPGA memory resources, this allows the implementer to trade delay compensation resolution for spectral resolution, for example, by shifting resource allocation from the delay compensation components into the F-engine's FFT.

Many of the CASPER blocks (including the ones modified and developed

for this work), allow the implementer to decide how certain functions are implemented. For example, a multiplier may be implemented as a lookup table, using the FPGA's DSP slices or constructed using generic logic slices. The FFT and X-engine blocks, for example, allows the implementation to be configured on a per-stage basis. By allowing such fine-grain control over the implementation (not a global selection or optimisation), all FPGA resources can be utilised.

5.4.2 Adding functionality to an existing system

Functionality can be added to existing systems easily in two ways:

Firstly, additional processing nodes can be added to the system to perform the new function. Data can be routed arbitrarily through the switch to these new nodes. The Ethernet interconnect and CASPER hardware allows for the appropriate amount of compute hardware to be deployed, and for the system to grow or shrink incrementally as additional functionality is needed, even after deployment. The Ethernet interconnect allows such nodes to be added while the existing system is operating, thereby avoiding any downtime.

Secondly, the existing FPGA code can be changed to add functionality to existing bitstreams. Most CASPER hardware is able to be reprogrammed remotely so that additional functionality can be added. The modular approach adopted here limits impact to the rest of the system when such changes are required. This was demonstrated when beamforming functionality was added to the KAT-7 correlator, which only required a recompile of the X-engines (to which the beamformers were added). F-engines remained unchanged.

The FPGA platforms require a recompile to change their programming. Loading the new bitstream remotely takes milliseconds on modern devices, fast enough so as to be insignificant over a typical observation, and enabling on-the-fly modification. While it is technically possible to avoid even these few milliseconds of downtime and add functionality to an existing bitstream in a completely hitless fashion, this would require partial FPGA reconfiguration – a complication I avoided in this work and a feature not supported by the current CASPER tools.

5.4.3 Heterogeneous systems

PAPER and LEDA have also demonstrated the ability to exchange the FPGA X-engine processors for GPUs and in so doing have created a hybrid machine consisting of FPGAs and GPUs. Users are referred to Clark *et al.* (2011) for details of the GPU implementation. This has effectively demonstrated the design's ability to host multiple processor types concurrently. The implementor is free to select the appropriate technology for a given task and seamlessly integrate it into the system.

5.5 Principles of generic FPGA-based board design for radio astronomy instrumentation

The progression through three generations of hardware platforms has allowed for iterative improvements to the design and optimisation for typical radio astronomy instrumentation applications. This section details some of the lessons learnt and caveats encountered when trying to design a generic FPGA processing platform for radio astronomy instrumentation.

For reference, Table 5.1 shows the hardware resources and on/off chip resource proportions for various CASPER boards.

5.5.1 Simplicity

Albert Einstein is famously quoted as saying that *everything should be made as simple as possible, but not simpler.* Engineers have a tendency to want to add complexity to offset risk and also increase the device's feature-set. But to limit costs, ease usability, reduce failure modes and improve manufacturability, hardware platforms should be kept as simple as possible to achieve the required functionality.

The iBOB was an extreme example of such simplification as it was just an FPGA with non-volatile memory to hold the device's configuration. This board was too simple as it did not allow for efficient remote control in large systems or for remote reprogramming without a JTAG tool. Even so, it

Table 5.1: **A summarised listing of on and off chip compute resource distributions of various CASPER hardware generations.**

Resource	iBOB	BEE2 (/FPGA)	ROACH	ROACH -II
Logic cells	53K	74K	94K	476K
DSP slices	232	328	640	2016
BRAM capacity	4.2Mb	5.8Mb	8.8Mb	38Mb
SRAM capacity	2x18Mb	-	2x36Mb	4x144Mb
SRAM bandwidth	9Gbps	-	43Gbps	200Gbps
DDR capacity	-	4x8Gb	1x8Gb	1x16Gb
DDR bandwidth	-	115Gbps	38Gbps	50Gbps
10GbE ports	2	4	4	8

was a very popular platform within the CASPER community, validating the simplicity argument. ROACH arguably lies at the opposite end of the sophistication spectrum, and even includes hardware features that were never used and these created difficulties in manufacturing and support.

While software solutions are generally preferred, these should not be adopted if they require additional hardware that adds complexity. For example, ROACH is populated with an early-generation Actel Fusion mixed-signal FPGA to allow for system power control and health monitoring, either through the PPC, or else, independently through a Lantronix XPort Ethernet to RS232 bridge device. At the time of design, this was felt to offer maximum flexibility should system configuration need to be changed in the field, such as reconfiguring automated safety shutdown logic or to allow for possible software work-arounds for hardware faults. However, configuration and administration of both the Fusion and the XPORT has proven troublesome and difficult – at one stage requiring reverse-engineering of the XPort's programming protocol to allow for automated configuration of the device on the production line. The system is also not reliable, periodically not responding to commands and so is useless for its desired role as an emergency interrogation and recovery interface (for example, to hard-reboot the power). As a

result, datacentre-grade Ethernet controlled power distribution units are still required for reliable remote power control. This entire onboard subsystem is now unused by SKA-SA, though it required a significant amount of development time which could have been better spent elsewhere or in reducing the board's time to market.

When designing hardware, requirements specifications should be carefully considered for real-world advantages and only added if it is a reliable, robust solution. Beware of components added "in case". In these situations, it might be more cost effective and faster to rather first evaluate a simplified circuit in a prototype board, with a planned second or third hardware revision to make any required changes for the final production units.

As it was, ROACH-1 progressed through three hardware revisions anyway and these reprogrammable devices are statically configured in all known deployments, negating the need for a complex reconfigurable subsystem. ROACH-II does not feature a programmable power supply subsystem, nor an out-of-band external interface. A prototype board, shown in Figure 5.8, was manufactured to verify the new design and to correct any bugs in the hardware design before full production of a second revision began. ROACH-1's Actel has been discarded and the entire power supply subsystem simplified. Power sequencing now takes place without any software or gateware control.

Furthermore, all implemented functions should be reliable and easy to use on all boards. Some ROACH-1 boards' PPCs' 1GbE ports are unable to maintain stable links at 1000Mbps speeds. This problem appears to be due to bad or out-of-spec PHY parts and replacing them on the production line fixes the issue. However, neither PAPER nor KAT-7 need 1Gbps speeds and, indeed, the PPC is unable to deliver 1Gbps throughput, so this feature has limited usefulness. All Ethernet ports are manually configured to connect at 100Mbps for deployments in order to ensure reliability (which requires either an older switch or an expensive managed unit, since overriding the PHY's auto-negotiation does not solve the problem). It would have been cheaper and more reliable to adopt proven-technology 100Mbps PHYs from the outset, without compromising performance. ROACH-II uses an alternate 1Gbps PHY together with a wider FPGA-PPC bus to boost data throughput in order to effectively leverage the 1000Mbps port.

5.5.2 On-chip resources

The choice of which FPGA device to use should be determined not only by the overall logic capacity, but also by the ratio of logic to other on-chip resources (BRAM and DSP slices).

The Virtex-II Pro based devices (used in the iBOB and BEE2) had a well-balanced mix of on-chip resources for radio astronomy instrumentation purposes, but designs would sometimes exhaust DSP and BRAM before fully utilising all onboard logic. For this reason, when selecting between the Virtex-5 LX (favouring logic resources) and SX (favouring DSP and BRAM resources) families for ROACH, the SX range was chosen. With ROACH-1, however, I am often unable to make full use of the DSP capabilities as routing (logic) resources are exhausted before all DSP slices are used, especially for X-engines. F-engines typically exhaust all on-chip BRAM for larger designs with sharp filters or many FFT channels before utilising all DSP and logic slices.

It thus follows that no single chip will ever be optimal for both F and X-engines and this is the price paid when using a single hardware platform for multiple purposes. However, it is my belief that the benefits of this single-board approach still outweigh the small loss in hardware utilisation efficiency.

Co-locating different engine types (such as F and X engines) on a single FPGA might better distribute the use of on-chip resources.

Extra BRAM is of particular benefit to channelisation operations by allowing longer FFTs and additional or larger lookup tables for filter coefficients, since memory requirements scale linearly with length (as opposed to logic and multiplication which scales $O(NlogN)$. BRAM also is generally useful for debugging by hosting test vectors or capturing snapshot data, and so it is unlikely that any design will ever have this resource go unused. In general, then, I choose the FPGA with the greatest BRAM resources.

The larger Virtex-6 SX devices (as used in ROACH-II) have a better distribution of on-chip resources for X-engines but will likely still be BRAM limited for large F-engines.

5.5.3 Off-chip resources

Off-chip resources also need to be considered carefully. The iBOB and ROACH boards aimed to match bandwidths throughout all off-chip devices. The concept was to support streaming designs where data rates in and out of all peripherals could be matched so that no single resource limited the system's processing capacity.

ROACH supports memory bandwidth into or out of the QDR and DDR parts at 36Gbps when clocked at a typical 250MHz. After adding headers to these streams, data rates increase closer to 40Gbps - well matched to the four 10GbE ports. However, this is not always ideal. QDR is typically used for matrix transpose operations and so requires both a read and write on every clock, and so should support at least twice the bandwidth. ROACH-1's QDR capacity is also limited and is often exhausted before computing resources are completely utilised.

The layout of the memory on the board is also worth considering. For some applications, such as for large synchronous corner-turn operations, a single wide and deep memory is required. This is typical in the F-engines.

For asynchronous designs, each engine core on the FPGA runs independently of the others. Designs are much simpler if each engine has access to its own storage area (for example, each X-engine's vector accumulator). While intra-engine clocks are synchronous and data alignment ensured within that chain, synchronisation across multiple engines (even on the same FPGA) requires additional control logic and potentially larger buffers, which is to be avoided when possible. As an example, ROACH has two QDR parts, limiting the number of co-located independent engines to two. For smaller designs, such as KAT-7's eight antenna X-engines, ROACH has sufficient logic for four engines but insufficient independent QDR memory resources for the vector accumulators. Compute resources are thus wasted as only two engines can be efficiently placed on a board. To make full use of the logic resources in small systems with insufficient off-chip resources, on-chip BRAM can be effectively used for additional vector accumulators since the vector lengths are short anyway, and so would make poor use of large off-chip memory. For designs of over 16 antennas, ROACH's on-chip resources limit the number of engines

to two, and so maximum use can be made of all available compute resources at all times.

To support multiple layouts, ROACH-2 features as many independent memories as possible. In this case, FPGA I/O pin counts limit ROACH-2 to four QDR parts. These can be connected serially for deeper memory, in parallel for wider bandwidths or used independently as required by the application, giving much greater flexibility.

Looking ahead, this matched bandwidth model is becoming increasingly difficult to maintain. While on-chip FPGA resources are scaling aggressively, off-device IO is not. ROACH-2 already uses over 99% of available pins on the largest package, mostly for QDR memory. Considering the six fold increase in processing capacity over the four year period between Virtex-5 and Virtex-7 devices, there is not a corresponding increase in the number of pins (1759 to 1930) to allow for additional memories, nor faster clocking speeds on these pins (1Gb/s to about 1.5Gb/s) to allow for faster memories.

FPGA processing capabilities are increasing at a higher rate than the number of FPGA IO pins and this trend is likely to continue with the industry moving towards 3-D and layered packages. This forces a higher percentage of the IO pins to support high speed serial protocols to maintain the IO:compute balance. Future boards will thus require a higher number of high speed serial interfaces to peripherals such as ADCs and external memories. For this reason, I believe the high speed serialised memory interfaces will become standard connectivity options in future, though such commercial devices (such as the MoSys Bandwidth Engine[2] or the Hybrid Memory Cube concept) are currently rare.

5.5.4 Interconnect

This research advocates the use of a packetised interconnect between all processing nodes.

The BEE2, an early CASPER board, placed multiple FPGAs on a single LRU. In many practical radio astronomy applications of this platform, the onboard interconnect between these processors were eschewed in favour of

[2]http://www.mosys.com/products.php?pid=33

the Ethernet interfaces, which allowed the programming abstraction where all processing nodes communicate using the same mechanism, rather than managing on-board and off-board communications differently.

It is important that each compute element be connected to the network switch to ensure maximum flexibility, rather than forming local chains. While it is evident from §2.2 that all instruments employ some degree of pre-processing on individual inputs, the data traffic patterns are very different and the network must remain flexible to accommodate this re-routing if the compute machine is to be generic.

For this reason, if a generic processing machine is to be built, then it is recommended that the system-wide unit of measure be a single processing element which connects to a crossbar switch for all data exchange. Any onboard, inter-chip links should not come at the expense of switch network ports.

5.5.5 ROACH-2

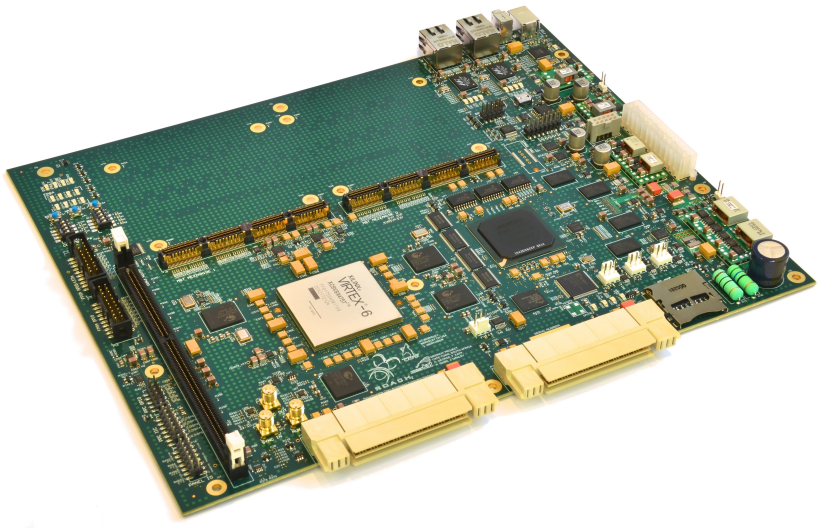


Figure 5.8: An early prototype ROACH-2 board, showing two ZDOK connectors in the foreground and two mezzanine interfaces for mounting cards on top of ROACH-2, towards the rear.

The concept of a generic, reprogrammable, single-FPGA board proved successful with ROACH-1 (CASPER's most successful board to date) and this concept has been carried-over to the next hardware processing platform, ROACH-2. ROACH-2 has been simplified compared to its predecessor, ROACH-1, and upgraded in respect of all signal processing paths. In keeping with the rapid hardware design concept, ROACH-2 is an iterative improvement on the ROACH-1 platform used in this project, rather than a ground-up re-design. This has reduced development time between conceptualisation and production. The high-level block diagram was completed in 2010, whereafter detailed design began. Prototypes were ready in 2011, less than one year after availability of the FPGAs. The time to market is decreasing between CASPER hardware generations as the reprogrammable platform concept matures.

ROACH-2 improves upon ROACH-1 not only in its digital signal processing capacity and performance but also, through simplification, in manufacturability, usability and reliability. The introduction of multi-gigabit transceiver (MGT) mezzanine cards has added an additional dimension to the platform's flexibility.

Figure 5.8 shows a first revision board. At the time of writing, ROACH-2's second revision board has entered production, which corrected a number of minor bugs.

Differences to ROACH-1

ROACH-2 builds on the successes of ROACH-1 while adding larger, faster memories to a larger FPGA with modular SERDES interfaces. It is designed around a Virtex-6 SX475T main FPGA which offers approximately four times the compute power of the ROACH-1's Virtex-5 SX95T together with a better ratio of on-chip memory to DSP and logic resources for the purposes of radio astronomy signal processing.

ROACH-2 features over ROACH-1:

- Five times the FPGA logic resources, three times the DSP and four times the BRAM of ROACH-1.
- Mezzanine cards for high-speed SERDES interfaces

- A wider, 32-bit bus between the PPC and FPGA (up from 16 bits)
- Four 36-bit wide, 72Mb QDR memories (up from two 18-bit 36Mb)
- A DDR-3 DIMM (as opposed to ROACH-1's DDR-2 DIMM)
- Simplified, all-hardware power supplies and supervisors
- Fan speed regulation and monitoring
- Onboard FTDI USB interface for JTAG and serial communications
- Additional 1GbE FPGA interface
- Marvell 1GbE PHYs
- Improved PCB layout

Interfaces

ROACH-2 does not host any native high speed Ethernet links, as was done on ROACH-1. While the Ethernet standard has been historically long-lived, there are various physical interfaces available, especially in the faster 10GbE, 40GbE and 100GbE implementations. ROACH-1's CX4 connectors did not allow optical links without adaptors, nor are they robust, with numerous failures reported in early (iBOB and BEE2) hardware systems.

ROACH-2's FPGA's high speed serial transceivers are available to mezzanine cards. This allows for the physical cable interface (for example, CX4, QSFP, SFP+ or RJ45) to be chosen independently of the processing platform. As with the Z-DOK boards, it is envisaged that these mezzanine cards will be reuseable across hardware generations. Alternatively, these interfaces can be used to host ADCs and DACs that conform to the JESD204 JEDEC standard for high-speed serial interfacing.

An additional 1Gbps Ethernet connector is provided to the FPGA for those mid-range applications that do not need the faster, more expensive 10GbE interfaces but need higher data rates than the PPC's CAM interface can provide.

For control and monitoring, the flexible and reliable text-based KATCP[3]

[3]https://casper.berkeley.edu/wiki/KATCP

was retained from ROACH-1, ensuring that all developed applications are forwards compatible. Rajan (2012) describes how the BORPH layer has been removed to increase performance and the KATCP server now interacts with the FPGA in user-space over a memory-mapped driver.

Power supplies and health monitoring

ROACH-2 simplifies aspects of the ROACH-1 design such as the hardware monitoring and power control circuitry, replacing ROACH-1's fully programmable Actel Fusion device with non-programmable units after lessons learnt in §5.5.1.

ROACH-2's power supplies sequence off each other and are internally current-limited, largely negating the need for external sequencing and hardware supervisory functions to prevent damage and catastrophic failures. Function-specific voltage, current and fan monitoring ICs were placed on the PPC's IIC bus to accommodate the additional functions previously performed by the Actel Fusion on ROACH-1.

The Ethernet out-of-band hardware monitoring XPORT device has also been dropped to simplify the design.

Manufacturability and maintainability

The initial board bringup was streamlined and support for in-field firmware upgradability added. This was achieved by populating ROACH-2 with an FTDI USB-JTAG bridge. This device also allows for simplified automated configuration and testing as the boards exit the assembly line, using a unified open-source software environment over a single USB interface.

Not only does this simplify and accelerate board testing but it also allows users to recover "bricked" boards after failed firmware updates by using an ordinary laptop without any specialist equipment. In contrast, ROACH-1 required three vendor-specific JTAG programmers to flash the hardware monitoring chip, bus-management CPLD and the bring up the CPU. Each manufacturer-provided JTAG programmer employs a proprietary software tool which can be difficult to automate reliably.

Additional minor improvements were made to the PCB layout to further

ease manufacturability and thermal characteristics. For example, all front panel interfaces are now hosted on two co-located ribbon connectors near the left of the board (as opposed to three connectors located on both extremes of the ROACH-1 board). Functions were split across two separate ribbon cables, rather than unified, to maintain compatibility with standard ATX interfaces. The second connector augments the two front-panel LEDs on conventional ATX cases with eight FPGA-connected LEDs and two more PPC LEDs.

All external interfaces have ESD protection and fan headers feature self-resetting fuses to protect against shorts. Cooling requirements and thermal simulations were considered by SKA-SA and partially motivated for the PPC's external DRAM memory to be soldered directly onto the PCB, in order to prevent the upright DIMM from shielding the PPC from the airflow. Air cooling provides a simple, low-maintenance, low-risk and low cost cooling solution. CFD simulations also indicated that while a high cooling capacity was required when operating at full load, for most envisaged applications where the boards are housed in temperature-regulated rooms, reduced fan speeds could be accommodated, thereby reducing ambient noise in the server room and extending the operating life of the chassis fans.

5.6 Development environment

The CASPER Xilinx Platform Studio (XPS) toolflow has been highly successful, so much so, that maintaining compatibility with this framework was a design constraint for ROACH-1 and ROACH-2, which motivated for the use of a Xilinx FPGA.

While the CASPER XPS toolflow works well for streaming DSP designs typically employed in radio astronomy instrumentation, it is not without its flaws and limitations. This section will outline some identified advantages and disadvantages encountered during the implementation of this work.

5.6.1 Context

Before considering the advantages and disadvantages of the CASPER XPS flow, it is necessary to first put it in context.

The CASPER DSP library currently only targets FPGA platforms, primarily due to the reliance on Xilinx's System Generator blockset. Alternative blocksets, such as Mathworks' native Simulink blocks, allow for compilation for FPGAs (through Mathworks' HDL Coder product) or even to GPUs and CPUs. These blocksets are not yet supported.

While the DSP library is parameterised, the compile-to-static-bitstream nature of the solution precludes runtime adjustment of any parameters. The user must thus configure his system at compile time. Recompiles can be lengthy, with users reporting compile times from 5 minutes to 14 hours depending on the design complexity and the compile computer's specifications. Once compiled and made available on the ROACH boards, bitstreams can be swapped very quickly ($\sim$20ms for ROACH-2).

This implies that telescope operators would be able to select from a list of available instruments with pre-defined functions and parameters such as processed bandwidth or spectral resolution, rather than using a single instrument where these parameters can be selected while operating. This is not necessarily a drawback, as each instrument can thus be kept simple with operation of its single function thoroughly verified.

5.6.2 CASPER XPS advantages

The CASPER Xilinx Platform Studio toolflow offers the following advantages:

- Data-oriented design
- Rapid development
- Intuitive GUI environment
- Cross-platform Linux and Windows support
- Existing DSP library is extensive, configurable and parameterised

- Clock-cycle accurate simulations
- Tunable for resource limitations (can often trade-off logic resources for BRAM or DSP slices, for example)
- Ability to abstract-away low-level interfaces and common operations
- A true single-button, end-to-end solution for FPGA bitstream generation

5.6.3 CASPER XPS disadvantages

The CASPER Xilinx Platform Studio toolflow presents the following disadvantages:

No backwards compatibility guarantee for existing DSP designs. The CASPER MATLAB scripts, which are used to dynamically redraw blocks, use Xilinx primitives and connect them together and configure their parameters as appropriate. Unfortunately, Xilinx periodically change their parameter names and functions, thereby breaking the automated scripts. This was encountered by CASPER during the migration from Xilinx version 7 to version 11 tools, when a library port to the new blockset was required. Existing designs must then be updated to work with the new libraries, which is partially automated.

Inability to support legacy devices. Virtex-II (iBOB and BEE2) is no longer supported from Xilinx ISE 11 onwards. Radio astronomy instruments and hardware generally have lifetimes exceeding those of consumer products. Manufacturers deprecate support for legacy devices long before the device stops becoming useful for RA purposes. Existing iBOB and BEE2 users are now forced to use older tools that are no longer supported by Xilinx or the CASPER toolflow. This is not a reflection on the CASPER tools or hardware in particular, but rather the commercial electronics industry in general.

Version incompatibilities between different vendors' products prevents incremental or piecemeal upgrades of the toolflow. The older versions

of the Xilinx tools, such as ISE10 (required for BEE2s and iBOBs), are not compatible with the latest versions of MATLAB. Using older versions of software is problematic when obscure errors are encountered as vendors are often unwilling to support their older products and encourage users to upgrade.

Instabilities. Especially with larger designs, which require large amounts of RAM on the compile machine, a number of unexplained segmentation faults and crashes have been observed. Perhaps most worrisome is that older versions of Simulink will on occasion not open certain existing files, crashing on load. If the user does not have a backup of this file, they are unable to recover it. The causes of the corruption of these files is unknown, but seems to have been rectified in later versions.

Vendor-locked. Because the toolflow uses Xilinx System Generator primitives, the toolflow is only able to target devices supported by these primitives; that is, Xilinx FPGA devices.

No testbench facility. To ensure stability and reliability of the libraries after submissions from multiple collaborators, an automated unit testing framework is desirable. However, there is no known simple way to implement this on current Matlab Simulink models.

Lack of source-code control. Not all source code is under CASPER's control. FPGA cores within the Xilinx framework are utilised for many low-level interfaces and operations. These are sometimes deprecated, as demonstrated in ISE11 when Xilinx moved from PPC to ARM processors and deprecated the entire IBM OPB bus infrastructure that all CASPER boards employ. Version 12 tools, and later, no longer support this bus infrastructure, requiring CASPER to re-architect their busses across all peripherals.

5.6.4 Porting designs to new FPGA platforms

It is assumed here that new processing platforms will become available that can leverage the latest technological innovations. Writing software for mod-

ern CPUs and GPUs to interface to new devices is normally easy as the manufacturer provides drivers and infrastructure and this programming interface is usually held static, or at least backwards compatible, for as long as possible.

FPGA based systems, however, have significantly less manufacturer support, with regularly changing interfaces to standard hardware IP modules. This is especially evident in the memory interfaces, for example, as each DRAM generation (DDR, DDR-2, DDR-3) requires a new interface.

As the port from CASPER's Virtex2Pro to Virtex5 series boards revealed, this can have unforeseen complications for the application designer. In the correlator application case, the F-engines used the iBOB's external single-ported SRAM for a matrix transpose operation which had to be ported to the ROACH's dual-ported QDR. While the existing iBOB DSP design could have been used with ROACH, it would have been inefficient as each secondary QDR interface would have been wasted. It is unlikely that all the benefits of a new compute platform will be leveragable without some updates or modification to the source code to take advantage of new features.

CASPER accepts but aims to minimise the effort required during this process by making incremental changes to hardware platforms and maintaining backwards compatibility whenever possible. This implies the use of standardised interfaces wherever possible.

5.7 Conclusion

In this chapter, I considered the implemented design from §4 critically, and start by answering the primary research questions.

1. I showed that it is possible to construct an Ethernet interconnected radio astronomy signal processor by successfully designing, constructing and deploying an operational correlator and later a beamformer.

2. I found that the cost of using an Ethernet switch is modest, when considering each of the purchase cost, running costs and resource requirements of these components. I also considered the total ownership cost of a packetised, FPGA-based solution and found that FPGAs are

cost-competitive only if operated over longer periods (and that in some cases, technologies with lower purchase costs, such as GPUs, would be more cost effective for shorter-lived instruments), but that in all cases, deployed technologies should be refreshed regularly and not operated indefinitely.

3. Scalability was analysed for each component in the system. The architecture itself is highly scalable, allowing any number of processing devices to be connected, and limited only by the selected switch's ability to host more ports. Certain choices made for this reference implementations, however, imposed practical limitations on the bandwidth (such as choice of processing platform and resulting ADC interface bandwidth), number of antennas (chosen packet format limitations, for example) or achievable number of spectral channels (memory limitations on the hardware platform, for example). In all cases, however, the design could be modified to support greater scalability, should this become necessary.

4. The implementation was shown to be flexible by demonstrating deployment at multiple telescope facilities, each with varying instrument requirements. In all cases, the design was simply and quickly adapted to the new requirements.

After answering the research questions, some principles of generic FPGA-based processing board design for the purposes of radio astronomy instrumentation are considered. Focus is placed on the desirables for simplicity to achieve a fast time-to-market, the need for a good balance of the on-FPGA resources (and how there can be no perfect "one-ratio-suits-all" for all applications; ensuring that there will always be some inefficiencies and wasted resources when designing a generic processing platform), the requirements for off-chip resources (such as memory capacity and bandwidths) as well as the interconnect desirables. From this, a potential design for ROACH-2 is mapped for use on MeerKAT.

The chapter concludes with some thoughts on the development environment. The CASPER XPS flow is found to be a significant advantage over

conventional HDL coding for the purposes of radio astronomy instrumentation. However, it is not without fault and some identified advantages and disadvantages are listed for reference.

Chapter 6

Conclusion

The hypothesis that it is possible to develop a scalable, flexible, general purpose radio astronomy imaging instrument for array signal processing using reprogrammable hardware along with a commercial Ethernet interconnect appears to be valid. All objectives of this project were met.

After a review of existing instrumentation systems, a suitable hardware platform was selected and a flexible instrument developed, using the correlator as the first example instrument implementation. This design was successfully deployed at a number of telescope facilities, namely, PAPER, KAT-7, GMRT and Medicina. I also successfully demonstrated low development costs and a reduced development time when compared to other systems constructed using traditional instrumentation techniques.

Ultimately, the architecture described in this work has been a positive evolution of telescope instrumentation. The single-platform, reprogrammable, Ethernet-connected ROACH platform has also proven successful, and has found widespread use in radio astronomy instrumentation. This platform has since been refreshed with a more modern FPGA and additional, faster memories on ROACH-2. ROACH-3 is now under development and will follow the same design principles as its predecessors.

This section will now summarise this work's accomplishments.

6.1 Reduced costs

Section 5.2, shows that instrumentation based on this work had low costs as a result of fast time-to-market and the consequential use of the latest hardware. Key enablers were the use of a single mass-produced hardware platform (rather than specialised, function-specific boards historically deployed) and a common, low-cost commodity interconnect in the form of Ethernet.

By using the same commodity components throughout the system, hardware development costs are amortised more widely and it is possible to leverage volume production more easily, even for smaller systems. Furthermore, by employing open-source DSP libraries from the CASPER community, engineering costs for the programming of these boards were very low.

I also showed that switching and Ethernet costs were modest. In large systems, the switching infrastructure represents less than 10% of the initial system cost and approximately 6% of the system's power budget. The Ethernet cores are not too onerous for modern FPGAs, currently representing approximately 8% of the FPGA's resources for an un-optimised core. And I note that the logic requirements will become ever smaller on future, larger devices.

Taking a holistic view, I showed that telescope facilities with long lifetimes should budget for technology refreshes, and not continue operating the existing hardware indefinitely. This will reduce the overall lifetime cost of the facility.

6.2 Reduced development time

The packetised interconnect simplifies the overall system design and accelerates the design of additional features and functions, which can leverage the existing logic. Ethernet cores and interfaces are also popular, allowing for considerable re-use of intellectual property from other disciplines. It greatly simplifies the design of the inter-processor interfaces.

However, initial development of asynchronous components can take longer than a similar synchronous system as packetised designs must cope with additional failure mechanisms (out of order or missing packets, for example).

The packetised interconnect itself comes at a cost of increased FPGA logic requirements and some network overhead. However, these costs are justified and correctly managing these failure modes results in a low cost, scalable, robust and flexible system. Also, once developed, this logic can be re-used in future designs and does not require re-engineering.

This project's first packetised correlator took nearly a year to develop. New intellectual property was developed, in addition to leveraging existing CASPER open-source DSP libraries. A one-year development time is already a significant improvement on many other systems. But once these building blocks were in place, subsequent designs were constructed in much shorter timeframes, determined primarily by a project's required customisation and procurement practices. A system was assembled and deployed at INAF's Northern Cross facility in Medicina in just 10 days. KAT-7's correlator was designed, constructed and deployed in three months. PAPER and KAT-7 have already demonstrated world-class science from these instruments, vetting its performance.

The reduced development time was achieved by:

- Standardising the processing platform (ROACH)
- Standardising the interconnect (Ethernet)
- Using a well-suited programming environment (Simulink dataflow model)
- Leveraging existing IP wherever possible (CASPER libraries)
- Employing a simple, flexible, scalable architecture able to adapt and expand incrementally, as necessary.

6.3 Improved flexibility through a universal architecture

It is easy to customise the design for a given telescope. DSP components throughout the signal processing chain are parameterised for typical radio astronomy instrumentation requirements, such as number of inputs, bandwidth and number of frequency channels. Features such as Walsh switching,

polarisation correction or spectrum equalisation can be included or excluded depending on a given installation's functionality requirements.

In addition to modifying existing instruments, new instruments are also easily added. This was demonstrated by the addition of a beamformer to the correlator on KAT-7. It is possible to add pulsar machines, transient buffers and other, as yet unforeseen, features to the design. MeerKAT will take this concept a step further, and make all intermediate data products available on a multicast network switch, including raw digitised time-domain data.

The architecture is universal. By placing all reprogrammable processing nodes in a flat logical space, and interconnected by a full crossbar switch, the machine is able to host any instrument topology and implementers are not constrained by hard-wired processing flows. Furthermore, the architecture allows for heterogeneous computing by exchanging any network-connected processing elements at will. With defined interfaces, exchanging processing elements is transparent to the rest of the system.

6.4 Scalability

PAPER alone has already demonstrated seamless scalability of the design from 16 inputs to 256 inputs, including design compatibility across FPGA hardware generations. LEDA has scaled the design up to 1024 inputs, using GPU-based X-engines. Section 5.3 discusses the scaling implications and, although some limitations are noted with this implementation, solutions have been proposed to overcome all of them and no significant hurdles are noted for scaling to much larger arrays, including SKA-sized systems with many thousands of elements.

MeerKAT is the largest planned deployment of this design. It is scheduled to come online in 2016, and will eventually process over 2GHz bandwidth for 128 inputs. The multicasting concept will be leveraged to simultaneously perform correlation for broadband and spectral line work, beamforming and pulsar searching or timing operations, raw data recording etc. Like PAPER, MeerKAT's digital processor is expected to scale with the array as additional antennas come online. This work is particularly well-matched to these array requirements, allowing for cost-effective growth by incrementally growing the

network and adding processing elements.

6.5 Impact and applicability

This work has already had a significant impact on the radio astronomy instrumentation community. The architecture is highly flexible and will find use in many telescope facilities.

The hardware, programming and interconnect concepts have already seen rapid adoption by the community. As of 2011, over 300 ROACH boards were in circulation at more than 30 institutions. LEDA, PAPER, KAT-7, GMRT and Medicina already all use a packetised interconnect (with FPGA and/or GPU compute nodes) for their imagers, with similar architectures to the one described in this work. A packetised FPGA/GPU heterogeneous processor has since also been proposed for SKA-1 (Faulkner, 2012).

Apart from imaging applications, the ROACH boards are also already used in pulsar machines at Green Bank, Parkes and Effelsberg, for example. The Green Bank Telescope's new Ultimate Pulsar Processor Instrument (GUPPi) is a highly flexible machine featuring ROACH boards feeding a bank of GPU processors through a 10Gbps Ethernet network.

Growth and interest in these concepts continue, as can be judged by the attendance at the annual CASPER radio astronomy instrumentation workshop which was inaugurated in 2006 with 30 participants. The 2009 edition, hosted in Cape Town, South Africa, attracted more than 100 attendees for the first time.

The application space is also broadening: at the 2010 CASPER workshop, Haystack and NRAO (Mcwhirter, 2010) announced an upgrade for the international VLBI project's data receivers, which would also feature a ROACH board for its primary processing, with Ethernet output to a diskpack recorder. This system did not make use of CASPER's programming environment and the development period was longer than most other CASPER instruments at $\sim$5 years. But this is still significantly faster than it would have been if constructing a full custom machine.

While these international stories are testament to the success of the concept, and the concept continues to gain traction, most of the existing deploy-

ments are in smaller systems. The decision by SKA-SA to base the Meerkat digital back-end (one of the largest signal processing systems of its kind) on this work represents the single biggest commitment and will demonstrate the system's applicability for next generation facilities such as the SKA.

6.6 Further work

While a working correlator and beamformer have already been demonstrated, the system is capable of hosting many other instruments, including time-domain beamformers, VLBI data formatters and pulsar machines. The full capability of this architecture has not yet been realised and multicasting has not yet been implemented.

In the following subsections, I consider implications for multicasting and the possibility for adding recirculation to the correlator.

6.6.1 Multicasting demonstrated but undeployed

Both PAPER and KAT-7 are using the point-to-point UDP protocol over unicast IPv4 for data exchange. I have been unable to truly realise my goal of sharing all antenna, frequency and intermediate data products in an IP multicasting environment due to current switch limitations which do not support large numbers of concurrent multicast groups.

The original intent was to have the packetised correlator exchange multicast data in the network switch and to allow any other instrument to subscribe to this data to enable commensal observations. Ideally, data processors would be able to subscribe to any frequency slice(s) from any of the array's component antennas. So I would like to have a separate multicast group for each frequency channel on each antenna. Even for a modest 32-input, 1024 channel correlator, this would require $32 * 1024 = 32768$ groups, not considering output data products.

Early 10GbE switches, such as those used in this project, supported very few simultaneous multicast groups. For example, the Fujitsu XG700 and XG2000 are limited to 32, which is grossly insufficient for my goal of multicasting each frequency channel of each antenna of each instrument in a

separate group. Even the latest switch models do not support such large numbers. The Arista 7050 series, for example, is limited to 8192 simultaneous multicast groups. These limitations are due to lookup table restrictions within the switch. While additional groups can be accommodated in some designs, in these cases the switch's processor swaps-out lookup table entries for the switching ASIC as other multicast groups are encountered, drastically reducing performance with larger numbers of simultaneous groups.

As antenna arrays grow ever larger, this goal of per-channel, per-antenna addressing becomes even more untenable. Either specialist switches are required or concessions need to be made to collect data together in coarser subscribable groups.

Special-order, custom switches can be constructed to support larger numbers of channels, but that defeats the purpose of using standard parts to leverage the benefits of volume production consumer products, saving money and reducing development time.

Even if a custom switch were to be designed, IPv4 multicasting itself is very restrictive, only allocating 28 bits of unique multicast address space, with only 24 bits available for local, organisational scope (see §3.3.5). IPv6 increases useful multicast address space to 112 bits which would be sufficient for all foreseeable needs on a layer-3 switch. But the Ethernet MAC address space remains limited to only 23 bits on a Layer-2 switch.

Fortunately, most scientifically-useful instruments (correlators, beamformers, pulsar processors etc) are designed to process groups of adjacent channels, and so even in the case where each frequency channel were in an independent group, these instruments would simultaneously subscribe to multiple multicast groups. So in most cases, no functionality is lost by collapsing frequency channels into coarser multicast groups consisting of multiple frequency channels (or bands). In this way, it is possible to reduce the number of simultaneous multicast channels while retaining all practical commensal multicast benefits. Such a configuration has already been prototyped for deployment on MeerKAT and demonstrated to work well with commercial switches.

In the case of MeerKAT, which will likely be the first telescope to deploy multicasting instrumentation, it is likely that the F-engines' frequency chan-

nels, at least, will be collected into more coarse multicast groups. Similar concessions will likely be made with the beamformer and correlator outputs. While this will limit the granularity to which consumers can subscribe to data products for commensal observations, it will still allow for a machine of unprecedented flexibility. The MeerKAT system is further discussed in Appendix C.

6.6.2 Recirculation

It is possible to recirculate the output from an F-engine back into another F-engine to obtain finer spectral resolutions. If the output of an F-engine were transposed and fed back into the same F-engine, processed bandwidth could be traded for higher spectral resolution.

A packetised design can simply redirect F-engines' output back into those same F-engines. A counter in the packets' header section can be added to keep track of how many times the data has been recirculated and hence the spectral resolution of the contained data. The data must also be transposed before reprocessing which requires memory. However, this happens anyway, regardless of recirculation, as the chosen packet format for the X-engines contain multiple time samples of a single frequency channel. These packets can then be operated on independently by X- or F-engines. Recirculation can thus be had for free in this packetised design.

There is a complication in that Ethernet switches will not forward a packet back out on the inbound link (see §4.5.7). Thus, it is not possible to send one F-engine's data back to that same F-engine through the switch. Either a local copy must be kept, or the output must be sent to another F-engine board. The other F-engine can, in turn, send it's output data to this F-engine. Practically, this is not a significant limitation as most systems will have more than one F-engine board.

Carlson (2000a) has analysed the effects and requirements for recirculation in the FXF WIDAR correlator and lists a number of complexities associated with data re-alignment and delays in that system. A true packetised approach, such as is being designed for MeerKAT (see Appendix C), naturally avoids these difficulties as the receivers already include logic for

re-organising and re-aligning data packets, based on timestamp headers.

6.7 Conclusion

This project aimed to address the instrumentation problem faced by many large radio telescope facilities: that it takes a long time and costs a lot of money to develop typical facility instrumentation. This is partially due to continuous wheel re-invention. I have shown that a flexible, generic machine, capable of hosting correlators, beamformers and other radio astronomy array instrumentation can successfully address this problem. A single deployed set of hardware is able to host instrumentation for multiple science experiments. A beamformer and various correlators were implemented and demonstrated.

This work used CASPER and other open-source tools to deliver a machine based on FPGAs, but able to host other processor technologies too, interconnected using Ethernet-based packetised switches. By using a full-crossbar, any-to-any switch, arbitrary data flow topologies are possible. This allows for a general-case solution where any desired algorithm can be simply mapped onto the system. Furthermore, multicasting allows for many instruments to run concurrently, and share intermediate products arbitrarily.

This architecture is somewhat future-proof; engineers are able to add functionality and cost-effective upgrades are possible in future. Piecemeal upgrades are also easily possible, by adding hardware of different technology generations and even of disparate processor types. All interfaces employ the historically long-lived Ethernet standard.

Modification and customisation of the design is possible quickly. Rapid design and deployment enables integrators to track the technology curve closely, and this brings benefits of lower total cost of ownership.

By using a reduced number of types of general purpose compute platforms, rather than many disparate application-specific processors, maintenance is eased and reliability improved.

These concepts have all been demonstrated successfully. A cost-effective, general-purpose radio astronomy instrument was realised on a flexible, scalable architecture, constructed from reprogrammable, FPGA-based processing platforms and interconnected using commercial Ethernet technologies.

References

Y Abhyankar, C Sajish, P Kulkarni, and C Subrahmanya. Design of a FPGA based data acquisition system for radio astronomy applications. In *The 16th International Conference on Microelectronics, 2004. ICM 2004 Proceedings.*, pages 555 – 557, Dec 2004.

D Backer, A Parsons, R Bradley, C Parashare, N Gugliucci, E Mastrantonio, D Herne, M Lynch, M Wright, D Werhimer, et al. PAPER: The precision array to probe the epoch of reionization. *BULLETIN-AMERICAN ASTRONOMICAL SOCIETY*, 39(4):133, 2007.

D Backer. EoR experiment - memo: Quantization with four bits. March 2007.

T Bastian and A Bridle. Proceedings of a science workshop: The VLA development plan. http://www.cv.nrao.edu/vla/upgrade/node111.html, January 1995.

A. O Benz, P. C Grigis, V Hungerbühler, H Meyer, C Monstein, B Stuber, and D Zardet. A broadband FFT spectrometer for radio and millimeter astronomy. *Astronomy and Astrophysics*, 442:767–773, November 2005.

A Bridle and F Schwab. Bandwidth and time-average smearing. *Synthesis Imaging in Radio Astronomy II, ASP Conference Series*, 180:371, 1999.

W Brisken. Cross correlators. Tenth Synthesis Imaging Summer School, June 2006.

J Bunton. An improved FX correlator. ALMA Memo 342, December 2000.

B Carlson. A more detailed analysis of 'recirculation' architecture, algorithms, and limitations in the proposed widar correlator for the evla. NRC-EVLA Memo 004, July 2000.

B Carlson. A proposed WIDAR correlator for the expansion very large array project: Discussion of capabilities, implementation, and signal processing. NRC-EVLA Memo 001, May 2000.

B Carlson. EVLA project book. Chapter 8, August 2007.

C Chang, J Wawrzynek, and R Brodersen. BEE2: A high-end reconfigurable computing system. *IEEE Design and Test of Computers*, 22(2):114–125, March-April 2005.

H Chen and M Wagner. External adjustment for Atmel/e2v iADC. Technical Report 40, CASPER, August 2010.

W. N Christiansen, R. H Frater, A Watkinson, J. D Osullivan, W. M Goss, and I. A Lockhart. Observations of 15 southern extragalactic sources with the Fleurs synthesis telescope. *Monthly Notices of the Royal Astronomical Society*, 181:183–202, October 1977.

M. A Clark, P. C. L Plante, and L. J Greenhill. Accelerating radio astronomy cross-correlation with graphics processing units. *International Journal of High Performance Computing Applications*, preprint 2011.

C Clos. A study of non-blocking switching networks. *Bell System Technical Journal*, 32(2):406–424, March 1953.

R. E Crochiere and L. R Rabiner. *Multirate Digital Signal Processing*. Prentice-Hall, 1983.

L D'Addario. Correlators: General design considerations. ATA memo 24, March 2001.

A Deller, S Tingay, M Bailes, and C West. DiFX: a software correlator for very long baseline interferometry using multiprocessor computing environments. *Publications of the Astronomical Society of the Pacific*, 119(853):318–336, 2007.

P Demorest, R Ramachandran, D Backer, R Ferdman, I Stairs, and D Nice. Precision Pulsar Timing and Gravity Waves: Recent Advances in Instrumentation. *Bulletin of the American Astronomical Society*, page 1598, December 2004.

A Enteshari. *HIGH-SPEED ACCESS OVER COPPER WIRING*. PhD thesis, The Pennsylvania State University, 2009.

Evironmental Protection Agency. Report to congress on server and data center energy efficiency. August 2007.

A Faulkner. Integrated SKA1 central processing system: Correlator, beamformer, gridding, imaging, searching and storage. SKA-1 Central processing system draft, June 2012.

J Fisher. Autocorrelator sampler level setting and transfer function. GBT Spectrometer note, 2002.

A Gunst. LOFAR architectural design document of the astronomical applications. LOFAR-ASTRON-ADD-006, February 2007.

P. L Herselman. Four-bit FX correlator van vleck calibration and correction. MeerKAT Systems Engineering Memo Series K0000-2002V1-03 TM REV A, SKA-SA, March 2010.

B. C Joshi, A. G Lyne, and M Kramer. Next Generation Software Based Instruments for Pulsar Astronomy. *Bulletin of the Astronomical Society of India*, 31:237–242, March 2003.

T Laakso, V Valimaki, M Karjalainen, and U Laine. Splitting the unit delay [FIR/all pass filters design]. *Signal Processing Magazine, IEEE*, 13(1):30–60, January 1996.

A Latour and K v.d. Schaaf. LOFAR central processing facility architecture description. LOFAR-ASTRON-ADD-012, March 2007.

D Lorimer and M Kramer. *Handbook of Pulsar Astronomy*. Cambridge University Press, 2005.

R Mcwhirter. Haystack/NRAO VLBI digital backend on the ROACH platform. CASPER workshop 2010, Harvard-Smithsonian Centre for Astrophysics, 2010.

V Nagpal and T Filiba. Beamforming for antenna arrays: BEE2 vs DSP processors. Memo, Center for Astronomy Signal Processing and Electronics Research, UCB., 2007.

A Networks. *Cloud Networking Portfolio Product Quick Reference Guide*, August 2011.

A Parsons, D Backer, C Chang, D Chapman, H Chen, P Droz, C de Jesus, D MacMahon, A Siemion, J Wawrzynek, D Werthimer, and M Wright. A New Approach to Radio Astronomy Signal Processing: Packet Switched, FPGA-based, Upgradeable, Modular Hardware and Reusable, Platform-Independent Signal Processing Libraries. *Proceedings of the XXXth General Assembly of the International Union of Radio Science, Boulder, Colorado*, September 2005.

A Parsons, D Backer, C Chang, D Chapman, H Chen, P Crescini, C de Jesus, C Dick, P Droz, D MacMahon, K Meder, J Mock, V Nagpal, B je Nikolic, A Parsa, B Richards, A Siemion, J Wawrzynek, D Werthimer, and M Wright. PetaOp/Second FPGA Signal Processing for SETI and Radio Astronomy. In *Signals, Systems and Computers, 2006. ACSSC '06. Fortieth Asilomar Conference on*, pages 2031 –2035, 29 2006-nov. 1 2006.

A Parsons, D Backer, A Siemion, H Chen, D Werthimer, P Droz, T Filiba, J Manley, P McMahon, A Parsa, D MacMahon, and M Wright. A Scalable Correlator Architecture Based on Modular FPGA Hardware, Reuseable Gateware, and Data Packetization. *Publications of the Astronomical Society of the Pacific*, 120:1207–1221, November 2008.

A Parsons, D Backer, R Bradley, J Aguirre, E Benoit, C Carilli, G Foster, N Gugliucci, D Herne, D Jacobs, et al. The precision array for probing the epoch of reionization: 8 station results. *Arxiv preprint arXiv:0904.2334*, 2009.

A Parsons. The symmetric group in data permutation, with applications to high-bandwidth pipelined FFT architectures. *IEEE SIGNAL PROCESSING LETTERS*, 16(6), June 2009.

A Peens-Hough. Technical requirements for the KAT-7 digital back-end. Technical Memo SET/SSD/004/K, SKA-SA, September 2010.

R. A Primiani, J Weintroub, and J deWerd. Polyphase filter-bank utilization. SMA Wideband Correlator: Memo 1, 2011.

S Rajan. Masters thesis in preparation. Master's thesis, University of Cape Town, 2012.

S. M Ransom, P Demorest, J Ford, R McCullough, J Ray, R DuPlain, and P Brandt. GUPPI: Green bank ultimate pulsar processing instrument. *American Astronomical Society Meeting Abstracts*, 214:605.08, December 2009.

J Romein, P. C Broekema, J. D Mol, and R van Nieuwpoort. The LOFAR correlator: Implementation and performance analysis. *Proceedings of the 15th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, Bangalore, India*, 45(5), May 2010.

J Roy, Y Gupta, U.-L Pen, J. B Peterson, S Kudale, and J Kodilkar. A real-time software backend for the gmrt. *Experimental Astronomy*, 28(25), 2010. A real-time software backend for the GMRT (Roy, J. et al., 2010, Experimental Astronomy, 28, 25).

H So and R Brodersen. Improving usability of FPGA-based reconfigurable computers through operating system support. In *16th International Conference on Field Programmable Logic and Applications (FPL'06)*, pages 349–354. CODES+ISSS, 2006.

H. K.-H So, A Tkachenko, and R Brodersen. A unified Hardware/Software Runtime Environment for FPGA-Based Reconfigurable Computers using BORPH, 2006.

D Thaler and C Hopps. Multipath issues in unicast and multicast next-hop selection. IETF RFC2991, November 2000.

A Thompson, J Moran, and G Swenson. *Interferometry and synthesis in radio astronomy*. Krieger Pub. Co., second edition, 1994.

W Urry, M Wright, M Dexter, and D MacMahon. The ATA correlator. ATA memo series, February 2007.

W. L Urry. The ATA imager. ATA memo 39, January 2002.

P Vaidyanathan. Multirate digital filters, filter banks, polyphase networks, and applications: A tutorial. *Proceedings of the IEEE*, 78(1):56–93, 1990.

R van Rooyen and A Rust. Kat-7 dbe atr. K0000-2011VA ATR, July 2011.

R van Rooyen and A Rust. Kat-7 dbe atp. K0000-2006V1 ATP, June 2012.

J. H van Vleck and D Middleton. The spectrum of clipped noise. *Proceedings of the IEEE*, 54(1), January 1966.

Various Authors. Wikipedia: Very large baseline array. as at 22 December 2010, December 2010.

K v.d. Schaaf. LOFAR system overview. NGI, 22 November 2005, Astron, Netherlands, November 2005.

D Werthimer. Spectrometers, Beam Formers, VLBI and Correlators using general purpose FPGA boards, tools and libraries (how to build eight radio astronomy instruments in two years). Annual CASPER workshop, July 2007.

A Woods. Accelerating software radio astronomy FX correlation with GPU and FPGA co-processors. Master's thesis, University of Cape Town, February 2010.

Appendix A

PAPER

This chapter will describe the correlators built for the PAPER array, which represent the first instruments constructed as part of this research. The correlators scaled together with the array, starting with 8 antennas and progressing to 64 antennas as at the time of writing in 2012. This exercised the scalability of the design. Moreover, PAPER was the first array to deploy the new ROACH platform, and these processing boards were used in conjunction with the existing iBOBs, demonstrating device inter-operability. The existing BEE2 X-engine design was recompiled for the ROACH platform with minimal changes required which demonstrated progression across technology generations.

The PAPER array is a targeted instrument designed to search for the Epoch of Reionisation (EoR) by detecting the highly redshifted hydrogen spectral line (Backer *et al.*, 2007). The EoR is expected to be found between a redshift of 7 and 10 which set the operating frequency of the array at 100MHz to 200MHz.

PAPER is a significant technical challenge as it tries to detect a signal well below the ambient cosmic microwave background noise. PAPER's initial approach is to perform wideband imaging over its full frequency range and then subtract a very accurate known sky model of the primary foreground objects to look for the EoR signal in the residual.

A large collecting area, comprised of many small antennas, is needed to reach the required sensitivity, along with new calibration and imaging tech-

niques. Figure A.1 shows a PAPER antenna, which is designed to be constructed using simple, low-cost materials. They consist of dual-polarisation dipoles with full sky view. A reflective base is used to provide some forward gain and flaps were later added to the sides which help reduce the response at the horizon and to terrestrial RFI sources.

PAPER is a compact array, with the longest baseline being less than 300m.

Figure A.1: A deployed PAPER antenna showing the dual polarised dipole receiver, reflective base and flaps. The LNA and balun is housed within the central support column.

Each antenna contains an amplifier and 75-ohm coaxial cable driver. The sky signal is transported back to a central processor over commercial television type coaxial cable. At the central processor, this analogue signal is filtered and further amplified before entering the digital system, without any downconversion.

An engineering test-platform array is operational at the NRAO site in Green Bank, West Virginia and the full array is currently being constructed in phases on the Karoo Astronomy Reserve in South Africa, with intermediate scientific observations occurring between these phased upgrades. The

correlator system is required to scale with the array as additional antennas are deployed.

While the number of antennas in the PAPER array continues to grow, processed bandwidth remains fixed at 100MHz. PAPER would like as many FFT channels as possible, but this is not critical. Because PAPER does not perform any delay tracking, this must be done in software during post processing. The increased spectral resolution is thus useful for limiting intra-channel smearing. Increased spectral resolution is also useful for RFI excision.

Details of the PAPER system, including antennas, analogue front-ends and receivers can be found in Parsons *et al.* (2009). This chapter will briefly summarise the relevant sections from the early designs, also published in Parsons *et al.* (2006) and Parsons *et al.* (2008), and then focus on the later ROACH-based packetised correlators.

A.1 PAPER's correlator

The PAPER packetised correlator consists primarily of four components:

- the F-engines,
- the X-engines,
- a central control and storage computer and,
- infrastructure (cabling, network, timing etc).

Centralised control of the system greatly simplifies system management. The central control computer performs the configuration and overseas operation of the system using a collection of Python scripts. These scripts leverage a Python-based ROACH communications API which was layered on top of the Karoo Array Telescope Control Protocol (KATCP) framework to simplify FPGA command exchanges. Various scripts have been written to initialise the system and monitor signal propagation. A significant effort was invested to make this system flexible and scalable. This effort was well spent as this framework now also serves KAT-7 and will provide a base for

MeerKAT. Realtime visualisation of the sky signals is enabled by an early release of the KAT-7 signal display system.

The TCP control network is kept physically and logically separate from the high speed 10GbE data network used to transport F-engine data to the X-engines. These high speed intra-correlator data streams are transported in self-contained IPv4 UDP datagrams (see Chapter 4). The correlator's output data products are assembled by BORPH on PPCs where a custom application layer protocol was initially in use. Later designs migrated to the Streaming Protocol for the Exchange of Astronomical Data (SPEAD, §4.8), though packetisation still occurs in software on the X-engine boards' PPCs. A stand-alone receiver daemon is currently responsible for capturing, reassembling and storing this data on the control computer.

Every aspect of the PAPER design is open-source and has been made available on the CASPER web server.

A.2 Early designs: the PoCo, synchronous and 16-input packetised correlators

PAPER's initial CASPER-based correlator was a four-input single-board "pocket correlator" (PoCo) based on an Internet Breakout Board (iBOB) and two dual 1Gsps ADC cards. This system was limited in that it was not scalable to more inputs (which would require multiple boards) and readout was slow through a serial interface.

The first multi-board FPGA correlator was designed for the initial eight antenna PAPER array using the BEE2 and iBOB platforms. This 16-input multiboard synchronous system was developed by Aaron Parsons (described in Parsons *et al.* (2006)) as a stepping-stone towards a fully packetised asynchronous design.

This first multi-board system was augmented with packetised interfaces, overcoming the final scalability hurdle. It was a full-Stokes, 16-input packetised system based on a 10GbE switch, four iBOBs and a BEE2. This system was in use by PAPER in Green Bank for nearly two years but also suffered from slow readout speeds, due to bus limitations on the BEE2. These early

Figure A.2: Aaron Parsons (left) and Jason Manley (right) capturing the first fringes with PAPER's 8 antenna packetised correlator at NRAO's Galford Meadow site in Green Bank. Eight iBOBs can be seen in the rack in the background.

systems are described by Parsons *et al.* (2008).

This first packetised correlator was later expanded to form a 32-input system by adding another four iBOBs and a BEE2, though this was more regularly used as two independent 16-input systems rather than as a single 32-input correlator, partly due to the slow readout speeds when outputting the extra baselines.

The remainder of this section will discuss this first 16-input packetised correlator.

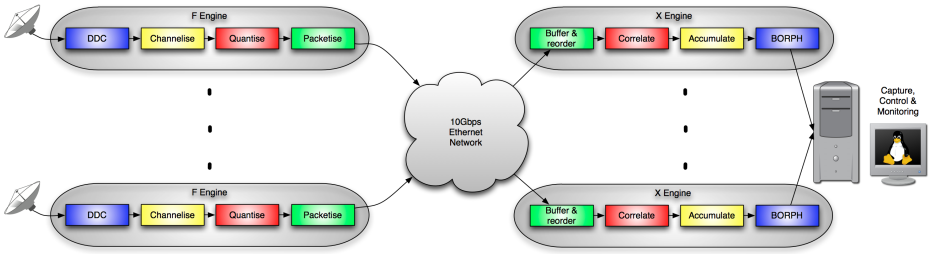


Figure A.3: A block diagram showing the basic signal processing stages in PAPER's first packetised correlator. This design is simpler than KAT-7 (Figure 4.1), without fringe rotation or delay compensation, and instrument readout through BORPH rather than natively through the 10GbE ports. But it features a Digital Downconverter (DDC) to select a sub-band of interest, which KAT-7 does not.

A.2.1 Communicating with the early hardware

Recall from §3.4.5 that while the BEE2s run BORPH, and so can be operated remotely over Ethernet, the iBOBs run TinySH on one of the Virtex-2 Pro Power PC cores. TinySH is a simple command-line interface for reading and writing registers on the FPGA. Without an Ethernet stack, TinySH is accessible only over a direct serial (RS232) connection. The "Lighweight IP" component, which allows for such remote Ethernet communication, requires a significant portion of FPGA resources and so was not incorporated in PAPER's packetised designs. It was considered clumsy and unmanageable to run an independent RS232 serial port to each of the connected iBOBs in

larger systems.

Recall that PAPER employs the loopback optimisation discussed in §4.2.4. This F-engine to X-engine link is also used for the control of IBOB devices by communicating in-band from the BEE2, using free upstream bandwidth across these point-to-point links. This mechanism proved successful and monitoring data was later injected into the downstream datastream too, which was extracted by the X-engines for forwarding by BORPH.

The iBOBs' EEPROMs are flashed with the F-engine bitstream and load themselves at power-on automatically. They cannot be reprogrammed without a JTAG tool making remote upgrades difficult. PAPER thus requires the presence of a technician on site when iBOB firmware updates are required.

The BEE2s also imposed certain restrictions. The slow FPGA communication bus (§3.4.5) made configuring the FPGAs and initialising the various registers a lengthy process and throttled the rate at which data could be output from the instrument.

A.2.2 F-engine and X-engine customisation

The first PAPER F-engines were hosted on hardware available at the time: iBOBs populated with two dual 1GSa/s iADCs operating at 600MHz, sampling the sky frequency directly. These designs included a DDC to select the band of 75-225MHz from the 300MHz digitised band.

PAPER uses length-matched cables and so delay-compensation is not necessary with integration times up to approximately seven seconds, even on the longest (300m) baselines.

The channeliser consists of a 4-tap polyphase filter bank across 2048 FFT channels. This sliding-window filter greatly increases channel isolation, reducing the typical sinc response of rectangular-windowed FFTs (see §4.3.4 for more info on PFB performance). The CASPER FFT has a runtime-configurable optional downshift at each butterfly stage to prevent overflows within the FFT. PAPER shifts at every stage, which means that the signal amplitude decreases through each FFT stage (by a factor of $\frac{\sqrt{2}}{2}$ for noisy input) and ensures that no overflows can occur, under any circumstances.

The 18 bit FFT output is then scaled using a complex multiplier before

quantising down to a 4 bit complex representation. The positioning of the signal within these four bits is critical for good quality data. PAPER attempts to set this scale coefficient to place the RMS level of the signal in the small portion of the 4-bit quantiser that offers a linear transfer function, while still maintaining a good signal to noise ratio. From Figure 4.11, the linear portion corresponds to a 4-bit power level between 5dB and 10dB, or an RMS voltage count of 1.8 to 3.2 (assuming symmetrical ±8 range) – a relatively low level, resulting in poor SNR. PAPER chooses to use average RMS voltage counts closer to 4.

Independent digital equalisation on each frequency channel allows for fine-grain bandpass shape correction. Each iBOB hosts a single set of coefficients which is applied to all four digitised inputs on that iBOB. These four analogue inputs are trimmed to within +-1dB by manually placing attenuators on the ADC inputs.

A packet length of 128 data words was chosen, and with double buffering, the 2048 frequency channels across the 4 inputs of 4 bit complex data on each iBOB requires 16 Mbit of corner-turn memory. This, by necessity, uses off-processor memory due to on-chip block-RAM capacity limitations. The ECC bits available in the SRAM are ignored and no data integrity checks are performed in the memories.

All the X-engine cores were hosted on a single BEE2 (two cores per corner FPGA). The loopback optimisation described in Section 4.2.4, whereby F-engine data is routed through the X-engines, was included to save switch ports. A single 10GbE link was used from each of the BEE2's corner FPGA to the switch. Each FPGA operated independently, with the BEE2's onboard inter-FPGA communication links unused.

A.2.3 Timestamping

A free-running signal generator was used to generate the 600MHz sampling clock, which was sufficiently accurate for PAPER's needs. This was then amplified and distributed passively to all ADCs using off-the-shelf RF components through length-matched cables. The F-engines' FPGA clocks are in turn derived from the attached ADCs.

The initial design used a stand-alone pulse generator to synchronise all the F-engines. An NTP-synchronised computer then received and timestamped the sequentially-numbered output packets from the correlator. This proved unreliable, with a changing offset every time the system was resynchronised, due to the uncertainty of when the synchronisation pulse occurred. The control computer's LPT port then replaced the pulse generator, allowing for accuracies better than 10 ms. Knowing the time of synchronisation eased system calibration significantly.

A further improvement was made by timestamping the data within the individual X-engines upon reception from the F-engines (as opposed to having this done by the receiving computer). This reduced variability in the integration timestamps.

However, if an error occurred on the XAUI cable linking the F- and X-engines, synchronisation of that antenna would still be lost until the system was resynchronised. This problem plagued early deployments until shorter cables were installed which improved signal integrity. This problem was finally solved with the next upgrade of the system, as described in Section A.3, where timestamping was moved into the F-engines so that lost data would not adversely affect the continued operation of the instrument.

A.3 32 and 64-input correlators

The 8-antenna system was upgraded in 2009 to support 16 dual-polarisation antennas (32 inputs) in preparation for the larger South African deployment. Later, this was upgraded again to support 64 inputs. It builds on the experiences gained by constructing the earlier generation systems. These iterations on the design maintained the same basic architecture and operation of the aforementioned 16-input system with some hardware changes. Primary differences include:

- Denser but lower sample rate ADCs better suited to PAPER's requirements.
- ROACH computing hardware in place of BEE2s. iBOBs are still used for the F-engines.

- New timing and synchronisation hardware constructed around a Thunderbolt E ovenised oscillator.
- Unified software control package.

A.3.1 Digitisation

The first PAPER packetised correlator used CASPER's iADC digitisers based on the Atmel (now E2V) AT84AD001; an 8 bit dual 1 Gsps part. These were mated to iBOBs and BEE2s for data processing. This ADC is costly and much faster than PAPER requires for directly sampling its 100-200MHz sky frequency. A slower, quad-input ADC, based on the Analogue Devices AD9480, was developed. This ADC allowed for more analogue inputs on the same processing hardware and so lowered overall system costs. The new ADC card digitises four streams at up to 250Msps with 8 bits per sample.

Recall that the signal is filtered using analogue filters to select the 100MHz to 200MHz band, allowing the use of a slower 200MHz sampling clock. This places the band of interest in the second Nyquist zone which gets aliased down to baseband. The spectrum must be mirrored before post processing. This is done in software before recording to disk. While the use of iADCs required a DDC in the F-engines, these are now no longer required with the slower sample clock.

The simplicity with which the ADCs were changed again demonstrates the flexibility of the design.

A.3.2 F-engine and X-engine modifications

Apart from an ADC change, the F-engine portion of the design has remained largely unchanged through the design iterations. Four (later eight) iBOB F-engine boards host two quad-ADC cards each, for a total of 32 inputs (later 64 inputs).

Four ROACH boards replaced the single BEE2 from the 16-input system. Each iBOB connects directly to a ROACH board using a point-to-point XAUI link as before.

The number of frequency channels was initially decreased to 1024 to save hardware resources under the assumption that the design with the additional inputs on each iBOB would not fit in the FPGA. The PFB core inside the F-engines was altered to allow a single streaming core running at 200MHz to process two 100MHz streams using the same compute resources, allowing the number of channels to be increased back to 2048.

The number of equalisation coefficients (to allow for digital amplitude correction) was reduced from fully-independent control of each frequency channel to 32 (later 64) across the band. But they were now independently configurable for each input (as opposed to one spectrum of coefficients applied all inputs on each iBOB board). This enabled automatic adjustment for mismatches in the individual analogue chains; the manual 1dB analogue matching adjustments were labour intensive and not precise enough for efficient 4-bit requantisation. The reduction of spectral coefficients is not due to hardware limits, but simply because the frequency-dependent adjustability had not been used to date and configuration time is saved when using fewer coefficients. The coefficients are runtime configurable to allow optimal selection as the sky temperature changes. Any resulting stepped adjustment across the spectrum must then be undone when the data is processed as part of the non-linear Van Vleck correction.

By populating each iBOB F-engine with two quad-ADCs, doubling of the number of inputs required twice as much corner-turner memory in the external SRAMs. This fully-utilised the iBOBs' SRAM capacity. No additional inputs can be accommodated on the iBOB unless spectral resolution or bit-widths are sacrificed. PAPER will be deprecating the iBOB F-engines in favour of ROACH-based units in future.

In PAPER's 32-input correlator case, each ROACH board is populated with two X-engine cores. These two X-engines share a single network interface. The X-engines are clocked at 210MHz and process the 100MHz analogue bandwidth from the 16 dual-polarisation inputs at 200MHz.

A.3.3 Timestamping improvements

A *Trimble Thunderbolt E* now provides the primary timing reference. It is a GPS-disciplined, ovenised quartz oscillator. Its 10MHz reference output is fed into a *Valon 5003* synthesiser which provides the 200MHz sampling frequency.

The Thunderbolt's 1PPS output has a 15ns (1-sigma) specification which means that system synchronisation to within one 200MHz clock (5ns) of the second boundary cannot be guaranteed. But it is sufficient to align inputs for the correlation operation to proceed in hardware. The remaining calibration takes place in software during post processing. This is a significant improvement over the software-based LPT solution which provided pulses only to within milliseconds of second boundaries.

The computer's system time is now locked to GPS time using an RS232 dataport, allowing for remote operation without NTP network connectivity.

Most significantly, the datastreams are now timestamped in the F-engines and these timestamps are propagated along with the data. This provides an absolute reference for all subsequent engines and allows the system to resume functioning normally should a failed communication link be restored while operating, without requiring resynchronisation.

A.3.4 FPGA resource utilisation

Tables A.1 and A.2 show the on-chip FPGA resource utilisation for PAPER's F- and X-engines on their respective hardware platforms. PAPER makes good use of FPGA resources, with DSP operations dominating the device utilisation.

This section will analyse the resource utilisation for PAPER's F- and X-engines.

F-engines

The *Control and Monitoring* (CAM) function on the iBOB F-engines include a multiply-accumulate (MAC) operation to calculate RMS input levels on each ADC input.

Table A.1: **PAPER's F-engine FPGA resource utilisation by function as a percentage of the iBOB's Xilinx Virtex-II Pro 50 FPGA's resources for the initial 1024 channel compile.**

Function	Logic	BRAM	DSP
1024-chan FFT	55%	32%	24%
4-tap windowing PFB FIR filter	6%	18%	13%
ADC interfaces and IBOB base	5%	13%	0%
Control and Monitoring	5%	2%	7%
Matrix transpose	2%	1%	0%
Packetiser and XAUI interface	2%	1%	0%
Equaliser & quantiser	1%	2%	6%
Total Design	76%	68%	50%

All external communications with the iBOB take place through the XAUI port, with commands utilising the uplink bandwidth from *X-engines* to *F-engines*. However, in addition, memory and peripherals are allocated in "ADC interfaces and IBOB base" for one of the iBOB's FPGA's onboard hardcore Power PCs which is accessible over an RS232 serial link. This PPC runs a basic shell (TinySH) which is not used in normal operation, but provides a fall-back debugging interface.

The F-engine's amplitude equaliser is independent on each input, but only allows for 32 scalar coefficients across the frequency band.

The excess FPGA capacity has subsequently been used to increase the spectral resolution back to 2048 channels.

X-engines

The ROACH X-engines were initially tightly packed, using 99% of available logic resources, and so some effort was invested to optimise this portion of the design. The majority of the resources were consumed by control and monitoring functions, which consisted primarily of on-bus software registers for device configuration and status reporting through BORPH. 38% of the FPGA's logic resources was considered excessive for this purpose and a subse-

Table A.2: **PAPER-32, 64-input X-engine FPGA resource utilisation by function as a percentage of ROACH's Xilinx Virtex 5 SX95T FPGA's resources. This table reflects the first compile of the design, before any optimisation.**

Function	Logic	BRAM	DSP
Control and Monitoring	38%	23%	0%
Two X-engine cores	29%	7%	45%
Two vector accumulators	11%	5%	0%
F-engine loopback	9%	4%	0%
10GbE transceiver	8%	7%	0%
Incoming packet re-order	3%	6%	0%
ROACH base infrastructure	1%	0%	0%
Total Design	99%	52%	45%

quent optimisation of this library component reduced control and monitoring logic utilisation to approximately 20%, which was considered reasonable. It is possible to further optimise control and monitoring usage by packing logic signals into a reduced set of 32-bit bus-attached registers or even memory-mapped BRAMs. Globally, further optimisation and resource shuffling is also possible, for example, by using the Virtex5's excess DSP48E slices as muxes and counters to save logic resources. These optimisations have since allowed PAPER to increase the number of inputs to 128 (doubling the X-engine core size) and also reduced the FPGA resources required for KAT-7 (see Table B.3).

Excess BRAM could be put to use for storing input test vectors which can be used for digital (bit-accurate) system verification. Such arbitrary pattern insertion abilities have not yet been implemented and all current test vectors are generated locally, using combinations of counters and logic gates.

The 9% logic used in the X-engine boards by the F-engine loopback operation includes a XAUI interface and loopback FIFO buffer. These resources could be saved if the loopback optimisation (§4.2.4) were forfeited. However, this would then require twice as many switch ports.

The figure quoted in Table A.2 for ROACH base infrastructure includes the instantiation of on-chip clocks, busses and bridges to allow the on-board microprocessor to communicate with the FPGA. The figures quoted for the vector accumulator include the underlying QDR memory controller.

A.3.5 Cost and power consumption

Table A.3 shows the cost and power consumption of PAPER's 32-input digital system.

Power consumption is becoming more of a concern for modern computer systems. The Evironmental Protection Agency (2007), in a report to congress, showed that large supercomputing systems' costs are often dominated by power and cooling costs over the lifetime of the system.

The current PAPER system is still relatively small and the entire digital back-end comfortably fits inside a single 19" rack which can be powered from a portable generator, allowing for remote operation. However, as the array scales to more inputs, the power requirements will become more significant. By projecting the power consumption figures shown in Table A.3 to larger arrays, future estimates for PAPER's power consumption were obtained. This motivated for the site to be connected to South Africa's national grid network, which was completed in 2011.

Table A.3: **PAPER-32 system cost and total out-the-wall power consumption.**

Function	Cost	AC Power (wall)
4 ROACHes	11720 USD	204W
4 iBOBs w/QuADCs	12400 USD	113W
Clocking and timing	2200 USD	30W
10GbE network switch	11300 USD	49W
Compute server	3311 USD	233W
Cables and adaptors	700 USD	N/A
Total	41631 USD	629W

The list prices for the iBOB and ROACH boards do not include the

purchase of the FPGA devices which were donated by Xilinx for this project. Paying list price for single quantities would add 20400 USD to the total cost.

The cost of the four iBOBs includes the eight quad ADC daughter boards at 1300 USD each and 1800 USD per iBOB (pricing for single quantities). It does not include the cost of the power supply or custom chassis. The total cost of the four ROACH boards includes purchase of rackmount chassis, power supplies and unused DRAM memory modules. Costs of the compute server include a disk storage array of 2TB and a 10GbE network card (which is not used in this design).

The network switch is a 20 port Fujitsu XG2000-CX4 unit. Only four ports are used for the 32-input design, with the additional empty slots reserved for future use (PAPER-64). The power consumed at idle is 48W, which increases to 49W when the four ports are driven. 10GbE CX4 cables continually transmit data when connected (in the form of idle sequences when no data is present), and there are only negligible savings in the switch's packet processing systems when not exchanging any correlator data.

The listed prices and power consumption for the clocking subsystem includes the GPS, RF amplifier, splitter and all associated power supplies and rackmount enclosures.

A fully-loaded ROACH board consumes 70W from AC mains power, much of which is lost in the commodity 200W ATX AC-DC power supply which runs with a very light load, making it inefficient. PAPER is not using all on-board ROACH peripherals and each of the X-engine ROACH boards only consume 51W out-the-wall.

Each iBOB together with two quad-ADC cards consumes 25W at 5V DC. All iBOBs share a single Meanwell SP-480-5 480W 5V power supply which operates at a measured 88% efficiency.

The power consumption of the complete digital signal processing chain is dominated by the single control computer which is responsible for control, monitoring, data acquisition and storage. The network switch draws less than 8% of the total power budget when using copper cables.

A.4 The future of PAPER's digital back-end

As of 2011, this author is no longer directly involved with the development of PAPER's digital back-end. This section describes the subsequent work on this system by other PAPER engineers for the purpose of demonstrating this work's further scalability and to provide a background for those wanting to further this work.

A.4.1 Up to 128 antennas

The PAPER array scaled to 64 antennas in 2011 by adding additional iBOBs and ROACH boards to the existing switch. The existing design scaled directly; the X-engines were recompiled for a larger number of inputs but no architectural changes were required. The limitations of the iBOBs will be exposed when PAPER scales to 128 antennas:

The corner-turn in the F-engines are currently double-buffered in the iBOBs' external SRAM chips. When PAPER reaches 128 antennas, a single iBOB will no longer have sufficient SRAM in which to perform the required matrix transpose, due to the required of the packet size increase (see §5.3). By this stage PAPER will already be performing in-place re-ordering in this transpose operation and no further memory optimisation will be possible. The number of inputs per board will need to be halved, data further quantised, or a different processing platform with more SRAM employed.

At the same time, ROACH will no longer provide sufficient logic capacity to host a monolithic 128-antenna X-engine core. A possible solution would be to retire the current iBOBs and introduce a new processing platform for the X-engines (possibly GPU or ROACH-2 based). ROACH-2 offers a four-fold increase in compute power over ROACH-1, with less than a two-fold increase in cost; ROACH-2s are thus more cost-effective and should be used in place of ROACH-1s wherever possible. Employing ROACH-2s in the X-engines would free the current ROACH-1 X-engine boards to be repurposed as F-engines, making it a very cost-effective upgrade.

A ROACH-2 X-engine solution should provide sufficient capacity for up to 256 inputs before a single ROACH-2 again runs out of logic capacity to host a 256-input X-engine core and the process is repeated; now introducing

ROACH-3 for X-engines and augmenting the ROACH-1 F-engines with the old ROACH-2 hardware.

A.4.2 GPUs

An alternative to ROACH-2 based X-engines is to use GPUs for the X-engines. This is a sensible approach for large-N designs, which become compute bound rather than IO bound. The $O(N^2)$ compute requirements begin to dominate the $O(N)$ input bandwidth and allows the GPU to be fully utilised efficiently, even with modest bus bandwidths of current commodity PC hardware. In 2013, PAPER deployed GPUs as X-engines for their 128-antenna array, in collaboration with LEDA. This resulted in a cheaper system. The primary drawback of a GPU-based approach is that a larger power and cooling budget is required.

The use of an Ethernet interconnect allowed for this simple exchange of FPGA for GPU. It again demonstrates the flexibility of the design.

A.5 Science

PAPER faces the challenge of attempting to detect a signal significantly weaker than the brightest radio sources in the sky. These bright sources must first be characterised and excised from the data before any science can be done. PAPER's initial arrays are designed to aid ongoing instrument characterisation and refinement and are used to design imaging techniques for such foreground extraction. Figure A.6 shows an all-sky image produced by PAPER in 2011, using this packetised correlator.

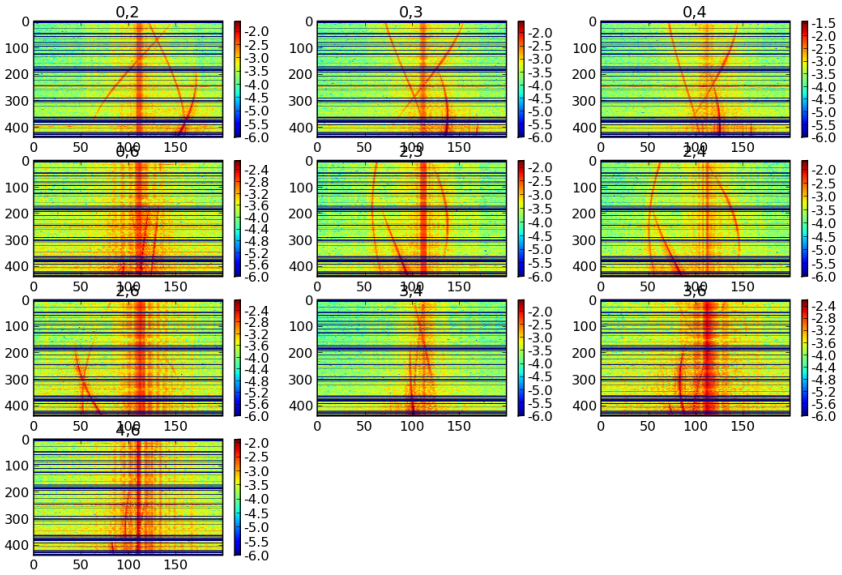


Figure A.4: The Fourier transform of the phase spectrum of each 16s accumulation over time. "Delay transform" on X axis vs time on Y axis. A given source appears at a delay that varies cosinusoidally with time. The curved lines thus show sources moving across the sky over a period of approximately two hours.

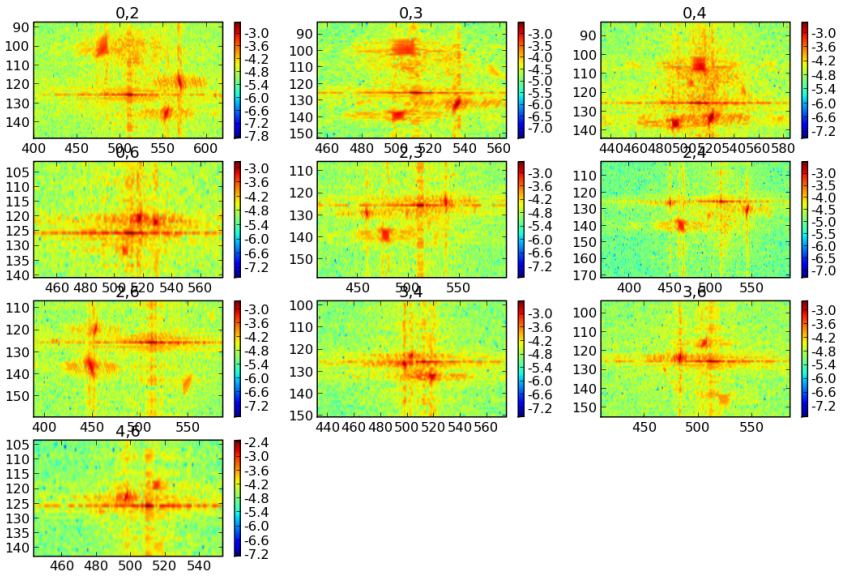


Figure A.5: Interference fringe rate: By taking the processing of Figure A.4 one step further and performing a Fourier transform over time, a crude image is revealed. For sources near the zenith, this is a map of celestial position.

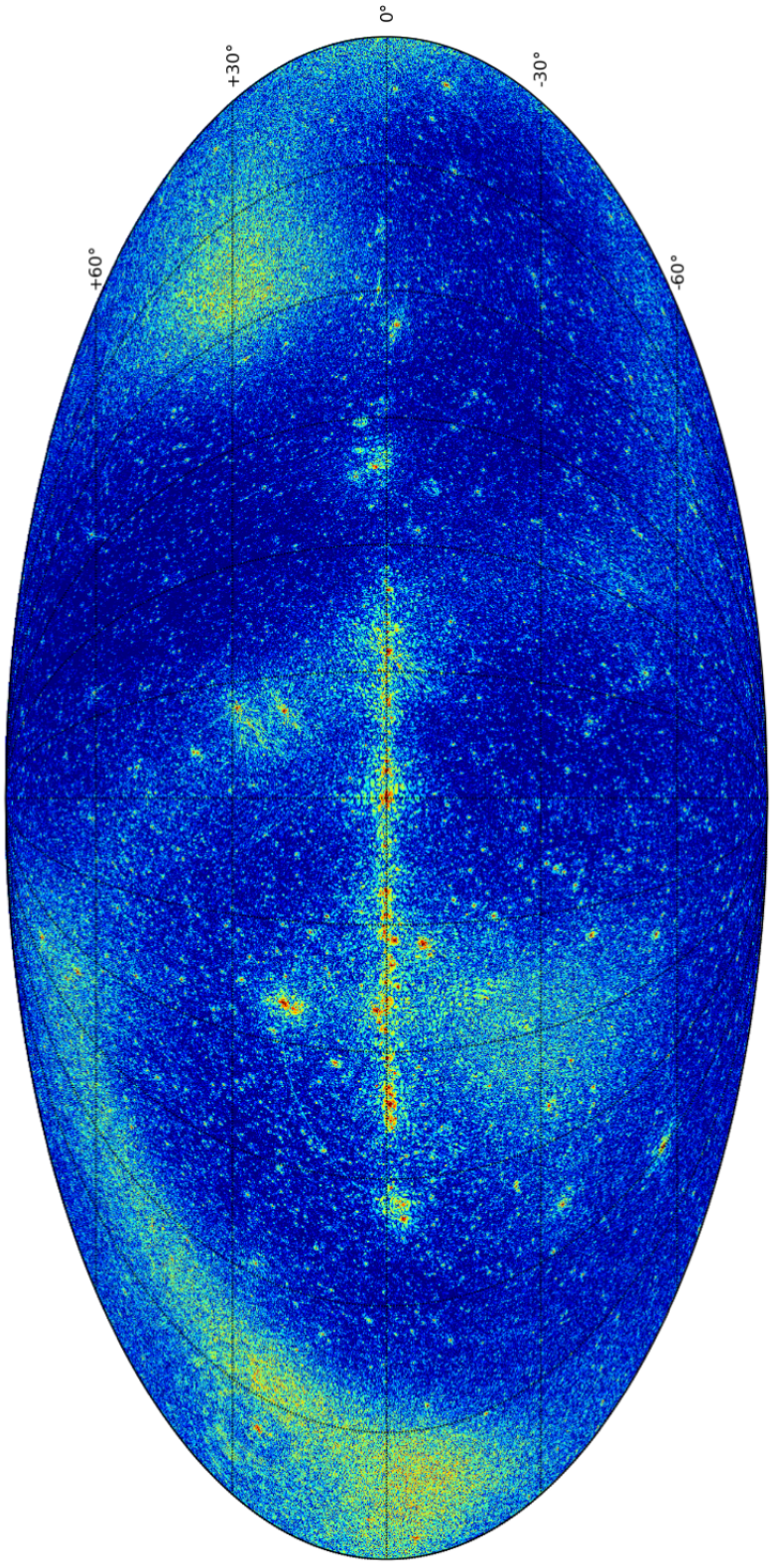


Figure A.6: An all-sky image produced by combining the data from the northern hemisphere PAPER array in Green Bank with the southern hemisphere array in the Karoo. Bright foreground sources have been removed.

A.6 Conclusion

The PAPER correlators represented the first deployments of this work. A number of designs based on different hardware platforms were successfully deployed, including the use of different ADCs. The central Ethernet switch remained a common theme throughout these deployments.

The PAPER correlator scaled with the array. More processing capacity was added as antennas were integrated. In this way, PAPER has demonstrated scalability. But the latest iteration of the design also employs GPUs; demonstrating the heterogeneous capability of the architecture.

The system was upgraded and improved with each deployment. PAPER was particularly valuable in assessing reliability and improving design robustness as their arrays operated remotely for a number of years. It resulted in improvements to the timing and synchronisation systems, debugging facilities and control software.

Two copies of the correlator remain operational on both their Karoo and Green Bank sites.

Appendix B

KAT-7

KAT-7 is a seven antenna precursor to the larger 64-antenna MeerKAT instrument. It is an engineering test array, allowing for experimentation, evaluation and validation of proposed MeerKAT designs.

This section will discuss the KAT-7 array with a specific focus on its digital back-end (DBE) and the customisations made to the design discussed in §4.

Figure B.1: The KAT-7 array in the Northern Cape, South Africa.

KAT-7's seven 12m prime-focus dishes were commissioned in 2010. Each dish is fitted with a dual, linearly-polarised, cooled, single-pixel, L-band feed. The received broadband, analogue sky signals are mixed onto optical fibres for transport to a central facility which performs the filtering, downconversion, amplification, digitisation and processing functions.

This processing facility is located 5km from the antennas, behind a hill, in an RFI shielded container. Here, analogue signal conditioning is applied before a 256MHz band is selected from the L-band sky frequencies of 1195MHz to 1950MHz by a dual-conversion superheterodyning down-converter, mixing to an intermediate frequency (IF) centred at 200MHz, which is then digitised.

B.1 The KAT-7 Digital Back-End

Eight F-engine boards and eight X-engine boards form the KAT-7 digital back-end. Although initial requirements for KAT-7 could have been met with half this processing capacity, the excess space on the FPGAs can be used for future upgrades or trialling new instruments.

16 analogue inputs are available, designed to digitise each antenna's two 72-328MHz IF signals. A single, dual-input, 8-bit KATADC, operating at 800Msps, is hosted by each ROACH F-engine board.

The system architecture remained true to original packetised concept, as shown in Figure 4.2. Each processing board has a 10GbE port connected to a central Ethernet switch, allowing for arbitrary traffic flows and patterns.

The back-end is constructed with eight dual-pol inputs. By keeping to 2^n scaling, addressing on the processing boards can be easily and efficiently achieved by slicing counters. The spare inputs are used by an automated test signal generator for continuous, online system testing and verification. Later, this can be used to evaluate the first MeerKAT antenna against the existing KAT-7 dishes.

The Fujitsu XG2000 switch, used for interconnecting these engines, has 20 10GbE ports in total, with 16 used by the processing nodes. DBE output data rates of up to 40Gbps are thus possible.

KAT-7 aims to keep operating modes simple and stand-alone. Rather than constructing a single highly configurable FPGA bitstream with runtime

Figure B.2: The KAT-7 digital back-end (DBE) rack prior to deployment. The F-engines can be seen on the right, with analogue inputs. The left rack hosts the X-engine boards and the 10GbE network switch.

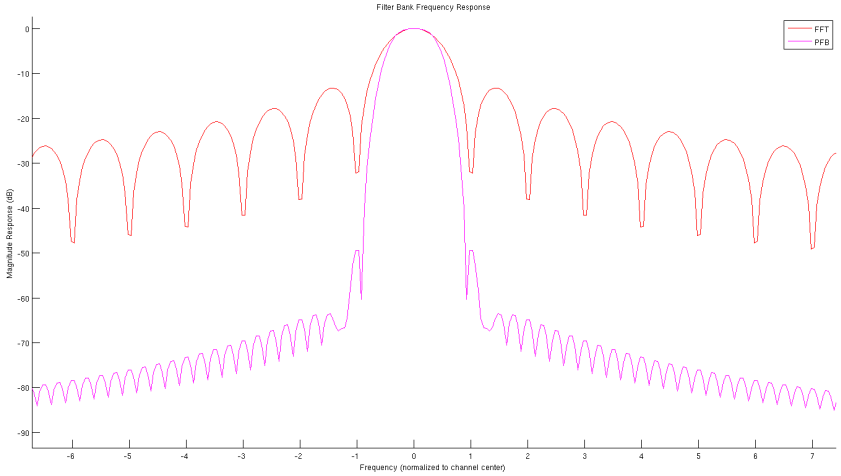


Figure B.3: The simulated channel response of a single channel from KAT-7's polyphase filterbank.

programmable parameters, I choose to construct a reprogrammable machine that can be reprogrammed to perform simple operations which are chosen at compile time. To demonstrate this operation, the following channellisation modes are supported on KAT-7:

OH Spectral line correlator 4096 channels over 1.5MHz

OH Spectral line correlator 4096 channels over 6.25MHz

OH Spectral line correlator 4096 channels over 12.5MHz

H1 Spectral line correlator 4096 channels over 25MHz

Wideband correlator and beamformer 1024 channels over 400MHz

High resolution wideband correlator 8192 channels over 400MHz

The following sections will discuss the design of these modes.

B.2 Wideband correlator

KAT-7's digital back-end (DBE) started with a single *wideband correlator* mode, deployed in December 2010 after 3 months of development. This mode channelises the entire digitised 400MHz wideband signal into 1024 frequency channels using an 8-tap polyphase filterbank and performs full correlation on 16 inputs. Later, a higher spectral resolution wideband mode was added with 8192 channels, but with a 2-tap PFB and lacking the fringe rotation and delay tracking features due to resource shortfalls in the F-engine ROACH boards.

B.3 KAT-7 hardware resource utilisation

The hardware allocation for KAT-7's wideband correlator mode was significantly over-specified. 16 ROACH-1 boards were allocated to the task of digitising and channelising the data (eight F-engines, two inputs each), and cross-multiplying and accumulating the data (eight X-engine boards, two cores on each). Extra hardware was purposefully deployed so that there would be reserve capacity for debugging functions and for adding future, unforeseen features. In the X-engines, some of this reserve capacity was later used for beamforming.

Tables B.2 and B.3 show the breakdown of the FPGA resources for the wideband correlator.

The KAT-7 DBE design further streamlined the PAPER system and optimised various components. This is clearly evident in Table B.3 from the X-engine's reduced control and monitoring logic requirements (KAT-7's 14%, down from PAPER's original 38%) and the 10GbE core (half the logic and BRAM requirements).

Since KAT-7 was specified to have eight ROACH-1 X-engine boards, the excess hardware resources were put to good use for additional debugging and characterisation functions. These components are not required for normal operation, but allow for sampling of the signal throughout the processing chain and for injection of various waveforms. If additional resources are required for DSP operations, these debugging functions can easily be progressively

Table B.1: KAT-7's wideband digital correlator specifications.

Digitised Bandwidth	400MHz
Processed Bandwidth	400MHz
Number of inputs	8 antennas × 2 polarisations
Number of frequency channels	1024
Frequency resolution	390.625KHz
Filterbank type	Polyphase, 8 tap
Filterbank window	16384 point Hamming
Spectral channel isolation	better than 51dB after 390kHz better than 80dB after 4MHz
Channel equaliser gain	-16384-16384i to 16383+16383i in linear steps of 0.5
Initial fringe phase Fringe rate range Internal interpolation	$-\pi$ to $+\pi$, steps of 9.59e-5 radians 0 - 47.683Hz in steps of 727.6μHz 3.638pHz
Delay compensation range Delay rate	0 to +81.92μs in steps of 3.638ps -32768 to +32767s/s, steps of 3.638ps/s
Correlator type	FX, 36 baselines, full cross-pol
Number of correlated bits	4-bits real + 4-bits imaginary
Vector accumulator	32-bits real + 32-bits imaginary
Accumulation period	0.65536ms - 32.578days, steps of 655.36μs (<224.39s to guarantee no overflows)
Output data format	SPEAD over 10GBASE-CX4

Table B.2: **KAT-7 Wideband F-engine FPGA resource utilisation by function as a percentage of ROACH's Xilinx Virtex 5 SX95T FPGA's resources.**

Function	Logic	BRAM	DSP
Two FIR and FFT cores	41%	47%	34%
Debug functions	18%	8%	2%
Control and Monitoring	12%	1%	2%
Fine delay and fringe compensation	4%	6%	4%
10GbE transmitter	4%	3%	0%
Output packetisation and data re-order	4%	3%	0%
ROACH base infrastructure	4%	0%	0%
Coarse delay	2%	7%	1%
Equaliser and requantiser	1%	0%	3%
Total Design	90%	75%	46%

Table B.3: **KAT-7 Wideband X-engine FPGA resource utilisation by function as a percentage of ROACH's Xilinx Virtex 5 SX95T FPGA's resources.**

Function	Logic	BRAM	DSP
Debug functions	16%	7%	2%
Control and Monitoring	14%	0%	0%
Two X-engine cores	9%	7%	25%
10GbE transceiver	8%	3%	0%
Two vector accumulators	6%	2%	0%
Input unpack and buffer	5%	3%	0%
Output packetisation	4%	2%	0%
ROACH base infrastructure	2%	0%	0%
Total Design	64%	24%	27%

traded-out.

There is an abundance of debug and monitoring and control points in the engines, which consume a significant portion of the FPGA's resources. However, these resource requirements remain static on each board when scaling the system (the same monitoring points are used, regardless of the number of antennas, bandwidth or number of channels processed by each engine). For small systems such as KAT-7, these monitoring and debug features are a significant overhead. But for larger designs, such as MeerKAT, they are projected to consume less than 2% of logic resources, even if all the monitoring points are retained.

KAT-7 has also added additional functionality to allow vector accumulators to start at arbitrary times. This is used to synchronise the correlator to external devices such as noise diode calibrators, at the added cost of some interface logic.

B.4 Narrowband modes

KAT-7 desired narrowband modes for spectroscopy. This requires a much higher spectral resolution (down to 530Hz), but is only needed over narrow bandwidths (typically 1 - 30MHz). It is typically used in spectral line work to observe the presence of ionised gasses at known frequencies.

An initial option for these narrowband modes, as employed by PAPER in their early systems, was to use a DDC to select a narrower band and to then further process this as normal. This allows arbitrary centre frequency tuning but only produces one tunable band. The required tuning resolution and filter specifications (ripple and rejection) determine the size of the DDC, which can be significant for very narrow bandwidths, if employing FIR filters. KAT-7 had a goal of providing multiple simultaneous spectral windows, and so experimentation with 2-stage FFTs was undertaken.

Three narrowband modes were added in 2012 based on a 2-stage filterbank architecture, where the first PFB selects a coarse channel which is further channelised by a subsequent PFB. Chaining filterbanks in this fashion allows for very high spectral resolutions to be realised across very wide bandwidths, at the cost of two very modest-sized FFTs and a matrix transpose memory.

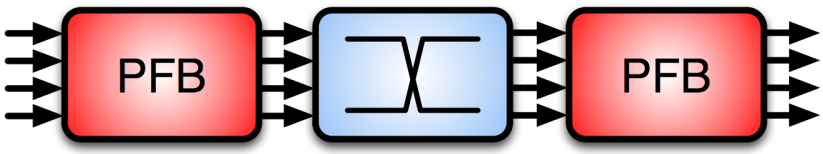


Figure B.4: High spectral resolution over wide bands can be obtained by performing the filterbank operation in two stages: first a coarse channelisation and then further finely channelising each coarse channel using a second filterbank.

By selecting individual channels from the first PFB for further channelisation, multiple spectral windows can be created, provided sufficient memory exists for the matrix transpose operation. The limitation is that these frequencies cannot be arbitrarily chosen, as the coarse channeliser is based on a PFB consisting of a FIR filter and FFT, creating equi-spaced channels of equal size.

There are a number of complications that arise with this implementation. Signals with spectral content near a coarse channel transition will have leakage into the adjacent coarse channel. As each of these coarse channels are subsequently finely channelised, in a critically-sampled filterbank, this leakage power will alias in adjacent coarse channels and appear mirrored about the coarse channel transition frequency. In this case, broadband artefacts are unavoidable and the sharpest possible filter rolloff should be used in the first PFB. The problem can be greatly alleviated if an oversampling PFB is employed, allowing the out-of-band aliased artefacts to be attenuated down to insignificant levels and overlapping channel portions would then simply be discarded.

There is also a concern for power and SNR loss towards the edge of each coarse channel. Half the signal power is lost at the band edges if the FFT channels are designed to intersect at half power points (see Figure 4.10). Widening the PFB's response is not a practical solution as this would increase the leakage from adjacent bins.

This problem can be simply overcome by processing more bandwidth in

the coarse channels than would otherwise be needed. Only the central portion of the channel is used and the outer portions of the band are discarded. However, quite apart from being wasteful, this introduces another problem: since the coarse FFT restricts selection to discrete centre frequency choices, portions of the band will necessarily lie in the discarded transition zone between coarse channels, preventing arbitrary frequency tuning. This can be partially overcome by selectively frequency shifting the incoming signal by half a channel width, for example.

KAT-7 experimented with a number of these techniques within the limitations of the existing DSP filterbank library blocks, all of which are critically sampled. Because KAT-7 features a tunable RF front-end, these digital modes are practically never tuned in frequency; all the tuning is done in the analogue front-end systems and so this functionality was ultimately removed from the digital systems. Also, due to memory restrictions on the F-engines, only a single spectral line is processed, which completely defeats the purpose of the 2-D FFT approach. MeerKAT will likely adopt independently-tunable DDCs for its narrowband modes. Very narrow bands can be selected by employing CIC filters rather than FIR filters.

KAT-7's narrowband H1 mode performs two sequential FFTs, as in figure B.4. The first, coarse FFT, channelises the 400MHz band into 32 equally spaced channels. The output from one of these channels is then buffered and supplied to a second, high resolution FFT which further channelises this sub-band. This architecture allows for cheap, simultaneous multiple narrowband observations. Additional buffer space is required to buffer a block for processing for each additional channel. The second FFT can be time-shared to cover the entire band. Alternatively, if the entire band is not needed, resources can be saved in the second FFT as the reduced data rates allow for serialised FFT implementations.

B.5 Adding the beamformer

Together with Andrew Martens, the simple boresight beamformer described in §4.6 was added in 2012 to demonstrate the beamforming capability.

This effectively demonstrated the reconfigurability and flexibility of this

work's architecture. The beamformer was added at minimal cost by sharing the channelisers, interconnect and all routing, timing and buffering logic with the correlator.

Table B.4: The incremental FPGA resource cost for adding a boresight beamformer to KAT-7's existing wideband correlator X-engines.

Function	Logic	BRAM	DSP
Two Beamformer engine cores	9%	6%	4%
Asynchronous packet stream merging	7%	8%	0%
Total additional resources for 2 B-engs	16%	14%	4%

Table B.4 shows the modest incremental cost for each beam. Additional B-engines can be added to fill the FPGA device. The buffering required to merge multiple streams over a single 10GbE port is significant. The BRAM requirements can be halved if smaller packets can be accommodated. In this case, KAT-7 emits jumbo packets of 8192 bytes. This could easily be halved (and thus also halving resource requirements) to 4096 bytes without affecting performance.

B.6 Verification and early results

KAT-7's digital back-end is tested according to System Engineering practices. Rigorous Qualification Test Procedures (QTPs) and Acceptance Test Procedures (ATPs) were defined to test functionality, performance and interfaces at both subsystem and system levels.

In the case of the KAT-7 correlator-beamformer (CBF) subsystem, these tests take place wholly within the laboratory until they pass all tests. Thereafter, it is integrated with other subsystems and the integrated KAT-7 system is tested as a whole.

The results of these tests are compiled into Acceptance Test Reports (ATRs). For KAT-7, separate ATPs are conducted for each CBF mode.

Many of the KAT-7 CBF ATPs relate directly to the effects of the underlying ADCs' responses and performance (such as VSWR, input cross-talk,

frequency response, Allan-Variance, DC offsets, non-linearities, compression points etc). These are of little concern for anyone reading this work, as designers are free to choose an appropriate ADC for their application, and performance will vary correspondingly.

Rather, this chapter will focus on effects of the DSP operations. The entire report is too large to append here, so I will extract key digital findings of the wideband correlation mode acceptance test, in order to demonstrate correct operation or illustrate some simple test methodologies. Should you be interested, KAT-7's complete ATP and ATR documents are available from SKA-SA upon request (van Rooyen and Rust, 2012, 2011).

B.6.1 Timing verification

Verifying the correct timestamping of signals throughout the signal chain, and that corrections are being applied at the specified timestamps pose a particular challenge, as the sample period (1.25ns) is too rapid for software to sample and process. Hardware mechanisms must thus be put in place to verify these aspects of the design.

Ultimately, the mechanism employed by KAT-7 to timestamp and apply corrections was qualified by injecting a 1PPS signal into the ADC inputs to create a known-input signal, and using an oscilloscope on the FPGA pins for verification. Logic was added to threshold the analogue signal and output it to a GPIO pin at various stages through the signal chain. This allows for delays through the signal chain to be measured precisely.

Additional GPIO pins are also used to indicate when a correction was applied. Oscilloscopes can then be connected to these pins to verify relative timing of the analogue signal and when any corrections were actually applied. None of these signals are used in the production system, but are merely present to ease formal verification.

FPGA logic was also added to sample the analogue signal together with the internal FPGA timestamp, without any integration. This allows timestamps to be easily verified.

The system operates as expected: data is timestamp accurately to within one sample period, and fringe stopping and delay tracking coefficients can be

loaded at arbitrary timestamps, with a timestamp resolution equal to a single FPGA clock period, and these are applied at the requested time, without any offsets.

B.6.2 Channelisation

Verification of the channeliser's performance was a critical part of the ATP. Many of these tests are limited by the correlator's 4-bit requantisation operation, which restricts the system's instantaneous dynamic range.

Measuring the channeliser's dynamic range by observing the output products is limited by the 4-bit requantisation operation. But by dithering with very little (approximately 1LSb of the 8-bit ADC) noise and then averaging, the channeliser's response becomes evident.

To measure a channel's dynamic range, a 72MHz CW tone was selected as a non-integer divisor (factor) of the sample clock (800Msps/72MHz = 11.11$\dot{1}$ samples), near the bottom end of the useable band. The CW tone's input power level is varied above and below the nominal input level to determine the available dynamic range. Naturally, the 4-bit quantisers must be re-adjusted regularly during the test to measure over the PFB's full dynamic range if the output products are referenced (alternatively, a chipscope-like tool could be used to capture data before quantisation). This test was repeated with selected additional frequencies at 200MHz (one fourth of the sample rate; exact centre of the band) and 328MHz (non-integer multiple near the upper end of the usable band).

The channel response test was performed by sweeping a near-FS CW tone through individual channels, and observing the power of adjacent individual channels. By design, all PFB channels have an identical response. Thus, we need not measure all channels. Some of these are plotted in Figure B.6.

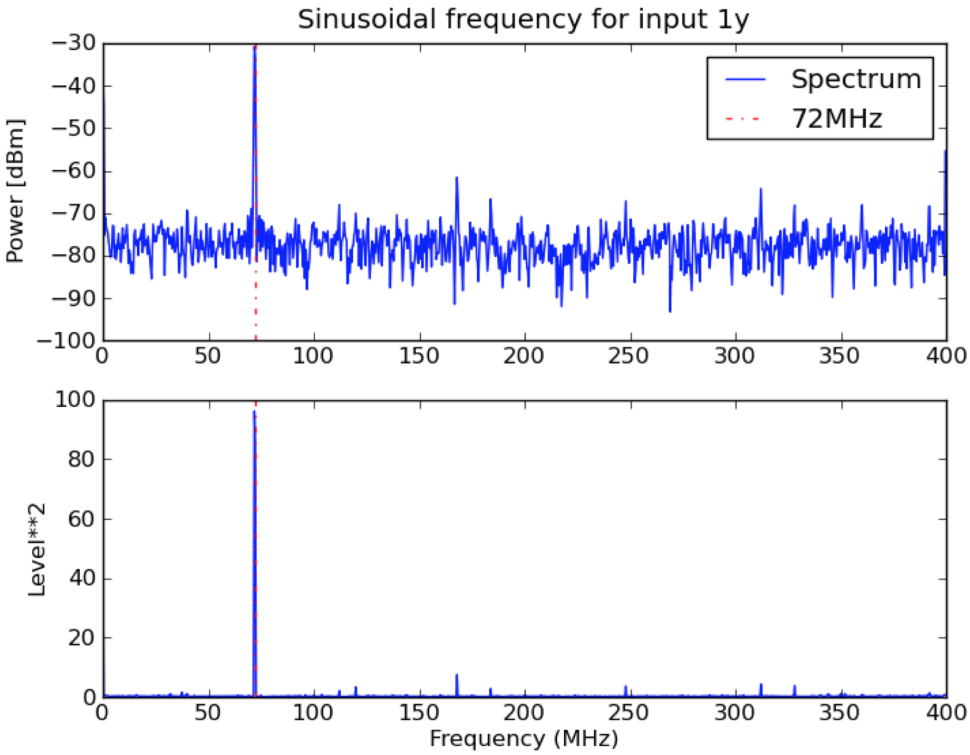
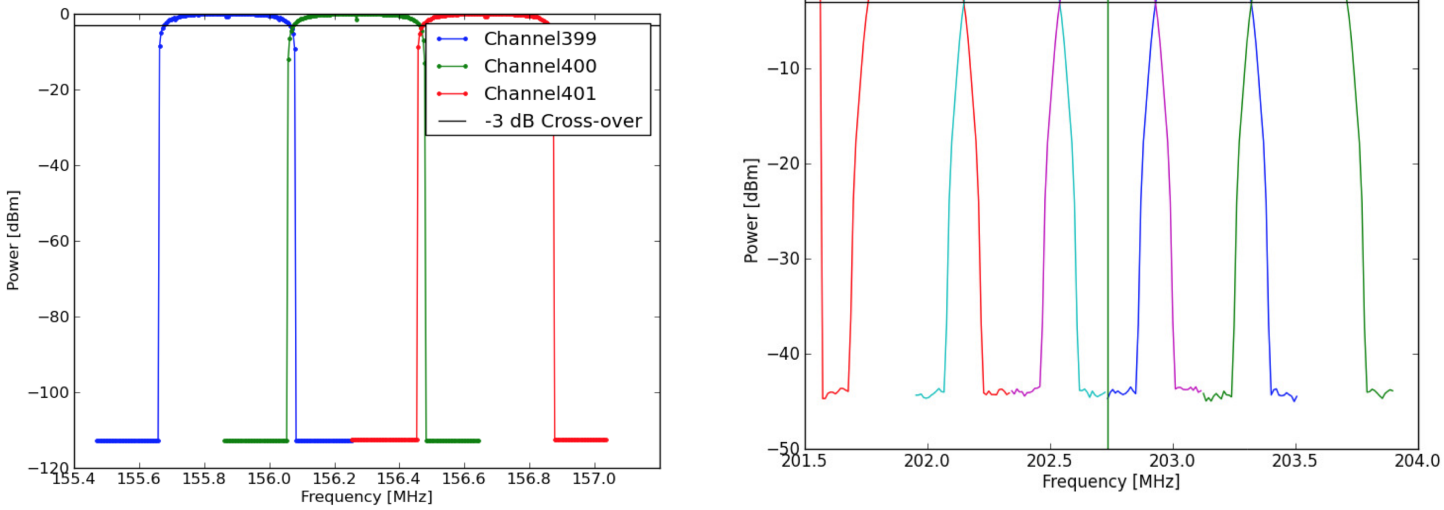


Figure B.5: Output spectrum from KAT-7's 1024-channel wideband correlator, plotted on both an uncalibrated linear Y-axis, and a calibrated logarithmic Y axis. Interleaving artefacts are visible. Reproduced directly from the ATP report.



(a) KAT-7 channelisation without noise.

(b) KAT-7 channelisation with noise.

Figure B.6: Measuring KAT-7's channeliser response. The leftmost figure illustrates the difficulty in attempting to measure a single channel's response by sweeping a CW tone through the channel. Due to 4-bit requantisation, instantaneous dynamic range is limited to only 16.9 dB (max level of 7, symmetrical about zero). Below -16.9dB FS, the output values simply drop to zero. The highly non-linear transfer function of the 4-bit quantisation also distorts the channel shape. The figure on the right adds noise to dither the measurement, and then averages each measurement for one second. The noise floor of each channel is now visible, along with an accurate representation of the channel rolloff characteristics. Unfortunately, this test now takes much longer to execute, as each measurement must be integrated for one second. Note difference in Y-axis scales. Reproduced directly from the ATP report.

KAT-7 System Engineers specified channels with uniform responses, with channel intercept points at the half power points (-3dB) in order to preserve total input:output power. Figure B.6 adds a horizontal line at -3dB, clearly showing the channel intercept points are at -3dB.

Simulations of KAT-7's channeliser predict channel isolation (the power leakage of all other channels into any one channel, or sometimes, the effect of a given channel on any other channel) in excess of 50dB away from the adjacent channels (from Figure B.3). Measuring the digital channeliser's contribution to the isolation is difficult, as we're again limited by 4-bit dynamic range and ADC performance (specifically, SFDR) – the leakage is well below the 4-bit measurable 16.9 dB measurable range. But dithering and averaging again allows us to achieve an increased resolution. Care must be taken not to inject too much noise, as this increases the system noise floor and can adversely affect the measurement. Since all PFB channels' responses are identical, if we avoid tallying harmonic channels (to limit ADC contributions), we can at least cite a minimum achievable figure. From Figure B.6, this is clearly better than 40dB.

Attempting to measure channeliser linearity proved fruitless, as this was dominated by contributions from the ADC and the 4-bit quantiser.

B.6.3 Correlation

The primary operation of KAT-7's digital backend is that of correlation. The instrument must be able to accurately measure the time difference between two input signals. This is simply demonstrated in the static case by injecting a common broadband signal into two analogue inputs, with a known time delay added to one (typically by using different length cables). To cross-check, the delay contribution of any given cable is easily measured using a common laboratory instrument, a Vector Network Analyser (VNA).

Figure B.7 shows a flat delay slope over the band when matched cables are used on inputs from a common noise-source on KAT-7. Figure B.8 shows the delay slope over the band when a delay difference of 7.5ns is introduced between input signals (measured using a VNA).

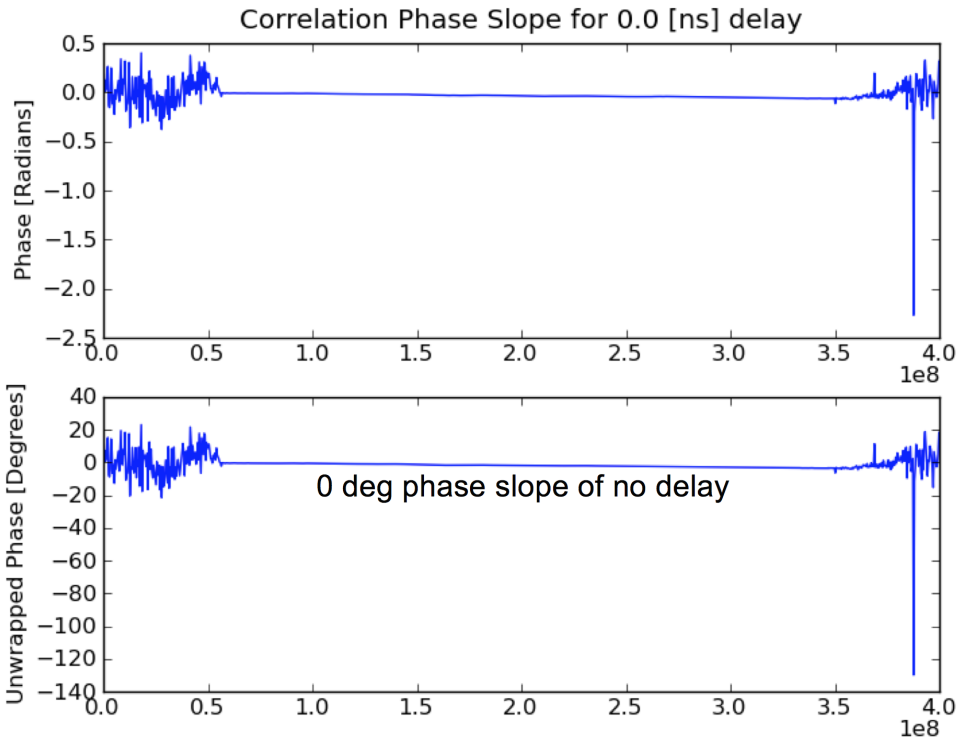


Figure B.7: The cross-correlation phase slope across the band when equal length cables are used is zero, shown in both radians and degrees. The X-axis is frequency in Hz (ie 0-400MHz). Note that over the 400MHz usable band, a bandpass filter on the inputs restricts the signal to the central 250MHz. The "scruff" at the edges is due to there being no signal present. Reproduced directly from the ATP report.

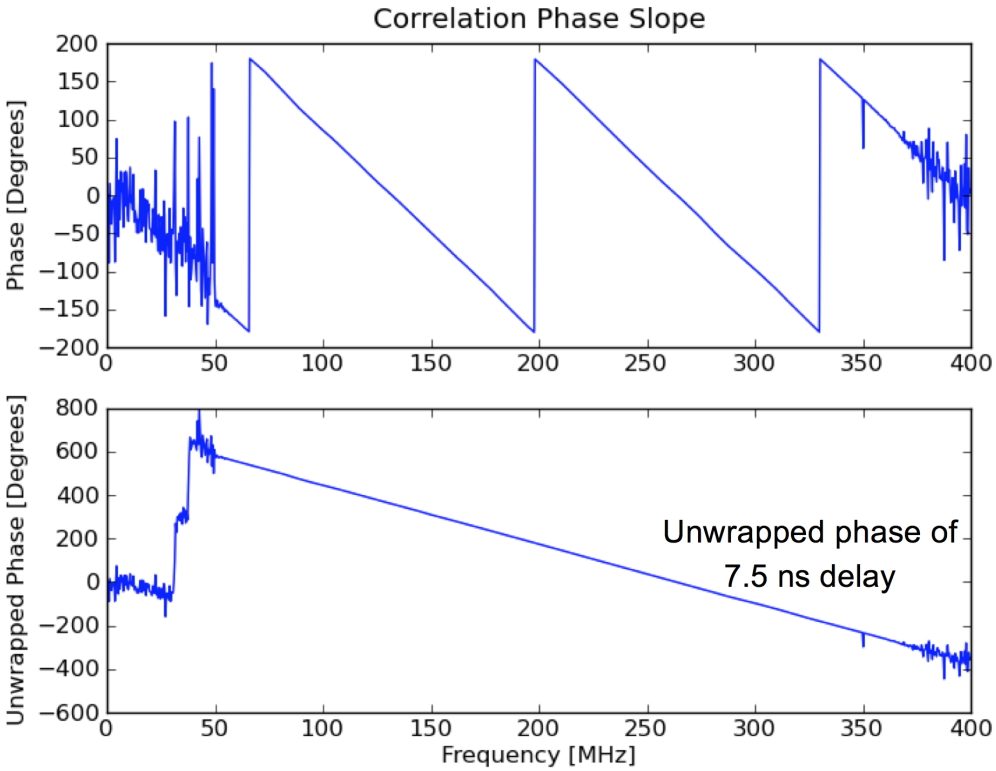


Figure B.8: The phase slope across the band when a cable delay of 7.5ns is inserted. Note that over the 400MHz usable band, a bandpass filter on the inputs restricts the signal to the central 250MHz. The "scruff" at the edges is due to there being no signal present. Reproduced directly from the ATP report.

B.6.4 Phase tracking

Testing the phase and delay tracking functionality is difficult without access to an actual telescope, because generating wideband, glitch-free, continuously variable analogue delays in the laboratory is impossible with standard laboratory equipment. Instead, we adopt the approach of keeping the inputs static, and then observing the effects on the output products of adjusting the various system parameters.

Figures B.9 and B.10 illustrate the effect of using hardware assistance for loading new coefficients at a precise time (in this test case, synchronously with the integration period), and also the value of continuous hardware interpolation (updated internally every FFT spectrum in the case of KAT-7).

B.6.5 Science results

The KAT-7 instrument was originally intended to be a technology demonstrator: an engineering test platform used to vet new construction techniques and allow for proof-of-concept implementations. However, at the time of writing, KAT-7 was in the process of being commissioned and actually doing basic science.

As an example of one such commissioning activity to demonstrate end-to-end imaging performance of KAT-7 with this correlator, Figures B.11 and B.12 were created, illustrating images captured during these phases. They establish that the system is able to produce scientifically useful images.

B.7 Conclusion

KAT-7's fully-featured wideband correlator was successfully deployed in a matter of months, with installation taking place in December 2010. It represented the second long-term deployment of this architecture (after PAPER). KAT-7 represented a refinement of the design, including modifications to the timing and synchronisation system and the monitoring and control software. The topology was also subtly different as it did not include the loopback optimisation that PAPER employs; all processing nodes are directly connected to the network switch.

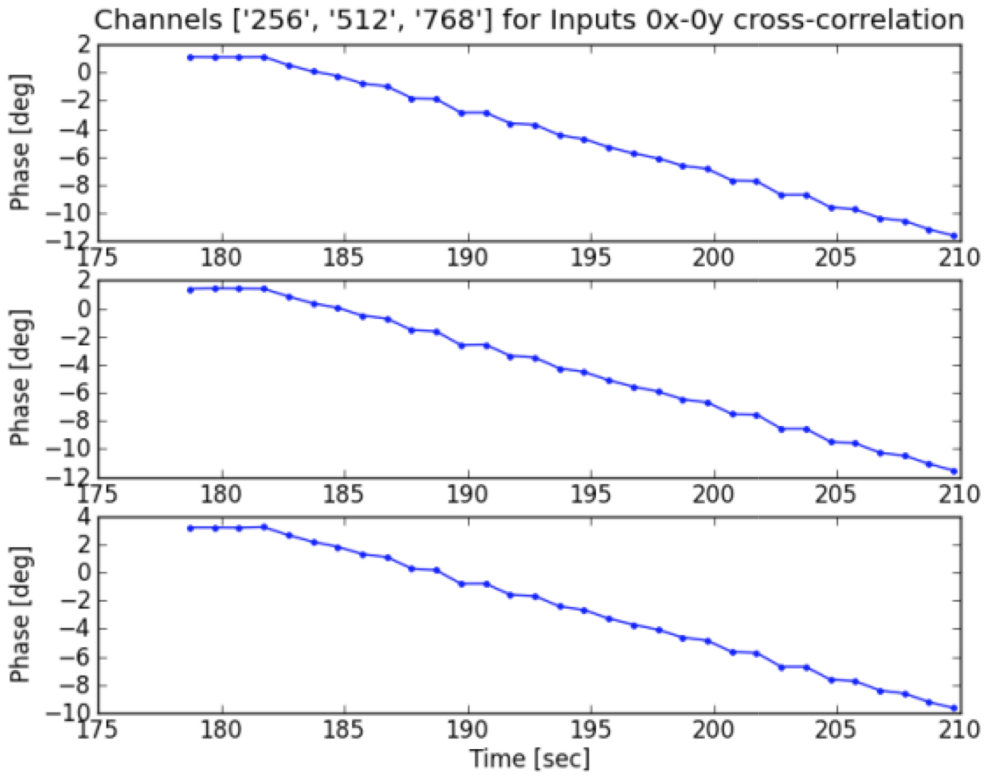


Figure B.9: This figure shows the phase of a single channel over time. Software steps the phase offset every second, by one degree, without any attempt to align the application of the new offset to an integration period, or to correct for network delays (commands are issued remotely). Notice that because the application of the coefficients are not synchronised, the slope is not linear. It is difficult to accurately apply delays in software, without hardware assistance. Reproduced directly from the ATP report.

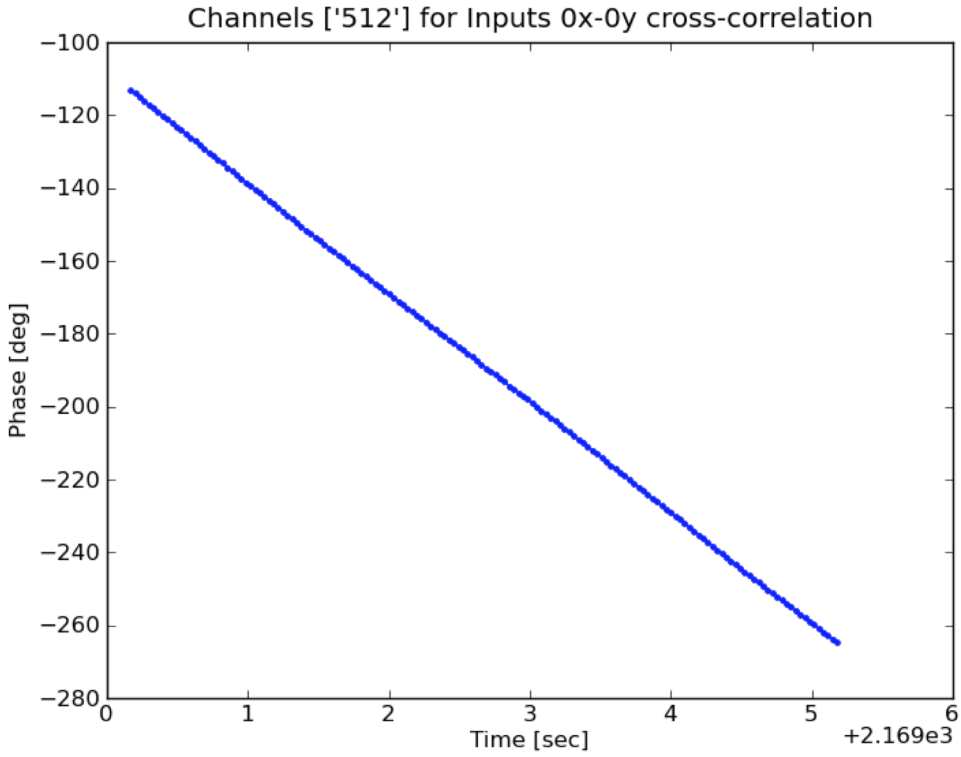
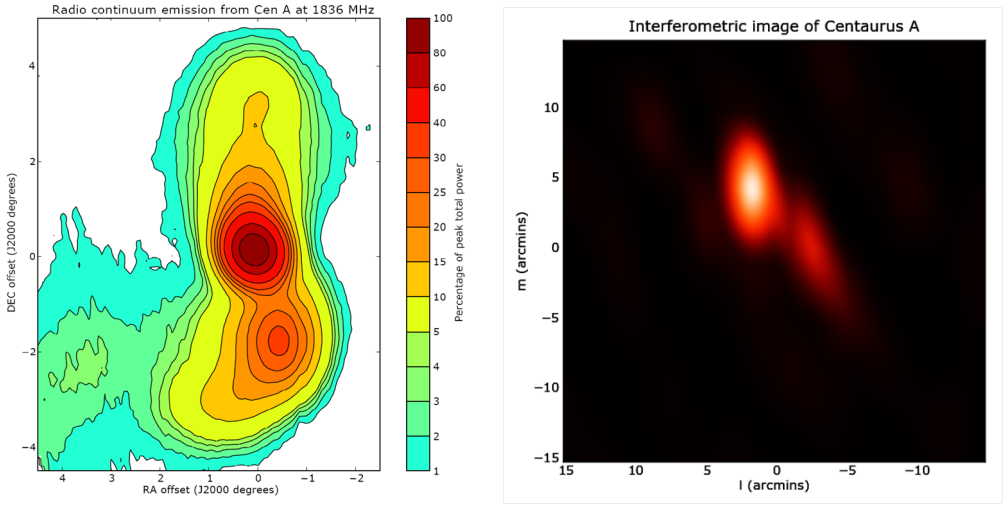


Figure B.10: In this plot, software issues a single command at the start of the recording, along with a linear rate of change for the delays. Hardware then continues to apply the required delay, and readjusts this by the required rate of change after every FFT. Even with a reduced integration time of 33ms, the plot is smooth. Hardware assistance such as this is necessary for good delay and phase tracking performance. Reproduced directly from the ATP report.



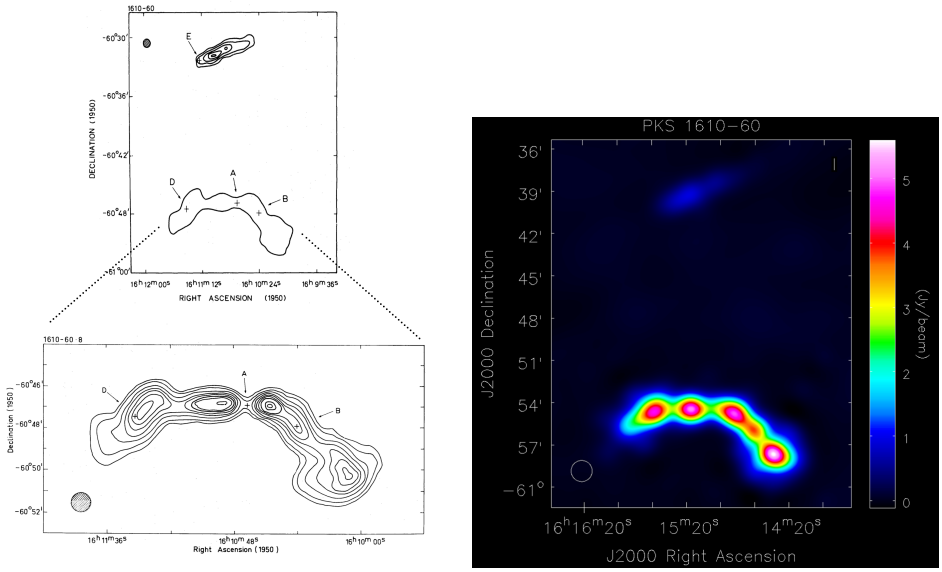
(a) Single dish scan

(b) Interferometric scan of centre

Figure B.11: In 2009, after the first KAT-7 dish was assembled, it was used to form the leftmost image of *Centaurus A* by scanning across it. Later, when four dishes were brought online, an interferometric image was formed with much higher angular resolution.

In addition to basic wideband correlation operations (with various channelisation options), KAT-7 trialled narrowband modes for spectral line work and also later added a beamformer mode. This coherent tied-array output is available simultaneously with the wideband correlator products, and the two instruments share channelisers. These instruments have all been successfully used. They were simple to deploy remotely, needing no physical intervention or additional hardware.

KAT-7 proved a success, and the role of the Digital Back-End (DBE) has increased significantly in MeerKAT, KAT-7's successor.



(a) Christiansen et al. at 1415MHz, 1977

(b) As observed with KAT-7

Figure B.12: Galaxies around PKS1610-60.8 observed using KAT-7, compared to Christiansen *et al.* (1977)'s results, including the weaker emission around PKS1610-60.5. This image was formed using cooled feeds on all 7 antennas with the CASA imaging package. Individual galaxies are resolved by KAT-7.

Appendix C

Looking ahead to MeerKAT

MeerKAT is a much more ambitious project than KAT-7, with specifications that will make it one of the most sensitive telescopes in the world. It will have a signal processing system that includes a number of novel concepts.

Table C.1 shows the primary differences between the KAT-7 and MeerKAT signal processing requirements. The design of this instrument has not yet been finalised but Figure C.1 shows a possible block diagram implementation, including the output products. The MeerKAT system needs to support a number of new instruments, many of which must operate simultaneously. MeerKAT's signal processor will thus be a much larger system than the KAT-7 prototype's. The correlation operations alone will need nearly two hundred times the compute power. The MeerKAT central processing facility will have a computational capacity of over two peta operations per second in its hundreds of FPGAs and GPUs, which will place it amongst the most powerful computers on the planet at the time. This processing facility is to be located on-site, in a partially bunkered building, behind a hill for added RFI shielding and will be directly supplied with digital, rather than analogue, signals.

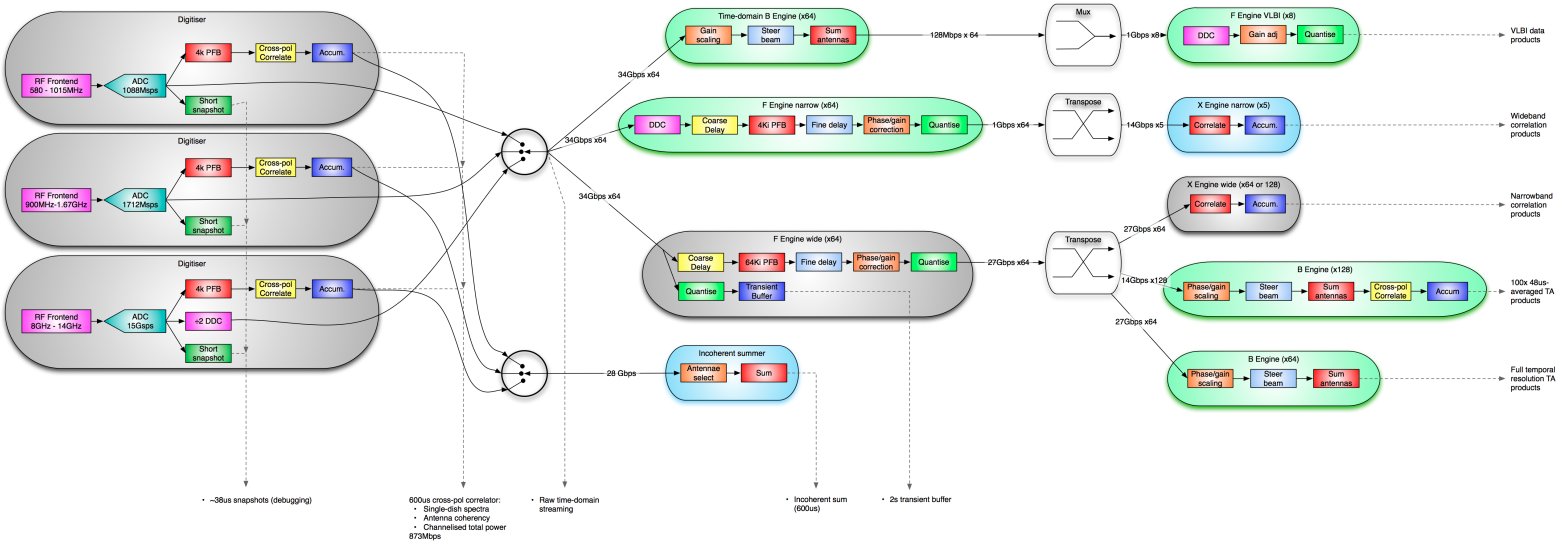


Figure C.1: The MeerKAT DBE design has not yet been finalised, but this diagram shows a possible signal processing pipeline and associated data rates between engines. The engines highlighted in green and blue are not required at all times, and so are able to share hardware resources.

	KAT-7	**MeerKAT**
Digitised frequency bands	72 - 328MHz	0.58-1.015GHz† 0.9-1.67GHz 8-14GHz†
Antennas	7	64
Polarisation	Dual	Dual
Processed BW	400MHz	over 2GHz ‡
Max. num. channels	8192	32768
Min. channel BW	530Hz	1.67KHz
Concurrent beams	1	4
Concurrent spectral lines	1	5
Concurrent modes	1	many
Transient search and time?	no	yes

†Only for MeerKAT phase two.

‡856MHz for MeerKAT phase one.

Table C.1: Comparison of KAT-7 and proposed MeerKAT specifications, as presented at the 2011 MeerKAT preliminary design review.

C.1 Early digitisation

The signal processing system will be an all-digital design, starting with an FPGA and ADC at each dish's feed. This *D-engine* will digitise the sky signal directly, without any analogue mixing, and then packetise and timestamp the data with sub-nanosecond precision for transport to the central processor over Ethernet optical fibres. Placing digital components so close to the feed has significant implications for self-generated radio interference and very careful design and testing will be required.

The advantage of sampling directly at the feed is that it reduces the cost and complexities associated with traditional RF chains. Transporting the signal digitally as opposed to RF over fibre also has the advantage that the

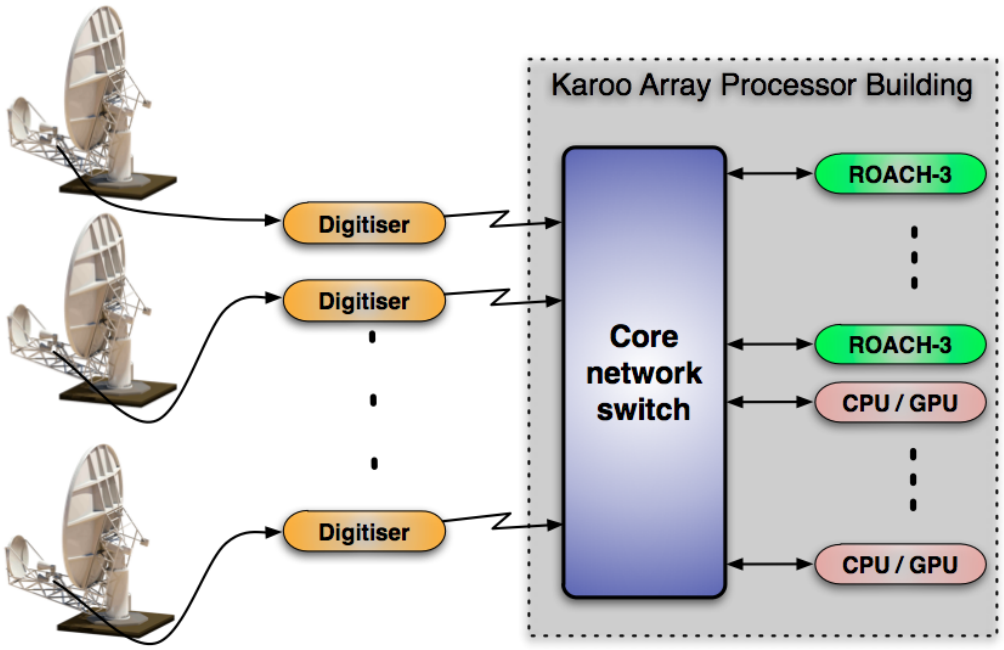


Figure C.2: A physical overview of the MeerKAT digital back-end architecture showing the digitisers located at the feeds, connecting directly to the core network switch. FPGAs, CPUs and GPUs are used as processing engines. A separate high speed network (not shown here) hosts a long-term data archive and offline processors.

optical receiver is no longer susceptible to amplitude and phase variation caused by the temperature changes of optical fibre. It also overcomes the typical ∼40dB linear dynamic range limitations of commercial RF-over-fibre solutions. In this way, MeerKAT hopes to provide a pristine passband with low noise levels and high dynamic range.

MeerKAT will operate in one of four RF bands at any given time: UHF (580MHz - 1015MHz), L-band (1GHz - 1.75GHz) or X-band (8GHz to 14.5GHz), with the fourth band being reserved for future use, possibly covering S-band. The initial deployment will be with the L-band feed. The specifications for both the UHF and L-bands allow for the use of commercially available 10-bit samplers operating at under 2GSa/s. The requirement to digitise at X-band,

however, will likely require ADCs sampling at 15Gsps, which are not currently commodity parts. Resolution requirements are eased somewhat in this wider, quieter band, possibly allowing for the use of a 5-bit part. Initial in-house testing suggests that interleaved ADC units might not be suitable for all astronomy applications due to spectral artefacts, even after calibration (see §3.4.5) and for this reason, a monolithic part has been selected for L-band use.

C.1.1 RFI concerns

The primary concern with sampling so close to the feed, is arguably for RFI, caused primarily by digital clocks and data switching. Figure C.3 shows that ROACH-1 radiated emissions need to be reduced by approximately 100dB before this hardware can be used in the MeerKAT digitiser's location.

To deal with the challenges surrounding RFI, various mitigation strategies are underway:

- The number of electrical interfaces to the digitiser are being minimised with the majority of the external interfaces converted to use optical signals.
- Reverse isolation provided by the RF conditioning unit that precedes the A/D converter ensures that little RFI generated by the A/D converter propagates back to the receiver via the RF interface.
- The digitiser is designed to be modular with each module being separately shielded and RFI qualified.
- Where possible, inter-module interfaces are designed to be optically isolated. This includes interfaces between the ADC and FPGA and serial control and monitoring lines to and from the analogue RF components.
- The use of switching power supplies will be restricted to the optically-isolated DSP processor module, with independent linear supplies in service everywhere else.

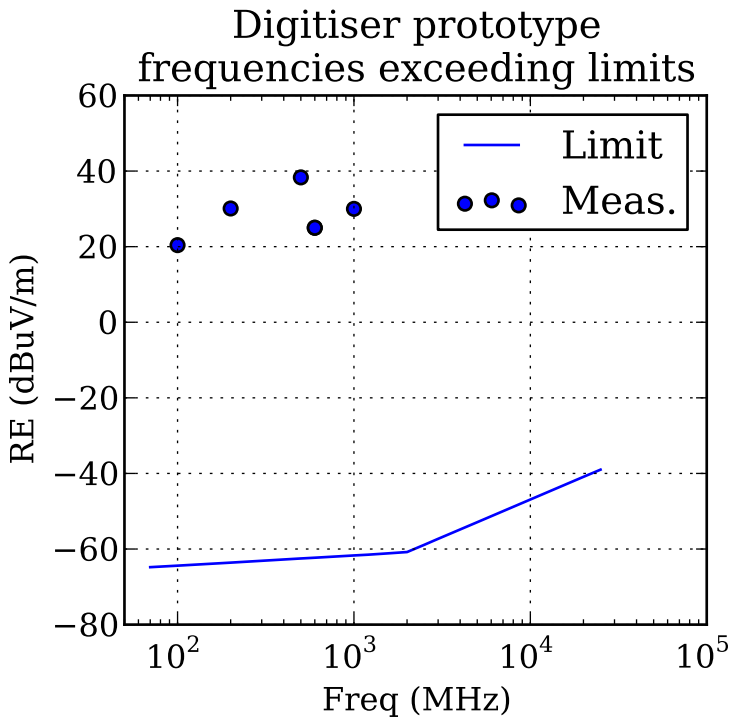


Figure C.3: A single ROACH-1 board, unshielded, radiates nearly 100dB above MeerKAT's threshold for devices located near the feed. The line shows the threshold limits and the dots indicate spot measurements taken of an unshielded ROACH board at known clock frequencies.

It is believed that an 80dB reduction will be possible through careful shielding and enclosure design, with the remaining 20dB being achieved through careful PCB design, layout and use of optical isolation techniques.

Measuring the radiated RFI levels from the digitiser for qualification will be difficult as most commercial test systems are not sensitive enough to detect such low radiated emissions. The MeerKAT engineering teams envisage that verification will need to be performed indirectly by measuring unshielded radiation levels from each module and adding the measured attenuation provided by the digitiser enclosures. Final, integrated verification is then to be performed on site using the first integrated MeerKAT dish assembly.

C.2 Array-wide adoption of SPEAD

SPEAD[1] has been adopted as the array-wide standard for MeerKAT. In some cases, such as communication between FPGA units, only a subset of the standard is employed to keep implemented state machines simple. Even with the reduced subset deployed, the use of SPEAD throughout the system ensures that any device can communicate with any other device and processing nodes can be easily interchanged or exchanged.

C.3 Switch port counts

Figure C.4 shows the traffic flow and assignment of hardware resources for the L-band case, if all instruments were to be used concurrently. It assumes the use of a next-generation processing platform (ROACH-3, or similar).[2] In this case, 389 40GbE ports are required. However, since not all instruments will be used concurrently, MeerKAT can accommodate some over-subscription. The narrowband (spectral line) correlator and beamformer will not be required concurrently, for example, saving 64 ports and 64 ROACH-3 boards.

Fringe stopping and delay compensation can be implemented in the time-domain on the D engines, or this function can be implemented in the F engines as was done with KAT-7. Since MeerKAT essentially has multiple correlators operating in parallel (wideband and various narrowband modes), in this latter case fringe stopping and delay compensation logic must be implemented multiple times in each of the narrowband and the wideband systems. If the D-engines have sufficient reserve logic capacity, it might be more effectively implemented here. Any instruments subscribing to the raw time data will then always get coherent data from all dishes, which is usually what is desired.

[1] `https://casper.berkeley.edu/wiki/SPEAD`

[2] board count, and hence port count, would increase if ROACH-1 or ROACH-2 were used; see Figure C.5.

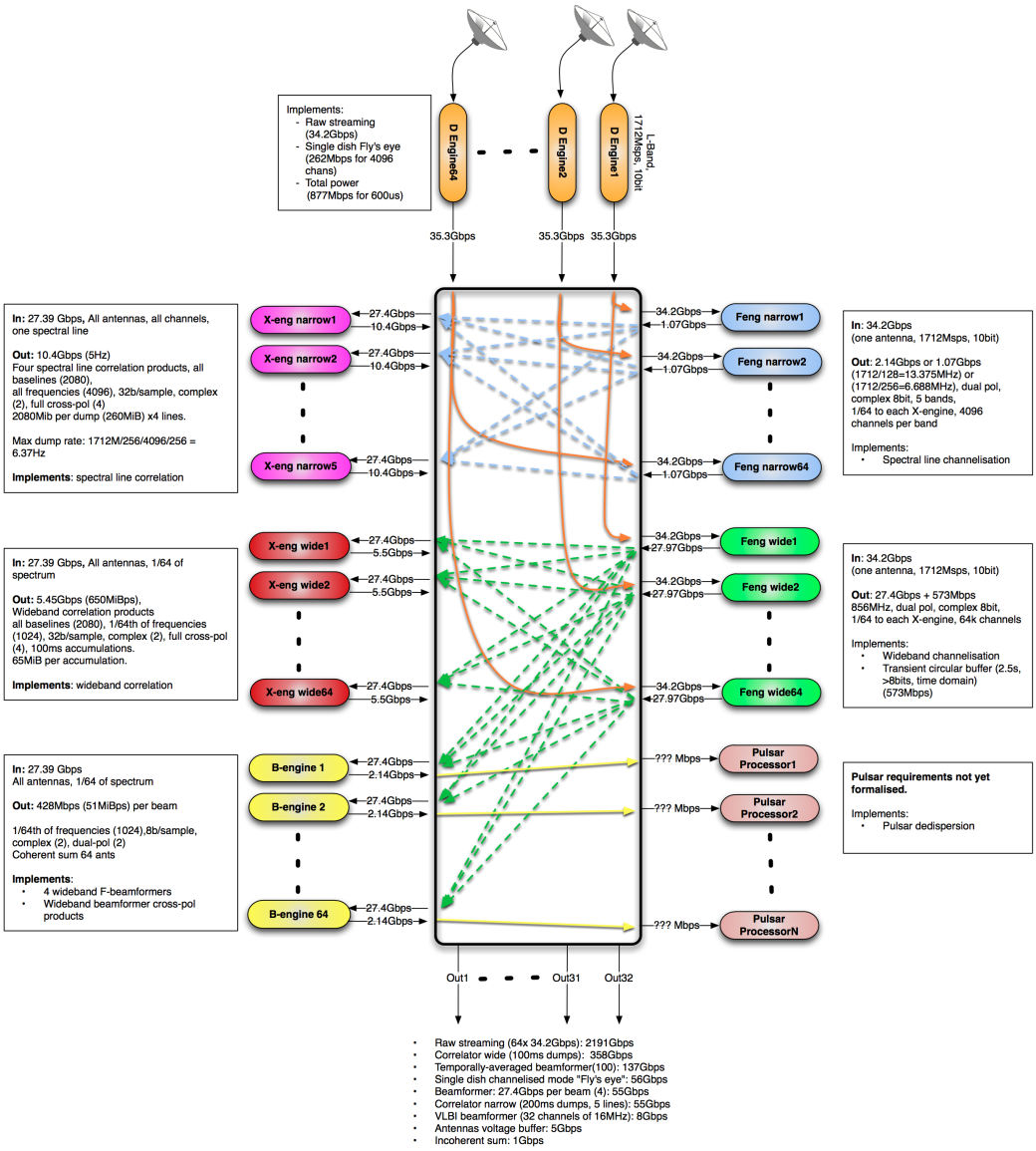


Figure C.4: The anticipated L-band switch traffic for the MeerKAT instrument, should all instruments be used concurrently. However, the beamformer and narrowband (spectral line) correlators, for example, are not required simultaneously and are likely to share hardware.

Function	40GbE ports
D-engine input streams	64
Wideband F-engines	64
Narrowband F-engines (5 lines)	64
Wideband X-engines	64
Narrowband X-engines	5
Beamformers (4)	64
Incoherent summer	1
CAM interfaces	2
40GbE output stream (SPT)	32
Total	**1866**

Table C.2: The estimated port counts for the L-band MeerKAT system illustrated in Figure C.1.

C.4 Multicasting

The incoming Ethernet links from all antennas connect directly to a large networking switch, which also hosts all the realtime compute resources. This allows data to be routed to and from any compute engine in the system. These processing resources will be allocated on-the-fly to signal processing jobs.

Multicasting is required in order to support the use of multiple instruments concurrently. An aim is to make all datastreams, including intermediate products, available in individual multicast groups allowing nodes to subscribe to any data of interest to them. With 64 antennas, 65536 frequency channels, 500 beams, 2080 baselines and any combination of subarrays, this is an ambitious goal requiring many millions of concurrent multicast groups. Current switches only allow for thousands of concurrent multicast groups and IP multicasting itself has address space limitations (some of which can be alleviated by switching to IPv6).

It is likely that the frequency channels, at least, will be collected into more coarse multicast groups. By grouping together the channels that a single X

engine would process into a single multicast group, each engine only needs to subscribe to a single multicast group, which also simplifies the FPGA interface. Similar concessions will likely be made with the beamformer's and correlator's outputs. Table C.3 shows estimated multicast channel requirements for each instrument, assuming 64 spectral groups. For the L-band case, this would provide $\frac{856MHz}{64} = 13.375$MHz subscribeable bands.

Function	Multicast Groups
Raw digitised stream	64
Transient buffer	64
Antenna coherency products	64
Wideband, 64 F-engs + 64 X/B-engs	
Channelised baseband	64
Beamformer (4 full time resolution beams)	256
Beamformer (100 time averaged beams)	1280
Correlator	64
Spectral line, 64 F-engs + 5 X/B-engs	
Channelised baseband (5 sub-bands)	5
Correlator	5
Total	**1866**

Table C.3: The estimated number of multicast groups required by the correlator and beamformer components, assuming 64 frequency channel groups over the full digitised bandwidth.

C.5 Datarates

When wide bandwidths are digitised for many antennas, the data rates required of the switching hardware quickly become very large. Table C.4 shows the estimated data rates produced by digital back-end subsystems when using MeerKAT at L-band.

The per-port and backplane requirements of the switching hardware for MeerKAT are significantly greater than the 10Gbps Ethernet system used for

Instrument	Maximum data rate
Raw digitised stream[a]	2191 Gbps
Channelised wideband[b]	1753 Gbps
Wideband correlator[c]	349 Gbps
Temporally-averaged beamformer[d]	137 Gbps
Channelised narrowband[e]	68 Gbps
Single-dish "Fly's eye"[f]	56 Gbps
Beamformer[g]	55 Gbps
Narrowband correlator[h]	55 Gbps
VLBI beamformer[i]	8 Gbps
Antennas voltage buffer[j]	5 Gbps
Channelised incoherent sum[k]	1 Gbps

Table C.4: **MeerKAT L-band data rates, excluding packetisation and protocol overhead. This realtime machine will need to switch nearly 5Tbps and will produce in excess of 600Gbps during normal operations.**

[a] 128 streams, 1712Msps, 10-bit
[b] 8-bit complex per sample
[c] Using a 100ms accumulation period, 65536 channels, 32-bit complex output.
[d] 100 beams, 4096 channels, 47.85μs averaging, 8-bit complex
[e] 5 bands of 13.375MHz, 8-bit complex per sample
[f] 598μs accumulations, 4096channels, 16-bit
[g] 856MHz, 8b complex, 4 beams
[h] 200ms accumulations, 5 spectral lines of 4096 channels each, 32-bit complex.
[i] 32 channels of up to 16MHz each, 8-bit complex
[j] 2048MiB, 128 buffers, 1/60Hz readouts, 8-bit real
[k] 598.13μs accumulations, 4096 channels, 32-bit complex, dual polarisation

KAT-7, even when processing the narrower L-band. While multiple 10Gbps links could be used in parallel, 40Gbps transceivers and switches are preferred throughout the digital back-end to reduce cabling requirements. In the X-band, the system is required to process at least 2GHz of analogue bandwidth, more than doubling all the numbers shown in Table C.4.

Large COTS 10Gbps switches, such as Arista's 7500 with 1088 ports or Cisco's Nexus 7000 series with 768 ports, are already available but in their current form, these switches do not meet the requirements for MeerKAT. Technological advancement in Ethernet switch technology from 10Gbps to 40Gbps and 100Gbps in large port counts supporting simultaneous full-crossbar traffic flows will be needed to implement MeerKAT's digital signal processing systems.

It is possible to custom-construct a scalable, distributed switch using smaller switching units to meet all MeerKAT's switching requirements but I expect advances to occur in the commodity, commercial sphere within MeerKAT's construction timeframe. For further details about large switch construction, see §5.3.3.

C.6 Scalability

Again considering the correlator component for MeerKAT, the increased number of antennas over KAT-7's design (from seven to 64) is the major scaling factor, resulting in a factor of 64 growth in the X-engines' compute requirements. Fortunately, the design of the X-engine cores allows this increased size to be split into two linear dimensions, requiring linearly more processing engines and also increasing the length of each engine linearly, which makes the problem more manageable. A linear increase in the number of switch ports is achieved by keeping the $O(N^2)$ routing problem inside the switch. These scaling concepts are outlined in Parsons *et al.* (2008).

The channelisation operations take place on a per-input basis, and thus scale linearly with the number of antennas. But specifications for finer spectral resolution over wide bandwidths result in a factor of 50 growth for the F-engines over KAT-7's F-engines. The increase in processed bandwidth (to over 2GHz) further increases the overall system size by another factor of five.

A detailed study of CASPER's filterbank scaling can be found in Primiani *et al.* (2011).

The architecture of the system allows for this required compute power to be sourced from additional processing nodes. Should it be necessary, individual processing tasks can also be pipelined and distributed over multiple boards (such as if the length of an X-engine were to grow beyond what a single FPGA could host). Using 2009 ROACH-1 technology, MeerKAT would require thousands of connected nodes, a significant electricity consumer, requiring large chiller plants in the hot Karoo as well as significant rack space. To reduce these requirements, the latest technology will be used whenever possible. The ROACH's successor, the ROACH-II (CASPER's latest 2011 board), is already available but MeerKAT will benefit from another technology iteration, further reducing the size of the switch, the number of boards and the power and cooling budget by almost a factor of four over a ROACH-2 solution.

C.7 New processing hardware

It is likely that MeerKAT will use a mixture of FPGAs, GPUs and CPUs for the bulk of the signal processing tasks. FPGAs are well matched to the foreseen IO vs compute ratios typically found in beamformers and correlators of this size. For pulsar processing, it is likely that the processing hardware will take the form of commercial GPU-accelerated CPUs. These machines offer dense compute power at a very low price-point. They offer a further advantage, that software to do the more complex de-dispersion algorithms is easier to develop and debug than on FPGAs.

MeerKAT's digital processing subsystem can benefit from advances in processor technology. While it is technically possible to construct MeerKAT with current processing platforms, such as ROACH-2, the use of more powerful processors will decrease the system size, lowering costs and reducing off-chip interconnects, which are expensive and bulky. This is demonstrated in Figure C.5. The next iteration in the line of data processing boards produced

by the CASPER collaboration[3] could have a production schedule making it available in 2015; a timeframe suitable for MeerKAT's later deployments.

Increasing the IO bandwidth to off-chip memories will be crucial in later phases of MeerKAT as the instantaneous bandwidth requirements increase for the X-band.

GPUs have also become an option for use as the X-engines in MeerKAT's correlator. Clark *et al.* (2011) has recently demonstrated that for 128 (and more) inputs, it is possible to achieve 80% compute efficiencies with current technologies.

C.8 Reliability

The telescope is specified to have an uptime of more than 95% for more than 90% of the array. Most of this margin has been allocated to the mechanical and cryogenic components of the system and so the digital processing subsystem and signal transport subsystem both require an uptime of well over 99%.

The central processor is to be designed for availability. The modular nature of this system makes it possible to keep redundant processing units (CPUs, GPUs, FPGA boards, etc) connected and available at all times. In the event of a board failure, its processing load can be switched on-the-fly to a standby unit using only a software command. The faulty board can then be replaced during the next scheduled maintenance period. This results in a very relaxed two month MTBF specification for the processing system.

The digitiser at the feed, however, must be designed for reliability as the entire antenna will be unavailable in the event of a failure on this piece of equipment. It has a more demanding 12 month MTBF requirement.

Most importantly, the core network switch is a single point of failure that would disable the entire facility should it fail. It is specified to have an MTBF of 15 years. But redundant power supplies, hot-swappable line units and the use of high reliability components are typical in large, commercial, chassis units. Should a very large CLOS network be constructed from smaller units,

[3]https://casper.berkeley.edu/wiki/Hardware

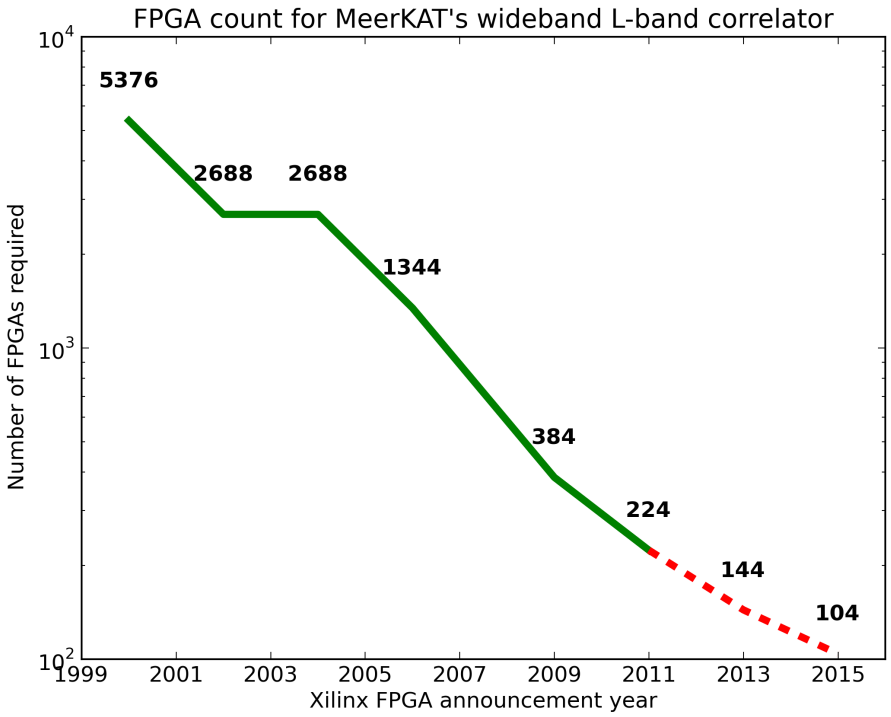


Figure C.5: The number of hardware boards required for MeerKAT's wideband correlator, including *D-engines*. MeerKAT can continue to take advantage of advances in FPGA technology. The benefits of deploying the latest technologies include reduced system costs, power savings, simplified maintenance and reliability increases through reduced connector counts.

N+1 redundancy can be added simply and cost-effectively.

The modular nature of the design and use of Ethernet interconnect allows for failed processing components to be replaced with a minimal effect on the rest of the system, even during operation. This reduces the impact of unscheduled maintenance. The commodity nature of most of the hardware means that spares and replacement parts are easily available and have short lead-times, reducing the required stock of spares. Portable IP allows for old, failed devices to be substituted with the latest models, negating the need to keep a lifetime supply of spare components.

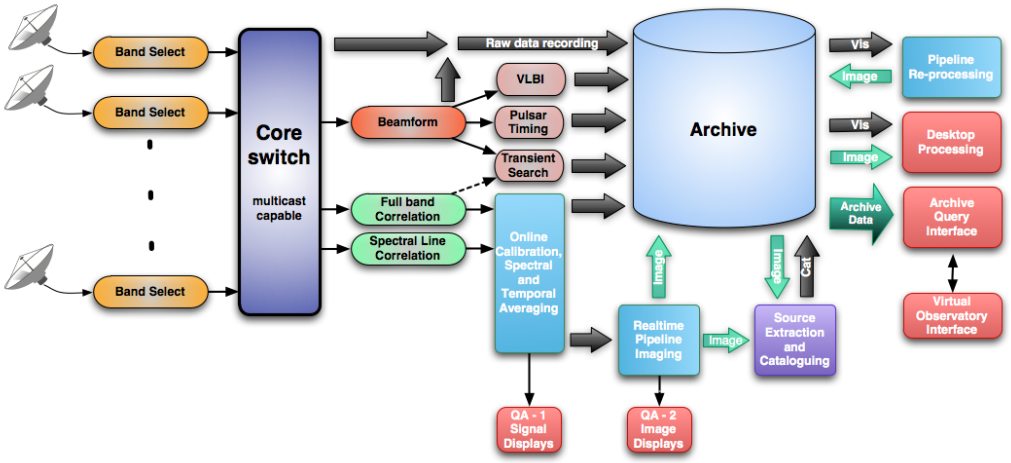


Figure C.6: A functional overview of the MeerKAT digital back-end and signal processing system. Digitised data is propagated into a distributed supercomputer for realtime analysis and storage. This machine consists of both online (realtime) and offline components, where interfaces are included for reprocessing stored products later.

C.9 Conclusion

MeerKAT will represent significant advance from KAT-7 in terms of science capability. Its digital signal processing subsystem reflects this, and will have hundreds of FPGAs deployed in a specially-designed building. This processor will receive pre-digitised signals, as each antenna includes all analogue and ADC components to output Ethernet packetised data products.

Reliability and maintainability are key aspects of the MeerKAT telescope and these issues are eased by using an Ethernet interconnect.

MeerKAT will also be the first telescope to demonstrate true multicasting and concurrent operation of multiple instruments that are hosted on different processing boards.

The use of this scalable processing solution means that MeerKAT could be scaled out to SKA-1 and beyond, should that option become attractive to the SKA organisation.